Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Securing information using a proposed reliable chaos-based stream cipher - with real-time FPGA-based wireless connection implementation

Lahcene Merah (l.merah@lagh-univ.dz)
Universite Amar Telidji Laghouat Faculte de Technologie https://orcid.org/0000-0001-6181-3273
Pascal Lorenz
Université de Haute-Alsace: Universite de Haute-Alsace
Chaib Noureddine
Universite Amar Telidji Laghouat Faculte des Sciences
Ali-Pacha Adda
USTO MB: Universite des Sciences et de la Technologie d'Oran Mohamed Boudiaf

Research Article

Keywords: Chaos, Cryptography, Telecommunication, FPGA, Real-time, Channel noise, dynamical degradation, Synchronization.

Posted Date: April 19th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1534133/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Noname manuscript No.
(will be inserted by the editor)

Securing information using a proposed reliable chaos-based stream cipher

with real-time FPGA-based wireless connection implementation

Merah Lahcene · Pascal Lorenz · Chaib Noureddine · Ali-Pacha Adda

Received: date / Accepted: date

Abstract In this paper, a robust Chaos Based Stream Cipher (CBSC) is proposed. The novelty of this work is that it addresses all challenges confronting chaos-based cryptography. The PCBSC (Proposed CBSC) has a robust synchronization circuit that mitigates the effect of channel noise, a perturbation block that overcomes the dynamical degradation, a robust encryption scheme, and an efficient control parameters' generator that generates strong keys. According to the complexity evaluation, the improved chaotic map provides good statistical properties. This can be confirmed by the obtained high values of the statistical metrics (Largest Lyapunov Exponent, Approximate Entropy, Permutation Entropy, and Sample Entropy) used for the evaluation. According to the security analysis; the PCBSC has good security features, the PCBSC provides strong keys that ensure confusion property, as well as enough space to withstand brute-force attacks. On the other hand, the proposed encryption scheme proves its efficiency; the result of the differential attack clearly shows that the diffusion property is guaranteed. Additionally, the original images' statistical properties are completely dispersed on the encrypted images. The obtained performance over noisy channels proves the synchronization circuit's efficiency. When compared to other proposals, the PCBSC provides the best results. In addition, the PCBSC is implemented on an FPGA and evaluated in real-time over a wireless link.

Keywords Chaos, Cryptography, Telecommunication, FPGA, Real-time, Channel noise, dynamical degradation, Synchronization.

Lahcene Merah (✉)
Department of Electronics, Faculty of Technology, University of Laghouat, Laghouat 03000, Algeria.
E-mail: l.merah@lagh-univ.dz

Pascal Lorenz
University of Haute Alsace IUT, 68008 Colmar, France.
E-mail: pascalorenz@gmail.com

Chaib Noureddine
Computer Science and mathematics Laboratory, University of Laghouat, Laghouat 03000, Algeria.
E-mail: n.chaib@lagh-univ.dz

Ali-Pacha Adda
Coding and Information Security Laboratory, university of Sciences and Technology of Oran, Oran 31000, Algeria.
E-mail: a.alipacha@gmail.com

1 Introduction

oday, the internet reaches all corners of the globe, and anyone can connect to it. It is a place where anyone (whether businesses or individuals) may provide and request products, services, assistance, news, and so on. As a result, there is an easy access to digital content, as well as higher demand for data sharing. On the other hand, wireless communication is a vital part of the telecommunications system's backbone, and its security has become a main priority. Concurrently, private content became vulnerable to illegal access, and so, protecting private content from unauthorized users became one of the most major challenges confronting owners. Over the years, numerous traditional cryptographic algorithms have shown their efficiency in protecting private digital content. However, as technology evolves, so does the demand for data exchange (military databases, banking transactions, medical imaging systems, paid TV streaming, and so on, with image and video accounting for the majority of it). This has necessitated the development of more sophisticated and reliable cryptographic algorithms to meet the growing needs of data security.

Recently, there has been a surge of interest in chaos-based cryptography. This is due to the fact that it is capable of meeting the growing demand for information security. Chaotic systems exhibit random behavior, are extremely sensitive to initial conditions, and exhibit a continuous broad-band power spectrum. The evolution of two chaotic signals generated from very similar initial conditions differs dramatically. In addition, the random-like behavior of such systems is given by simple and deterministic dynamical systems. Since the 1990s, numerous researchers have observed an intriguing link between chaos and cryptography: numerous features of chaotic systems have analogs in conventional cryptosystems. [1,2].

Due to chaotic systems' extreme sensitivity to their initial conditions, having two chaotic systems evolve in synchrony may appear odd. However, with the finding of Pecora and Carroll in 1990, the prospect of self-synchronization of chaotic oscillations became possible [3], This discovery was a watershed moment in the use of chaos in cryptography. Since that time, considerable effort has been made in this field of research.

Using chaotic systems in cryptography has a variety of implications. According to [1], two ways to construct chaos-based cryptosystems exist: analog and digital. Analog chaos-based crypto-systems are often built on synchronized setups; depending on the configuration, the message to be transmitted is either included into the analog chaotic signal or used to control multiple chaotic outputs. Some known examples of analog chaos-based crypto-systems including (but not limited to): Chaotic masking [4–8], chaotic switching [9,10], and chaotic modulation [11].

In fact, analog chaos-based cryptography has not received much attention compared to the digital approach. This is because, on the one hand, of the domination of digital computers and what they can offer as advantages, and on the other hand, of the vulnerability of analog crypto-systems. [12–16].

Digital chaos-based cryptosystems (a.k.a. digital chaotic ciphers) are developed for digital computers and encrypt plaintext in a variety of ways using one or more chaotic maps implemented with finite computational precision [1]. The majority of research efforts have been directed toward developing fast and trustworthy cryptosystems based on digital chaos. Due to the widespread availability of computers and digital devices, and the ease with which binary data can be handled (digital filtering, detection and correction of data, etc.), digital chaos-based cryptosystems are more efficient than analog ones. Studies on digital chaos generators have been reported to have more diversity compared to their analog counterparts [17]. Using chaotic systems as a source of randomness for digital cryptosystems is the most intuitive application of chaotic systems.

The following is a rough classification of recently proposed digital chaos-based cryptosystems:

- *Stream ciphers* :For most chaos-based stream ciphers, the chaotic maps act as PRNGs (Pseudo Random Number Generators), in which they are used for generating keystreams to mask bit by bit a plaintext (mostly using the XOR function) [17–25]. The security of such configurations is entirely dependent on the keystream's statistical properties and the key used to generate it.
- *Block ciphers* : Instead of encrypting one bit at a time, chaos-based block ciphers encrypt a collection of bits of plaintext called a "block". Since the only nonlinear component of these ciphers is the substitution box (S-box), chaotic systems are used to design it and determine its strength [26–30]. Some existing block ciphers, such as the AES (Advanced Encryption Standard), improve their security by designing their S-boxes with chaotic maps [31].
- *Other chaos-based ciphers* : There have been some proposals for a stream-and-block hybrid encryption based on chaos [32]. Using chaotic maps to create hash functions has taken the lion's share of the chaos application in cryptography [33–36]. Permutation of image pixels is a widely used encryption technique; a large number of works have used chaotic maps for image pixel permutation [37–39].

It's worth noting that the preceding lines barely present of what's out there in terms of proposed chaos-based cryptosystems; due to their vast number and diversity, classifying them is difficult. On the other hand, and in accordance with [1], it is difficult to assess the security and effectiveness of many digital chaos-based cryptosystems in a systematic manner, leaving them widely exposed to attacks. However, a variety of security evaluation measures may be utilized to analyze the security of any proposed chaos-based crypto-systems and, to some extent, establish its reliability.

Despite the huge number of chaos-based cryptosystems proposed recently, many of them have failed to meet established design and security requirements (see Section.2). Some proposals focus on certain problems while turning a blind eye to others, and we rarely come across a fully functional chaos-based crypto-system. As a result, each proposed crypto-system's reliability and robustness are dependent on taking into consideration all known design and security requirements.

The main goal of this work is to develop a robust chaos-based stream cipher that takes into account as much as possible the design and security requirements.

The Proposed Chaos-Based Stream Cipher (PCBSC) is made up of four main parts, one of which is a perturbation block designed to reduce the impact of the digitization process on the chaotic behavior of the digitized chaotic map. A synchronization circuit that increases the performance of the PCBSC in noisy channels. For every given secret key, a control parameter generator that keeps the chaotic map in a chaotic ragime. The PCBSC Also includes an encryption block, which adds another degree of security to the PCBSC and ensures the diffusion property as well as protection to various cryptographic attacks.

