Design, Implementation, and Analysis of a Block Cipher Based on a Secure Chaotic Generator

Fethi Dridi, Safwan El Assad, Wajih El Hadj Youssef, Mohsen Machhout, René Lozi

▶ **To cite this version:**

Fethi Dridi, Safwan El Assad, Wajih El Hadj Youssef, Mohsen Machhout, René Lozi. Design, Implementation, and Analysis of a Block Cipher Based on a Secure Chaotic Generator. Applied Sciences, 2022, 12 (19), pp.9252. 10.3390/app12199952 . hal-03799417v2

HAL Id: hal-03799417
https://hal.science/hal-03799417v2

Submitted on 9 Oct 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

applied sciences

MDPI

Article

Design, Implementation, and Analysis of a Block Cipher Based on a Secure Chaotic Generator

Fethi Dridi [1,2], Safwan El Assad [2,*], Wajih El Hadj Youssef [1], Mohsen Machhout [1] and René Lozi [3]

[1] Electronics and Microelectronics Laboratory (EμE), Faculty of Sciences of Monastir, University of Monastir, Monastir 5019, Tunisia
[2] IETR UMR 6164, CNRS, University Nantes, F-44000 Nantes, France
[3] LJAD, CNRS, Université Côte d'Azur, F-06000 Nice, France
* Correspondence: safwan.elassad@univ-nantes.fr

Abstract: This work proposes a new secure chaos-based encryption/decryption system, operating in cipher block chaining (CBC) mode, and analyze its performance. The cryptosystem includes a robust pseudorandom number generator of chaotic sequences (PRNG-CS). A strong chaos-based S-box is proposed to perform a circular substitution operation (confusion process). This PRNG-CS consists of four discrete 1-D chaotic maps, weakly coupled by a predefined coupling matrix M, to avoid, on the one hand, the divide-and-conquer attack and, on the other hand, to improve the generated sequence's randomness and lengths. The noun is also used in the construction of the S-box. Moreover, a 2-D modified cat map and a horizontal addition diffusion (HAD) preceded by a vertical addition diffusion (VAD) are introduced to perform the diffusion process. The security analysis and numerous simulation results of the main components (PRNG-CS and S-box) as well as the whole cryptosystem reveal that the proposed chaos-based cryptosystem holds up against various types of statistical and cryptographic attacks.

Keywords: chaos-based encryption/decryption system; PRNG-CS; S-box; modified 2-D cat map; performance; security analysis

check for updates

Citation: Dridi, F.; El Assad, S.; El Hadj Youssef, W.; Machhout, M.; Lozi, R. Design, Implementation, and Analysis of a Block Cipher Based on a Secure Chaotic Generator. *Appl. Sci.* **2022**, *12*, 9952. https://doi.org/10.3390/app12199952

Academic Editor: Arcangelo Castiglione

Received: 18 September 2022
Accepted: 29 September 2022
Published: 3 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Nowadays, the necessity to protect data has become a primary challenge. The security of communication of sensitive data is not only limited to military and diplomatic information and national defense issues, but also concerns the private lives of individuals and companies. With this regard, cryptography schemes have been developed. Indeed, cryptographic techniques leads to well-ensured messages and data confidentiality. Chaos was integrated into cryptography for the first time by Matthews in the 1990s [1]. Among the existing state-of-the-art approaches, chaos-based cryptosystems have proved their efficiency in data security [2–4] due, on the one hand, to the powerful proprieties of chaotic sequences such as deterministic, unpredictable, random-like behavior, and high sensitivity to initial conditions and control parameters (secret key) [5–7] and, on the other hand, to their efficient properties of confusion and diffusion performed by chaotic maps [8–12].

Symmetric chaos-based encryption/decryption systems are classified under two categories: block ciphers and stream ciphers. The former is the main symmetric key cryptosystem, as their design is related to Shannon's theory of information security based on confusion and diffusion operations [8,13]. Their security properties were well-studied, and these ciphers encrypt the plaintext on a block of bits: 128, 256, 512, 1024, etc. [14,15].

Compared to traditional symmetric cryptography, chaos-based encryption/decryption schemes are more flexible (generic design, variable block size, and secret key can be easily enlarged), easier to implement, and have intrinsic security related to the properties of chaotic sequences [14,16]. Moreover, some of them use a substitution layer based on key-

dependent S-boxes and not on fixed S-boxes as used in the Advanced Encryption Standard (AES) algorithm [17].

A key element of all chaos-based cryptosystems is the chaotic generator, which provides the dynamic keys (encryption keys) for the confusion and diffusion layer. The security, as well as the efficiency of the system, largely depend on the chaotic generator used.

However, most chaos-based cryptosystems reported in the literature refer to simple chaotic maps to supply the confusion and diffusion layer [18,19]. Such cryptosystems can be destroyed by side-channel attacks [20,21].

A fast chaos-based image encryption system with a dynamic state variables selection mechanism was proposed by Chen et al. [22] and it achieved a high confusion and diffusion process. Similar to the suggested cryptosystem by Fridrich [9] and Zhang et al. [23], three prototypes of a chaos-based cryptosystem were put up by Farajallah et al. [24]. The modified 2-D cat map was used to create the confusion process and 32-bit and 8-bit logistic maps were used as the diffusion layer for the two first versions, and a modified finite skew tent map (FSTM) in the third version.

A secure cryptosystem based on chaotic components was proposed by Qiao et al. [25]. In this cryptosystem, the AES S-box was used as a confusion layer. [17]. A global diffusion operating on the entire image was established using a horizontal addition diffusion (HAD), afterward, a vertical addition diffusion (VAD), introduced by Omrani et al. [26], then a permutation operation based on the modified 2-D cat map to reinforce the diffusion effect.

Wang et al. [27] published a block cipher cryptosystem using dynamic S-boxes based on a tent map. Their system operated on blocks using different generated S-boxes and used 32 iterations of substitution and left cyclic shift. A similar cryptosystem was proposed by Zhang et al. [28] based on a circular S-box as a confusion process and a keystream buffer as a diffusion layer. The system operates on the entire plain image. Another block cipher based on the chaotic logistic map was presented by Alawida et al. [29]. To enhance the performance of the logistic map, a new chaotification approach based on a multiplicative inverse function was applied. The diffusion and confusion process required data sequences, which were created by perturbing the chaotic variables with the secret key. The confusion process was controlled by the chaotic points themselves, whereas the diffusion process was controlled by an ergodic chaotic map.

Alshammari et al. [30] developed a new lightweight block cipher to protect the security of data acquired by Internet of things (IoT) sensor. It consisted of a customized AES algorithm with a chaos-based S-box and a chaotic map. Despite having good cryptographic characteristics and a high amount of randomness, this was related primarily to the AES underlying structure and not to the modified S-box. Additionally, AES was not well recognized for its lightweight implementation and could not be appropriate for IoT devices with constrained resource in terms of computing capabilities.

In this paper, we propose an efficient encryption/decryption system based on a secure pseudorandom number generator of chaotic sequences (PRNG-CS) and a strong key-dependent S-box. The proposed chaos-based cryptosystem design takes into account the weaknesses and strengths of the systems mentioned above. The proposed PRNG-CS is able to prevent the divide-and-conquer and the SCA attacks. This PRNG-CS is used to build the key-dependent S-box. It also provides the encryption/decryption keys necessary for the substitution and permutation operation [31–34].

The remainder of this paper is organized as follows: In Section 2, we present the proposed chaos-based cryptosystem architecture with a deep insight of the PRNG-CS, the S-box with the circular substitution process, and the diffusion process, then we give the pseudocode of the encryption/decryption algorithms. In Section 3, we present the experiments results and security analysis of the proposed cryptosystem and its computing performance. Finally, Section 4 summarizes the whole paper.

2. Proposed Chaos-Based Cryptosystem

The architecture of the proposed chaos-based cryptosystem is shown in Figure 1, for the encryption process and in Figure 2, for the decryption process. This structure achieves high confusion-diffusion effects. On the encryption side, the confusion is achieved by a circular substitution layer based on a proposed strong S-box, which is the non-linear element of the cryptosystem as in the Advanced Encryption Standard (AES) algorithm. The diffusion process consists of three steps: a permutation layer based on the modified 2-D-Cat map, a horizontal addition diffusion (HAD), and a vertical addition diffusion (VAD). The confusion and diffusion layers use the proposed PRNG-CS. On the decryption side, apart from the PRNG-CS, reverse diffusion and reverse substitution are achieved by reverse processes of those used in encryption, as indicated in Figure 2.

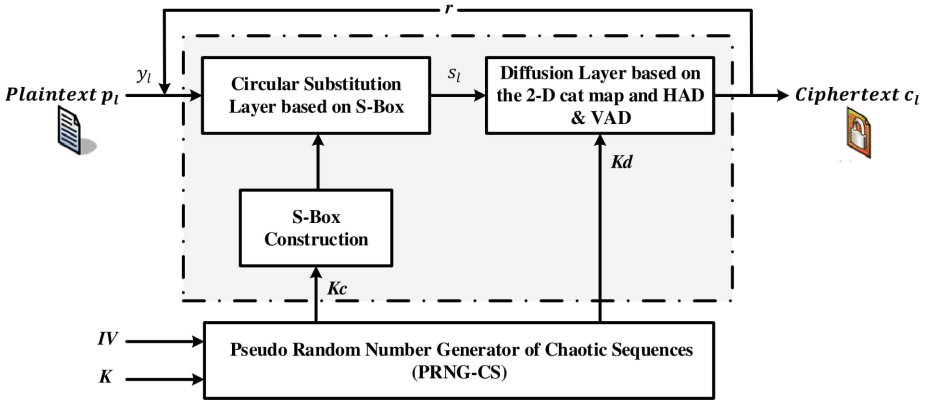


Figure 1. Diagram of the encryption process.

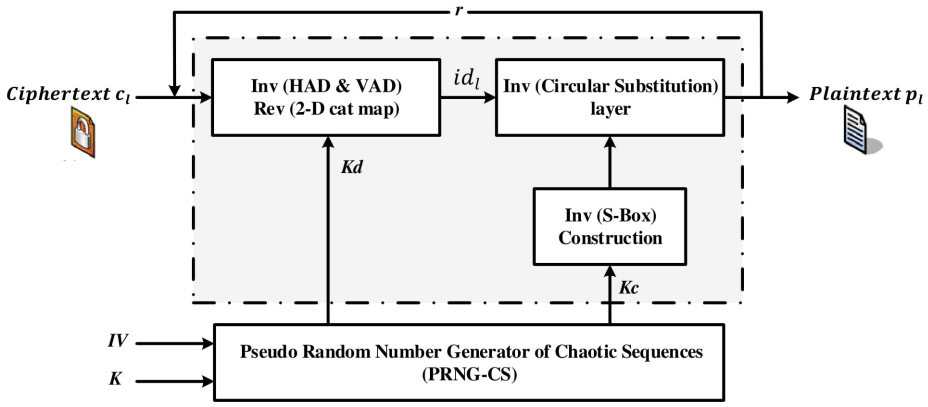


Figure 2. Diagram of the decryption process.

The block cipher operation can be repeated r times, if necessary, to obtain the final secure encrypted image.

Each component of the proposed cryptosystem is described in depth in the following sections.

