Journal of King Saud University – Computer and Information Sciences 34 (2022) 8136–8151

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

A chaos-based block cipher based on an enhanced logistic map and simultaneous confusion-diffusion operations

Moatsum Alawida [a,b,*], Je Sen Teh [b], Abid Mehmood [a], Abdulhadi Shoufan [c], Wafa' Hamdan Alshoura [b]

[a] Department of Computer Sciences, Abu Dhabi University, Abu Dhabi 59911, United Arab Emirates
[b] School of Computer Sciences, Universiti Sains Malaysia, 11800 USM, Pulau Pinang, Malaysia
[c] Department of Electrical Engineering and Computer Science, Khalifa University, Abu Dhabi, United Arab Emirates

ARTICLE INFO

Article history:
Received 8 May 2022
Revised 30 July 2022
Accepted 31 July 2022
Available online 3 August 2022

Keywords:
Chaotic map
Encryption
Block cipher
Image encryption
Cybersecurity

ABSTRACT

Over the years, there has been considerable interest in the area of chaos-based encryption due to the fact that cryptographic algorithms and chaotic maps share a wide-range of similar characteristics. The majority of chaos-based ciphers were designed for encrypting digital images and vast amounts of data. This paper proposes a new chaos-based block cipher algorithm (CBCA) based on an improved logistic chaotic map. To strengthen the performance of the classical logistic chaotic map, a new chaotification method based on a multiplicative inverse function is used which leads to improved properties such as ergodicity and entropy, both of which are desirable for cryptographic applications. The secret key is used to perturb the chaotic variables, and these perturbed variables are used to produce data sequences for the block cipher's diffusion and confusion structures. The permutation (diffusion) structure is controlled by an ergodic chaotic map, while the substitution (confusion) structure is controlled by the chaotic points themselves. This new ergodicity-based diffusion approach provides higher security and efficiency. The proposed algorithm not only possess good statistical properties provided by the enhanced chaotic map, but it is also key sensitive and uniformly distributed. Statistical evaluation of the proposed algorithm was performed using text and images as inputs to depict its capability to secure various types of media. The performance of the proposed cipher is then compared to other recently proposed algorithms in literature. The performance comparison indicates that CBCA is efficient, secure and that it can be used to encrypt both small and large amounts of data.

1. Introduction

The rising popularity of digital chaos, notably in the design of encryption algorithms, is due to a set of distinct properties. Firstly, digital chaos is a source of entropy that can be leveraged to provide confusion and diffusion properties. To achieve confusion, the relationship between the secret key and ciphertext must be obscured, such that each bit of the secret key must affect various parts of the ciphertext (ideally, all ciphertext bits should change if a single secret key bit is modified). Diffusion is where the relationship between the plaintext and ciphertext must be obscured, whereby changes to a single bit of plaintext should affect half of the bits in the ciphertext. In classical block ciphers such as substitution-permutation networks (SPN) or generalized Feistel networks (GFN), confusion and diffusion properties are achieved through the use of substitution boxes and permutation/shuffling respectively. Cryptosystems can use data sequences generated from a chaotic system to permute and substitute data values (Alshammari et al., 2021). Diffusion and confusion are fully controlled by users through the secret key that is used to generate chaotic parameters and initial conditions. In many chaos-based encryption algorithms, achieving confusion and diffusion involve operations that are slightly different than classical ciphers. Diffusion is generally achieved by substituting plaintext values with new ones generated based on prior plaintext values and chaotic maps. This is usually performed twice (from start to end, followed by end to start) to ensure that plaintext values are fully diffused. Confusion is generally fulfilled through key-dependent permutation, whereby the permutation patterns are generated using a chaotic system initialized using the secret key.

* Corresponding author at: Department of Computer Sciences, Abu Dhabi University, Abu Dhabi 59911, United Arab Emirates.
E-mail address: moatsum.alawida@adu.ac.ae (M. Alawida).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2022.07.025
1319-1578/

Secondly, the plaintext is either divided into set of non-overlapping blocking with limited size for conventional block ciphers or encrypted word by word in the case of a stream cipher. Chaos-based cryptosystems, on the other hand, were designed to process a single, large block for plaintext such as in previously proposed chaos-based image encryption algorithms (Niyat et al., 2017; Hua et al., 2019; Alawida et al., 2019), chaos-based video encryption algorithms (Valli and Ganesan, 2017; Wen et al., 2019), chaos-based image watermarking schemes (Alshoura et al., 2022) and chaos-based hash functions (Teh et al., 2020; Alawida et al., 2021; Alawida et al., 2020). To accommodate larger sizes of plaintext, the cryptosystem only needs to generate a longer data sequence by iterating its underlying chaotic system. Therefore, digital chaos can potentially be used to protect large amounts of data for different applications. Thirdly, the underlying chaotic systems are sources of entropy that can be easily replaced or swapped out for another. Thus, if newer, more secure chaotic systems are available, they can be easily implemented into existing chaos-based cryptosystems for improved security (Lian et al., 2005). This flexibility is an advantage of a chaos-based cryptosystem (Alawida et al., 2021). Lastly, the diffusion and confusion operations that are inherent in nearly all chaos-based cryptosystems can be integrated and performed simultaneously (Alawida et al., 2019). These properties imply that a chaos-based algorithms can be designed to encrypt large amounts of with reduced computational overheads.

As with other symmetric-key primitives, chaos-based cryptosystems also rely on secret or encryption keys which are used to derive initial conditions and control parameters. These parameters are then used to generate a chaotic data sequence which can be used in diffusion and confusion operations. A slight change to the chaotic variables will generate a completely new chaotic data sequence, which then leads to an entirely new output (a new ciphertext). Thus, the secret key in chaos-based cryptographic algorithms must be subjected to security analysis and have a key-space large enough to withstand brute force attacks (Alawida et al., 2021; Zhu and Zhu, 2021). The number of chaotic iterations required to generate a data sequence is proportionate to its length. As floating-point numbers are usually used to implement chaos-based cryptosystems in software, this incurs a computational overhead that is too costly for encrypting small amounts of data (such as text). Therefore, the use of chaos-based encryption for smaller data sizes remains an open problem, where there is still room for improvement in terms of reducing computational overhead.

On the other hand, public-key cryptography is another major direction in cryptography, which are usually based on mathematical hard problems. The main distinction between symmetric and asymmetric-key is that the latter uses distinct keys for encryption and decryption; a public key and private key. Depending on how the keys are used, asymmetric-key ciphers can not only provide confidentiality but also authentication and non-repudiation. Generally, public key ciphers are considered more secure than their symmetric-key counterparts due to their underlying mathematical hard problems. However, public-key ciphers are computationally inefficient, especially when used with longer messages. Thus, asymmetric encryption is usually used for key distribution or to exchange secret parameters in most security protocols. In chaos-based asymmetric encryption, chaotic maps such as Chebyshev and logistic map (Shakiba, 2021) have been used to generate two separate keys. Chaotic synchronization is another technique that has been used to improve security and to resist attacks such as the majority flipping attack (Pisarchik and Zanin, 2012). However, these algorithms still suffer from high computational costs.

The trade-off between security and computational efficiency is a well-documented, open problem for chaos-based encryption algorithms (Lian et al., 2005; Alawida et al., 2020), whereby efficiency is usually sacrificed to achieve optimal security. We address the aforementioned problem by proposing a chaotic block cipher algorithm (CBCA) that possesses desirable properties such as ergodicity, aperiodicity, randomness, and sensitive dependence on initial conditions. The classical logistic map is enhanced to possess improved security features which reflects upon the resulting diffusion and confusion operations in CBCA. CBCA achieves both diffusion and confusion properties simultaneously based on chaotic data sequences. A secret key is used to generate the chaotic variables (initial condition and two control parameters) to make CBCA highly sensitive to key values. The proposed algorithm encrypts a plaintext without dividing it into blocks to ensure that confusion and full diffusion can be achieved in only two rounds. One of the novel features of the proposed algorithm is that its diffusion property is achieved using a permutation pattern generated based on the chaotic map's ergodicity to provide higher security and remove correlations between ciphertext values. Both confusion and diffusion properties are also achieved simultaneously using a substitution process that involves prior plaintext words and chaotic values generated based on the secret key. Statistical results indicate that CBCA has better security characteristics than its peers, and can encrypt data of any size with high efficiency. Furthermore, the proposed algorithm is secure against different well-known attacks. The main contributions of this paper include:

- A new method to enhance the logistic map to achieve high chaotic complexity and a large chaotic range.
- A new chaos-based block cipher algorithm (CBCA) based on simultaneous diffusion and confusion operations.

The remaining sections of this paper are as follows: Section 2 provides a related work and Section 3 details the new enhanced chaotic map and its security analysis followed by CBCA in Section 4. A discussion of the experimental results is provided in Section 5. Application of CBCA in image encryption is discussed in Section 6 before the paper is concluded in Section 7.

2. Related work

In literature, cryptosystems based on digital chaos were mostly designed for a broad range of multimedia data types that includes image, audio, and video. Fewer algorithms were proposed specifically for textual data. In Albhrany et al. (2016), a chaotic map was used to design a new block cipher to encrypt textual contents. A plaintext was divided into a set of (8×8)-byte blocks, before permutation and S-boxes were employed to achieve the diffusion and confusion properties. Despite having a large key, the resulting cryptosystem has low entropy and security level. Murillo-Escobar et al. (4953) proposed a chaos-based text cipher, for which a set of rules were employed to achieve a higher throughput. Optimized logistic chaotic maps generated pseudo-random number sequences which were used in various operations. The scheme has a plaintext-dependent key generated using characteristics of the plaintext. Although the cipher achieved higher encryption speeds, it has a small chaotic parameter space, which leads to a smaller key space.

Yasser et al. (2020) proposed novel chaos-based multimedia encryption algorithms based on 2D chaotic maps. The algorithm relied on chaotic perturbation in both diffusion and confusion rounds, where different chaotic maps were used to encrypt different data types. The proposed cipher claims to achieve a throughput of 1.6 Megabytes per second. However, the scheme suffers from poor diffusion, whereby changes to one bit of the plaintext does not propagate throughout the ciphertext, making it susceptible to differential attacks. Another encryption algorithm for textual data

was proposed by Volos et al. (2013). The logistic map (which unfortunately has a limited chaotic range) was employed. The cipher relies on simple operations (exclusive-OR) that can be efficiently realized on a standard computer. Lian et al. (2005) proposed another block cipher based on the chaotic standard map which has three parts: the key generator, diffusion function and confusion based on the standard map. The skewed tent map was used to generate the secret key. Unfortunately, their design requires multiple rounds of confusion operations to achieve optimal security.

Abdullah and Khaleefah (2017) proposed a hybrid text encryption and image steganography algorithm based on chaos and image secret sharing. Firstly, a text message is encrypted and the ciphertext is converted into multiple shadows based on the secret sharing scheme. These shadows are then embedded in multiple host images. The designers claim that the embedding capacity is high and the proposed scheme has a large keyspace. However, the proposed algorithm uses a simple chaos-based algorithm to perform encryption. Its resistance to different well-known attacks was not analyzed. Kordov (2021) also proposed a chaos-based cipher for text-based data. Two chaotic maps were used to generate a pseudorandom number and the text is converted into a bitstream which is finally used to encrypt the text using an exclusive-OR (XOR) operation. The proposed cipher is essentially a stream cipher based on the chosen chaotic maps.