This paper is organized as follow : Section.2 discusses the most known design and security requirements for any proposed chaos-based cryptosystem. Section.3 introduces the Proposed Chaos-Based Stream Cipher (PCBSC) and provides sufficient details on its operational basis. In Section.4, the evaluation of the improved chaotic map in terms of complexity is presented, in which some statistical and mathematical tools are used for this purpose. Section.5 discusses the PCBSC's security analysis, in which the key strength, sensitivity, and space are all carefully analyzed. Additionally, the proposed encryption block is evaluated by examining the encrypted image's statistical properties and performing the differential attack on it to assess its strength. The performance of the PCBSC under noisy channel is reported in Section.6, where the proposed synchronization circuit is evaluated to confirm its efficiency. Section.7 discusses the implementation steps of the PCBSC on an FPGA and its real-time evaluation through wireless transmission. The obtained results comparison with other proposals is the subject of the Section.8. The paper is terminated by a conclusion about the achieved results in Section.9.

2 Chaos-based cryptography requirements

Designers of any chaos-based cryptosystem should emphasize many critical design challenges. Indeed, despite its effectiveness, applying chaos to cryptography is a tough process. A number of important issues must be addressed while constructing a chaos-based cryptosystem. Chaos-based encryption has encountered a number of issues that limit detract from its utility in cryptography. The following subsections discuss the important issues related to chaos-based cryptography, which should be considered by every designer.

2.1 Synchronization

In chaotic systems, synchronization means that the trajectories of the receiver (response or slave) system can track those of the emitter (master) system starting from arbitrary initial conditions. For chaos-based stream ciphers in particular, the first issue that should be taken into account is synchronization. That is, the emitter and the receiver should provide the same dynamics to easily recover the original plaintext.

Even though the possibility of synchronization in chaotic systems has been discovered [3], the high sensitivity to initial conditions that characterizes such systems precludes synchronization because even two identical systems cannot evolve in the same manner, even if their initial conditions are very close [21]. Additionally, the conventional synchronization method (continuously driving the response system) is very susceptible to channel noise. Thus, any proposed chaos-based stream encryption should consider synchronization over noisy channels.

2.2 Digitization effect on the dynamical properties of chaotic systems

Encryption algorithms that are based on the generation of random numbers should include strong PRNGs that provide good randomness quality, with good distribution and high unpredictability. Indeed, when chaotic systems are implemented on digital computers with finite computing precision, their dynamical properties deteriorate. A digitized chaotic system is qualitatively different from its analog counterpart and produces a predictable output (periodic) with poor statistical properties. This significant degradation diminishes the effectiveness of chaotic systems in cryptography considerably.

Many studies suggest that the cycle-length of a given digitized chaotic system's output gets longer as the arithmetic precision size gets larger. The digitized chaotic systems are assumed as periodic, and their longest orbit can never be more than 2^L (L represents the arithmetic precision size), more information about this can be found in [40].

Numerous remedies have been suggested to mitigate the dynamical degradation found in digitized chaotic systems; they can be broadly classified into three categories: (1) Using high arithmetic precision [41], which can significantly lengthen the cycle and improve statistical properties, (2) Cascading multiples chaotic systems [42], and combining their outputs to obtain the final chaotic output, and (3) using an external random or pseudo-random source to perturb the chaotic system's

orbit [43]. However, the first two remedies, despite the enhancements they can give us, are costly in terms of performance (implementation cost). The last remedy is more efficient and more supported by chaoticians.

Thus, any chaos-based cryptosystem designer should take into account the dynamical degradation found in the digitized chaotic systems and what solution might propose to overcome this problem.

2.3 Design Complexity and performance

Many applications do not suit cryptosystems that based on design complexity. Today, resource-constrained networks make up a significant portion of communication networks (military surveillance, medical sensing networks, wearable devices, etc.). These networks are characterized by their limited computation capability, limited storage space, and strict power utilization management [44]. Securing data over such networks is crucial. Consequently, complicated cryptosystems are not deemed ideal for securing data over resource-constrained networks.

Design complexity, on the other hand, entails increased hardware resource utilization, processing time (due to significant signal propagation delays), and heat dissipation. These difficulties have a direct impact on the design's performance, implementation costs, and energy usage.

When compared to their low-complexity equivalents, cryptosystems with complicated structures are more secure, reliable, and harder to break. As a result, any proposed chaos-based cryptosystem should strike a balance between security and complexity.

2.4 Security

Any encryption algorithm's reliability is determined by its resistance to different known attacks. A strong encryption algorithm should withstand any serious attack, regardless of whether the intruder has whole or partial knowledge of the encryption algorithm's structure.

Immunity from cryptographic attacks is intrinsically linked to the secret key. The key is by far the most major element of the encryption algorithms, security should be solely dependent on the key. No matter how robust and well-designed the encryption algorithm is, if the key is chosen incorrectly or the keyspace is too small, the cryptosystem will be easily broken [1].

There are three main important issues related to the key, namely, the key construction, the key length, and the key strength. For chaos-based cryptography, it is obvious that the key is made using the control parameters. Depending on the control parameters' intervals, a given chaotic system exhibits many behaviors between fixed points: periodic, quasi-periodic, and chaotic. What we are interested in is constructing the key from the intervals where the control parameters assure the chaotic behavior. The bifurcation diagrams can aid us in defining the intervals in which the control parameters result in chaotic behavior.

Another important issue concerning the key is the equality of strength. That is, each infinitesimal change in the key should result in the same ciphertext distribution. Flipping one bit in any position in the key should result in a significant change in the ciphertext. In general, the *Avalanche Effect*, as it is known in cryptography, can be used to evaluate the key's strength. The *Avalanche effect* is well known among cryptologists. It is one of the most important properties that characterizes the efficiency of encryption algorithms. Good encryption algorithms have high *avalanche effects*.

The *Avalanche Effect* can be measured by comparing the bit error rate (BER) of the plaintext with that of the recovered one as a function of the attempted keys. The BER should be around or more than 50%, regardless of how close or far we are to the correct key. The BER for weak encryption algorithms decreases as we come closer to the correct keys (low *Avalanche Effect*).

Confusion and diffusion are two more crucial properties that any encryption algorithm should provide, as outlined by Claude Shannon in his report [45]. The main objective of these properties is to make the relationship between the encryption algorithm's input and output as complicated as possible, so that any minor change in the input should be dispersed in the statistical properties of the output. Confusion indicates that a little change in the key (flipping one bit) should result in large changes in the ciphertext; at least 50% of the bits in the ciphertext should be affected. The same thing for the diffusion property; a minor change in the plaintext (flipping one bit) should lead to large changes in the ciphertext (a different ciphertext). Thus, for any proposed chaos-based cryptosystem, these two main properties should be ensured.

3 The Proposed Chaos-Based Stream Cipher (PCBSC)

The PCBSC's basic scheme (for both emitter and receiver) is depicted in Fig.1. On the emitter side, the system is made up of five main blocks: the first is the *Chaotic generator*; the second is the *Control parameter generator*; the third is the *Perturbation block*; the fourth is the *Encryption block*; and the fifth is the *Bit-Basher block*. On the emitter side, the system is identi-

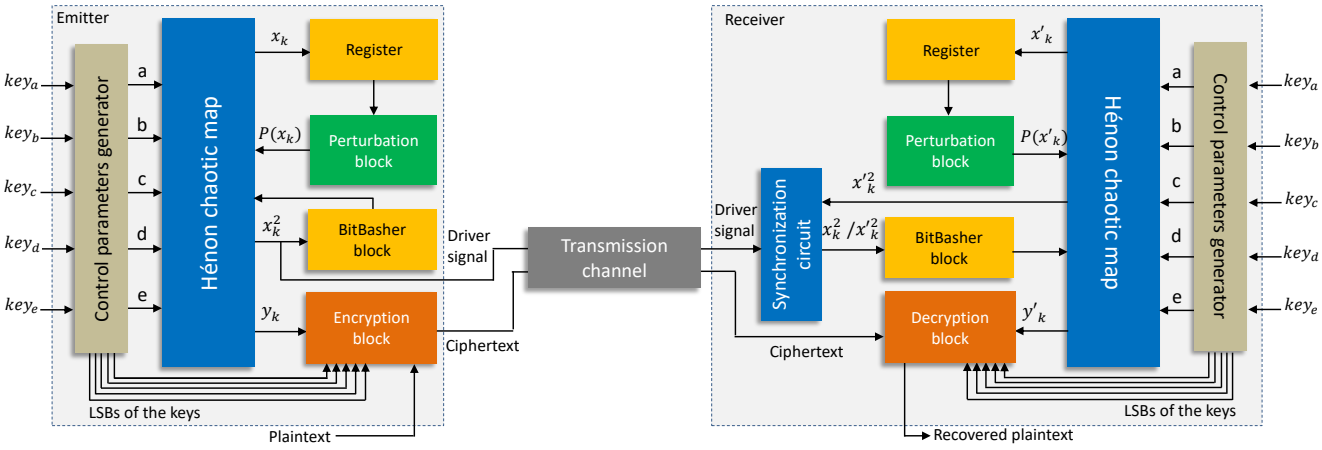

Fig. 1: The Proposed Chaos-Based Stream Cipher basic scheme.

cal to the emitter, with the exception of the *Decryption* and *Synchronization* blocks.