2.1. Description of the Proposed Pseudorandom Number Generator of Chaotic Sequences (PRNG-CS)

The proposed PRNG-CS (see Figure 3), which is a fundamental element of any chaos-based cryptosystem, uses an initial vector (IV) and a secret key (K) as inputs and outputs the dynamical keys (encryption/decryption keys) Kc for the confusing process and Kd for the diffusion process. The systems' IV supplies the initial vectors of the four chaotic maps IVP, IVS, IVL, and IVT each $N = 32$ bits in size, and the secret key K gives all

the initial conditions and parameters of the chaotic maps but also the initial value of the linear-feedback shift register (LFSR) and the parameters of the coupling matrix M and transient phase Tr, as summarized in Table 1.

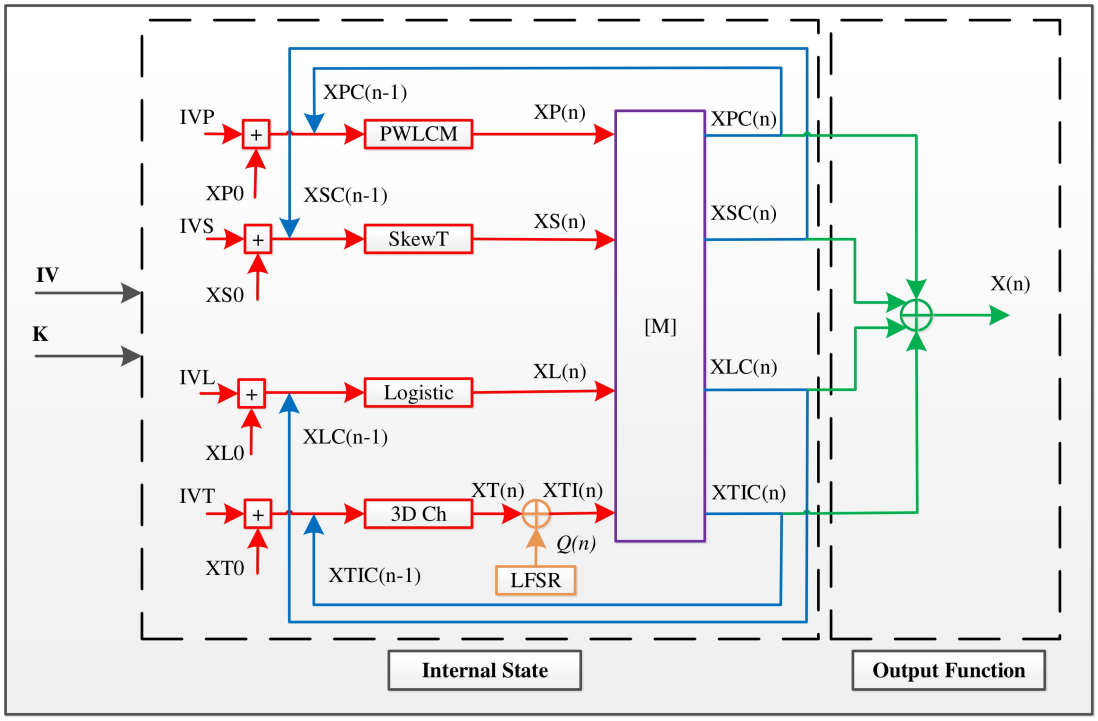


Figure 3. The architecture of the proposed pseudorandom number generator of chaotic sequences.

Table 1. Initial conditions and parameters that form the secret key.

Symbol	Definition
$XP0$, $XS0$, $XL0$, and $XT0$	Initial conditions of the four chaotic maps: PWLCM, skew tent, logistic, and 3-D Chebyshev in $[1,2^N-1]$.
P_p	PWLCM map control parameter in $[1,2^{N-1}-1]$.
P_s	Skew tent map control parameter in $[1,2^N-1]$.
ϵ_{ij}	Coupling matrix M parameters in $[1,2^k]$ with $k \leq 5$.
Q0	The initial value of the used LFSR is defined by: $Q(n) = x^{32} + x^{22} + x^2 + x + 1$.
T_r	The transient phase of 10 bits.

The four chaotic maps' initial conditions, $XP(0)$, $XS(0)$, $XL(0)$, and $XT(0)$ are provided by:

$$\begin{cases} XP(0) = IVP + XP0 \\ XS(0) = IVS + XS0 \\ XL(0) = IVL + XL0 \\ XT(0) = IVT + XT0 \end{cases} \quad (1)$$

The output $X(n)$ is calculated by:

$$X(n) = XPC(n) \oplus XSC(n) \oplus XLC(n) \oplus XTIC(n) \quad (2)$$

The coupling system is defined as follows:

$$\begin{bmatrix} XPC(n) \\ XSC(n) \\ XLC(n) \\ XTIC(n) \end{bmatrix} = M \times \begin{bmatrix} XP(n) \\ XS(n) \\ XL(n) \\ XTI(n) \end{bmatrix} \tag{3}$$

where

$$M = \begin{bmatrix} M_{11} & \epsilon_{12} & \epsilon_{13} & \epsilon_{14} \\ \epsilon_{21} & M_{22} & \epsilon_{23} & \epsilon_{24} \\ \epsilon_{31} & \epsilon_{32} & M_{33} & \epsilon_{34} \\ \epsilon_{41} & \epsilon_{42} & \epsilon_{43} & M_{44} \end{bmatrix} \tag{4}$$

with $M_{11} = (2^N - \epsilon_{12} - \epsilon_{13} - \epsilon_{14})$, $M_{22} = (2^N - \epsilon_{21} - \epsilon_{23} - \epsilon_{24})$, $M_{33} = (2^N - \epsilon_{31} - \epsilon_{32} - \epsilon_{34})$, and $M_{44} = (2^N - \epsilon_{41} - \epsilon_{42} - \epsilon_{43})$.

$XP(n)$, $XS(n)$, $XL(n)$, and $XT(n)$ are denoted as the maps' output values at instant n of the *PWLCM*, skew tent, logistic, and 3-D Chebyshev maps, respectively, and defined as follows:

The first sample is given by:

$$XP(1) = PWLCM\Big\{\mathrm{mod}\left(XP(0), 2^N\right),\ P_p\Big\} \tag{5}$$

$$XS(1) = SkewT\Big\{\mathrm{mod}\left(XS(0), 2^N\right),\ P_s\Big\} \tag{6}$$

$$XL(1) = Logistic\Big\{\mathrm{mod}\left(XL(0), 2^N\right)\Big\} \tag{7}$$

$$XT(1) = 3D\ Ch\Big\{\mathrm{mod}\left(XT(0), 2^N\right)\Big\} \tag{8}$$

Then, for $2 \leq n \leq N_s$, the samples are calculated by (N_s: the number of the desired samples):

$$XP(n) = PWLCM\Big\{\mathrm{mod}\Big(XPC(n-1), 2^N\Big),\ P_p\Big\} \tag{9}$$

$$XS(n) = SkewT\Big\{\mathrm{mod}\Big(XSC(n-1), 2^N\Big),\ P_s\Big\} \tag{10}$$

$$XL(n) = Logistic\Big\{\mathrm{mod}\Big(XLC(n-1), 2^N\Big)\Big\} \tag{11}$$

$$\begin{cases} XT(n) = 3D\ Ch\{\mathrm{mod}\left(XTIC(n-1),\ 2^N\right)\} \\ XTI(n) = XT(n) \oplus Q(n) \end{cases} \tag{12}$$

The discrete equations of *PWLCM*, skew tent, logistic, and 3-D Chebyshev maps are reported in the literature [35–38].

The size of the secret key of the proposed PRNG-SC is:

$$|K| = |XP0| + |XS0| + |XL0| + |XT0| + |Q0| + |P_p| + |P_s| + |T_r| + (9 \times |\varepsilon_{ij}|) = 278\ bits \tag{13}$$

where $|XP0| = |XS0| = |XL0| = |XT0| = |Q0| = |P_s| = 32\ bits$, $|P_p| = 31\ bits$, $|T_r| = 10\ bits$, and $|\varepsilon_{ij}| = 5\ bits$.

Therefore, the keyspace contains 2^{278} different values, which is large enough to make brute-force attacks infeasible.

2.2. Circular Substitution Process Based on the S-Box and Their Inverse Processes

In the following, we first describe the construction process of the S-box, then we give the equation of the circular substitution process based on this S-box. Recall that the S-box is generally the only component of a cryptosystem that gives rise to a nonlinear mapping

between inputs and outputs. Its main role is to make the cryptosystem immune against differential and linear cryptanalysis, so the security of the cryptosystem is strongly increased.

2.2.1. S-Box Construction

Mathematically, an $n \times n$ S-box is a nonlinear mapping $S : \{0, 1\}^n \rightarrow \{0, 1\}^n$, where $\{0, 1\}^n$ represents the vector spaces of elements from GF (2). Since the key-dependent S-boxes do not offer any specific properties to the attackers, then, they are more secure than fixed S-boxes. Therefore, many methods have been proposed to design and create key-dependent S-boxes in conventional cryptography, such as SEAL [39], which used a Secure Hash Algorithm (SHA) and especially in chaos-based cryptography [40–46]. All of these later methods are based on iterating discrete chaotic maps to generate a 1-D table of size 256, containing unique chaotic values belonging to 0 and 255. The proposed S-box is based on our PRNG-CS and uses the approach of Çavuşoğlu et al. [42].

The block diagram of the 8×8 S-box design algorithm is given in Figure 4.

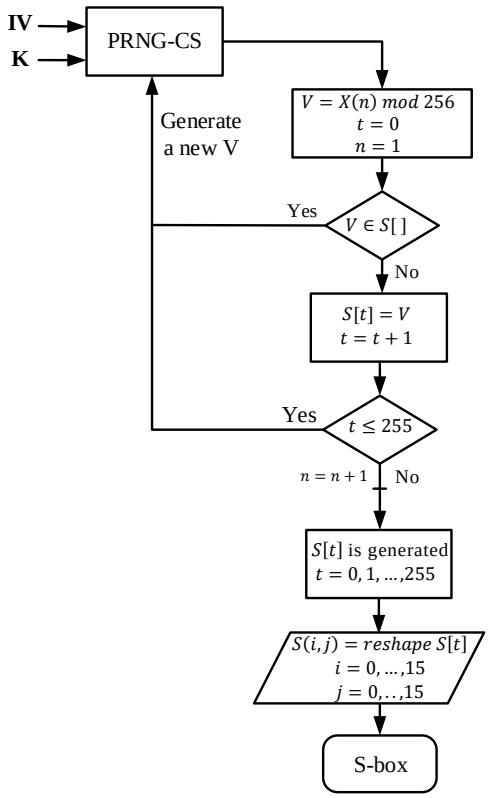


Figure 4. Block diagram of the proposed chaos-based 8×8 S-box construction.

The design steps of the S-box generation algorithm are as follows:

1. Specify a one-dimensional empty array called 1-D S-box $S[\,]$.
2. The eight-bit V-value is obtained from the PRNG-CS output sequence $X(n)$.
3. Check for the existence of the last obtained V-value in the content of the 1-D S-box $S[\,]$.
4. If this value exists, it is rejected. Otherwise, it is added to the S-box 1-D $S[\,]$.
5. Afterward, new 8-bit values are generated and the whole process is repeated until all 256 values are completed in the form of a sequence $S[t] = \{S[0], \ldots, S[255]\}$, where each value $\in [0, 255]$ is unique to others in it, so the constructed 8×8 S-box is obviously bijective (by construction).