Arifin et al. (2021) introduced a new cipher that improves upon the conventional hill cipher based on the Bernoulli chaotic map to increase its key length. The Bernoulli map is used to build a unimodular matrix. Text is then converted to integers and then permutated using the improved hill cipher. As the main goal is just to enhance the conventional hill cipher, its feasibility to secure real-world data still requires further study. Alshammari et al. (2021) proposed a new lightweight encryption algorithm to protect data collected Internet of Things (IoT) sensors. It was essentially a modified AES algorithm that uses a new chaotic S-box. Despite having good cryptographic properties and a high level of randomness, this was mainly due to the underlying structure of AES rather than the modified S-box. In addition, AES is not known for its lightweight implementation and may not be suitable for IoT devices that have limited processing capabilities.

3. Chaotic map and analysis

Chaotic maps have been used in many security-related applications such as image encryption, hash function, and watermarking algorithms. One-dimensional (1D) chaotic maps have one system variable and one control parameter. Hence, they are very fast and are easy to implement in software. However, 1D classical chaotic maps such as the logistic map have limited chaotic range and low chaotic performance. Directly using these maps in cryptographic applications may result in security issues. Therefore, many remedial techniques such as the cascade, perturbation, delayed chaotic and bit reversal systems were proposed to improve the chaotic properties of 1D chaotic maps. In our work we introduce a generic or flexible model that enhances the chaotic properties and performance of 1D chaotic maps such as the logistic map, then use it in the proposed cipher. Desirable properties such as high ergodicity is very important to provide uniform distribution for chaos-based cryptosystems.

3.1. Enhanced logistic map

The logistic map is a common chaotic map used in many applications. Despite having only one system variable and control parameter, it has the capability to produce chaotic behaviors. Due to its simplicity, it is faster than most 1-dimensional maps and can depict chaotic behavior when its control parameter is set close to 4. The logistic map is mathematically defined as

$$F_L(x_{n+1}, r) = F_L(r \times x_n \times (1 - x_n)) \tag{1}$$

where x_n is the system variable, for which x_0 is the initial condition, and r is a control parameter with a range of [0,4]. However, the chaotic parameter range is very small and the overall map has low complexity. Thus, there have been a number of chaotification methods proposed to enhance its chaotic behavior and widen its chaotic parameter range (Alawida et al., 2019; Alawida et al., 2020). Some other approaches still lead to small chaotic ranges or require the use of other chaotic maps to enhance the logistic map, leading to increased computational overhead. In this paper, we propose a new chaotification method based on perturbing the underlying chaotic map. This is achieved by using the chaotic sequences generated by the latter in a generic equation with high sensitivity due to the use of a multiplicative inverse function. The proposed chaotification model has a generic (i.e. not specific to a particular chaotic map) mathematical structure described as

$$x_{n+1} = \left(\frac{2^k}{2^{F_L(x_n, r)}}\right) mod 1 \tag{2}$$

where x_n is system variable and bounded between 0 and 1, r is a control parameter between 0 and ∞ (for a fair comparison with the classical logistic map, we will limit r between 0 and 4 in our following experiments). The logistic map function, F_L can be replaced with any other chaotic map. The n is the number of iterations while k is a new parameter that balances the new map's computational complexity and its statistical properties. If k is large, the chaotic system will have improved chaotic behavior but takes longer to compute. As a result, we recommend limiting the range to $k \in (5, 15)$. For our experiments, we select $k = 10$ as it falls right in the middle of the recommended range. The result of computing the logistic map is used as an exponent of the denominator to increase sensitivity. The final modulo operation ensures that the chaotic point remains within the range of $x \in [0, 1]$.

The proposed function is designed to be a fraction as it leads to improved statistical properties (Alawida et al., 2019). $\frac{1}{x_n}$ always results in a value larger than 1 when $x_n \in (0, 1)$. Performing the modulo operation will further increase the function's sensitivity and randomness. The fraction, $\frac{2^k}{2^{F_L(x_n, r)}}$ is a generalization of the function introduced in Alawida et al. (2019). To depict the chaoticity of the proposed model, we provide the following mathematical proof:

Lemma 1. *The fractional function $\frac{1}{x}$ is chaotic and highly sensitive to initial conditions whereas the logistic map only has chaotic behavior in the range of $r \in (3.67, 4)$. $\frac{1}{x}$ is chaotic under the modulo operation and has high sensitivity because a small change to x_n will result in a large value (after solving for $\frac{1}{x}$) prior to applying the modulo operation.*

Proof. Let C be the difference between two values, x_0 and y_0. If C is a small real number (implemented using the IEEE 754 double floating point number), then the result after the first iteration of function $\frac{1}{x_n}$ will be large ($\gg 1$). This is mathematically depicted as

$$\begin{aligned} |x_0 - y_0| &= |C| \\ |\tfrac{1}{x_0 - y_0}| &= |\tfrac{1}{C}| \\ \text{Then, } |C| &< |\tfrac{1}{C}|, \text{ and } C \text{ has two cases} \\ 1: |C| \text{ is a small value, } |\tfrac{1}{C}| &= C \times 10^P, \text{ where } P \text{ is the number of digits after the decimal point} \\ 2: |C| \text{ is a large value, } |\tfrac{1}{C}| &> 1 \end{aligned} \tag{3}$$

The result after only one iteration is a large difference between two consecutive state variables. The modulo operation removes the integer portion of the final result. As the state variables x_0 and y_0 are real numbers between 0 and 1, the inverse of their difference, $\frac{1}{C}$ will always be a large value, thus amplifying the sensitivity of the overall chaotic model. □

3.2. Chaotic analysis

3.2.1. Bifurcation diagram

Logistic map has a small chaotic parameter range and contains many windows of periodicity, indicated by unshaded regions in its bifurcation diagram shown in Fig. 1(a). When used for cryptographic applications, these periodic regions must be avoided. Thus, when the control parameter value is used as part of the secret key, the overall keyspace is greatly reduced. Therefore, increasing the chaotic parameter range helps to overcome brute force attacks due to the increased keyspace. To study chaotic parameter range, bifurcation diagrams can be used to represent the relationship between chaotic points and control parameters. Fig. 1(a) and (b) show the bifurcation diagrams of the logistic map and its proposed enhancement. We can see that the new map no longer has windows of periodicity and the state variable, x is evenly distributed between 0 and 1 regardless of r. This implies that the proposed model has not only eliminated the periodic windows of the logistic map but also enlarged its chaotic parameter range.

3.2.2. Lyapunov exponent

Sensitivity to small changes to initial conditions and control parameters play a big role in chaos-based cryptographic applications. LE is a mathematical indicator that can be calculated under different initial conditions and parameter settings. A chaotic map with large and positive LE values have chaotic points that diverge after a shorter period of time (fewer iterations) as compared to chaotic maps with smaller or negative LE values. In other words, larger LE values imply increased unpredictability and sensitivity. In contrast, negative and zero LE values indicate periodic behaviour and chaotic points that converge after a number of iterations. We calculate the LE values of both the classical and enhanced logistic maps under the same initial conditions and parameter settings. Fig. 2 shows the LE values of two chaotic maps. We can observe that the new enhanced logistic map has larger positive LE values for nearly the entirety of the $r \in [0, 4]$ range whereas the logistic map only has a small region of positive LE values. These results confirm that the enhanced logistic map has higher unpredictability and sensitivity as compared to its classical counterpart.

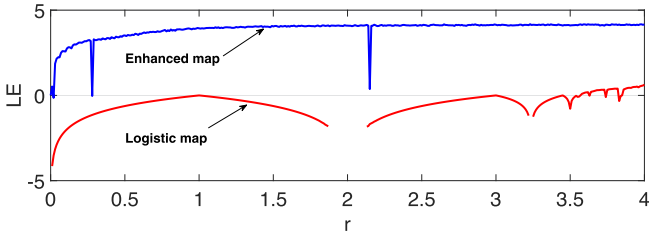

Fig. 2. LE values of the logistic map and enhanced logistic map under the same parameter settings.

There are some of values of the control parameters are produce non-chaos such as 0.28 and 2.15, these parameters produce the fixed points after a few iterations of the map. A fixed point means that the chaotic map generate the same input $x_j = f(x_j, r)$. Each chaotic map can generate fixed point, but the type of point is determined if the chaotic behavior attracted to the fixed or diverge on it. To analyze the stability of fixed points, we calculate the Jacobian matrix from the derivative of Eq. (2). To obtain accurate results, mod operation is not used in the Jacobian matrix. The Jacobian matrix of Eq. (2) is calculated as

$$f'(x) = \ln(2)\, r \cdot (2x - 1) \cdot 2^{10 - r \cdot (1-x)x}$$

where $f'(x)$ is derivative result for any value of x and r.

When the Jacobian matrix of fixed points goes outside the interval [-1,1], the fixed points are unstable and all surrounding states will not attract them. When the Jacobian matrix of fixed points falls within the interval [0,1], the fixed points are attractors to adjacent states, indicating that the system is stable. We use $r = 0.28$ and the fixed point is $x_n = (0.502405747468856)$. The

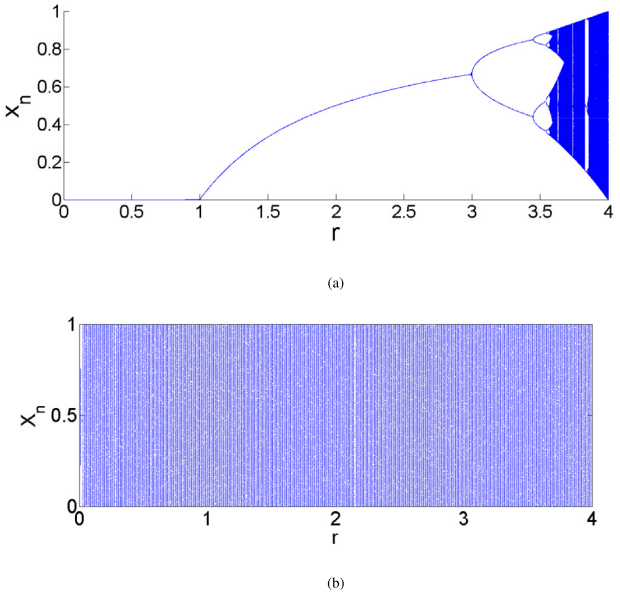

Fig. 1. Bifurcation diagrams of the (a) Logistic map, (b) Enhanced logistic map.

jacobian matrix of the fixed points is 0.9109, which indicates that the proposed model at this fixed point is attract other states, indicating non-chaotic behaviour. The second control parameter $r = 2.15$ is used to generate the chaotic sequence, the chaotic sequence after a few iterations generate neared points, which are very near to each other. Fig. 3 show the chaotic sequence when $r = 2.15$ and $x_0 = 0.2$. It shows the chaotic points are very close to each other. As such, we should ensure that the control parameters are always taken from within the chaotic range.

3.2.3. Fuzzy entropy

A chaotic system is described as complex when a lot of information is needed to explain its behavior. A highly complex system can produce random outputs with no observable statistical patterns. To evaluate complexity for a chaotic map, we rely on the fuzzy entropy (FuzzyEn) metric because it uses the Gaussian function instead of Heaviside function to precisely compute the complexity estimate (Xie et al., 2011). The Gaussian function is written as

$$\Theta(w_{ij}^m, t) = exp(\frac{-(w_{ij}^m)^2}{t}), \tag{5}$$

where m is embedding dimension that is set to 2 for a standard estimation, t is the tolerance value, $w = \max_{i,j\in(0,m-1)}|x(i) - x(j)|$ is the maximum distance between two sequences equivalent to m.