The PCBSC is a stream cipher cryptosystem in which the plaintext is encrypted bit by bit using the chaotic sequence (using the XOR logical operator) and transferred across the transmission channel with the synchronization signal. When the receiver system synchronizes with the emitter, the original plaintext can be easily recovered on the receiver side. Each block's functionality is described in detail in the subsections that follow.

3.1 The chaotic generator

In our case, we have used the Hénon chaotic map because of its simplicity, the Hénon map is a 2D chaotic map that given by the following recursive system:

$$\begin{cases} x_{k+1} = 1 + y_k - a \times x_k^2 \\ y_{k+1} = b \times x_k \end{cases} \quad (1)$$

The Hénon map has been studied in depth for $a = 1.4$ and $b = 0.3$, where numerical evidence of chaotic behavior was found. Fig.7 (a) presents the Hénon attractor for $a = 1.4$ and $b = 0.3$.

3.2 The Control parameters generator

The control parameters generator is a vital part of the PCBSC. In cryptography, the chaotic generator's strength is determined by its chaotic behavior. This chaotic behavior is only possible within a restricted ranges of control parameter values. Bifurcation diagrams can help to define the ranges within which the system is chaotic or not (see the PCBSC security evaluation section). Due to the fact that secret keys are generated using control parameters, this block's internal structure allows to retain the chaotic behavior of the chaotic generator for any given key.

Besides the control parameters a and b in the Hénon chaotic map, we add two more control parameters, c and d, and assume the constant 1 in the system (1) as a new variable, e. This is done to make the key size larger. Thereby, the system (1) can be written like this:

$$\begin{cases} x_{k+1} = [e + y_k - a \times x_k^2] \times c \\ y_{k+1} = [b \times x_k] \times d \end{cases} \quad (2)$$

Later, we'll look at how to set appropriate ranges for the newly added control parameters in order to attain chaotic behavior. To facilitate comprehension, we will defer discussing the block's funcrionality to the PCBSC security evaluation section. This is because we need visual evidence of the resulting bifurcation diagrams in order to identify the proper control parameters' ranges.

3.3 The Perturbation and BitBasher blocks

The perturbation block's basic scheme is presented in Fig.2. It consists of a simple but efficient circuit. The main advantage of this block is that it requires no external perturbation source; it provides a self-perturbation mechanism. As shown in Fig.1, the perturbation is performed on the feedback signal x_k, so the system (2) can be rewritten as follows, where $P(x_k)$ denotes the perturbed signal:

$$\begin{cases} x_{k+1} = [e + y_k - a \times P(x_k)^2] \times c \\ y_{k+1} = [b \times P(x_k)] \times d \end{cases} \quad (3)$$

The perturbation is performed in the following manner: initially, the chaotic signal x_k is routed through two

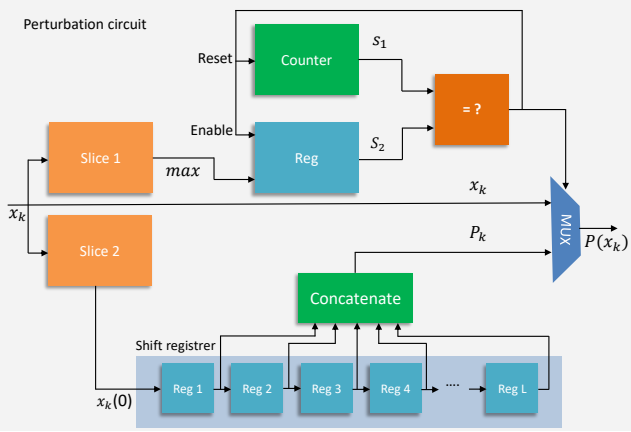

Fig. 2: The basic scheme of the proposed perturbation block.

blocks, *Slice 1* and *Slice 2*. The *Slice 2* block extracts the LSB (Lowest Significant Bit) from the binary vector of the chaotic signal x_k and serially inserts it into L registers (as mentioned previously, L represents the arithmetic precision size and at the same time here the number of the internal registers contained in the shift register). The *Concatenate* block is used to collect the outputs of all internal registers to create the perturbation signal P_k.

The perturbation process is done by feeding the perturbation signal P_k to the chaotic system. The perturbation is not continuous but occurs at random intervals. The process is as follows: the *Slice 1* block extracts the m lowest bits from the chaotic signal x_k, where m is defined by the maximum period that the original chaotic map can provide with the arithmetic precision size L. For example (see Table.1), the longest period that the original Hénon map may give is 280 for $L = 16$. As a result, $m = round(log_2(280)/log_2(2)) = 8$.

The output signal max is an m-bit integer. During the counting process, one value of this signal is stored on the register. The counter is incremented until this value is reached. When the counter hits this maximum, the comparison block generates a value of 1 to reset it, controls the multiplexer to output the perturbation signal instead of the signal x_k, and enables the register to load a new value of the signal max. Thus, as previously stated, the perturbation occurs when the counter reaches its maximum value. Because the maximum value is variable, the period of perturbation is not constant but random. This is a pivotal feature that characterizes the proposed perturbation circuit. In Section.4, we will show that the proposed perturbation circuit is capable of effectively increasing the cycle length of the digitized chaotic map.

The *BitBasher* is another straightforward but efficient block in the PCBSC. This block is included because it serves the same objective as the previous one, which is to enhance the chaotic map's statistical properties. The *BitBasher* block is responsible for reversing the bit order of the chaotic signal input. Following a series of experiments, we concluded that the *BitBasher* block should be placed as illustrated in Fig.1. We shall see how such a simple block can significantly increase the randomness of chaotic systems and expand the range of control parameters to have a chaotic behavior.

3.4 The Encryption/Decryption Blocks

Keep in mind that the PCBSC is a symmetric cipher key, which means that encryption is done bit by bit (one bit at a time) and that the same encryption key is used on the emitter side should be regenerated and used on the receiver side.

As previously stated, two important tasks of any cryptosystem should be assured, namely the confusion and diffusion. Because the control parameters are used as keys in the PCBSC, the confusion property may be guaranteed (as we will see next) due to the chaotic system's high sensitivity to the initial parameters. To assure the diffusion property and to complicate and obfuscate the relationship between the plaintext and the ciphertext, we propose an efficient encryption/decryption mechanism. The ciphertext C_k is obtained as follow:

$$C_k = O[C_{k-1} \oplus P_k \oplus R_k] \tag{4}$$

Where P_k, R_k, and C_{k-1} represent respectively the plaintext, the chaotic sequence, and the previous ciphertext. "O" denotes the process by which the final ciphertext is obtained; it comprises a sequence of blocks (from one to five) through which the ciphertext will be subject to many changes, as shown in Fig.3.

Each of these five blocks shuffles the bits positions of its input in a distinctive manner. There are two possible scenarios for the final output of each block, depending on the signal that controls the correspnding multiplexer: the same input signal and the shuffled input signal. The LSBs of the secret keys key_a, key_b, key_c, key_d, and key_e are the signals that control every multiplexer (keys representing each control parameter).

For the decryption process, the order of the blocks is inverted so that the original plaintext can be easly recovered, as shown in Fig.3. The recovered plaintext rP_k is obtained as follow:

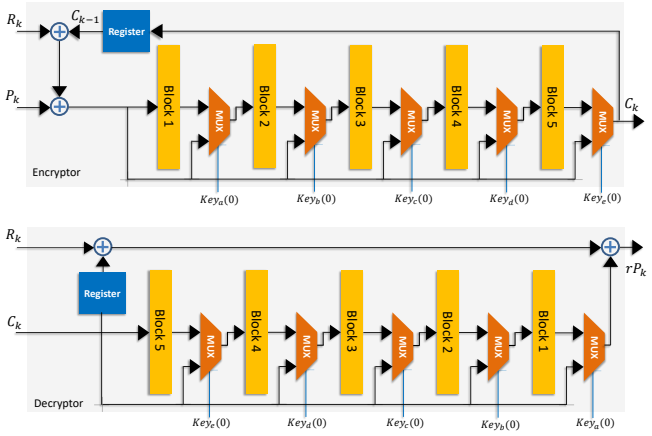

Fig. 3: The proposed Encryption/Decryption mechanism.