The number n of needed samples to construct the S-box is the size of the dynamic key K_c: $|K_c| = n \times 32$-bit.

In Table 2, we give an example of an S-box obtained by our proposed algorithm, presented in its usual form, a 16×16 matrix of byte values in hexadecimal presentation.

Table 2. An example of an S-box is obtained by the proposed algorithm.

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	2E	3E	5D	03	7C	B7	AD	9B	E0	62	67	50	95	BC	24	83
1	A6	CB	DD	F5	BE	39	9A	C3	2C	64	EA	EC	05	3B	26	2A
2	F8	B4	D0	C5	30	FD	E6	69	70	74	7A	77	F7	1A	E5	DF
3	15	7F	8B	32	51	F1	C9	93	40	5C	48	8E	2F	65	E2	CC
4	FB	D8	BD	04	C7	0B	E1	06	90	B6	A1	EF	14	2B	D5	82
5	4B	D1	B8	59	92	E3	CD	B9	38	1F	71	73	EE	F9	46	45
6	C0	A8	00	6E	9C	6A	9F	A9	78	6F	87	5E	D9	BA	47	42
7	68	DA	F0	53	58	31	D3	36	34	20	33	8D	D4	8C	4C	3D
8	1D	4E	61	E9	5A	5F	4D	6B	AA	C4	91	3F	4A	1E	02	0F
9	09	72	88	CA	F2	85	99	9D	FF	CF	B2	2D	52	23	E4	3C
A	C8	EB	AF	81	1C	FE	9E	B5	7B	55	ED	07	41	89	94	7E
B	12	11	D7	49	AB	4F	21	DE	0C	D6	17	BF	56	6D	F3	01
C	1B	A5	3A	8A	27	E7	8F	0E	A3	75	BB	84	76	CE	25	B1
D	FA	DB	18	B0	A0	C1	44	54	A7	28	43	13	22	A2	35	FC
E	C6	C2	66	63	AC	A4	7D	F6	96	29	10	86	16	D2	37	6C
F	79	08	97	E8	DC	0A	19	0D	B3	98	80	AE	F4	57	5B	60

2.2.2. Circular Substitution Equation Based on the Constructed S-Box

Once the S-box is constructed, then it is used to perform the substitution operation. Unlike the circular substitution used by Zhang et al. [28], which was performed in ECB mode on the entire plain image, we achieved here the circular substitution in CBC mode on blocks of size 1024 pixels each. The advantages of proceeding in CBC mode are, on the one hand, because it is a secure mode (ECB mode is not secure), and on the other hand, the encryption/decryption processes are carried out on blocks that can be stored and computed on IoT devices which are constrained resource in terms of computing capabilities, and also energy and memory capacities. In addition, in ECB mode, the decryption process requires receiving the entire ciphered image before decrypting it.

To start the substitution process, the input plain image is split into b_l blocks p_l, each of size $|p_l| = 1024$ pixels (32×32 bytes). Notice that the elements of the S-box are considered to be circular (the last element followed the first element) with a head pointer h initialized to a constant (or a random) value in the range of 0 to 255. Each pixel $p_l(k)$ of a given block l is substituted by an element of the S-box, $s_l(k)$, according to the pixel value $p_l(k)$, the previous substituted pixel $q_{l-1}(k)$, and the pointer h (see Equations (14)–(16)). After a pixel is substituted, the value of the pointer h is adapted to a new value using the second row of Equation (14), where $m_h \in [0,\ 255]$ is a random value.

$$\begin{cases} s_l(k) = S[y_l(k) + h] \ mod\ 256 \\ \\ h = s_l(k) \oplus m_h \end{cases} \tag{14}$$

where

$$y_l(k) = p_l(k) + q_{l-1}(k) \tag{15}$$

$$q_{l-1}(k) = \begin{cases} ivb & if\ l = 0 \\ \\ c_{l-1}(k) & if\ l > 0 \end{cases} \tag{16}$$

where $c_{l-1}(k)$ is the previous ciphered pixel obtained after the horizontal and vertical addition diffusion of the block sp_{l-1}, which is the permuted of the substituted block s_{l-1}, $k = 1, 2, \ldots, |p_l|$ $l = 1, 2, \ldots, b_l$

$b_l = (image\ size/block\ size) = (R \times C \times P/|p_l|)$ (R: number of rows, C: number of columns, and P: number of planes ($P = 1$ for a grayscale image and $P = 3$ for an RGB image).

From Equations (14)–(16), we can observe that all the same pixel values of the plain image are substituted by different values, and then the security is strengthened against cryptographic attacks. In addition, there is an intrinsic diffusion effect in each substituted block.

2.2.3. Inverse Substitution Process

The reverse substitution operation is carried out by first, the construction of the inverse S-box, IS, which is derived from the S-box $S(t)$ by a simple operation satisfying the following equation:

$$IS[S(t)] = t \tag{17}$$

used during the inverse substitution process, then, performing the inverse substitution itself using the following formula to recover each plain pixel $p_l(k)$:

$$\begin{cases} p_l(k) = (IS[s_l(k)] + 256 - h)\ mod\ 256 \\ \\ h = s_l(k) \oplus m_h \end{cases} \tag{18}$$

2.3. Diffusion Process and Its Inverse

A permutation layer based on a modified 2-D cat map followed by a horizontal addition diffusion (HAD) and a vertical addition diffusion (VAD) is used to achieve the diffusion process of the proposed cryptosystem (VAD).

2.3.1. Permutation Layer Based on the Modified 2-D Cat Map and Its Reverse

To permute a substituted block $s_l(k)$, utilizing the improved 2-D cat map, we transform it to a matrix form $s_l(i,\ j)$ such that [24]:

$$s_l(i,\ j) = s_l(k),\ \ k = (i-1) \times M + j;\ \ i = 1, \cdots M;\ \ \ j = 1, \cdots M \tag{19}$$

The equation of the modified 2-D cat map [47] and its optimized implementation [24] are given below:

$$\begin{bmatrix} i_n \\ \\ j_n \end{bmatrix} = Mod\left(\begin{bmatrix} 1 & u_d \\ \\ v_d & (1 + u_d v_d) \end{bmatrix} \begin{bmatrix} i \\ \\ j \end{bmatrix} + \begin{bmatrix} r_{id} + r_{jd} \\ \\ r_{jd} \end{bmatrix}, \begin{bmatrix} M \\ \\ M \end{bmatrix} \right) \tag{20}$$

$$\begin{cases} i_n = Mod((i + u_d \times j + Z_1), M) \\ \\ j_n = Mod((Z_3 + Z_2 \times j), M) \end{cases} \tag{21}$$

with $Z_1 = r_{id} + r_{jd}$, $Z_2 = u_d \times v_d + 1$, and $Z_3 = v_d \times i + r_{jd}$.

The values of Z_1 and Z_2 are calculated once per round and the value of Z_3 is calculated M times per round ($with\ M = \sqrt{|p_l|} = 32 = 2^q - 1$).

The 2-D cat map is bijective since each permuted element's (i_n, j_n) position is unique. Despite it being reversible, the modulo operation on the cat map prevents it from being an invertible function. Thus, the same following relation can be used to perform the permutation and the reverse permutation:

$$sp_l(i_n,\ j_n) = s_l(i,\ j),\ \ i = 1, \cdots M,\ \ \ j = 1, \cdots M \tag{22}$$

The dynamic key Kd is formed by four elements $Kd = \left\{u_d, v_d,\ r_{id}, r_{jd}\right\}$, $each \in [1,\ M-1]$ and it is fed by the PRNG-CS. The size Kd is $|Kd| = 4 \times q =$ 20-bit.

2.3.2. Horizontal and Vertical Addition Diffusion HAD and VAD and Their Inverses

The operations of horizontal and vertical addition diffusion [26] on a given permuted block $sp_l(i,j)$ are described by the following mathematical model, and shown in Figure 5:

$$sph_l(i,j) = HAD[sp_l(i,j)] = \begin{cases} sp_l(i,j) + sp_l(i,j-1) \ mod\ 256 & for\ 1 \le i \le M\ and\ 1 < j \le M \\ sp_l(i,j) + sp_l(i-1,\ M) \ mod\ 256 & for\ 1 < i \le M\ and\ j = 1 \\ sp_l(i,j) + sp_l(M,M) mod\ 256 & for\ i = j = 1 \end{cases} \quad (23)$$

$$c_l(i,j) = VAD[sph_l(i,j)] = \begin{cases} sph_l(i,j) + sph_l(i-1,j) \ mod\ 256 & for\ 1 < i \le M\ and\ 1 \le j \le M \\ sph_l(i,j) + sph_l(M,\ j-1) \ mod\ 256 & for\ i = 1\ and\ 1 < j \le M \\ sph_l(i,j) + sph_l(M,M) mod\ 256 & for\ i = j = 1 \end{cases} \quad (24)$$

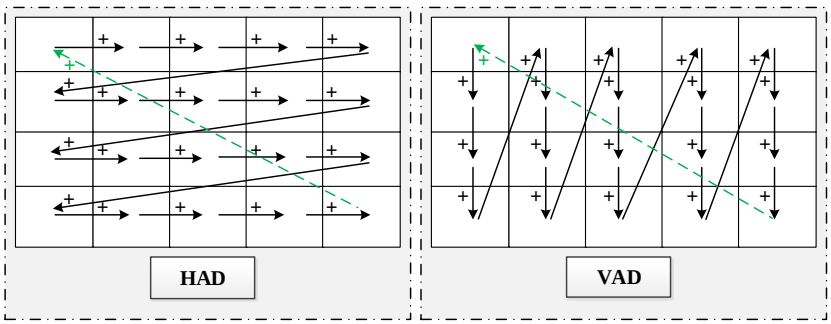


Figure 5. Horizontal and vertical addition diffusion.

The inverse relations of the vertical and horizontal addition diffusion are given by:

$$sph_l(i,j) = InvVAD[c_l(i,j)] = \begin{cases} c_l(i,j) - c_l(i-1,j) \ mod\ 256 & for\ 1 < i \le M\ and\ 1 \le j \le M \\ c_l(i,j) - c_l(M,j-1) \ mod\ 256 & for\ i = 1\ and\ 1 < j \le M \\ c_l(i,j) - c_l(M,M) \ mod\ 256 & for\ i = j = 1 \end{cases} \quad (25)$$

$$sp_l(i,j) = InvHAD[sph_l(i,j)] = \begin{cases} sph_l(i,j) - sph_l(i,j-1) \ mod\ 256 & for\ 1 \le i \le M\ and\ 1 < j \le M \\ sph_l(i,j) - sph_l(i-1,M) \ mod\ 256 & for\ 1 < i \le M\ and\ j = 1\ \ and\ j = 1 \\ sph_l(i,j) - sph_l(M,M) mod\ 256 & for\ i = j = 1 \end{cases} \quad (26)$$

Finally, we summarize in Algorithms 1 and 2, the full operation of encryption and decryption process of the proposed chaos-based cryptosystem, respectively.

Algorithm 1 Encryption process.

Input : IV = Initial Vector of $PRNG - CS$;
K = Secret key of the $PRNG - CS$;
ivb = Initialization vector used in the first block;
b_l = Number of blocks;
p_l = Plaintext block;
$PRNG - CS(IV, K)$ = (Kc, Kd).