For the two chaotic maps, logistic and enhanced logistic map, FuzzyEn values are estimated under the same parameter settings and initial conditions. Fig. 4 shows the FuzzyEn values for both maps. We can see that the enhanced logistic map has large FuzzyEn values as compared to the classical logistic map which indicates elevated complexity. Thus, it is more difficult to identify patterns that can describe the enhanced map's behavior.

3.2.4. Local Shanonn Entropy (LSE)

Entropy is a good measure of randomness and unpredictability. For a chaotic system, entropy can also be used as an indicator of the ergodicity property. If a chaotic trajectory can be visited equally all sub-regions in the phase space, then a chaotic map is ergodic, where each sub-region has equal effect on chaotic behaviors. On the other hand, a chaotic trajectory that only visits certain sub-regions and remains in particular sub-regions for long periods of time (multiple iterations), this implies that the system is more predictable, has a biased distribution, strong correlation between chaotic points, and is susceptible to chaotic parameter estimation attacks.

In this experiment, we use local Shannon entropy (LSE) to quantitatively evaluate the ergodicity of both the enhanced and classical logistic map. LSE values are obtained by calculating Shannon Entropy (SE) for non-overlapping blocks of the chaotic trajectories (Wu et al., 2013). SE is calculated as

$$H_j(S) = \sum_{i=0}^{L-1} p(S_i) log_2 \frac{1}{p(S_i)}, \tag{6}$$

where $j < K$ is the index of the current non-overlapping block, K is the total number of non-overlapping blocks being tested, $p(S_i)$ is the probability of occurrence of a symbol, $S_i \in [0, 1]$, and $L = 256$ is number of intervals. LSE is the mean of the SE values calculated as

$$H = \sum_{j=1}^{K} \frac{H_j(S)}{K}, \tag{7}$$

where K is the number of non-overlapping blocks being tested. A chaotic trajectory can successfully pass the LSE test if the LSE value falls between two critical values, H_{left} and H_{right}. This implies that the chaotic trajectory can visit all intervals of the phase space. H_{left} and H_{right} are calculated as

$$\begin{aligned} H_{left} &= \mu_H - \Phi^{-1}(\tfrac{\alpha}{2})\sigma_H/\sqrt{K} \\ H_{right} &= \mu_H + \Phi^{-1}(\tfrac{\alpha}{2})\sigma_H/\sqrt{K} \end{aligned} \tag{8}$$

where μ_H and σ_H are the mean and the standard deviation of the LSE values, $\Phi^{-1}(.)$ is the inverse cumulative distribution function (CDF) of the standard normal distribution $N(0, 1)$, and α is the significance level. In our experiments, we set $\alpha = 0.001$ and $K = 30$.

The enhanced logistic map and logistic map are used to generate 10^5 chaotic points each, and the phase space is divided into 256 intervals. Next, K non-overlapping sequences with 1936 chaotic points are randomly selected for testing. The critical values are calculated as $H_{left} = 7.9015156987$ and $H_{right} = 7.903422936$. Fig. 5 shows the LSE values for the enhanced logistic chaotic map and the underlying logistic chaotic maps for different control parameters. Results show that the enhanced map has high LSE values which implies higher ergodicity and randomness.

The enhanced map managed to pass the LSE test for most of its control parameter settings, where the LSE values fall between the critical points. In contrast, the classical logistic map was not able to pass the LSE test for the entirety of its control parameter range. These results point out that the enhanced logistic map can provide higher ergodicity than the classical logistic map and is better suited for sensitive applications such as symmetric-key encryption.

Next, we compare LSE values for different K when $\alpha = 0.001$. These LSE values were been calculated for a randomly generated image which is basically the equivalent of a random number sequence. A comparison is made between the enhanced map and logistic map and the standard LSE values as reported in Wu et al. (2013). LSE values that fall between these standard LSE values

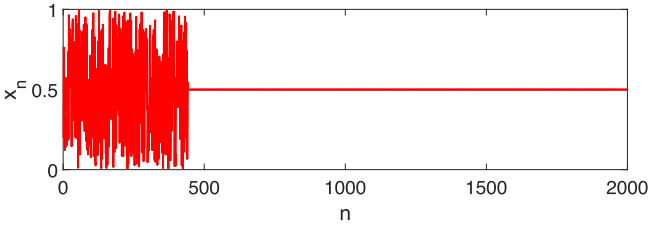

Fig. 3. Chaotic sequence when $r = 2.15$ and $x_0 = 0.2$.

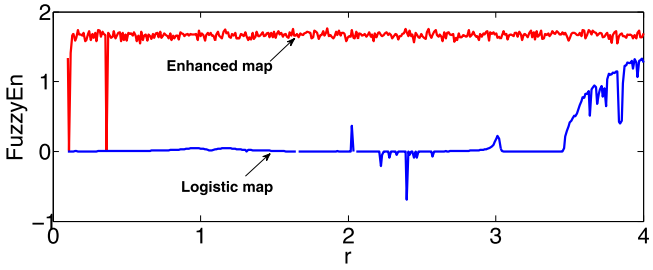

Fig. 4. FuzzyEn values of the logistic map and enhanced logistic map under the same parameter settings.

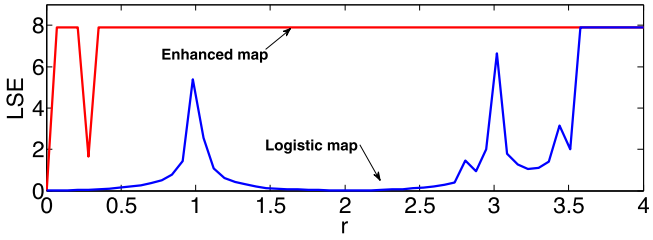

Fig. 5. LSE values of the enhanced logistic map and logistic map under the same parameters settings.

H_{left} and H_{right} are considered to have passed the LSE test. The chaotic map trajectories were generated using $r = 3.999$and $x_0 = 0.33$ for a fair comparison. Table 1 shows the LSE results for varying K values. The enhanced logistic map has LSE values that passed the LSE test for different K values while the logistic map failed in all instances. Again, the results imply that the enhanced map leads to improved randomness and ergodicity as compared to its peer.

However, the enhanced logistic map has some low FuzzyEn values especially in the region of $r \in (0, 0.5)$. The reason behind this phenomenon is the fixed point that occurs when using a control parameter value of $r = 0.28$. The enhanced chaotic map under this control parameter falls into a fixed point in fewer than 287 iterations, and the fixed point attracts other states. In contrast, control parameter 2.15 creates a fixed point after large number of iterations that exceed 100,000 iterations, which allows to calculate FuzzyEn in the normal way.

3.3. Comparison and discussion

The newly enhanced chaotic map has better chaotic performance than the logistic chaotic map in all security aspects. We now compare the proposed map with other enhanced chaotic maps in literature, using LE and FuzzyEn as metrics for evaluating sensitivity and complexity respectively. We implement all of the improved chaotic maps on the same device and platform for a fair comparison. We calculate the mean values of these indicators for all the selected chaotic maps and tabulate them in Table 2. One can observe that the proposed enhanced logistic map outperforms the other chaotic maps in both aspects.

Results suggest that the enhanced chaotic map outperforms the standard logistic map in terms of bifurcation properties, LE and FuzzyEn measures. The proposed chaotification model is the driving force behind the improved performance, which relies on the multiplicative inverse function to increase the sensitivity of the chaotic map. The output of the classical chaotic map (logistic map) is also employed as an exponent of the denominator, specifically in the multiplicative inverse function to further boost sensitivity. The final output will serve as the modulo function $\frac{1}{x}$, which exhibits more chaotic behaviour than the standard chaotic map (Alawida et al., 2019).

4. Chaotic block cipher algorithm (CBCA)

In this section, we use the enhanced chaotic map to design a new block cipher. To showcase the capability of the proposed map in cryptographic applications and to overcome some drawbacks of prior chaos-based encryption algorithms. The chaotic map will be involved in both diffusion and confusion operations. A chaotic map with good statistical properties contributes towards the cipher's resistance to attacks such as correlation, brute force and differential attacks. In the proposed encryption algorithm, we use the enhanced logistic map to achieve diffusion and confusion properties simultaneously. The map's ergodicity property is leveraged in the permutation process while its chaotic sequence is used for substitution. CBCA can encrypt a plaintext of any size, represented as 8-bit unsigned integers. CBCA has two stages: the first is to generate the chaotic variables using a secret key, which will then be used to generate a data sequence used in the encryption process. This is equivalent to the key scheduling algorithm of conventional block ciphers. This second stage is the encryption process itself. Both stages are detailed in the following subsections.

Table 2
Chaotic map performance comparison.

Chaotic maps	LE	FuzzyEn
Enhanced logistic map	**4.121**	**1.832**
Logistic map	0.212	0.654
TM-DFSM (Alawida et al., 2019)	1.387	1.421
Delay logistic map (Liu and Miao, 2017)	1.728	1.622
LSC (Hua et al., 2019)	1.412	1.532
TSTS (Alawida et al., 2019)	1.822	1.638
TLM (Zhou et al., 2015)	0.812	0.898
LCT (Hua and Zhou, 2016)	0.464	0.536
C-P map (Zhu et al., 2018)	0.356	0.235
Sine-G (Alawida et al., 2019)	1.151	1.535
LS-S map (Lan et al., 2018)	0.345	0.457

4.1. Chaotic variable generation

In this section, we use the secret key to generate the initial condition and control parameter of the chaotic map, both of which are henceforth referred to as chaotic variables. A secret key should have a length of at least 128 bits to resist brute force attacks. The main goal is to design an algorithm that generates the chaotic variables with high sensitivity to the secret key. Prior chaos-based algorithms generally use secret key bits to construct the chaotic variables in a direct manner: using the key bits to modify the fixed-point or floating-point representation of these variables. In the proposed algorithm, we still rely on both floating-point and fixed-point numbers, $U(i,f)$, where i is the number of bits are used to represent the integer portion of a real number while f is the number of bits required to represent the fractional portion of the real number. The resulting fixed-point number can be calculated as $\sum_{n=0}^{52} f_n \times 2^{-n}$.

Rather than using the key bits to modify or instantiate the chaotic variables directly, we use the secret key to perturb their values using one of the chaotic perturbation methods. We first divide the secret key into consecutive, overlapping 52-bit blocks, each of which presents one chaotic point (system variable) bounded between 0 and 1. Each 52-bit value is used to perturb the control parameter and initial chaotic point (initial condition). These divided key points are used to create new chaotic points and control parameters before using them to generate new chaotic points (data sequence). The last chaotic point and two control parameter values will be used for next steps in CBCA. The steps of key generation are below:

1. Divide the secret key into consecutive, overlapping 52-bit blocks. For example, Block 1 comprises bits 1 to 52 of the secret key and Block 2 comprises bits 2 to 53 of the secret key. This will continue until the final block which is made up of bits 77 to 128. Thus, the total number of blocks is 77. Using these overlapping blocks increases the sensitivity of the chaotic variables to the secret key.

Table 1
Comparison of LSE values of the enhanced map and logistic map with distinct K values, $r = 3.999$ and $x0 = 0.33$ for two maps.