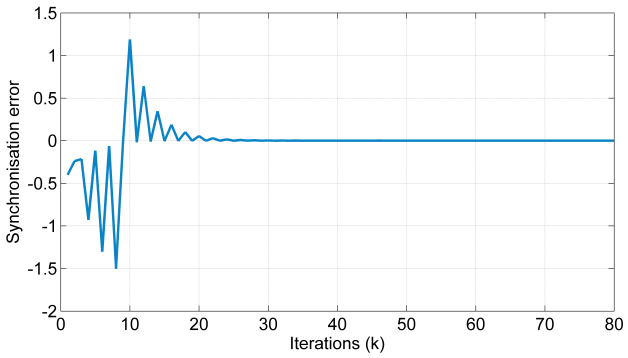

Fig. 4: Evolution of the synchronization error between the emitter (master) system and the receiver (response) one.

$$rP_k = C_{k-1} \oplus R_k \oplus O^{-1}[C_k] \tag{5}$$

Where O^{-1} denotes the process of the inverting order of the blocks.

The proposed encryption scheme assures that a little change in the plaintext or the multiplexers' inputs results in significant changes in the ciphertext. From a cryptography viewpoint, the plaintext's statistical properties will be completely distributed in the ciphertext (More details can be found in Section.5).

3.5 Synchronization block

The synchronization block is one of the most critical components of our PCBCS. The original plaintext can never be recovered without synchronization, even with the correct keys. As the driving signal, we used the signal x_k^2. Experiments revealed that when using this signal, synchronization between the emitter (master) and receiver (response or slave) systems happens promptly after only few iterations. Given that the master system is indicated in (3), the response system is as follows:

$$\begin{cases} x'_{k+1} = [e + y'_k - a \times x_k^2] \times c \\ y'_{k+1} = [b \times P(x'_k)] \times d \end{cases} \tag{6}$$

$(')$ denotes the signals generated by the response system to distinguish between them and those generated by the master system. Starting from different initial conditions ($x_0 = 0.1$, $y_0 = 0.1$, $x'_0 = 0.5$, $y'_0 = 0.9$), the evolution of the synchronization error (Fig.4) demonstrates that after a few iterations, the response system synchronizes with the master, and the receiver follows the emitter's dynamics.

Experiments have shown that in noisy channels, continuous driving of the response system is not the right approach. That is, the existence of noise in the picture adds another challenge due to the chaotic systems' high sensitivity to the initial settings. To protect the PCBSC from channel noise as much as possible, we propose a simple solution that consists of driving the response system just during specified periods rather than continuously.

In fact, the noise power fluctuates continuously between high and low values. The proposed synchronization circuit (Fig.5) tries to track the moments where the noise power is low. Practically, it compares the value of the driver signal x_k^2 with the same signal $x_k'^2$ generated by the response system. When the two signals are equal, it means that the two systems are synchronized and have the same outputs. To be sure that this equality is not accidental, each signal is passed through 3 registers, and each register's output of the first signal (x_k^2) is compared to the register's output of the second signal $x_k'^2$. The outputs of the comparators are loaded into a logical AND gate, so the signal S_1 of the output of the logical AND gate will take the value 1 only if the two signals are equal during three successive periods.

The S_1 signal is fed into a logical OR gate, which then controls a counter and a multiplexer. When $S_1 = 1$ $(E = 1)$, the counter starts counting, and the multiplexer outputs $x_k'^2$ rather than x_k^2. In this case, the response system works independently from the master system, and will be protected from the channel noise. The signal S_2, on the other hand, is activated until the counter reaches its maximum value. The counter can reach a maximum of 2^m, where m is the counter's size in bits. The process will be restarted when the counter reaches its maximum.

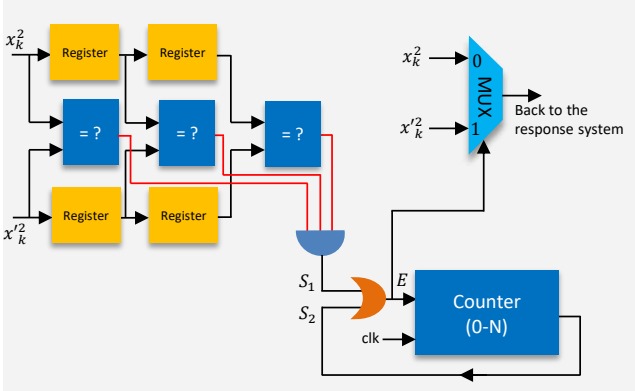

Fig. 5: The proposed synchronization circuit basic scheme.

The evaluation of the performance of the proposed synchronization circuit under a noisy channel is given in Section.6.

4 Evaluation of the improved Hénon map

Given that the PCBSC's strength is related to the complexity of the chaotic generator, this section will focus on the evaluation of the improved Hénon map using the proposed perturbation block. The resulting chaotic sequence is evaluated based on two key factors: its complexity and its statistical properties. Some mathematical and statistical tools are used to perform the evaluation.

4.1 Complexity evaluation

This section deals with the evaluation of the improved Hénon map in terms of its complexity, where some mathematical tools are used for this purpose.

4.1.1 Trajectory and phase space analysis

The improved Hénon map's complexity can be visually assessed by visualizing the output and phase space trajectories. The resultant trajectories and phase spaces for both the original and improved Hénon maps are shown in Fig.6 and Fig.7, respectively.

It is clear that the trajectory of the original Hénon map has a regular-like aspect, whereas the improved map has a random-like aspect. On the other hand, the phase space analysis shows clearly the good distribution of the improved Hénon map output over the whole space, in contrast to the original map where the output is confined to a specific form. These obtained results

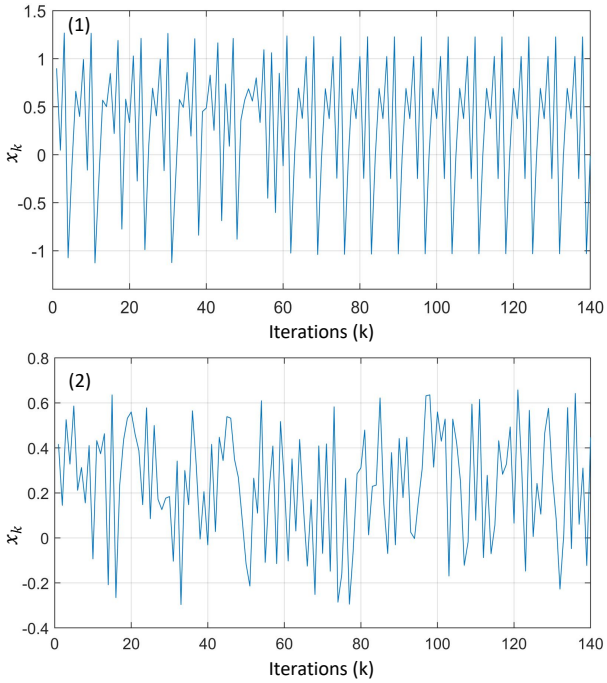

Fig. 6: The resultant trajectories of : (1) the original Hénon map, (2) the improved Hénon map.

confirm that the complexity of the modified Hénon map has been improved.

4.1.2 Bifurcation diagram analysis

A bifurcation diagram is an extremely valuable tool for visualizing the values that are visited or approached asymptotically in phase space when the control parameter evolves. It simplifies the visual distinction between regular and chaotic zones. [46].

By fixing L to 16 , the bifurcation diagrams of both the original and improved Hénon maps are shown in Fig.8. As can be seen, the improved Hénon map provides far better results. The ranges of control parameters within which the system exhibits chaotic behavior are greatly expanded in comparison to the original Hénon map. Additionally, the improved map's output has a well-defined distribution throughout the interval [0,1].