Output: The block under test is encrypted.

begin
  Generate the needed dynamic keys $|Kc|$ to construct the S-box and $|Kd|$ to perform the permutation;
  Produce the initial vector ivb with a prng to encrypt the first block and produce h, m_h values (with a prng);
  Construct the S-box using the four steps of Figure 4;
  for $m = 1$ *to* r **do**
    for $l = 1$ *to* b_l **do**
      Substitution process
      for $k = 1$ *to* $|p_l|$ **do**
        Calculate $y_l(k)$ by Equations (15) and (16);
        Perform the substitution process using Equation (14) to obtain $s_l(k)$;
      end
      Permutation process for the diffusion
      $M = \sqrt{|p_l|}$;
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate $k = (i-1) \times M + j$;
          Perform the permutation of pixels: $sp_l(i_n, j_n) = s_l(i, j)$;
        end
      end
      HAD diffusion
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate horizontal diffusion using Equation (23);
        end
      end
      VAD diffusion
      for $i = 1$ *to* M **do**
        for $j = 1$ *to* M **do**
          Calculate vertical diffusion using Equation (24);
        end
      end
    end
  end
end

Algorithm 2 Decryption process.

Input : IV = Initial Vector of $PRNG - CS$;
K = Secret key of the $PRNG - CS$;
ivb = Initialization vector used in the first block;
b_l = Number of blocks;
p_l = Plaintext block;
$PRNG - CS(IV, K) = (Kc, Kd)$.

Output: The block under test is decrypted.

```
begin
    Generate the needed dynamic keys |Kc| to construct the S-box and |Kd| to perform the permutation;
    Produce the initial vector ivb with a prng to encrypt the first block and produce h, m_h values (with a prng);
    Construct the S-box using the four steps of Figure 4;
    for m = r to 1 do
        for l = 1 to b_l do
            Inverse VAD diffusion
            for i = M to 1 do
                for j = M to 1 do
                    Calculate inverse vertical diffusion using Equation (25);
                end
            end
            Inverse HAD diffusion
            for i = M to 1 do
                for j = M to 1 do
                    Calculate inverse horizontal diffusion using Equation (26);
                end
            end
            Reverse Permutation process for the diffusion
            M = sqrt(|p_l|);
            for i = 1 to M do
                for j = 1 to M do
                    Calculate k = (i − 1) × M + j;
                    Perform c_l(i, j) = c_l(k);
                    Calculate [i_n, j_n] using Equation (21);
                    Perform the permutation of pixels: c_l(i_n, j_n) = id_l(i, j);
                end
            end
            Inverse Substitution process
            for k = 1 to |p_l| do
                Calculate y_l(k) by Equation (17);
                Perform the substitution process using relation (18) to obtain p_l(k);
            end
        end
    end
end
```

3. Experiments Results and Security Analysis

Hereafter, we evaluate the proposed chaos-based cryptosystem's performance. First, we estimate the number r of rounds required to have a secure system, then we assess the

performance of the proposed chaotic generator, histogram chi-square, NIST tests on the generated sequences, and finally, we examine the security of the cryptosystem up against the usual cryptographic attacks.

All the simulations were implemented using MATLAB (R2016b) on a PC with a 2.5 GHz processor Intel Core i5-7300HQ CPU, 16 GB RAM, under Windows 10 Professional, and a 64-bit operating system.

3.1. Estimation of the Number of Rounds Required

In this section, we provide the number r of rounds that was used in the chaos-based cryptosystem implementation. For this, we determined the average Hamming distance (HD) between two ciphered images C_1, using 100 secret keys, and C_2 using the same secret keys, each with a different LSB-bit. This test was applied to 10 different plain images. The $HD(C_1,\ C_2)$ was defined by the following equation:

$$HD(C_1,\ C_2) = \frac{1}{Nb}\sum_{i=1}^{Nb}(C_1(i) \oplus\ C_2(i)) \tag{27}$$

where Nb was the number of bits in an ciphered image. In Table 3, a list of the Hamming distance outcomes for $r = 1,\ 2,$ and 3 is provided.

Table 3. HD versus the round times in the encryption process.

Image	Size	HD (%)		
		$r = 1$	$r = 2$	$r = 3$
Airplane	$512 \times 512 \times 3$	50.0010	49.9974	49.9998
Black	$256 \times 256 \times 1$	50.0027	50.0050	49.9977
Bridge	$512 \times 512 \times 1$	50.0039	49.9962	50.0021
Cameraman	$256 \times 256 \times 1$	49.9936	50.0018	50.0000
Flowers	$256 \times 256 \times 3$	49.9993	50.0034	49.9955
Goldhill	$512 \times 512 \times 3$	49.9999	49.9988	49.9996
Kiel	$512 \times 512 \times 1$	50.0010	49.9989	50.0036
Lena	$512 \times 512 \times 3$	50.0001	49.9997	49.9972
Sailboat	$512 \times 512 \times 3$	49.9961	50.0050	50.0000
White	$256 \times 256 \times 1$	50.0107	49.9948	50.0009

It is noteworthy from Table 3 that for $r \geq 1$, the HDs for the various plain images encrypted by the proposed encryption algorithm were close to the optimal value of 50%. Therefore, for all subsequent tests, we used $r = 1$.

This result was also confirmed by the plaintext sensitivity test carried out in Section 3.3.3.

3.2. Performance Analysis of the Proposed PRNG-CS

We give below the performance obtained by the proposed chaotic generator.

3.2.1. Histogram and Chi-Square Test

A histogram describes the distribution of numerical data in the form of a graph. It forms an estimation of the probability distribution of a random or pseudorandom variable.

The distribution uniformity of the generated sequences was examined. A uniform distribution over the whole phase space must be provided by the proposed PRNG-CS. Figure 6 shows an example of a histogram of a produced sequence, formed by 3,125,100 samples, in which the first 100 samples were not used (the transient regime).

Visually, we observe that the generated sequence is nearly uniformly distributed.

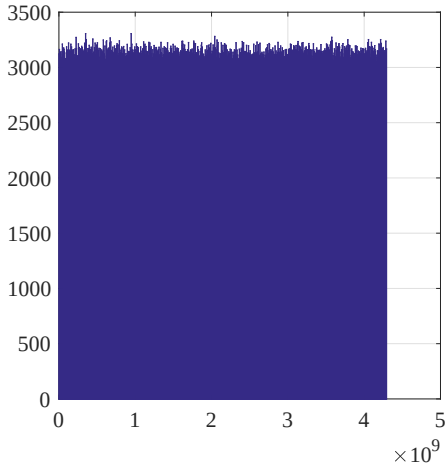


Figure 6. An example of a histogram of a generated sequence $X(n)$.

Then, to assert the uniformity of this sequence, we applied the chi-square test. The experimental chi-square χ^2 value was given by:

$$\chi^2_{exp} = \sum_{i=0}^{N_c-1} \frac{(O_i - E_i)^2}{E_i} \tag{28}$$

where $N_c = 1000$, O_i, and $E_i = Ns/Nc$ are the number of classes, the number of calculated samples in the ith class E_i, and the anticipated number of samples of a uniform distribution. Table 4 shows the experimental and theoretical values for the chi-square values obtained. The theoretical value of the chi-square was greater than the obtained experimental one, which proved that the sequence generated by the proposed PRNG-CS was uniform.

Table 4. Chi-square results on the tested histograms.

Chi-Square Test	PRNG-CS
χ^2_{ex}	967.700
χ^2_{th} (1000, 0.05)	1073.642

Note that this test was repeated on one hundred various sequences, each was generated using a different secret key. All the sequences passed the chi-square test.

3.2.2. NIST

The NIST Test Suite [48] is a statistical package consisting of 15 tests that were developed to test the randomness of binary sequences produced by PRNGs. Some tests are decomposable into a variety of subtests. To determine the result of each test, a p-value is computed. A p-value ≥ 0.01 means that the sequence is considered to be random with a confidence of 99%. A p-value ≤ 0.01 means that the sequence is nonrandom with a confidence of 99%.

To carry out the NIST test, we produced 100 different sequences of 3,125,100 32-bit samples each, using 100 random secret keys. Only 3,125,000 samples per sequence (i.e., 10^8 bits) were used (the first 100 samples for each sequence, corresponding to the transient regime, were not useful for the various robustness tests).

We give in Table 5 the p-values obtained for the 15 tests by the proposed PRNG-SC. All p-values were distinctly larger than the critical value 0.01, which proved the high degree of randomness of the generated sequences. Therefore, from the statistical results obtained previously, we could confirm that the proposed PRNG-CS was robust against statistical attacks.

Table 5. *P*-values and proportion results of NIST tests.

Test	*p*-Value	Proportion (%)
Frequency test	0.494	98.000
Block-frequency test	0.514	99.000
Cumulative sums test	0.504	98.000
Runs test	0.494	99.000
Longest-run test	0.262	97.000
Rank test	0.868	99.000
FFT test	0.384	99.000
Nonperiodic templates	0.482	99.061
Overlapping templates	0.596	98.000
Universal	0.946	99.000
Approximate entropy	0.419	100.000
Random excursions	0.439	98.214
Random excursion variant	0.488	98.651
Serial test	0.415	100.000
Linear complexity	0.817	97.000

3.3. Security Analysis of the Proposed Chaos-Based Cryptosystem

In this section, we first give the performance of the proposed S-box, then we evaluate the security of the proposed chaotic encryption system regarding statistical attacks and a cryptanalytic analysis. Eventually, we estimate the computing performance.

3.3.1. Performance Analysis of the Proposed Key-Dependent S-Box

A strong S-box should have some important properties, based on an information theory analysis. The main properties, besides the bijectivity, are nonlinearity, strict avalanche criterion (SAC), output bits independence criterion (BIC), equiprobable input/output XOR distribution, differential approximation probability (DP), and linear approximation probability (LP). To demonstrate the performance of the proposed S-box, we quantified these properties and compared the results obtained with those obtained from several S-boxes in the literature.

From Table 6, we can observe that the constructed S-box met all the requirements of a robust S-box and the performance obtained were close to those obtained by [42–44,49,50].

Table 6. Performance comparison of different S-boxes.

S-Box	Nonlinearity			SAC			BIC-SAC	BIC Nonlinearity	DP	LP
	Min	Avg	Max	Min	Avg	Max				
Proposed S-box	102	104	108	0.4375	0.4948	0.5625	0.4991	103.4286	10	0.1094
Çavuşoğlu et al. [42]	104	106	110	0.4218	0.5039	0.5937	0.5058	103.40	10	0.1406
Lambić et al. [43]	106	106.75	108	-	0.5034	-	0.5014	103.78	10	0.1328
Ahmed et al. [44]	106	107.5	108	-	0.4943	-	0.4982	104.35	10	0.1250
Lai et al. [49]	104	105	110	0.3906	0.5014	0.5937	0.5028	102.75	10	0.1250
Al Solami et al. [50]	106	108.5	110	-	0.5017	-	0.5026	104	10	0.1094

You can see from Table 2 that the constructed S-box had all the different output values of the interval $[0, 255]$, so it satisfied the requirement of bijectivity.