K	30	40	50	60	70
H_{left}	7.901515698	7.901754103	7.901897145	7.901992507	7.902060623
H_{right}	7.903422936	7.903184531	7.903041489	7.902946127	7.902878011
Enhanced map	**7.902684751**	**7.902487545**	**7.902854625**	**7.901999857**	**7.902711354**
Logistic map	7.676137736	7.679071335	7.677496852	7.6766976452	7.676048960

2. The 52-bit values are fixed-point real numbers between 0 and 1 used as seed values (or initial values), and also to perturb the control parameter. The functions used to modify the chaotic variables are as follows:

$$x_i = (x + key_i) \, mod 0.99999 \tag{9}$$

$$r_i = ((r + x + key_i) \times 2^{32}) \, mod 3.99999 \tag{10}$$

where x_i and r_i denote the seed values and control parameters respectively. key_i is the i-th 52-bit block obtained from the secret key. We select 0.99999 and 3.99999 to ensure the resulting chaotic variables remain in the appropriate phase space, thus avoiding weak key scenarios (Teh et al., 2020).
3. Iterate the enhanced logistic map $77 + R$ times, where the value of R depends on whether the resulting chaotic point and control parameter values are being generated for the first or second round of CBCA. When generating the chaotic variables for the first round of encryption, we set $R = 10$. Then, we iterate the map an additional 10 times to produce the second set of chaotic variables for the second round of encryption (i.e. $R = 20$ in total). The two resulting data sequences are completely different. As mention in the Section 3.2.2, the enhanced map has excellent LE values on the various parameter settings.
4. The last seed value and two control parameters are used in the next CBCA round. The generated chaotic variables are used in the block cipher to generate data sequences that will use in the permutation and confusion operations.

The steps involved in generating the chaotic variables (and subsequently, their corresponding chaotic sequences and permutation patterns) only need to be performed once for each secret key. Thus, similar to regular encryption algorithms, this can be pre-computed and does not incur additional computational overhead.

Algorithm 1: Generation of chaotic variables (GCV)

Data: Secret key *key*, Number of blocks M, Round .
Result: Seed value x, control parameter r

$N = length(key)$;
$M = N - 51$;
$x = 0.33$;
$r(1) = 3.999$;
for *i=1 to M* **do**
  $key_1(1:52) = key(i : i+51)$;
  $key_2(i) = 0$;
  for *j=1 to 52:+1* **do**
    if $key_1(j) == 1$ **then**
      $key_2(i) = key_2(i) + 1/2^j$;
if *Round* $== 1$ **then**
  $R = 10$;
  $S = 1$;
else
  $R = 20$;
  $S = M$;
for *h=S to (M+R)* **do**
  if $h > M$ **then**
    $key_2(h) = key_2(M)$
  $x = (x + key_2(h)) \; mod \; 0.99999$;
  $r(h) = ((r(h) + x + key_2(h)) \times 2^{32}) \; mod \; 3.99999$;
  $x = 2^{10}/(2^{(r(h)\times x\times(1-x))}) \; mod \; 1$;

Algorithm 1 summarizes the steps involved in generating the new chaotic variables (seed chaotic point and list of the control parameters). The proposed cipher is a symmetric-key cipher, so the same secret key is used by both sender and receiver. The secret key should be exchanged securely using conventional methods such as public-key encryption. Unlike conventional block ciphers, the secret key is flexible, whereby a user can select any key size depending on their application's requirement.

4.2. Chaotic block cipher algorithm

In this section, we describe the design of the block cipher itself. CBCA can achieve both diffusion and confusion simultaneously based on the ergodicity of the enhanced chaotic map and its chaotic points. The secret key is used to generate chaotic variables that will be used to generate data sequences for encryption and decryption. Let N_p denote the number of 8-bit words in a plaintext. The enhanced logistic map is iterated N_p times and the resulting chaotic trajectory (data sequence) x_n, where $n = \{1, 2, 3, \ldots, N_p\}$ is obtained. The chaotic trajectory should be highly random, aperiodic and have a high LSE which implies ergodic behavior. y_n, which denotes random indexes of the 8-bit words in the data sequence, is calculated from the chaotic points X_n. These random indexes form a set, e.g., $y_n = \{8, 29, 77, 1, \ldots, N_p, 4, 5\}$. Also, let $Perm_n$, where $n = \{1, 2, 3, \ldots, N_p\}$, represent an array of size N_p that represents the permutation pattern used in the proposed block cipher. The steps of the proposed CBCA are as follows:

1. The phase space of the enhanced logistic map is divided into a set of non-overlapping intervals I_i, where $i = \{1, 2, 3, \ldots, N_p\}$.
2. The interval that each chaotic point generated by the chaotic map falls into will be noted. For example, assume the 1st chaotic point, y_1 falls into interval number four, I_4. Thus, we set the fourth permutation array element as the index of y_1, which is $Perm_4 = 1$. If a chaotic point falls into the intervals that have been visited before, then the index of that chaotic point is put into the another 1D array, E_n. The chaotic map will be iterated until it generates its final $(N - p)$-th chaotic point.
3. Array values stored in E_n is used to fill up the remaining array entries in *Perm* (in order of how the values are stored in E_n), completing the final permutation pattern, *Perm*. By generating *Perm* based on the ergodic nature of the enhanced map, we can ensure that the permutation pattern is sufficiently randomized and evenly distributed. Fig. 6 provides a numerical example for the permutation process, whereby 10 chaotic points $(x_1, x_2, \ldots, x_{10})$ are generated. For example, $x_1 = 0.81$ falls under interval I_9. Thus, the value stored in $Perm_9 = 1$. Next, $x_2 = 0.76$ falls under interval I_8, thus $Perm_8 = 2$. If another chaotic point falls under the same interval, e.g. $x_4 = 0.73$ in I_8, the index of the chaotic point will be stored in E. After all chaotic points are generated and their corresponding indexes are mapped to *Perm* and E, the values stored in E will be used to fill up the remaining empty entries in *Perm*.
4. To change (or effectively substitute) each 8-bit word in the plaintext, they are XOR-ed with chaotic points after they are permuted. These operations can be mathematically summarized as

$$swap(C_i, C_{Perm_i}) \tag{11}$$

$$C_i = (C_i \oplus x_i \oplus G) \tag{12}$$

where C_i is the i-th plaintext/intermediate ciphertext character and the $swap(val_1, val_2)$ function swaps the character at position val_1 with the one at position val_2. This permutation and substi-

tution process is repeated for each word in the plaintext/intermediate ciphertext. x_i are data points that are obtained after being converted into 8-bit unsigned integers by $x_i = uint8((x_i \times 2^{32}) mod 256)$. G is an 8-bit unsigned integer that represents a prior plaintext/ciphertext word. Depending on the encryption round, selection of G is based on the following:

$$G = \begin{cases} i = 1 \text{ and } round = 1, & 0 \\ i = 1 \text{ and } round = 2, & (C_{N_p} + i)^3 \ mod 256 \\ Otherwise, & (C_{i-1} + i)^3 \ mod 256 \end{cases} \quad (13)$$

The term, $(C + i)^3 mod 256$, was selected to increase the impact of prior ciphertext values to achieve diffusion. A small change in the prior plaintext/ciphertext values will be amplified and significantly impact the encryption of the remaining values.

5. The diffusion and confusion are simultaneously achieved by permuting and substituting each word in the plaintext. The ciphertext obtained is after two encryption rounds.

For the second round, the second set of chaotic variables obtained after $77 + K$ where $K = 20$ iterations of the key generation algorithm are used to generate another data sequence of length N_p. The same encryption steps are repeated. Fig. 7 shows the flowchart of the steps involved in CBCA. Both confusion (and to some extent, diffusion) are achieved by XOR-ing the plaintext with chaotic points and previous plaintext words. Further diffusion is achieved through permutation, influenced by the ergodicity of the enhanced chaotic map. CBCA only requires two rounds to achieve good diffusion and confusion properties while maintaining a high encryption throughput. Further details of CBCA can be obtained from Algorithm 2.

	1	2	3	4	5	6	7	8	9	10
I_i	0 - 0.09	0.1 – 0.19	0.2 - 0.29	0.3 – 0.39	0.4 – 0.49	0.5 – 0.59	0.6 – 0.69	0.7 – 0.79	0.8 – 0.89	0.9 - 1
x_n	0.81	0.76	0.33	0.73	0.98	0.22	0.67	0.08	0.99	0.23
$Perm_n$	8		6	3			7	2	1	5
E_n	4	9	10							
$Perm_n$	8	4	6	3	9	10	7	2	1	5

Fig. 6. Permutation example.

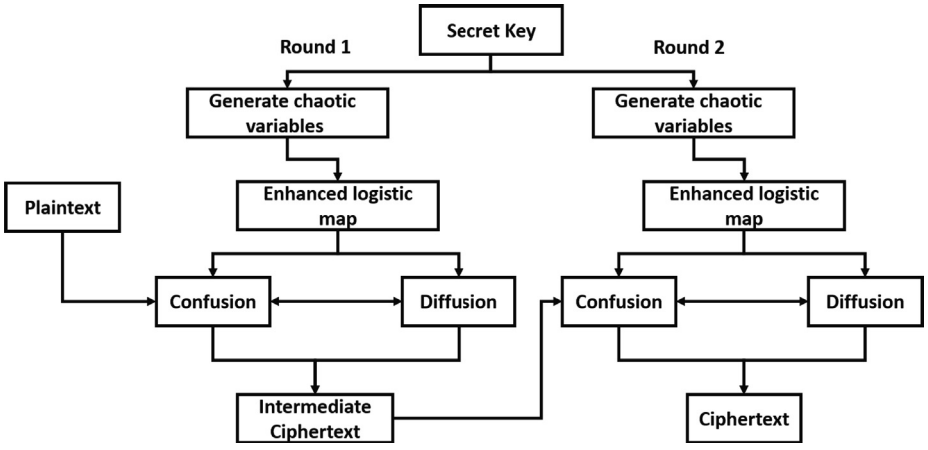

Fig. 7. CBCA flowchart of encryption side.

Plaintext	Ciphertext
ABCDEFGHIJKLMNOPQRSTUVWXYZ	'BHûÄCA□&□ÉÄëÏbw□°EDA,□EPá□'
BBCDEFGHIJKLMNOPQRSTUVWXYZ	'□OäLF□GD±□!RåÃ(¶ég□□Ñf ¶FÒp'
ABCDEFGHIJKLMNOPQRSTUVWXYY	'LHDGjC¹□o□□¤FCB□□Ç3^+wNà¢É2□'
ABCDEFGHIJKLNNOPQRSTUVWXYZ	'¯Br□Ý□µ÷Ú3ã□□¿¹N9çf□ÞÓF□□□Ù ö'

Fig. 8. Plaintexts and corresponding ciphertexts.

Algorithm 2: CBCA algorithm

Data: Secret key key, Plaintext P
Result: Ciphertext C

$C = dec2bin(P, 8)$;
$[N_p, L] = size(C)$;
for *round=1 to 2* **do**
  $[r_1, r_2, x_0] = GCV(key, M, round)$;
  $Perm = zeros(1, N_p)$; $k = 0$;
  for *i=1 to* N_p **do**
    if x_0 *mod* 2 $== 0$ **then**
      $r = r_1$;
    else
      $r = r_2$;
    $x_0 = (2^{10}/(2^{(r \times x_0 \times (1-x_0))}))$ *mod* 1;
    $y_i = fix((x_0 \times 2^{17.5} \ mod \ N_p) + 1)$;
    $x_i = uint8(x_0 \times 2^{32} \ mod \ 256)$;
    if $Perm(y_i)$ ==*0* **then**
      $Perm(y_i) = i$;
    else
      $k{+}{+}$; $E_k = i$;
  $k = 0$;
  for *i=1 to* N_p **do**
    if $Perm_i$==*0* **then**
      $k{+}{+}$; $Perm_i = E_k$;
  for *i=1 to* N_p **do**
    $temp = C_i$;
    $C_i = C_{Perm_i}$;
    $C_{Perm_i} = temp$;
    if *i==1 and round==1* **then**
      $G = 0$;
    else
      if *i==1 and round==2* **then**
        $G = (C_{N_p} + i)^3 \ mod \ 256$;
      else
        $G = (C_{i-1} + i)^3 \ mod \ 256$;
    $C_i = (C_i \oplus x_i \oplus G)$;

4.3. Chaotic block cipher decryption algorithm

Decryption merely requires the same operations as encryption but in reverse order. To decrypt the ciphertext, we use the same Algorithm 1 but instead starting from round 2, followed by round 1. The first three steps of the encryption algorithm are reused without modifications. In Step 4 and 5, the ciphertext is decrypted in reverse order (from the last word to the first word) for both rounds. The same secret key (used for encryption) must be used for decryption to successfully recover the plaintext.