4.1.3 Autocorrelation and period length analysis

In this section, the complexity of the improved Hénon map is evaluated by measuring the autocorrelation and the obtained period length. Autocorrelation is an important mathematical tool that can help to measure the relationship between samples of the same signal for the purpose of detecting similarities. It should be noted

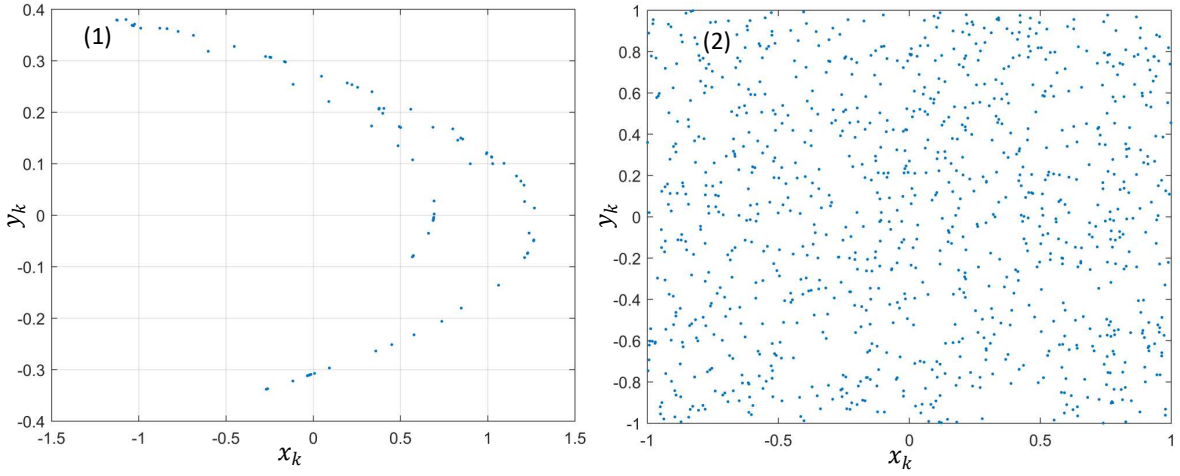

Fig. 7: The resultant phase spaces of : (1) the original Hénon map, (2) the improved Hénon map. The simulation result was obtained using a fixed arithmetic precision of $LQL-4$, where L represents the precision size ($L=12$ in this case). On the other hand, $LQL-4$ means that the fractional part has a size of $L-4$ (the integer part has 4 bits of size).

that a truly random process has very low autocorrelation coefficients, so the autocorrelation effectively evaluates the complexity of a given chaotic system. Autocorrelation can also help to determine the period of a chaotic system by identifying repeated patterns within the chaotic sequence.

The autocorrelation test was performed on both the original and improved Hénon maps for L = 16. The results are shown in Fig.9. The improved Hénon map has a poor correlation between its output samples and has a noise-like autocorrelation result. The original Hénon map, on the other hand, falls swiftly in the cycle and has a high correlation between its output samples.

The comparison of the autocorrelation coefficient values obtained from the improved Hénon map and the original map is another comparison proof that confirms the improvement made on the Hénon map; Fig.9 (c) represents the comparison result of the autocorrelation coefficients obtained from the improved Hénon map and the original map. Fig.9 (d) represents the comparison result of the autocorrelation coefficients obtained from the improved Hénon map and the Matlab *rand*.

The improved Hénon map exhibits the lowest autocorrelation coefficients when compared to the original map, as shown in Fig.9 (c). This means that the Hénon map's complexity has increased dramatically.

The Matlab *rand* function is used to generate pseudorandom sequences with uniformly distributed elements in the interval (0,1); it is a true random-like process because it is carried out with high arithmetic precision (64 bits). When the complexity analysis findings of the improved Hénon map are compared to the *rand* function, it is possible to get a good idea of the improved map's complexity quality. Although the improved Hénon map was carried out using low arithmetic precision (16 bits), it yielded better results than the *rand* function, which was carried out using high arithmetic precision. This is proven by the improved Hénon map's low autocorrelation coefficients when compared to the rand function (Fig.9 (d)).

The period length of the improved Hénon map has been effectively extended. The computed period length for both the original and improved Hénon maps is shown in the Table.1. It should be noted that for the improved Hénon map, the process was carried out with a set of control parameters that were chosen randomly. On the one hand, the results reveal that any control parameter for the improved map yields the same result (in which the period length is extended effectively). On the other hand, the period for $L=16$ has been prolonged by 16569 times compared to the original map, while the periods for $L=24$ and $L=32$ cannot be computed with our available computing platform.

4.1.4 Largest Lyapunov exponent analysis

The complexity of a given dynamical chaotic system refers also to its high sensitivity to the initial conditions; for chaotic systems, two trajectories started from very close points (initial conditions) diverge quickly. The largest LE (Lyapunov Exponent) is the mathematical tool that helps measure the rate of the divergence between the two trajectories. A chaotic system with a positive LE will have completely diverged trajectories

Table 1: The computed period length using different arithmetic precision, (U: Undefined)

Control parameters	Period length (Improved Hénon map)		
	L = 16, N=8	L = 24, N=11	L = 32, N =19
a= 0.1, b=0.1, c=0.1, d=0.1, e=0.9	1,986,905	108,917,260	U
a= 0.8, b=0.3, c=1.1, d=1.4, e=0.1	5,031,060	U	U
a= 1.8, b=2, c=1.8, d=0.3, e=0.3	4,351,739	U	U
a= 1.4, b=0.3,c=1.8,d=1.8, e=0.5	5,222,360	U	U
a= 0.9,b=1.3,c=0.43,d=0.5, e=0.7	6,169,873	U	U
a= 1.9, b=0.5,c=1.4,d=1.7, e=1	5,103,403	U	U
Control parameters	Period length (Original Hénon map)		
	L = 16	L = 24	L = 32
a =1.4, b=0.3	280	1798	536792

after a certain number of iterations, while the largest LE value is an indicator of higher unpredictability and sensitivity. On the other hand, a negative or zero value indicates periodic behaviour [47].

The original Hénon map and the improved one were subjected to the largest LE analysis as a function of the control parameters. The largest LE analysis on the Matlab *rand* function was undertaken to confirm the efficiency of the improvement made on the Hénon map. Because the *rand* function generates a different state each time the is run, the largest LE was applied to the sequence generated as a function of the round number (the rounds here replace the control parameters in the Hénon maps). The obtained results are shown in Fig.10. As we can see:

- For the original Hénon map, the largest LE takes negative values for $a < 1$ and $b < 0.07$, on the other hand, the maximum of the largest LE cannot exceed 0.48 for the first case (Fig.10 (1)) and 0.7 for the second case (Fig.10 (2)).
- In the case of the improved Hénon map; the obtained results were quite better. According to Fig.10 from (4) to (8), the largest LE is always positive (contrary to the original map) over the whole intervals of the control parameters, on the other hand, the largest LE had higher values compared to the original Hénon map, the largest LE approaches the value 2.35.

Based on what has been mentioned thus far, we may conclude that the improved Hénon has the highest largest LE value and thus more unpredictability and sensitivity to the initial conditions. These findings are also significantly better than those obtained using the rand function (Fig.10 (3)), despite the fact that this function is efficient and implemented high arithmetic precision.

4.2 Statistical properties analysis

In this section, the improved Hénon map is evaluated in terms of the statistical properties of its output. For this purpose, some statistical metrics are used, such as the approximate entropy, the sample entropy, and the permutation entropy. The well-known NIST statistical test set is also used to evaluate the randomness of the improved Hénon map's output. Matlab also has an efficient function for evaluating the complexity of a given random sequence, which is used to evaluate the output of the improved Hénon map.

4.2.1 The Approximate Entropy analysis

The Approximate Entropy ($ApEn$) is a mathematical tool that is used to evaluate the non-regularity and unpredictability in time series data. This test has been proposed first by Steve M. Pincus [48] for the purpose of detecting similar patterns in time series. A comprehensive tutorial on the $ApEn$ with theoretical background can be found in [49]. Roughly speaking, high values of $ApAn$ mean the signal is more random and unpredictable, whereas low values mean the signal is more regular and easy to predict.

The $ApEn$ is performed on the original and improved Hénon maps' outputs for a set of control parameters, the $ApEn$ is also performed on the Matlab *rand* function as a function of a number of rounds, this is for the purpose of making a comparison between the obtained results from the improved map and the Matlab *rand* function. The obtained results are presented in Fig.11.