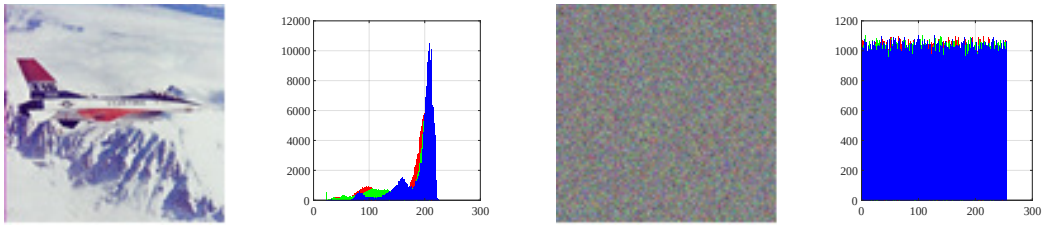
By comparing the properties of our S-box with other S-boxes, we can claim that the proposed chaos-based S-box was better than others in some measures. Therefore, the proposed S-box was suitable for use in the proposed encryption scheme.

3.3.2. Statistical Analysis

To prove the robustness of the proposed chaos-based cryptosystem against statistical attacks, we performed the following experiments on various plain images: histogram, chi-square test, correlation, and entropy analysis.

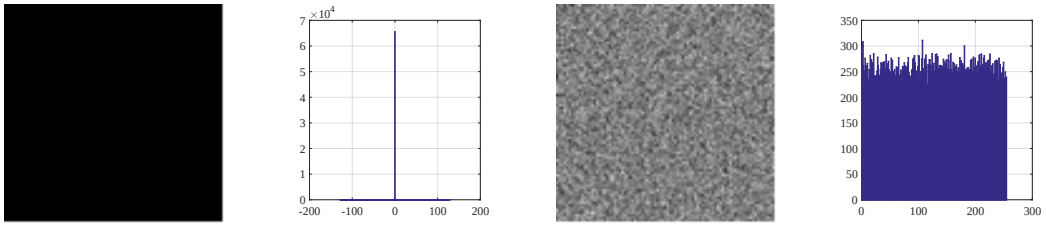
Histograms of encrypted images

The ciphered image must have a uniform distribution and this is a basic condition for any cryptosystem to be strong against statistical attacks. We encrypted 10 color and gray plain images with varying features and sizes and then we deducted the histograms of the plain images and corresponding ciphered images. The obtained results are given in Figures 7–16, and on each figure, we show: (a) the plain image, (b) the histogram of the plain image, (c) the corresponding ciphered image, and (d) the histogram of the ciphered image.



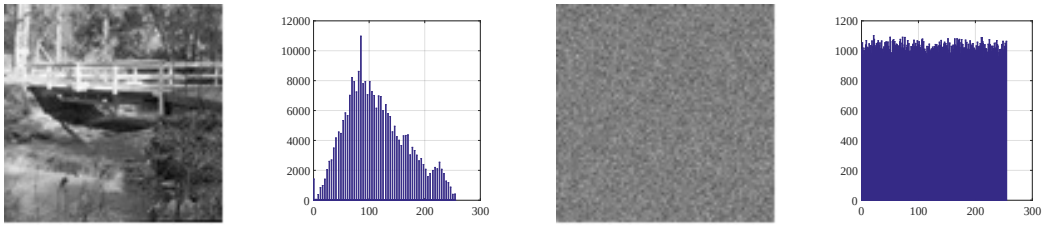
(**a**) (**b**) (**c**) (**d**)

Figure 7. Histograms: (**a**) Airplane image, (**b**) histogram of plain Airplane, (**c**) encrypted Airplane, and (**d**) histogram of encrypted Airplane.



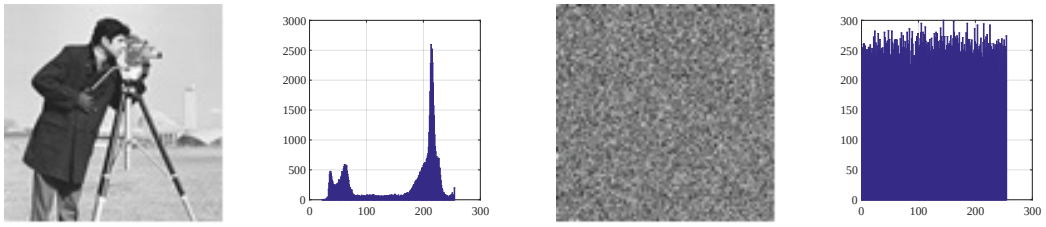
(**a**) (**b**) (**c**) (**d**)

Figure 8. Histograms: (**a**) Black image, (**b**) histogram of plain Black, (**c**) encrypted Black, and (**d**) histogram of encrypted Black.



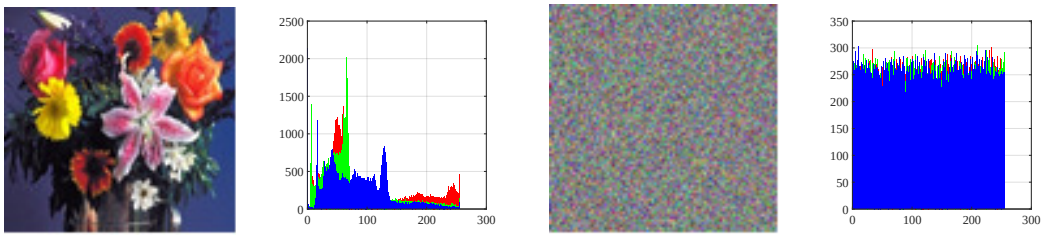
(**a**) (**b**) (**c**) (**d**)

Figure 9. Histograms: (**a**) Bridge image, (**b**) histogram of plain Bridge, (**c**) encrypted Bridge, and (**d**) histogram of encrypted Bridge.



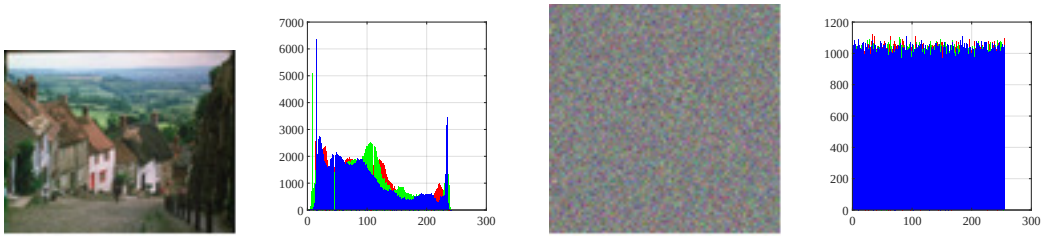
(**a**) (**b**) (**c**) (**d**)

Figure 10. Histograms: (**a**) Cameraman image, (**b**) histogram of plain Cameraman, (**c**) encrypted Cameraman, and (**d**) histogram of encrypted Cameraman.



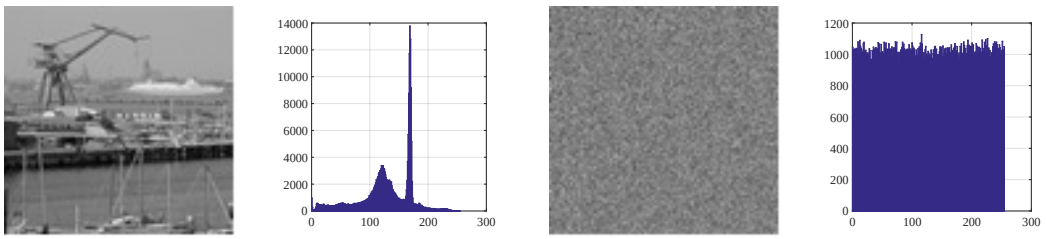
(**a**) (**b**) (**c**) (**d**)

Figure 11. Histograms: (**a**) Flowers image, (**b**) histogram of plain Flowers, (**c**) encrypted Flowers, and (**d**) histogram of encrypted Flowers.



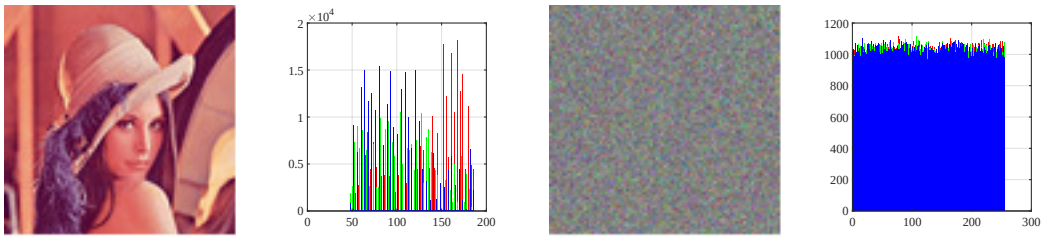
(**a**) (**b**) (**c**) (**d**)

Figure 12. Histograms: (**a**) Goldhill image, (**b**) histogram of plain Goldhill, (**c**) encrypted Goldhill, and (**d**) histogram of encrypted Goldhill.



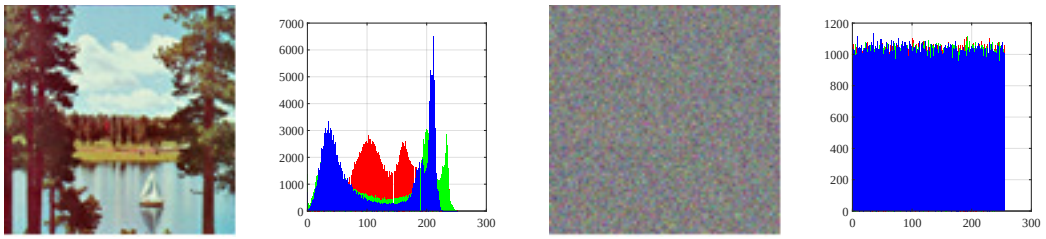
(**a**) (**b**) (**c**) (**d**)

Figure 13. Histograms: (**a**) Kiel image, (**b**) histogram of plain Kiel, (**c**) encrypted Kiel, and (**d**) histogram of encrypted Kiel.



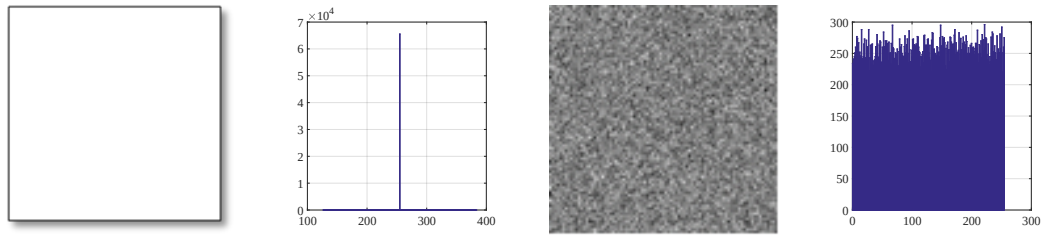
(**a**) (**b**) (**c**) (**d**)

Figure 14. Histograms: (**a**) Lena image, (**b**) histogram of plain Lena, (**c**) encrypted Lena, and (**d**) histogram of encrypted Lena.



(**a**) (**b**) (**c**) (**d**)

Figure 15. Histograms: (**a**) Sailboat image, (**b**) histogram of plain Sailboat, (**c**) encrypted Sailboat, and (**d**) histogram of encrypted Sailboat.



(**a**) (**b**) (**c**) (**d**)

Figure 16. Histograms: (**a**) White image, (**b**) histogram of plain White, (**c**) encrypted White, and (**d**) histogram of encrypted White.