4.4. Discussion

CBCA is proposed to encrypt the plaintexts of different sizes and provide a high security margin by using digital chaos. As Diffusion and confusion properties are very important encryption algorithms, CBCA was designed to ensure both properties are achieved successfully. The advantages of CBCA are as follows:

- The enhanced logistic map has high LE and LSE values that ensures highly chaotic behaviour and ergodicity, which then reflects on the permutation/confusion process.
- Diffusion and confusion operations can be performed simultaneously each round. Many existing chaos-based schemes have distinct diffusion and confusion rounds, leading to higher computational complexity.
- In contrast to conventional block ciphers that require many rounds (a recent cipher, SPEEDY, requires at least 5 rounds for practical security (Leander et al., 2021)) to ensure strong diffusion and confusion properties, CBCA requires at least two rounds.
- The secret key has an effect on both diffusion and confusion properties, unlike in conventional cryptosystems where operations such as substitution or permutation are static and not key-dependent.
- The chaotic variables generated from the secret key are highly sensitive to slight key changes due to the chaotic perturbation operations involved. In total, two initial chaotic points and four control parameters are generated for the cipher's 2-round encryption process. Each chaotic point and control parameter are represented as a 52-bit fixed-point numbers. This means that CBCA depends on 260 bits in final chaotic variables to generate data sequences. Thus, the cipher can provide a security level of up to 260 bits with a flexible key length.

5. Results and analysis

In this section, we experimentally evaluate the security, statistical properties, and overall performance of CBCA. The various metrics involved include randomness, key sensitivity, plaintext sensitivity, correlation and entropy tests. Apart from that, we also analyze the encryption speed of CBCA for both small and large amounts of data. Fig. 8 shows different plaintexts and their corresponding ciphertexts; a small difference (as shown in red fonts) in the plaintexts, lead to an entirely different ciphertext which is noise-like. Note that the ciphertext is being displayed using Unicode symbols.

5.1. Ciphertext randomness testing

Block ciphers are ubiquitous primitives that can be used as building blocks for other cryptographic primitives such as hash functions and pseudo-random number generators. Regardless of its application, the ciphertexts generated from a block cipher should depict essential properties such as high complexity, long cycle length (non-periodic behaviour), uniform distribution, and efficiency. To test for statistical randomness, we apply the NIST SP 800–22 test suite on CBCA. To generate the test samples, we produce random ciphertexts by randomly toggling a single bit of an arbitrary plaintext. A test sample is considered to have passed each sub-test when $P_{value} \geqslant \alpha$ where $\alpha = 0.01$ is the significance level. The number of test samples (ciphertexts) used for testing is 10^3, where each test sample has a length of $n = 10^6$ bits. A plaintext consisting of interleaved bit values of 1s and 0s, 10101.....01 is encrypted to produce the first test sample. The other test samples are generated by toggling 1 bit randomly in the plaintext. Based on results shown in Table 3, CBCA successfully passed all 15-sub-tests of the NIST test suite. This implies that the outputs of the block cipher are sufficiently random, evenly distributed, and complex.

5.2. Correlation test

The correlation between adjacent characters can be used to evaluate the diffusion property of a cryptosystem. We encrypt a plaintext of length 2^{16} bits and obtain its corresponding ciphertext. The plaintext consists entirely of the same 8-bit word repeated N times. The ciphertext should be uniformly distributed, such that there will no longer be any correlation between adjacent characters. We compute the correlation coefficient among characters of the ciphertext. The experiment is repeated for multiple plaintexts to study further the behavior of the algorithm. Fig. 9 shows the correlation correction results for 100 different plaintexts that are obtained by toggling random bits in an original plaintext consisting of interleaved 1s and 0s (101010...1010). It can be seen that CBCA has no correlation between adjacent words as the correlation values are approximately close to zero.

5.3. Keyspace and sensitivity

The proposed cipher supports the secret keys equal to or larger than 128 bits. Thus, the keyspace is at least 2^{128} bits or larger. The initial chaotic variables that are constructed using the secret key include the initial condition and control parameters, where 52-bit sections of the secret key are used in the fractional portion of their fixed-point representation. When testing the proposed cipher, we use a standard secret key length of 128 bits. Rather than using the secret key to initialize the chaotic variables directly, perturbation methods were used to ensure that every key bit has equal effect on all chaotic variables. This increases the overall sensitivity to the secret key and avoids weak key problems that affect many existing chaos-based encryption algorithms. Thus, the overall keyspace is 2^{128}, which is large enough to resist brute force attacks (based on current computational capabilities).

To analyze key sensitivity of the proposed cipher, we observe the effect of a 1-bit change to the secret key on the resulting ciphertext. Firstly, the enhanced logistic map has high LE values as mentioned in the Section 3.2.2, which reflects high sensitivity to small change to chaotic variables. Secondly a small change to the secret key will generate entirely new chaotic variables, and subsequently lead to different chaotic sequences. In this experiment, we change 1 bit of the secret key to obtain two closely related secret keys. These keys are then used to encrypt the same plaintext to obtain two corresponding ciphertexts. The test is repeated for all 128 possible bit locations of the secret key. We denote H as the variance ratio of each bit. Fig. 10 shows the results of the key sensitivity for the different secret keys. We can see that our algorithm has variance ratios of approximately 50% for the secret keys indicating that our algorithm is extremely sensitive to the secret key.

In the decryption algorithm, the key sensitivity is a very significant property. Decryption of a ciphertext even with a slightly modified secret key should generate a completely different message. In our algorithm, if a secret key is wrong, then the decryption of the ciphertext results in noisy data. To depict the sensitivity of the decryption algorithm to the secret key, we toggle one bit of the secret key and decrypt the ciphertext. Correlation testing of the decrypted ciphertexts (using both the original and wrong key) is performed. The experiment is repeated for all the bits of the secret key. Fig. 11 shows the results of the correlation testing. We can see that the cipher's decryption algorithm also has high sensitivity to small changes to the secret key, whereby all correlation values are close to zero.

Table 3
NIST SP 800-22 results.

Sub-tests	ciphertexts
Frequency	0.4251
Block frequency	0.3321
Cumulative Sums*	0.7354
Runs test	0.2152
Longest run	0.1243
Binary matrix rank	0.2564
FFT	0.5421
Non-overlapping template.*	0.3215
Overlapping template.*	0.6532
Overlapping template.	0.5241
Universal	0.3265
Approximate entropy	0.1782
Random excursions.*	0.3965
Random excursions variant.*	0.3854
Serial*	0.1265
Linear Complexity	0.9865
Success Counts	15/15

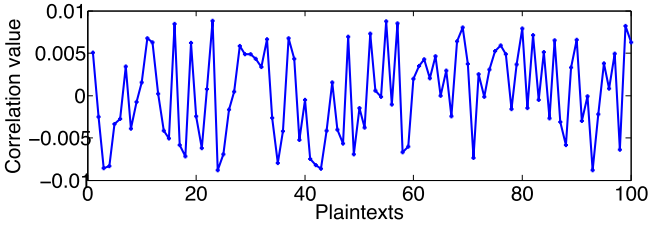

Fig. 9. Correlation values of 100 ciphertexts.

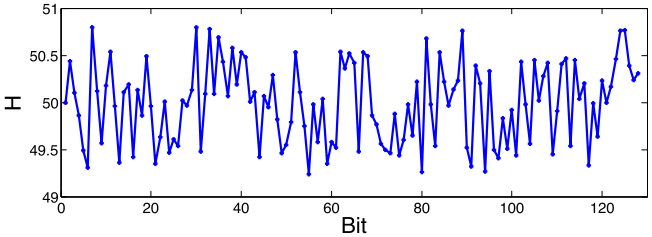

Fig. 10. Secret key sensitivity to small change.

5.4. Resistance to cryptanalytic attacks

Another advantage of the proposed cipher is that both its substitution and permutation operations are key-dependent. Thus, any attacks that rely on approximating the behavior of the cipher's operations, such as differential, linear or algebraic attacks, are prevented. Even if an attacker is able to obtain an approximation of the substitution or permutation operations, the approximation is only valid for that particular secret key. For example, to perform a differential attack, the difference propagation and its corresponding differential probability for nonlinear operations, such as S-boxes or modulo addition, must be known. For ciphers relying on S-boxes (e.g. SPN or GFN ciphers), the differential distribution table

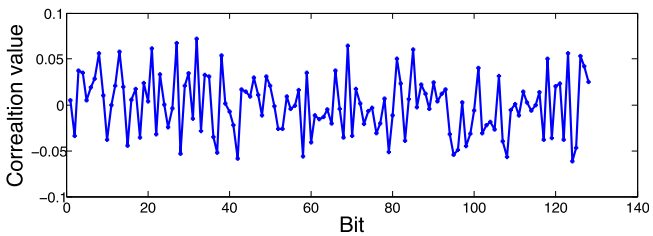

Fig. 11. Secret key sensitivity to small change.

for the S-boxes can be calculated (Biham and Shamir, 1991) whereas for ciphers with additions modulo 2^n operations (e.g. ARX ciphers), the differential properties are evaluated with respect to XOR-differences (Lipmaa and Moriai, 2001).

In the proposed cipher, the nonlinear operation can be found in Eq. 12, when the plaintext/intermediate ciphertext word, C_i is XOR-ed with x_i. Although XOR is a linear operation, the value of x_i is generated from the chaotic map, which involves multiple nonlinear operations and is indirectly dictated by the secret key value. In essence, $C_i \oplus x_i$ is equivalent to an 8-bit key-dependent substitution operation, which substitutes C_i with another value. Unlike a regular substitution operation, this operation is non-bijective (one-to-many). To even begin estimating the differential properties of this operation, an attacker needs to know the key value or equivalent key value (values of the chaotic variables), which needs to be recovered via exhaustive key search or brute force attack. In other words, differential cryptanalysis will have no computational advantage over brute force. Due to the duality between differential cryptanalysis and linear cryptanalysis, the same argument can be applied to the latter technique (Matsui, 1994).

For algebraic attacks, nonlinear operations such as S-boxes are represented as an over defined system of algebraic equations (Courtois and Bard, 2007). The proposed cipher can be seen as using an 8-bit key-dependent substitution operation that substitutes each plaintext word in a dynamic manner, whereby the same plaintext word in different locations in the plaintext will be substituted differently. Thus, an attacker cannot formulate the system of equations to describe the substitution process. Even if it were possible, the system of equations need to be generated for different lengths of plaintexts. Thus, due to the random and dynamic nature of the substitution process (Courtois, 2004), we conjecture that the proposed cipher is resistant to algebraic attacks.

Next, we provide some observations of the proposed cipher's security against the following broad classes of attacks:

Ciphertext-only attack (COA): COA is where an adversary attempts to obtain information about the secret key or plaintext merely by observing ciphertexts. As long as a block cipher generates statistically random outputs (bias-free), the adversary will not be able to deduce any further information about the cipher. In Section 5.1, it was already shown that CBCA successfully passed the NIST statistical test suite, which implies security against COA.