The obtained results show that:

- The best $ApEn$ result for the original Hénon map is 0.47 for $a = 1.4$, and 0.48 for $b = 0.3$, which are the only two best cases.
- For the improved Hénon map, the obtained results are quite better, the $ApEn$ exceeds the value 1.7 for the control parameter a and maintained around

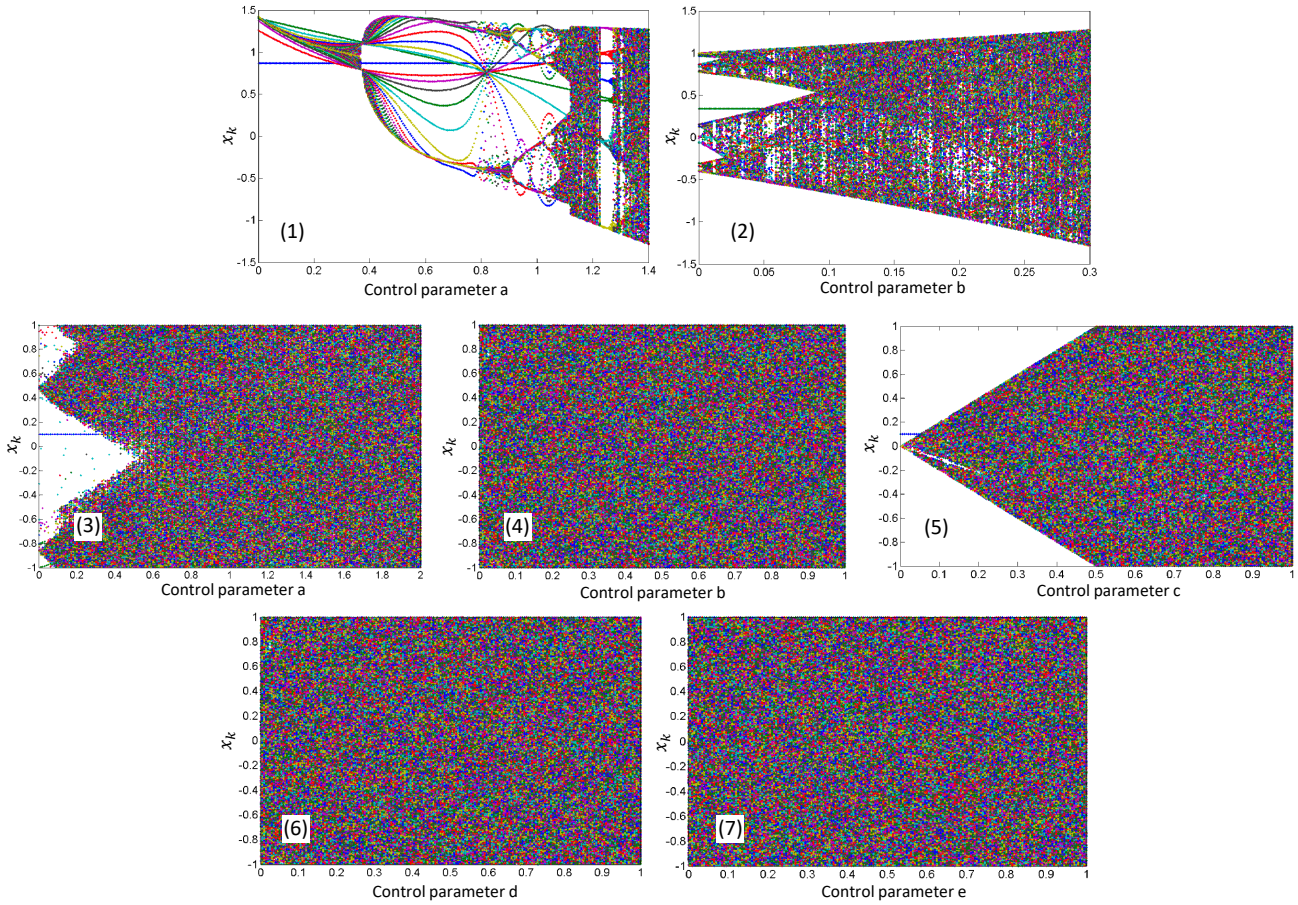

Fig. 8: The bifurcation diagrams for both the original and the modified Hénon maps: (1) and (2) the bifurcation diagrams of the original Hénon map respectively as a function of the control parameters a and b, (3) to (7) the bifurcation diagrams of the modified Hénon map respectively as a function of the control parameters a, b, c, d, and e.

this value over the whole interval. For the control parameter b; the $ApEn$ exceeds the value 1.82 and varies between this value and 1.52 over the whole interval.

– The obtained results for the improved Hénon map are much better; the $ApEn$ exceeds the value of 1.7 for the control parameter a and remains around this value over the entire interval. The $ApEn$ exceeds the value of 1.82 and ranges between this value and the value 1.52 during the entire interval of the control parameter b. For the control parameter c, the $ApEn$ varies between 1.75 and 1.23 over the entire interval (if we exclude the part between 0 and 0.07). For the control parameter d, the $ApEn$ varies between 1.82 and 1.5, and finally, for the control parameter e, the $ApEn$ varies between 1.73 and 1.45 over the entire interval. These results are quite better when compared with the original map, and they are close to those obtained using the Matlab *rand* function.

We may deduce from the obtained results that, first, the high $ApEn$ values corroborate the improved Hénon map's high unpredictability and randomness, and second, the high values are preserved over the entire intervals of the control parameters. This also confirms that for the improved Hénon map, the control parameter intervals in which the system is chaotic are extended.

4.2.2 The Permutation Entropy Analysis

The Permutation Entropy (PE) is an efficient tool that quantifies effectively the complexity in a dynamic system, it bases on the principle of capturing the order relations between values of a time series and extracting a probability distribution of the ordinal patterns, this tool is characterized by its conceptual simplicity

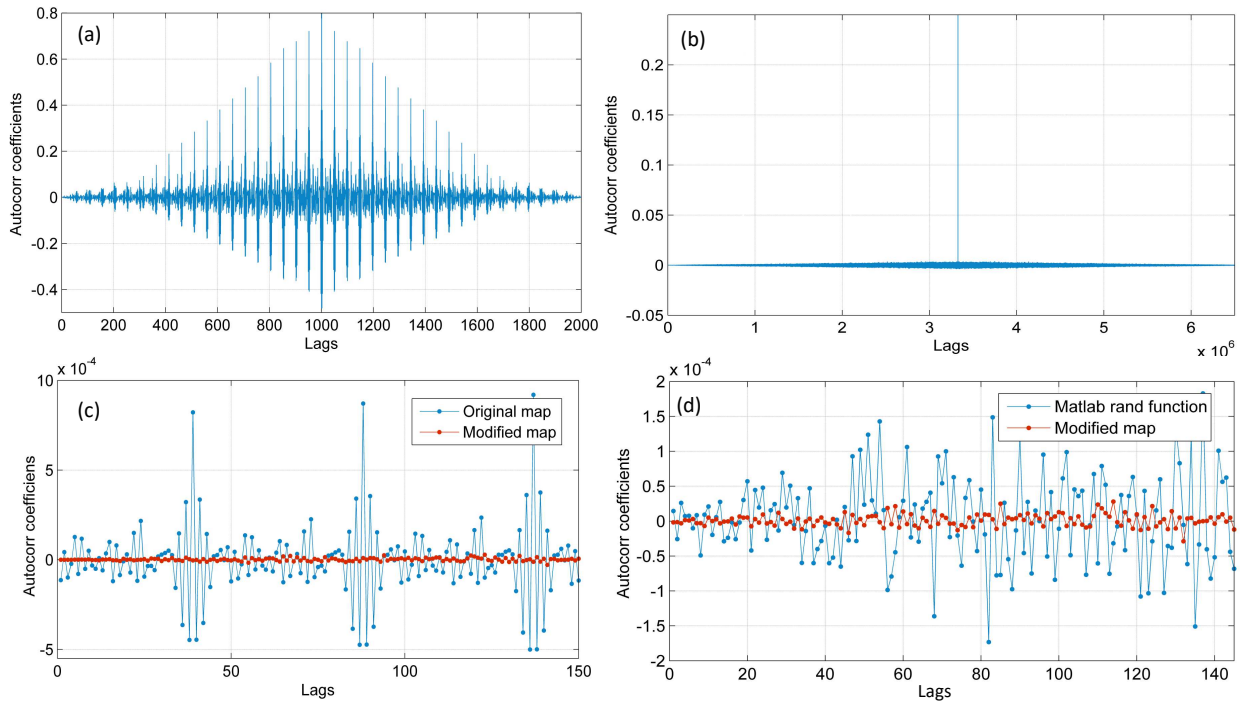

Fig. 9: The result of the autocorrelation test performed on : (a) the original Hénon map, (b) the improved Hénon map. Autocorrelation coefficients comparison: (c) the improved map versus the original one, (d) the modified map versus the Matlab *rand* function.

and computational speed and provide many advantages [50, 51]. The smaller the PE is, the more regular and more deterministic the time series is. Contrarily, the closer to 1 the PE is, the more noisy and random the time series is. For an in-depth understanding of how the PE test is performed, a comprehensive example is given in [50].

The PE test is carried out on both the original and improved Hénon maps as a function of a set of control parameters. The PE test is also performed on the Matlab rand function, as previously done, and the results are shown in Fig.12. The following is how the obtained results can be interpreted:

- For the original Hénon map, there are only two cases where the PE is high; the first for $a = 1.116$, where $PE \simeq 0.998196$, and the second for $b = 0.97$, where $\simeq 0.999351$; in the other cases, the PE varies between 0.254765to 0.998196 for a and 0.970927 to 0.998196 for b.
- For all control parameters in the improved Hénon map, the PE ranges from 0.991231 to 0.999931. In contrast to the original map, these results are preserved over the entire intervals of the control parameters. The PE values obtained are quite close to 1 and are consistent with the results obtained using the Matlab *rand* function. This confirms that the improved Hénon map has better randomization quality than the original one.