Note that visually the histograms of the encrypted images appeared to have a uniform distribution and were very different from those of the single images. Therefore, no visual information could be detected from the encrypted images by the proposed chaos-based cryptosystem.

To check the uniformity of the histogram, we applied the chi-square test (using Equation (28)) to statistically confirm it. However, here, N_c was the number of values (256), O_i were the detected occurrence frequencies of each color value $\in [0, 255]$ in the histogram of the encrypted image, and E_i was the anticipated occurrence frequency of the uniform distribution, provided by $E_i = (L \times C \times P)/N_c$. The distribution of the tested histogram was uniform if it met the following condition: $\chi^2_{exp} < \chi^2_{th}(N_c - 1, \alpha) = 293.2478$ (for $N_c = 256$ and $\alpha = 0.05$) [12].

In Table 7, we reported the experimental values of the chi-square test for the 10 ciphered images. The values of the experimental chi-square obtained confirmed the uniformity of the histograms of the tested encrypted images.

Table 7. Chi-square test.

Image	Size	Chi-Square Test
Airplane	$512 \times 512 \times 3$	256.7715
Black	$256 \times 256 \times 1$	234.0391
Bridge	$512 \times 512 \times 1$	256.6113
Cameraman	$256 \times 256 \times 1$	275.5625
Flowers	$256 \times 256 \times 3$	236.6458
Goldhill	$512 \times 512 \times 3$	239.6576
Kiel	$512 \times 512 \times 1$	243.4629
Lena	$512 \times 512 \times 3$	249.1270
Sailboat	$512 \times 512 \times 3$	242.1380
White	$256 \times 256 \times 1$	273.6641

Correlation Analysis

Another key property of a secure cryptosystem is that the correlation of the encrypted image should be close to zero and very different from the correlation of the plain image, which is generally close to one. To measure this property, $N = 8000$ pairs of two adjacent pixels in the horizontal (HC), vertical (VC), and diagonal (DC) directions from the original and the ciphered images were randomly selected. Then, the correlation coefficients were calculated using the following equation:

$$\rho_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{29}$$

with:

$$cov(x,y) = \frac{1}{N}\sum_{i=1}^{N}([x_i - E(x)][y_i - E(y)]) \tag{30}$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{31}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2 \tag{32}$$

Furthermore, x and y were the gray-level values of two adjacent pixels in the image.

In Table 8, we give the mean correlation coefficient value based on 10 images ciphered by 100 different secret keys. From this table, the correlation coefficients in all directions of the plain images were close to one, indicating that the pixels were highly correlated, while those of encrypted images were near to zero, which proved that there was no correlation between the plain and ciphered images.

Table 8. The correlation coefficient of 8000 pairs of two adjacent pixels of the plain and ciphered images.

Image	Size		Plain Image			Encrypted Image		
			HC	VC	DC	HC	VC	DC
Airplane	512 × 512 × 3	R	0.97191	0.96051	0.93848	0.00006	−0.00153	0.00024
		G	0.95150	0.96915	0.92863	−0.00154	−0.00041	−0.00098
		B	0.96226	0.94538	0.92419	−0.00033	−0.00056	−0.00004
Black	256 × 256 × 1		-	-	-	−0.00167	−0.00044	−0.00074
Bridge	512 × 512 × 1		0.93981	0.92720	0.89716	−0.00130	0.00172	0.00084
Cameraman	256 × 256 × 1		0.92013	0.95496	0.89590	0.00110	0.00193	0.00028
Flowers	256 × 256 × 3	R	0.95049	0.96800	0.93340	0.00197	−0.00128	0.00153
		G	0.91532	0.94200	0.88863	−0.00082	−0.00019	0.00071
		B	0.92085	0.94834	0.89510	−0.00092	0.00009	0.00182
Goldhill	512 × 512 × 3	R	0.97767	0.97649	0.95975	0.00039	0.00052	−0.00058
		G	0.98180	0.98496	0.96998	−0.00047	0.00028	0.00148
		B	0.98453	0.98641	0.97339	−0.00140	0.00084	−0.00024
Kiel	512 × 512 × 1		0.90591	0.82571	0.78020	0.00168	−0.00127	−0.00044
Lena	512 × 512 × 3	R	0.97529	0.98540	0.96464	0.00055	−0.00041	0.00001
		G	0.96666	0.98011	0.95321	−0.00162	0.00049	−0.00023
		B	0.93357	0.95552	0.91819	0.00077	−0.00006	0.00042
Sailboat	512 × 512 × 3	R	0.95595	0.95302	0.94030	−0.00041	0.00070	−0.00001
		G	0.97088	0.96404	0.95027	−0.00096	0.00038	−0.00015
		B	0.97054	0.96872	0.95208	−0.00164	0.00048	−0.00018
White	256 × 256 × 1		-	-	-	−0.00165	0.00025	0.00036

These results can be observed visually in Figure 17, in which we show the correlation of adjacent pixels of the Goldhill (512 × 512 × 3) and encrypted Goldhill images in the three directions: horizontal, vertical, and diagonal, respectively.

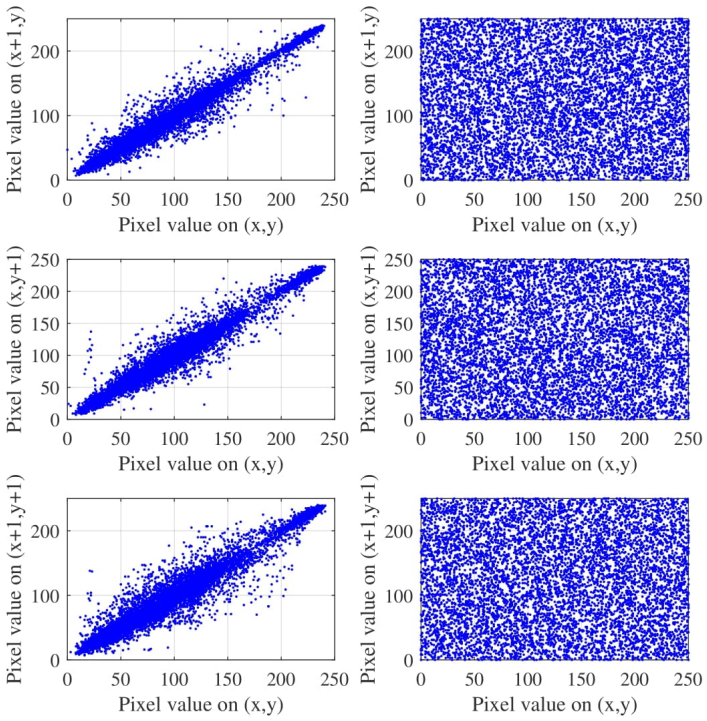


Figure 17. Distribution of adjacent pixels in the plain and encrypted images of Goldhill in the three directions: horizontal, vertical, and diagonal, respectively.

Entropy Analysis

The random behavior of the encrypted image can also be evaluated using the entropy information defined by:

$$H(C) = \sum_{i=0}^{N_c-1} P(c_i) \times log_2 \frac{1}{P(c_i)} \tag{33}$$

where $H(C)$ is the entropy of the encrypted image C, $P(c_i)$ is the occurrence value of each level ($i = 0, 1, 2, ..., 255$). In the condition of equal probability levels ($P(c_i) = 2^{-8}$), the entropy is maximum, $H(C) = 8$.

We show in Table 9 the entropy test results of the original and encrypted images. It is clear that the results for the encrypted images were close to the optimal value. Based on these results, the proposed chaos-based cryptosystem had a high degree level of resilience.

Table 9. Entropy test results.

Image	Size	Plain	Encrypted
Airplane	$512 \times 512 \times 3$	6.6639	7.9997
Black	$256 \times 256 \times 1$	0.0000	7.9974
Bridge	$512 \times 512 \times 1$	5.7056	7.9992
Cameraman	$256 \times 256 \times 1$	6.9046	7.9977
Flowers	$256 \times 256 \times 3$	7.5434	7.9991
Goldhill	$512 \times 512 \times 3$	7.6220	7.9997
Kiel	$512 \times 512 \times 1$	6.9589	7.9994
Lena	$512 \times 512 \times 3$	5.6822	7.9998
Sailboat	$512 \times 512 \times 3$	7.7632	7.9998
White	$256 \times 256 \times 1$	0.0000	7.9968

Based on all of these results of the histogram, chi-square, correlation, and information entropy analysis, the proposed chaos-based cryptosystem was secure against statistical attacks.

3.3.3. Cryptanalytic Analysis

In the following part, we perform some habitual cryptanalytic analyses such as the key sensitivity and the plaintext sensitivity test.

Key Sensitivity

A secure encryption/decryption system should be very sensitive to the secret key, i.e., even a one-bit change in the secret key should produce a completely different encrypted image. The testing scenario of the key sensitivity was as follows: first, a plain-image I was encrypted using a secret key K_1 to obtain C_1, then the same plain-image I was encrypted using a secret key K_2 different from K_1 by just a one bit LSB to obtain C_2. Three parameters, namely the number of pixels change rate (NPCR), the unified average changing intensity (UACI) [51], and the Hamming distance (see Equation (27)) were used to evaluate the key sensitivity and were defined as follows:

$$NPCR = \frac{1}{R \times C \times P} \sum_{i=1}^{R} \sum_{j=1}^{C} \sum_{p=1}^{P} D(i,j,p) \times 100\% \tag{34}$$

$$D(i,j,p) = \begin{cases} 0 & if\ C_1(i,j,p) = C_2(i,j,p) \\ 1 & if\ C_1(i,j,p) \neq C_2(i,j,p) \end{cases} \tag{35}$$

$$UACI = \frac{1}{R \times C \times P \times 255} \sum_{i=1}^{R} \sum_{j=1}^{C} \sum_{p=1}^{P} |C_1 - C_2| \times 100\% \tag{36}$$

In the above relation, i, j, and p were, respectively, the row, column, and plane indexes of the image. R, C, and P were its length, width, and plane sizes.

This test was repeated over 100 different secret keys.

Table 10 presents the average results obtained for the NPCR, UACI, and HD.

Table 10. Average NPCR, UACI, and HD key sensitivity tests.

Image	Size	Key Sensitivity Test		
		NPCR (%)	UACI (%)	HD (%)
Airplane	512 × 512 × 3	99.6065	33.4666	50.0052
Black	256 × 256 × 1	99.6099	33.4323	50.0094
Bridge	512 × 512 × 1	99.6098	33.5052	50.0019
Cameraman	256 × 256 × 1	99.5992	33.4234	49.9844
Flowers	256 × 256 × 3	99.6040	33.4772	49.9941
Goldhill	512 × 512 × 3	99.6069	33.4464	49.9942
Kiel	512 × 512 × 1	99.6042	33.4650	49.9976
Lena	512 × 512 × 3	99.6088	33.4664	49.9907
Sailboat	512 × 512 × 3	99.6077	33.4680	50.0090
White	256 × 256 × 1	99.6231	33.5524	50.0000

The resulting values obtained were near the optimal values of NPCR, UACI, and HD which were 99.6094%, 33.4635%, and 50%, respectively. Therefore, the proposed scheme was very sensitive to small changes in the secret key.

Plaintext sensitivity test

To resist known and chosen plaintext attacks, which are related to the differential attack, the cryptosystem must be sensitive to a single bit change in the plaintext. The plaintext sensitivity attack test case is almost identical to the key sensitivity attack. Indeed, after

having encrypted a plain image, we chose 21-pixel positions [26] and we changed, in turn, their LSB bit, then we encrypted them and calculate the 21 HDs.