Known-plaintext attack (KPA) and chosen-plaintext attacks (CPA): KPA is where an adversary has a quantity of plaintexts (that cannot be specifically chosen) and the corresponding ciphertexts. CPA is similar to a KPA with the added advantage (for the adversary) to choose plaintexts. KPA is less practical than a chosen-ciphertext attack. If a block cipher is shown to be insecure against KPA, it is also insecure against CPA (Biryukov et al., 2011). Earlier in this section, we highlighted the CBCA's security against differential, linear and algebraic attacks, all of which fall under the CPA class of attacks. We also further analyze CBCA's security against both of these classes of attacks in Section 6.

Chosen-ciphertext attacks (CCA): CCA is the inverse of CPA, where an adversary can select a ciphertext and can obtain its corresponding plaintext. CCA is commonly deemed too strong of an attack assumption, whereby mots ciphers that are not CCA secure are still sufficiently secure for practical purposes. Generally, all block ciphers (including AES) are susceptible to CCA when used with common modes of operation such a cipher block chaining (CBC), cipher feedback (CFB) and output feedback (OFB) due to ciphertext malleability (Canetti et al., 2003). A CCA-secure encryption scheme can only be obtained by ensuring changes to the ciphertext will result in decryption not producing any message such as authenticated encryption modes like CBC-MAC (Rogaway, 2011).

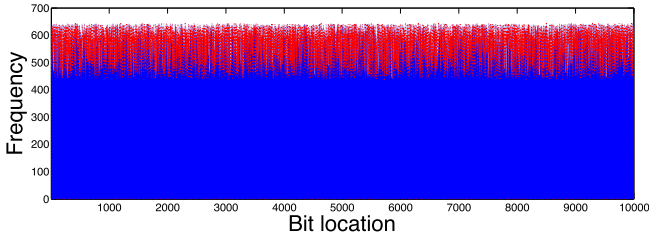

Fig. 12. Distribution of ciphertexts for 10,000 bits of plaintexts.

5.5. Distribution of ciphertext

A ciphertext has to be uniformly distributed to be secure against different statistical attacks. To analyze CBCA for uniform distribution, the following experiment was performed: First, an arbitrary plaintext is generated and one bit is selected at random to be flipped to generate a second plaintext. Both plaintexts are encrypted, and their resulting ciphertexts are compared to identify how many locations of bits have been changed. The experiment is repeated 10,000 times, and the minimum, maximum, and mean changed bit number is recorded. The plaintext size is 10,000 bits and each location should be close to the ideal value to be uniform distribution. Fig. 12 shows the distribution plot. It can be seen that the toggled bit number varies around the range of 4390 and 6420. The ideal value is 5000, which means the bit has been flipped approximately 50% of the time. The proposed cryptosystem has a mean flipped bit number of 566.2987, which is close to the ideal value. That implies that the proposed cryptosystem has uniform distribution even when a single bit of the plaintext has changed, making it highly resistant to statistical attacks.

5.6. Speed analysis

The efficiency of an encryption algorithm is an important criterion for cryptographic applications to minimize latency. Unfortunately, for nearly all encryption algorithms, there exists a trade-off between efficiency and security margins. To analyze the speed of the proposed algorithm, we select DES for comparison. Generally, floating point numbers that are used in chaotic maps computations are much slower than binary operations. Thus, the proposed cryptosystem was designed to achieve the Diffusion and confusion in only two rounds to reduce the overall runtime of the algorithm. The proposed cryptosystem and DES were implemented on the same platform and Matlab (matlab.mathworks.-com). The DES code source is used from Mahajan and Sachdeva (2013). The runtime of both algorithms when encrypting plaintexts of varying sizes using the same key is recorded. Fig. 13 shows the time vs plaintext size plot for both algorithms. In general, we can see that encryption time increases in proportion to plaintext size, with CBCA having faster encryption speed than DES. CBCA encrypts the plaintext as one block while DES requires the plaintext to be

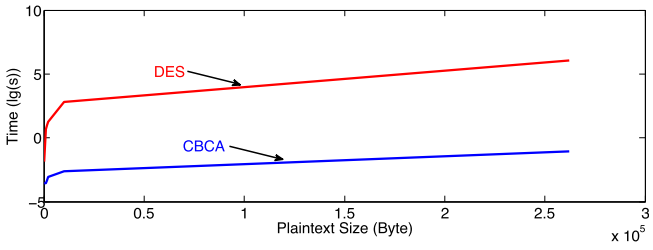

Fig. 13. Encryption speed comparison.

Table 4
Comparison of the proposed cipher against other ciphers.

	NPCR	UACI	SE	CC
CBCA	99.61	**33.32**	**7.9994**	**0.00021**
Ref. Albhrany et al. (2016)	99.55	32.79	7.9964	−0.0193
Ref. Abdullah and Khaleefah (2017)	99.55	24.95	7.0215	0.0939
Ref. Yasser et al. (2020)	99.33	33.42	7.9993	0.3361
RC5	98.83	31.20	7.9683	0.0041
RC6	98.92	31.21	7.9754	0.0038
AES	99.61	33.54	7.9758	0.00052

Table 5
Comparison between RC5, RC6, AES, and CBCA at different aspects.

	RC5	RC6	AES	CBCA
Number of round (r)	1–255	20	10,12,14	2
Key space	0 to 2040 bits	128, 192, or 256 bits	128, 192, or 256 bits	$\geqslant$ 128
Block size in bits	32, 64, or 128 bits	64,128, or 256	128	Unlimited
Max block size in bits	128	256	128	Unlimited
Number of keys derived from key schedule	2r + 2	2r + 4	4(r + 1)	1
Algorithm Structure	Feistel-like network	Feistel-like network	Substitution and permutation network	Simultaneous confusion-diffusion operation
Chaotic map	Does not exist	Does not exist	Does not exist	Enhanced Logistic chaotic map

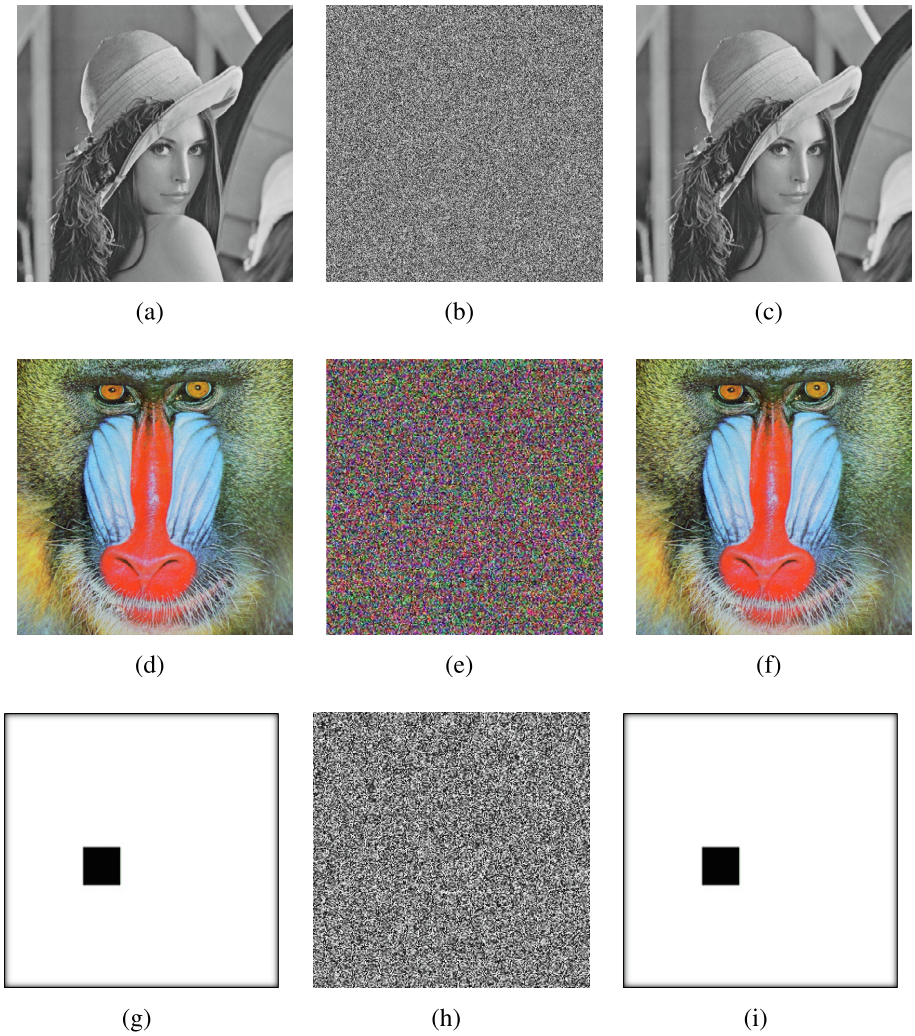

Fig. 14. (a) Lena plainimage (size 256 × 256), (b) Lena Cipherimage, (c) Decrypted Lena, (d) Baboon plainimage (size 512 × 512), (e) Baboon Cipherimage, (f) Decrypted Baboon , (g) Small black box plainimage (size 1024 × 1024), (h) Small black box Cipherimage, (i) Decrypted Small black box.

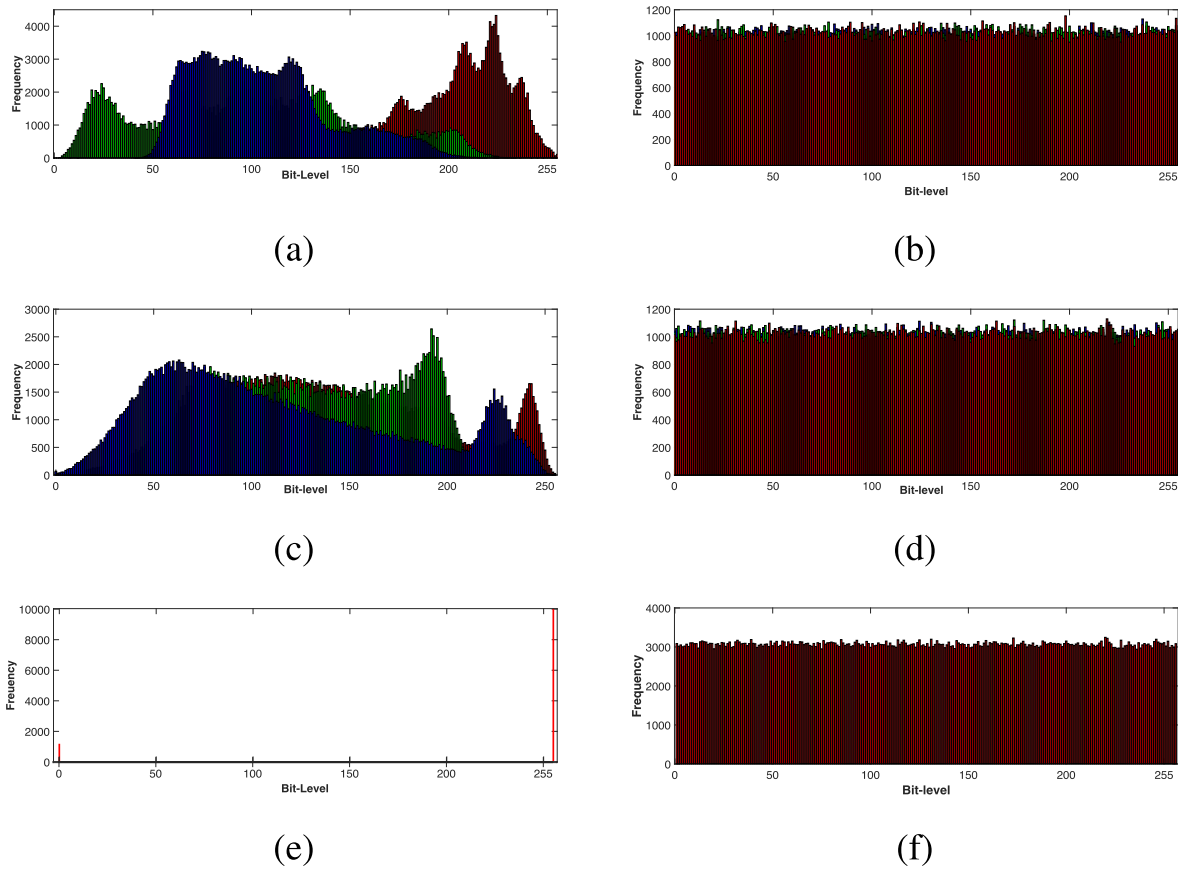

Fig. 15. (a) Histogram of Lena plainimage , (b) Histogram of Lena Cipherimage, (c) Histogram of Baboon plainimage, (d) Histogram of Baboon plainimage, (e) Histogram of Small black box plainimage, (f) Histogram of Small black box plainimage.