4.2.3 The Sample Entropy Analysis

Sample Entropy ($SampEn$) has the same concept as the $ApEn$ (it is a modified version of the $ApEn$). It was introduced by the authors in [52] and has some advantages over the $ApEn$, such as accuracy, data length independence, and implementation simplicity. More details about the SampEn can be found in [49]. High $SampEn$ values imply high complexity of the evaluated system's output.

The $SampEn$ test was performed on the original Hénon map, the improved one, and the Matlab *rand* function for comparison. The obtained results are shown in Fig.13. We can readily see that:

- For the original Hénon map; the $SampEn$ varies between 0.47 and 0.54 in the interval $a \in [1.15, 1.4]$, and between 0.32 and 0.55 for $b \in [0.06, 0.3]$. In fact, these are the best results for the original Hénon map.
- In the case of the improved Hénon map; the results are quite good, the $SampEn$ varies from 0.79 to 0.99 for the control parameter a, from 0.82 to 0.97 for the control parameter b, from 0.72 to 0.99 for the

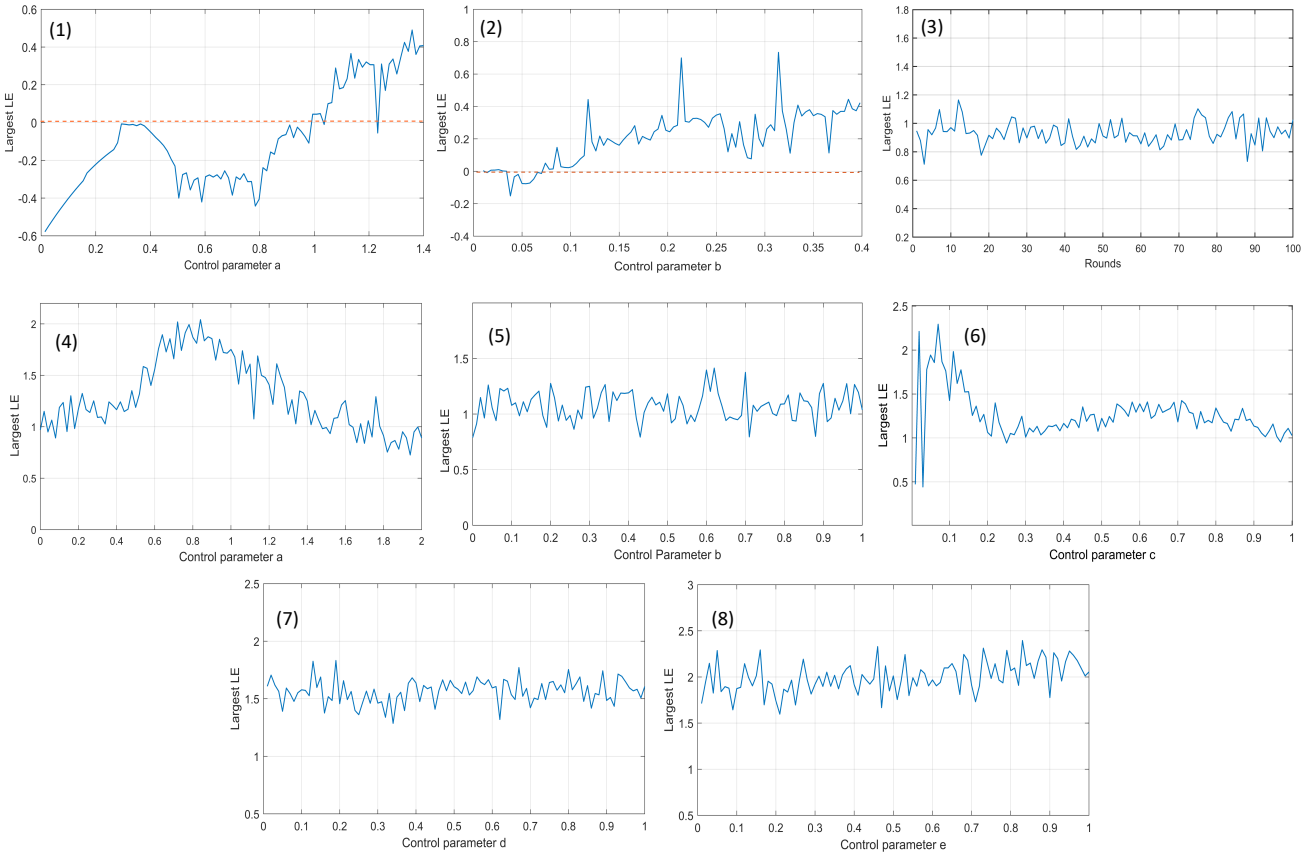

Fig. 10: The Largest *LE* analysis : (1),(2) for the original Hénon map as a function of the control parameters a and b, (3) for the Matlab *rand* function versus the round number, and (4),(5),(6),(7),(8) for the improved Hénon map as a function of the control parameters a, b, c, d, and e.

control parameter c, from 0.8 to 0.98 for the control parameter d, and from 0.83 to 0.99 for the control parameter e. These results are achieved across the whole range of control parameters intervals.

The obtained results show that the *SampEn* has high values for the improved Hénon map when compared to the original one; these values are approximately equivalent to those obtained for the Matlab *rand* function, despite the latter is implemented using high arithmetic precision (64 bits). This confirms the improvements made to the Hénon map in terms of complexity and control parameter interval extension.

4.2.4 Matlab Runstest function analysis

Matlab has a convenient function for determining the randomness of a given time series. It is about the *runstest* function, which is based on the number of consecutive runs with values above or below the variable's mean (the signal to be evaluated). If the test rejects the null hypothesis at the 5% significance level, the result is 1, else it is 0. The test can additionally provide a numerical value for the p_{value} (significance level), which must be greater than or equal to 0.05.

The runstest results are shown in Fig.14. It's worth noting that all of the runs' test results are presented in a single figure to facilitate comparison. The abscissa indicates the position of the control parameter being employed within its interval. As illustrated in Fig.14, the success of having a value greater than 0.05 is as follows:

- For the original Hénon map, the success rate is 20.83% for control parameter a, and 16.67% for control parameter b.
- For the improved Hénon map, the success rates for the control parameters a, b , c, d, and e are, respectively, 97.92%, 97.92% , 96.88%, 97.92%, and 98.95%.

In comparison to the original Hénon map, we can conclude that the improved Hénon map has better randomness properties. The improved Hénon map, on the other hand, gets better results than the Matlab *rand*