In Figure 18, we show for each position the HDs of the 10 test images (red dots) as well as their average values (green line). As we can see, for each position, the average HD was close to the optimal value of 50%. Furthermore, the maximum value of the HD was equal to 50.1602% and the minimum value was equal to 49.8039% which was close to the optimal value.

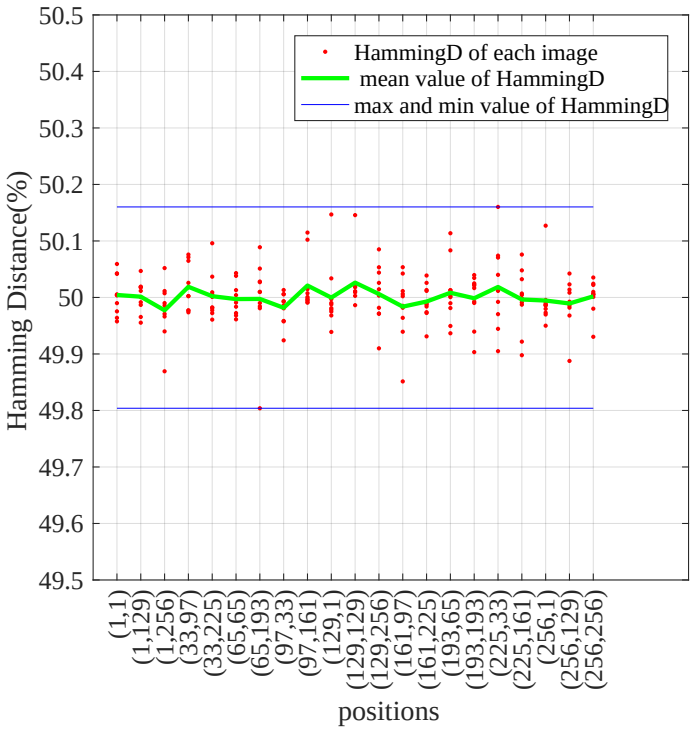


Figure 18. Plaintext sensitivity test evaluated by the Hamming distance.

Moreover, in Table 11, we give the average NPCR, UACI, and HD on 21 positions for each image. According to these results, the average values of the obtained NPCR, UACI, and HD were almost equal to their optimal value. These values proved that the proposed chaos-based cryptosystem was highly sensitive to small changes in the plaintext.

Table 11. NPCR, UACI, and HD of the plaintext sensitivity test.

Image	Size	Plaintext Sensitivity		
		NPCR (%)	UACI (%)	HD (%)
Airplane	512 × 512 × 3	99.6102	33.4659	49.9972
Black	256 × 256 × 1	99.6067	33.4840	50.0143
Bridge	512 × 512 × 1	99.6114	33.4960	50.0060
Cameraman	256 × 256 × 1	99.6068	33.4616	49.9907
Flowers	256 × 256 × 3	99.6016	33.4563	49.9954
Goldhill	512 × 512 × 3	99.6081	33.4460	50.0003
Kiel	512 × 512 × 1	99.6075	33.4658	49.9974
Lena	512 × 512 × 3	99.6092	33.4669	50.0028
Sailboat	512 × 512 × 3	99.6058	33.4837	50.0062
White	256 × 256 × 1	99.6008	33.5139	49.9981

3.3.4. Robustness against Noise and Occlusion

Encrypted images transmitted may be affected by channel noise and data loss. In this paragraph, we study the resistance capacity of the proposed cryptosystem against noise and occlusion. We utilized salt and pepper noise and an occlusion attack [25].

We added salt and pepper noise with different intensities (0.01, 0.02, 0.05, and 0.1) to the encrypted Cameraman image (see Figure 10c), as shown in Figure 19a–d, respectively, and then we decrypted them. The recovered images are shown in Figure 19e–h, respectively. It can be seen that under limited noise, the decrypted images were always identified. Thus, our proposed decryption system had good robustness against noise attacks.

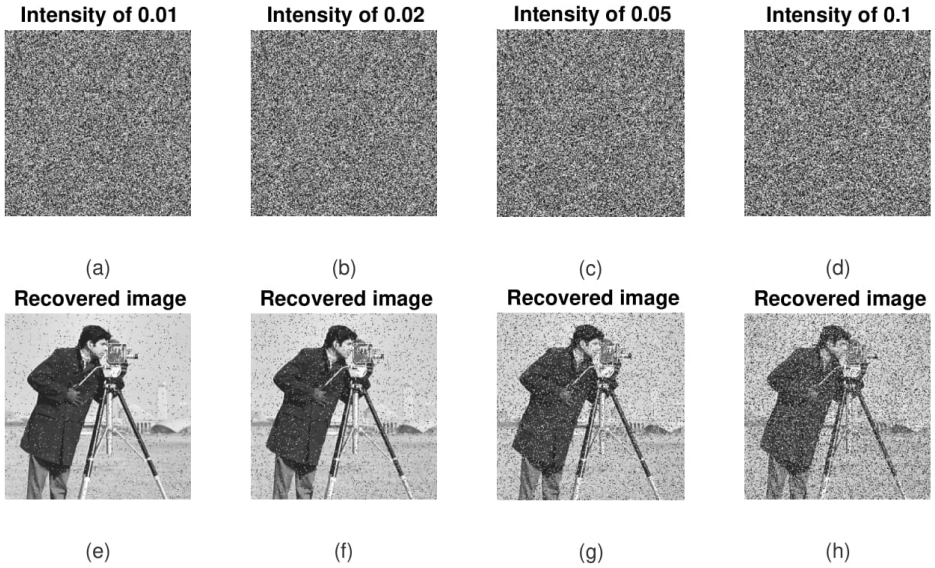


Figure 19. Robustness against salt and pepper noise.

Another kind of attack is occlusion. To assess the robustness of the cryptosystem against such an attack, first, we occluded the Cameraman encrypted image (see Figure 10c) with different data loss sizes and positions: 1/16, 1/8, 1/4, and 1/2, as shown in Figure 20a–d, respectively, then, we decrypted and shown them in Figure 20e–h. We can observe that all the recovered images were recognized but their qualities decreased with the increase of the occlusion size. Therefore, the proposed cryptosystem had a high level of robustness against occlusion attacks.

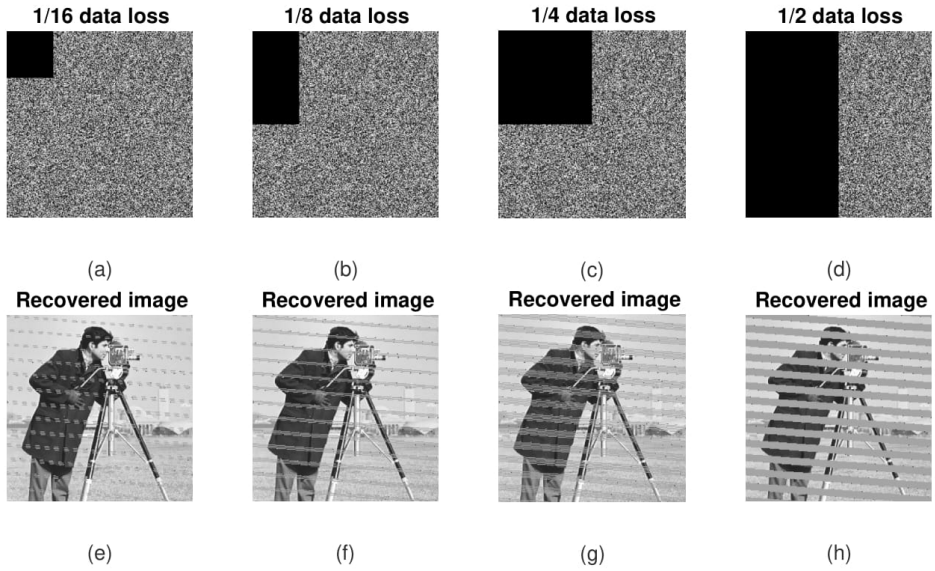


Figure 20. Robustness against occlusion attack.

3.3.5. The Speed Performance of the Proposed Chaos-Based Cryptosystem

Computing performance is an important issue for practical applications and largely depends on the programming languages used. The Matlab language is not a good candidate to achieve high computing performance compared to C language, VHDL description language, etc. However, with Matlab, we can design and realize cryptosystems faster than the other languages mentioned.

We evaluated the computational performance: average encryption time, average encryption throughput (*ET*), and average number of needed cycles per bytes (*NCpB*) of the proposed cryptosystem and we compared its performance (*NCpB*) with other cryptosystems designed in the Matlab language.

The *ET* and *NCpB* are given by the following equation:

$$ET = \frac{Image\ Size\ (Bytes)}{Average\ Encryption\ Time\ (second)} \tag{37}$$

$$NCpB = \frac{CPU\ Speed\ (Hertz)}{ET\ (Bytes/s)} \tag{38}$$

In Table 12, we give the computing performance obtained by the proposed encryption system using 100 different secret keys, for four plain images.

Table 12. The proposed encryption system's computing performance.

Image	Size	Encryption Time (Milliseconds)	*ET* (Mbps)	*NCpB* (Cycles/Byte)
Cameraman	$256 \times 256 \times 1$	39.3	1.589	1501
Flowers	$256 \times 256 \times 3$	119.2	1.5726	1516
Kiel	$512 \times 512 \times 1$	163.9	1.5249	1564
Lena	$512 \times 512 \times 3$	602.3	1.2451	1915

In Table 13, we compare the *NCpB* of our algorithm for the Lena image of size $512 \times 512 \times 1$ with the *NCpBs* of other chaos-based encryption algorithms.

Table 13. Comparison of NCpBs from different encryption schemes.

Cryptosystem	*NCpB*
Proposed	1568
Qiao et al. [25]	77,386
Luo et al. [52]	95,368
Huang et al. [53]	15,642

As Table 13 shows, the proposed encryption system had higher computational performance efficiency than the others.

4. Conclusions

This paper proposed a new secure chaos-based cryptosystem system for a block cipher in CBC mode and evaluated its performance. The proposed cryptosystem achieved high confusion diffusion effects thanks to a robust pseudorandom number generator of chaotic sequences, a strong circular substitution based on the proposed S-box, and a solid diffusion layer. The proposed PRNG-CS was formed by four discrete 1-D chaotic maps weakly coupled by a predefined coupling matrix M, to avoid the divide-and-conquer attack, and to increase the randomness of the sequences produced as well as their lengths. The diffusion layer was performed by a 2-D modified cat map and a horizontal addition diffusion (HAD) followed by a vertical addition diffusion (VAD). The experimental results and the security analysis demonstrated that the proposed cryptosystem could successfully resist various attacks known in the literature, such as statistical attacks, brute force attacks, noise attacks,

and occlusion attacks, and therefore could be used to secure sensitive data. In a future study, we plan to address the hardware implementation of the system.