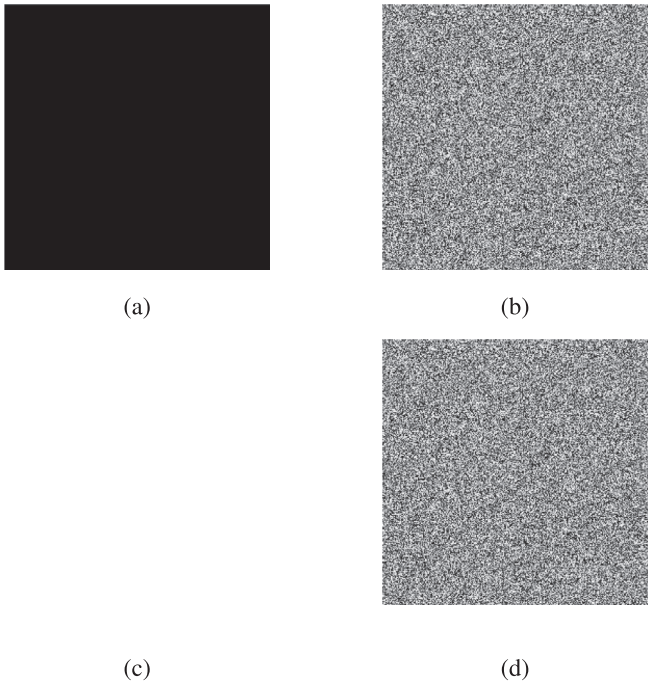

Fig. 16. Encryption results of the black and white images, (a) Black plain-image, (b) Black cipher-image, (c) White plain-image and (d) White cipher-image.

divided into multiple blocks which are processed separately using a mode of operation such as cipher block chaining. Overall, CBCA is faster than DES and can encrypt a large amount of data in less than one second, making it suitable for large multimedia files such as images or videos.

5.7. Comparison

There are not many chaos-based ciphers designed specifically for text-based data since the majority of them were proposed for images or videos. Chaotic ciphers are suitable for encrypting large amounts of data and can generate chaotic points to accommodate varying plaintext lengths. When it comes to textual data, classical encryption algorithms are usually adopted rather than any chaos-based alternatives. Therefore, in this section, we compare the proposed cipher to other chaos-based ciphers as well as classical ones.

We selected SE and resistance to differential attacks as a measure of randomness and diffusion. These two characteristics play a significant role in the secure ciphers. Randomness is a measure unpredictability whereas diffusion ensures that a cipher is highly sensitive to small changes in the plaintext or key. We use the number of pixel change rate (NPCR) and the unified average change intensity (UACI) to measure diffusion. High NPCR and UACI values imply a high degree of sensitivity to small changes in the plaintext. As for SE, high values close to 8 are desired and indicate a high degree of randomness in the ciphertext.

Table 4 provides a comparison between three chaos-based text ciphers and three classical ciphers. One can see that CBCA has superior performance to its chaos-based peers as well as the classical ciphers. NPCR value more than 99.56 indicates that the cipher has high sensitivity. Both CBCA and AES have two values higher than the critical point. UACI has range to indicate if a cipher has

Table 6
Comparison of the proposed cipher against other image ciphers.

Algorithms	GSE	CC	NPCR	UACI	Key space
CBCA (Lena)	7.9977	0.0003	99.61	33.48	2^{128}(and can be extended)
CBCA (Baboon (Red))	7.9991	−0.0015	99.65	33.43	–
CBCA (Small black box)	7.9993	0.0154	99.62	33.64	–
Ref. Liu et al. (2016)	7.9976	−0.003	99.61	33.33	2^{256}
Ref. Niyat et al. (2017)	7.9972	0.0022	99.65	33.44	2^{128}
Ref. Chai et al. (2017)	7.9971	−0.0016	99.62	33.45	2^{128}
Ref. Tao et al. (2020)	7.9021	−0.0017	99.60	33.45	2^{400}
Ref. Belazi et al. (2016)	7.9971	0.0113	99.61	33.66	2^{624}
Ref. Wang et al. (2021)	7.9978	−0.0012	99.60	33.45	2^{688}
Ref. Hosny et al. (2021)	7.9972	−0.0019	99.61	33.43	2^{320}
Ref. Xian et al. (2022)	7.9999	0.0004	99.61	33.46	2^{512}
Ref. Sayed et al. (2021)	7.9991	−0.0634	99.34	33.51	2^{396}
Ref. Broumandnia (2019)	7.9993	0.02694	99.64	33.47	6.13×10^{501}

good diffusion, which is between [33.28 – 33.56]. CBCA, the cipher from Ref. Yasser et al. (2020) and AES have UACI scores within that range. It indicates that CBCA provides similar diffusion to AES. In addition, CBCA produces ciphertexts with higher randomness than other ciphers. Moreover, the correlation between CBCA's ciphertexts is lowest one.

Table 5 provides another comparison between classical ciphers and CBCA using different design parameters. As we see that CBCA only requires two rounds to produce ciphertext with high diffusion and confusion proprieties. The keyspace is larger than 128 bits and users have the flexibility to select varying key sizes. The main purpose of the key is to generate chaotic variables to generate chaotic sequence. Thus, the keyspace can be extended to any size, It is not limited to specific lengths like in classical ciphers. CBCA also accommodates flexible block sizes by including the entire plaintext in one block. Therefore, there is no need to divide the plaintext into set of blocks. As previously mentioned, dividing to non-overlapping blocks incurs some overhead, and the use of a block cipher mode of operation is required to link the plaintext blocks together to achieve good diffusion.

CBCA converts the key into chaotic variables using the GCV process and the chaotic variable are then used to generate chaotic sequences. Thus, the a separate key expansion (key schedule) algorithm is not necessary in CBCA. The GCV process creates the chaotic variables, which have high sensitive to small changes. Therefore, the confusion is achieved through the GCV process and permutation operation. Classical ciphers use key expansion to derive different version of keys from the master key to use each one in each round to achieve the confusion property. CBCA has simultaneous confusion and diffusion operations which leads to faster encryption time and generates random ciphertexts. Finally, the enhanced logistic chaotic map is used in CBCA which also contributes to improved security.

6. Application in image encryption

In this section, we encrypt images using CBCA to showcase its feasibility for multimedia security. Various images were selected with different sizes, with each one converted to a single dimension prior to encryption and decryption. Both color and gray images were used in our experiments. Fig. 14 shows the plainimages and corresponding cipherimages. CBCA can efficiently encrypt images to produce noise-like outputs that contains no visual clue about the original plainimage. To further evaluate CBCA as an image cipher, we conducted multiple statistical tests that include histogram, correlation coefficient (CC), and global Shannon entropy (GSE). When CC values are near-zero, a ciphertext has no correlation between adjacent pixels. GSE to measure the randomness and distribution bits in the pixels. As each pixel is represented by 8 bits, the ideal GSE value is 8.

Fig. 15 shows the histogram of plainimages and their corresponding cipherimages. We can see that the histogram of the cipherimage is uniformly distributed (the peaks are almost equal). Due to the uniform distribution, it is difficult to extract information from the cipherimages. We generate histograms for three images (Lena, Baboon, and small black box) encrypted using CBCA, all of which have a uniform distribution.

In a good cipher, even when a specific plainimage is used, no discernible patterns are produced, even in extreme cases when the plainimage is entirely black or white. An encryption algorithm is key-dependent as well as pixel-dependent. CBCA used two operations at the same time and the effect of both keys and pixels is applied in both directions (start to end, end to start) during the two rounds of encryption. Pixel locations and pixel values are changed at the same time and depend on the secret key. A small change to the secret key will lead to an entirely new cipherimage. We choose black and white images to test CBCA against known plaintext attacks (KPA) and chosen plaintext attacks (CPA). Fig. 16 depicts the noise-like outputs after the black and white images were encrypted. That confirms that CBCA is highly resistant to KPA and CPA.

Users can also choose to generate secret keys related to the plainimages for additional security, since having a plainimage-related key would be closer to a one-time pad construction (one key for one message). This can be easily performed by hashing the plainimage, and using the message digest as (or to derive) the secret key. However, similar to a one-time pad, using a message-dependent key is less practical since it would incur a significant computational overhead, which leads to increased latency in communication. This stems from the fact that the key would always need to be recomputed and redistributed each time a new message needs to be sent.

Table 6 illustrates the statistical results for the Lena, Baboon and black box images (images commonly used to evaluate image ciphers). Results show that the proposed algorithm has good statistical results, which implies a high level security. To study an image cipher's resistance to differential attacks and its plainimage sensitivity, NPCR and UACI are widely used metrics. We evaluate CBCA using both metrics and tabulated the results in Table 6. The NPCR and UACI results show that the proposed algorithm is highly sensitive to small changes to the plainimage, which leads to a completely different cipherimage. Furthermore, Table 6 shows the comparison between the proposed cipher and other chaos-based image encryption algorithms, where our algorithm has a good trade-off in both security and efficiency as compared to its peers.

Overall, results show that CBCA is suitable for securing all data types that include binary data (e.g. regular text) and multimedia (e.g. images).

7. Conclusion

In this paper, a new chaotic block cipher suitable for various data types was proposed. The traditional logistic chaotic map was enhanced by using a straightforward chaotification approach to be more robust, notably in terms of its ergodicity. The enhanced logistic map generates data sequences with a high degree of randomness and entropy. The cipher's chaotic variables are generated using a flexible-length secret key, which are then used in both diffusion and confusion operations. The resulting chaotic variables are highly sensitive to secret key changes. As these chaotic variables are used in both permutation and substitution, both operations are thus key-dependent, making the cipher highly resistant to cryptanalytic attacks. The proposed algorithm only requires two rounds of encryption, and encrypts an entire plaintext block without dividing it into smaller blocks. The cipher supports different plaintext sizes with minimal effect on efficiency and security. We demonstrate the effectiveness of the cipher for two data types (text and images), whereby statistical and performance results show that the cipher is both highly secure and efficient when compared to other chaos-based encryption algorithms. For future work, we will look into the combination between classical ciphers such as AES and DES and improved chaotic maps to achieve improved security and low computational complexity. Also, encryption of data from sensors and drones is also a promising direction.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work has been fully supported by Abu Dhabi University under Grant No 19300635.