Author Contributions: Conceptualization, F.D.; Formal analysis, R.L.; Methodology, F.D.; Writing—original draft, F.D.; Writing—review & editing, F.D. and S.E.A.; Validation, W.E.H.Y., M.M. and R.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Matthews, R. On the derivation of a "chaotic" encryption algorithm. *Cryptologia* **1989**, *13*, 29–41. [CrossRef]
2. Liu, Y.; Tong, X.; Hu, S. A family of new complex number chaotic maps based image encryption algorithm. *Signal Process. Image Commun.* **2013**, *28*, 1548–1559. [CrossRef]
3. Zhang, X.; Shao, L.; Zhao, Z.; Liang, Z. An image encryption scheme based on constructing large permutation with chaotic sequence. *Comput. Electr. Eng.* **2014**, *40*, 931–941. [CrossRef]
4. Fouda, J.A.E.; Effa, J.Y.; Sabat, S.L.; Ali, M. A fast chaotic block cipher for image encryption. *Commun. Nonlinear Sci. Numer. Simul.* **2014**, *19*, 578–588. [CrossRef]
5. Caragata, D.; El Assad, S.; Luduena, M. An improved fragile watermarking algorithm for JPEG images. *AEU Int. J. Electron. Commun.* **2015**, *69*, 1783–1794. [CrossRef]
6. Pichler, F.; Scharinger, J. Finite dimensional generalized baker dynamical systems for cryptographic applications. In Proceedings of the International Conference on Computer Aided Systems Theory, Innsbruck, Austria, 22–25 May 1995; Springer: Berlin/Heidelberg, Germany, 1995; pp. 465–476. [CrossRef]
7. Masuda, N.; Aihara, K. Cryptosystems with discretized chaotic maps. *IEEE Trans. Circuits Syst. Fundam. Theory Appl.* **2002**, *49*, 28–40. [CrossRef]
8. Shannon, C.E. Communication theory of secrecy systems. *Bell Syst. Tech. J.* **1949**, *28*, 656–715. [CrossRef]
9. Fridrich, J. Symmetric ciphers based on two-dimensional chaotic maps. *Int. J. Bifurc. Chaos* **1998**, *8*, 1259–1284. [CrossRef]
10. Pareek, N.K.; Patidar, V.; Sud, K.K. Image encryption using chaotic logistic map. *Image Vis. Comput.* **2006**, *24*, 926–934. [CrossRef]
11. Zhu, Z.L.; Zhang, W.; Wong, K.W.; Yu, H. A chaos-based symmetric image encryption scheme using a bit-level permutation. *Inf. Sci.* **2011**, *181*, 1171–1186. [CrossRef]
12. El Assad, S.; Farajallah, M. A new chaos-based image encryption system. *Signal Process. Image Commun.* **2016**, *41*, 144–157. [CrossRef]
13. El Assad, S.; Farajallah, M.; Vladeanu, C. Chaos-based block ciphers: An overview. In Proceedings of the 2014 10th International Conference on Communications (COMM), Bucharest, Romania, 29–31 May 2014; pp. 1–4. [CrossRef]
14. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based cryptosystems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
15. Lian, S.; Sun, J.; Wang, Z. Security analysis of a chaos-based image encryption algorithm. *Phys. A Stat. Mech. Appl.* **2005**, *351*, 645–661. [CrossRef]
16. Bouteghrine, B.; Tanougast, C.; Sadoudi, S. A Survey on Chaos-Based Cryptosystems: Implementations and Applications. In Proceedings of the Chaotic Modeling and Simulation International Conference, Florence, Italy, 9–12 June 2022; pp. 65–80. [CrossRef]
17. Daemen, J.; Rijmen, V. *The Design of Rijndael*; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2.
18. Munir, N.; Khan, M.; Hussain, I.; Alanazi, A.S. Cryptanalysis of encryption scheme based on compound coupled logistic map and anti-codifying technique for secure data transmission. *Optik* **2022**, *267*, 169628. [CrossRef]
19. Rahman, Z.; Yi, X.; Billah, M.; Sumi, M.; Anwar, A. Enhancing AES Using Chaos and Logistic Map-Based Key Generation Technique for Securing IoT-Based Smart Home. *Electronics* **2022**, *11*, 1083. [CrossRef]
20. Nguyen, R. *Penetration Testing on a c-Software Implementation a-1709rns006-c*; Internal Report; Department of Informatics: Munich, Germany, 2018.
21. Nguyen, R.; Facon, A.; Guilley, S.; Gautier, G.; El Assad, S. Speed-up of SCA Attacks on 32-bit Multiplications. In Proceedings of the International Conference on Codes, Cryptology, and Information Security, Rabat, Morocco, 22–24 April 2019; pp. 31–39. [CrossRef]
22. Chen, J.X.; Zhu, Z.L.; Fu, C.; Yu, H.; Zhang, L.B. A fast chaos-based image encryption scheme with a dynamic state variables selection mechanism. *Commun. Nonlinear Sci. Numer. Simul.* **2015**, *20*, 846–860. [CrossRef]
23. Zhang, W.; Wong, K.W.; Yu, H.; Zhu, Z.L. An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion. *Commun. Nonlinear Sci. Numer. Simul.* **2013**, *18*, 2066–2080. [CrossRef]
24. Farajallah, M.; El Assad, S.; Deforges, O. Fast and secure chaos-based cryptosystem for images. *Int. J. Bifurc. Chaos* **2016**, *26*, 1650021. [CrossRef]

25. Qiao, Z.; El Assad, S.; Taralova, I. Design of secure cryptosystem based on chaotic components and AES S-Box. *AEU Int. J. Electron. Commun.* **2020**, *121*, 153205. [CrossRef]
26. Omrani, T.; Rhouma, R.; Becheikh, R. LICID: A lightweight image cryptosystem for IoT devices. *Cryptologia* **2019**, *43*, 313–343. [CrossRef]
27. Wang, Y.; Wong, K.W.; Liao, X.; Xiang, T. A block cipher with dynamic S-boxes based on tent map. *Commun. Nonlinear Sci. Numer. Simul.* **2009**, *14*, 3089–3099. [CrossRef]
28. Zhang, X.; Zhao, Z.; Wang, J. Chaotic image encryption based on circular substitution box and key stream buffer. *Signal Process. Image Commun.* **2014**, *29*, 902–913. [CrossRef]
29. Alawida, M.; Teh, J.S.; Mehmood, A.; Shoufan, A. A chaos-based block cipher based on an enhanced logistic map and simultaneous confusion-diffusion operations. *J. King Saud.-Univ.-Comput. Inf. Sci.* **2022**. [CrossRef]
30. Alshammari, B.M.; Guesmi, R.; Guesmi, T.; Alsaif, H.; Alzamil, A. Implementing a symmetric lightweight cryptosystem in highly constrained IoT devices by using a chaotic S-box. *Symmetry* **2021**, *13*, 129. [CrossRef]
31. Lozi, R. Emergence of randomness from chaos. *Int. J. Bifurc. Chaos* **2012**, *22*, 1250021. [CrossRef]
32. El Assad, S.; Noura, H. Generator of Chaotic Sequences and Corresponding Generating System. U.S. Patent 20,130,170,641, 6 October 2011.
33. Hamza, R. A novel pseudo random sequence generator for image-cryptographic applications. *J. Inf. Secur. Appl.* **2017**, *35*, 119–127. [CrossRef]
34. Dridi, F.; El Assad, S.; El Hadj Youssef, W.; Machhout, M.; Lozi, R. The Design and FPGA-Based Implementation of a Stream Cipher Based on a Secure Chaotic Generator. *Appl. Sci.* **2021**, *11*, 625. [CrossRef]
35. Lian, S.; Sun, J.; Wang, J.; Wang, Z. A chaotic stream cipher and the usage in video protection. *Chaos Solitons Fractals* **2007**, *34*, 851–859. [CrossRef]
36. Masuda, N.; Jakimoski, G.; Aihara, K.; Kocarev, L. Chaotic block ciphers: From theory to practical algorithms. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2006**, *53*, 1341–1352. [CrossRef]
37. Peng, J.; You, M.; Yang, Z.; Jin, S. Research on a block encryption cipher based on chaotic dynamical system. In Proceedings of the Third International Conference on Natural Computation (ICNC 2007), Haikou, China, 24–27 August 2007; Volume 5; pp. 744–748. [CrossRef]
38. El Assad, S.; Lozi, R. *Chaos-Based Cryptography*; Internal Report; Nantes Université: Nantes, France, 2019.
39. Bruce, S. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed.; John Wiley & Sons: New York, NY, USA, 1996.
40. Özkaynak, F.; Özer, A.B. A method for designing strong S-Boxes based on chaotic Lorenz system. *Phys. Lett. A* **2010**, *374*, 3733–3738. [CrossRef]
41. Wang, X.; Wang, Q. A novel image encryption algorithm based on dynamic S-boxes constructed by chaos. *Nonlinear Dyn.* **2014**, *75*, 567–576. [CrossRef]
42. Çavuşoğlu, Ü.; Zengin, A.; Pehlivan, I.; Kaçar, S. A novel approach for strong S-Box generation algorithm design based on chaotic scaled Zhongtang system. *Nonlinear Dyn.* **2017**, *87*, 1081–1094. [CrossRef]
43. Lambić, D. A novel method of S-box design based on discrete chaotic map. *Nonlinear Dyn.* **2017**, *87*, 2407–2413. [CrossRef]
44. Ahmed, H.A.; Zolkipli, M.F.; Ahmad, M. A novel efficient substitution-box design based on firefly algorithm and discrete chaotic map. *Neural Comput. Appl.* **2019**, *31*, 7201–7210. [CrossRef]
45. Shah, T.; Qureshi, A.; Usman, M. A Novel Color Image Encryption Scheme Based on Arnold's Cat Map and 16-Byte S-box. *Appl. Appl. Math. Int. J. AAM* **2021**, *16*, 33.
46. Belazi, A.; Abd El-Latif, A.A.; Belghith, S. A novel image encryption scheme based on substitution-permutation network and chaos. *Signal Process.* **2016**, *128*, 155–170. [CrossRef]
47. Wong, K.W.; Kwok, B.S.H.; Law, W.S. A fast image encryption scheme based on chaotic standard map. *Phys. Lett. A* **2008**, *372*, 2645–2652. [CrossRef]
48. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*; Technical Report; Booz-Allen and Hamilton Inc.: Mclean, VA, USA, 2001.
49. Lai, Q.; Akgul, A.; Li, C.; Xu, G.; Çavuşoğlu, Ü. A new chaotic system with multiple attractors: Dynamic analysis, circuit realization and S-Box design. *Entropy* **2018**, *20*, 12. [CrossRef]
50. Al Solami, E.; Ahmad, M.; Volos, C.; Doja, M.N.; Beg, M.M.S. A new hyperchaotic system-based design for efficient bijective substitution-boxes. *Entropy* **2018**, *20*, 525. [CrossRef]
51. Wu, Y.; Noonan, J.P.; Agaian, S. NPCR and UACI randomness tests for image encryption. *Cyber J. Multidiscip. J. Sci. Technol. J. Sel. Areas Telecommun. JSAT* **2011**, *1*, 31–38.
52. Luo, Y.; Zhou, R.; Liu, J.; Qiu, S.; Cao, Y. An efficient and self-adapting colour-image encryption algorithm based on chaos and interactions among multiple layers. *Multimed. Tools Appl.* **2018**, *77*, 26191–26217. [CrossRef]
53. Huang, L.; Cai, S.; Xiong, X.; Xiao, M. On symmetric color image encryption system with permutation-diffusion simultaneous operation. *Opt. Lasers Eng.* **2019**, *115*, 7–20. [CrossRef]