References

Abdullah, M.Z., Khaleefah, Z.J., 2017. Design and implement of a hybrid cryptography textual system. In: 2017 International Conference on Engineering and Technology (ICET), IEEE, pp. 1–6.
Alawida, M., Samsudin, A., Teh, J.S., 2019. Enhancing unimodal digital chaotic maps through hybridisation. Nonlinear Dyn. 96 (1), 601–613.
Alawida, M., Samsudin, A., Teh, J.S., Alkhawaldeh, R.S., 2019. A new hybrid digital chaotic system with applications in image encryption. Signal Process. 160, 45–58.
Alawida, M., Teh, J.S., Samsudin, A., Alshoura, W.H., 2019. An image encryption scheme based on hybridizing digital chaos and finite state machine. Signal Process. 164, 249–266.
Alawida, M., Samsudin, A., Teh, J.S., 2020. Enhanced digital chaotic maps based on bit reversal with applications in random bit generators. Inf. Sci. 512, 1155–1169.
Alawida, M., Teh, J.S., Oyinloye, D.P., Alshoura, W.H., Ahmad, M., Alkhawaldeh, R.S., 2020. A new hash function based on chaotic maps and deterministic finite state automata. IEEE Access 8, 113163–113174. https://doi.org/10.1109/ACCESS.2020.3002763.
Alawida, M., Samsudin, A., Alajarmeh, N., Teh, J.S., Ahmad, M., Alshoura, W.H., 2021. A novel hash function based on a chaotic sponge and dna sequence. IEEE Access 9, 17882–17897. https://doi.org/10.1109/ACCESS.2021.3049881.
Albhrany, E.A., Jalil, L.F., Saleh, H.H., 2016. New text encryption algorithm based on block cipher and chaotic maps. Int. J. Sci. Res. Sci. Eng. Technol. (IJSRSET) 2, 67–73.
Alshammari, B.M., Guesmi, R., Guesmi, T., Alsaif, H., Alzamil, A., 2021. Implementing a symmetric lightweight cryptosystem in highly constrained iot devices by using a chaotic s-box. Symmetry 13 (1), 129.
Alshammari, B.M., Guesmi, R., Guesmi, T., Alsaif, H., Alzamil, A., 2021. Implementing a symmetric lightweight cryptosystem in highly constrained iot devices by using a chaotic s-box. Symmetry 13 (1), 129.
Alshoura, W.H., Zainol, Z., Teh, J.S., Alawida, M., 2022. An fpp-resistant svd-based image watermarking scheme based on chaotic control. Alexandria Eng. J. 61 (7), 5713–5734. https://doi.org/10.1016/j.aej.2021.10.052.
Arifin, S., Muktyas, I.B., Prasetyo, P.W., Abdillah, A.A., 2021. Unimodular matrix and bernoulli map on text encryption algorithm using python, Al-Jabar: Jurnal Pendidikan. Matematika 12 (2), 447–455.
Belazi, A., Abd El-Latif, A.A., Belghith, S., 2016. A novel image encryption scheme based on substitution-permutation network and chaos. Signal Process. 128, 155–170. https://doi.org/10.1016/j.sigpro.2016.03.021.
Biham, E., Shamir, A., 1991. Differential cryptanalysis of des-like cryptosystems. J. Cryptol. 4 (1), 3–72.
Biryukov, A., 2011. Chosen ciphertext attack. In: van Tilborg, H.C.A., Jajodia, S. (Eds.), Encyclopedia of Cryptography and Security. second ed. Springer, p. 205. https://doi.org/10.1007/978-1-4419-5906-5_556.
Broumandnia, A., 2019. Designing digital image encryption using 2d and 3d reversible modular chaotic maps. J. Inf. Secur. Appl. 47, 188–198. https://doi.org/10.1016/j.jisa.2019.05.004.
Canetti, R., Krawczyk, H., Nielsen, J.B., 2003. Relaxing chosen-ciphertext security. In: Boneh, D. (Ed.), Advances in Cryptology – CRYPTO 2003. Springer, Berlin Heidelberg, Berlin, Heidelberg, pp. 565–582.
Chai, X., Gan, Z., Yang, K., Chen, Y., Liu, X., 2017. An image encryption algorithm based on the memristive hyperchaotic system, cellular automata and dna) sequence operations. Signal Process.: Image Commun. 52, 6 – 19. doi:10.1016/j.image.2016.12.007.
Courtois, N.T., 2004. General principles of algebraic attacks and new design criteria for cipher components. International Conference on Advanced Encryption Standard, Springer, 67–83.
Courtois, N.T., Bard, G.V., 2007. Algebraic cryptanalysis of the data encryption standard. IMA International Conference on Cryptography and Coding, Springer, 152–169.
Hosny, K.M., Kamal, S.T., Darwish, M.M., Papakostas, G.A., 2021. New image encryption algorithm using hyperchaotic system and fibonacci q-matrix. Electronics 10 (9), 1066.
Hua, Z., Zhou, Y., 2016. Dynamic parameter-control chaotic system. IEEE Trans. Cybern. 46 (12), 3330–3341. https://doi.org/10.1109/tcyb.2015.2504180.
Hua, Z., Zhou, Y., Huang, H., 2019. Cosine-transform-based chaotic system for image encryption. Inf. Sci. 480, 403–419. https://doi.org/10.1016/j.ins.2018.12.048.
Kordov, K., 2021. Text encryption algorithm for secure communication. Int. J. Appl. Math. 34 (4), 705.
Lan, R., He, J., Wang, S., Gu, T., Luo, X., 2018. Integrated chaotic systems for image encryption. Signal Process. 147, 133–145. https://doi.org/10.1016/j.sigpro.2018.01.026.
Leander, G., Moos, T., Moradi, A., Rasoolzadeh, S., 2021. The SPEEDY family of block ciphers engineering an ultra low-latency cipher from gate level for secure processor architectures. IACR Trans. Cryptogr. Hardw. Embed. Syst. 2021 (4), 510–545. https://doi.org/10.46586/tches.v2021.i4.510-545.
Lian, S., Sun, J., Wang, Z., 2005. A block cipher based on a suitable use of the chaotic standard map. Chaos Solitons Fractals 26 (1), 117–129.
Lipmaa, H., Moriai, S., 2001. Efficient algorithms for computing differential properties of addition. In: International Workshop on Fast Software Encryption. Springer. pp. 336–350.
Liu, L., Miao, S., 2017. Delay-introducing method to improve the dynamical degradation of a digital chaotic map. Inf. Sci. 396, 1–13. https://doi.org/10.1016/j.ins.2017.02.031.
Liu, W., Sun, K., Zhu, C., 2016. A fast image encryption algorithm based on chaotic map. Opt. Lasers Eng. 84, 26–36. https://doi.org/10.1016/j.optlaseng.2016.03.019.
Mahajan, P., Sachdeva, A., 2013. A study of encryption algorithms aes, des and rsa for security. Global J. Comput. Sci. Technol.
Matsui, M., 1994. On correlation between the order of s-boxes and the strength of des. In: Workshop on the Theory and Application of Cryptographic Techniques. Springer. pp. 366–375.
Murillo-Escobar, M., Abundiz-Pérez, F., Cruz-Hernández, C., López-Gutiérrez, R., 2014. A novel symmetric text encryption algorithm based on logistic map, in: Proceedings of the international conference on communications, signal processing and computers, vol. 4953.
Niyat, A.Y., Moattar, M.H., Torshiz, M.N., 2017. Color image encryption based on hybrid hyper-chaotic system and cellular automata. Opt. Lasers Eng. 90, 225–237. https://doi.org/10.1016/j.optlaseng.2016.10.019.
Pisarchik, A.N., Zanin, M., 2012. Chaotic map cryptography and security. Int. J. Comput. Res. 19 (1), 49.
Rogaway, P., 2011. Evaluation of some blockcipher modes of operation – uc davis. URL: https://www.cs.ucdavis.edu/rogaway/papers/modes.pdf.
Sayed, W.S., Radwan, A.G., Fahmy, H.A., Elsedeek, A., 2021. Trajectory control and image encryption using affine transformation of lorenz system. Egypt. Inf. J. 22 (2), 155–166. https://doi.org/10.1016/j.eij.2020.07.002.
Shakiba, A., 2021. A randomized cpa-secure asymmetric-key chaotic color image encryption scheme based on the chebyshev mappings and one-time pad. J. King Saud Univ.-Comput. Inf. Sci. 33 (5), 562–571.
Tao, Y., Cui, W., Zhang, Z., 2020. Spatiotemporal chaos in multiple dynamically coupled map lattices and its application in a novel image encryption algorithm. J. Inf. Secur. Appl. 55,. https://doi.org/10.1016/j.jisa.2020.102650 102650.
Teh, J.S., Alawida, M., Sii, Y.C., 2020. Implementation and practical problems of chaos-based cryptography revisited. J. Inf. Secur. Appl. 50, 102421.
Teh, J.S., Alawida, M., Ho, J.J., 2020. Unkeyed hash function based on chaotic sponge construction and fixed-point arithmetic. Nonlinear Dyn., 1–17

Valli, D., Ganesan, K., 2017. Chaos based video encryption using maps and ikeda time delay system. Eur. Phys. J. Plus 132 (12), 1–18.
Volos, C.K., Kyprianidis, I., Stouboulos, I. Text encryption scheme realized with a chaotic pseudo-random bit generator. J. Eng. Sci. Technol. Rev. 6(4).
Wang, M., Wang, X., Zhao, T., Zhang, C., Xia, Z., Yao, N., 2021. Spatiotemporal chaos in improved cross coupled map lattice and its application in a bit-level image encryption scheme. Inf. Sci. 544, 1–24. https://doi.org/10.1016/j.ins.2020.07.051.
Wen, W., Tu, R., Wei, K., 2019. Video frames encryption based on dna sequences and chaos. In: Eleventh International Conference on Digital Image Processing (ICDIP 2019), vol. 11179, International Society for Optics and Photonics. p. 111792T.
Wu, Y., Zhou, Y., Saveriades, G., Agaian, S., Noonan, J.P., Natarajan, P., 2013. Local shannon entropy measure with statistical tests for image randomness. Inf. Sci. 222, 323–342.
Xian, Y., Wang, X., Teng, L., Yan, X., Li, Q., Wang, X., 2022. Cryptographic system based on double parameters fractal sorting vector and new spatiotemporal chaotic system. Inf. Sci. 596, 304–320. https://doi.org/10.1016/j.ins.2022.03.025.
Xie, H.-B., Chen, W.-T., He, W.-X., Liu, H., 2011. Complexity analysis of the biomedical signal using fuzzy entropy measurement. Appl. Soft Comput. 11 (2), 2871–2879.
Yasser, I., Mohamed, M.A., Samra, A.S., Khalifa, F., 2020. A chaotic-based encryption/decryption framework for secure multimedia communications. Entropy 22 (11), 1253.
Zhou, Y., Hua, Z., Pun, C.-M., Chen, C.L.P., 2015. Cascade chaotic system with applications. IEEE Trans. Cybern. 45 (9), 2001–2012. https://doi.org/10.1109/tcyb.2014.2363168.
Zhu, S., Zhu, C., 2021. Security analysis and improvement of an image encryption cryptosystem based on bit plane extraction and multi chaos. Entropy 23 (5), 505.
Zhu, H., Qi, W., Ge, J., Liu, Y., 2018. Analyzing devaney chaos of a sine-cosine compound function system. Int. J. Bifurcation Chaos 28 (14), 1850176.