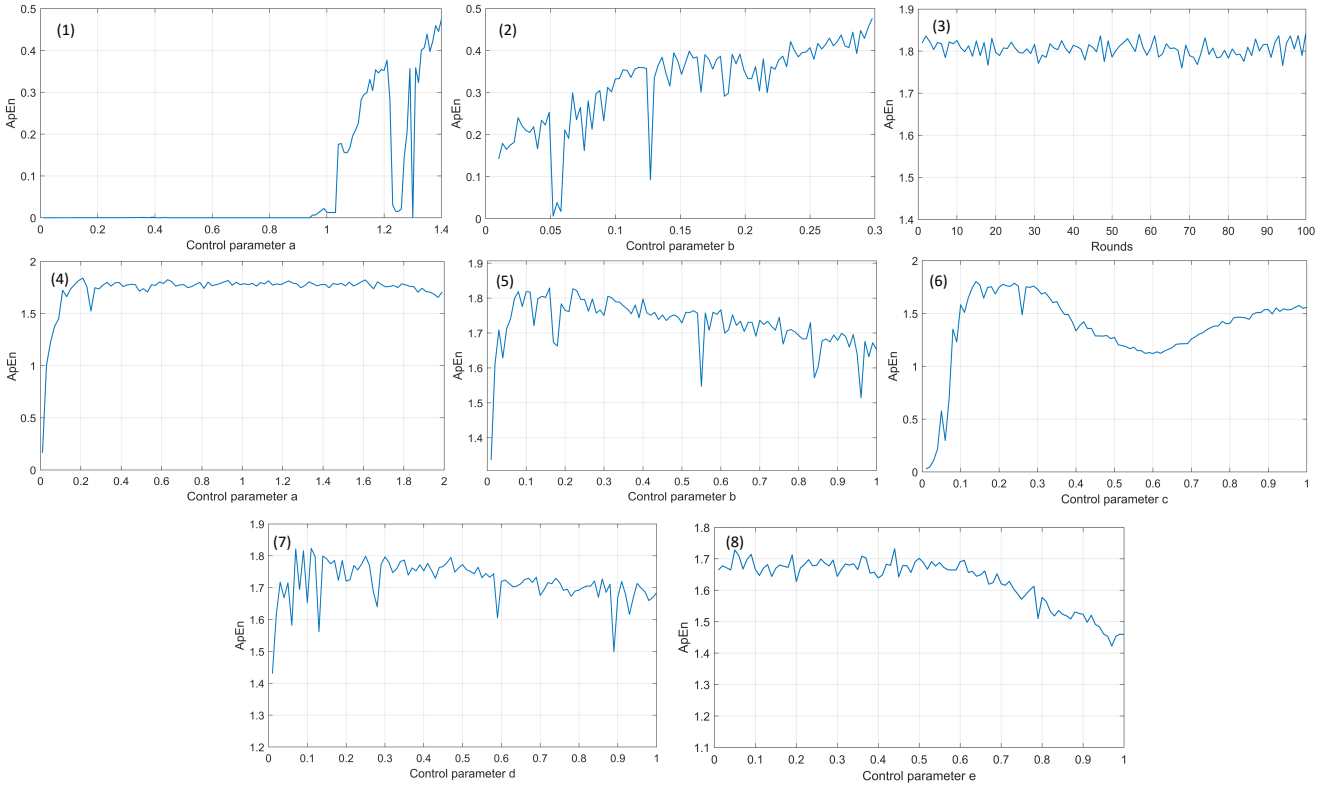

Fig. 11: The *ApEn* analysis : (1),(2) for the original Hénon map as a function of the control parameters a and b, (3) for the Matlab *rand* function versus the round number, and (4),(5),(6),(7),(8) for the improved Hénon map as a function of the control parameters a, b, c, d, and e.

function. The *rand* function has a success rate of 94.79% (despite being implemented using high arithmetic precision), which is less than the improved Hénon map in all cases. Thus, the Matlab *runstest* function's result show that the Hénon map has been improved.

4.2.5 The NIST test suite analysis

The NIST (National Institute of Standards and Technology) Test Suite is a statistical package comprised of 15 tests designed to check the randomness of (arbitrarily long) binary sequences generated by cryptographic random or pseudorandom number generators based on hardware or software [53]. Each test generates a p_{value}, which should be equal to or greater than 0.01, to indicate that the test was successfully passed.

The binary output sequences of both the original and improved Hénon maps are evaluated using the NIST statistical tests. In the case of the improved Hénon map, the tests are performed for different randomly chosen control parameters sets (eight cases) to be sure about the given results.

The obtained results, as well as the success overage, are shown in Table.2. It is obvious that the improved Hénon map has high randomness properties; more than 92.19% of the tests are passed successfully, whereas only 18.75% percent of the tests are passed successfully in the case of the original Hénon map (and this is the best case).

5 Security evaluation of the PCBSC

This section deals with the evaluation of the PCBSC in terms of its security and reliability. It is the most significant part of this work because it determines the PCBSC's security level. We begin by analyzing the key strength and space, then the confusion and diffusion properties, and finally the statistical properties of the encrypted image to demonstrate how the plaintext's statistical properties are dispersed throughout the ciphertext. The PCBSC was also subjected to a number of well-known cryptographic attacks in order to assess its reliability.

It's worth noting that the implementation precision has been increased ($L = 32$ in this case), which increases the size of the system's key while also raising the chaotic generator's complexity. The improved Hénon

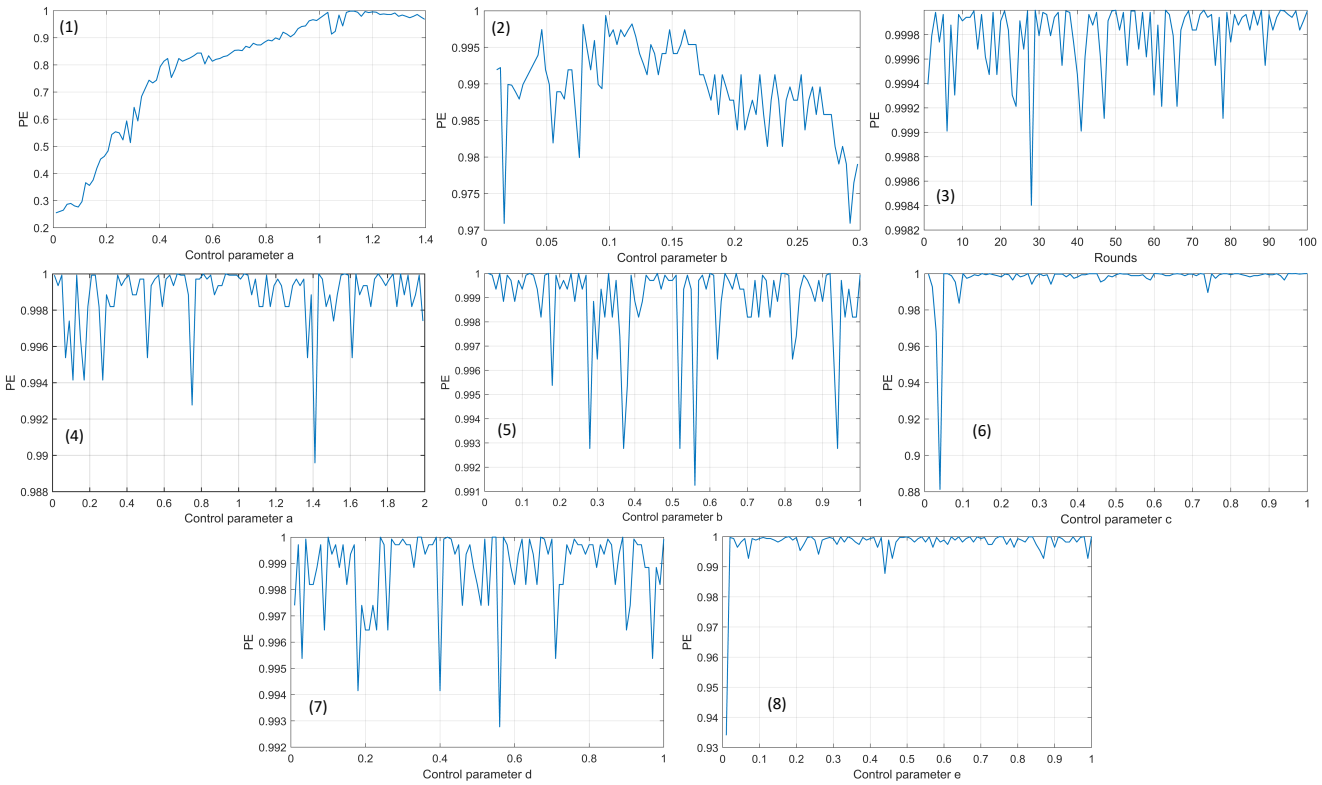

Fig. 12: The *PE* analysis : (1),(2) for the original Hénon map as a function of the control parameters a and b, (3) for the Matlab *rand* function versus the round number, and (4),(5),(6),(7),(8) for the improved Hénon map as a function of the control parameters a, b, c, d, and e.

map achieved excellent results when $L = 16$ was used. As a result, using $L = 32$ increases more the chaotic generator's complexity and statistical properties.

5.1 The system's keys

In the encryption algorithm, the key is the most crucial component. The security of the system is directly dependent on this key: how it is made, how long it is, and how strong it is. The length of the key, which can be expressed in bits, determines the level of security given by an encryption algorithm. The maximum number of operations required for decryption is defined by the length of the key.

The control parameters are commonly used to generate keys in chaos-based cryptography. The system only exhibits chaotic behavior for certain and restricted ranges of control parameters, which poses a significant difficulty. Selecting keys outside of these specific ranges is pointless because the system will behave in a predictable manner rather than chaotically. This almost surely results in a vulnerable encryption system. As a result, the keys should be chosen from the ranges in which the system behaves chaotically.

5.1.1 The keys construction

Bifurcation diagrams can be used to identify the control parameter ranges in which the system exhibits chaotic behavior. By referring to the bifurcation diagrams (Fig.8); we can see that the ranges of control parameters where the improved map exhibits chaotic behavior are extended. The control parameter a can be set to any value in [0,2] Over the interval [0,1], the control parameters b, c, d, and e can take any value. As a result, it should be evident that the keys can be chosen from any of these intervals, but there are some critical considerations to make:

- It would be preferable if we avoided using very small control parameter values. Experiments have shown that setting all control parameters to small values (> 0.05) at the same time leads in quasi-regular behavior. As a result, the control parameter intervals should start from values greater than 0.05.
- Experiments also shown that setting the control parameter b to a value greater than 0.55 lengthens the time it takes for the systems (emitter and receiver) to synchronize. Because it is preferable for the sys-

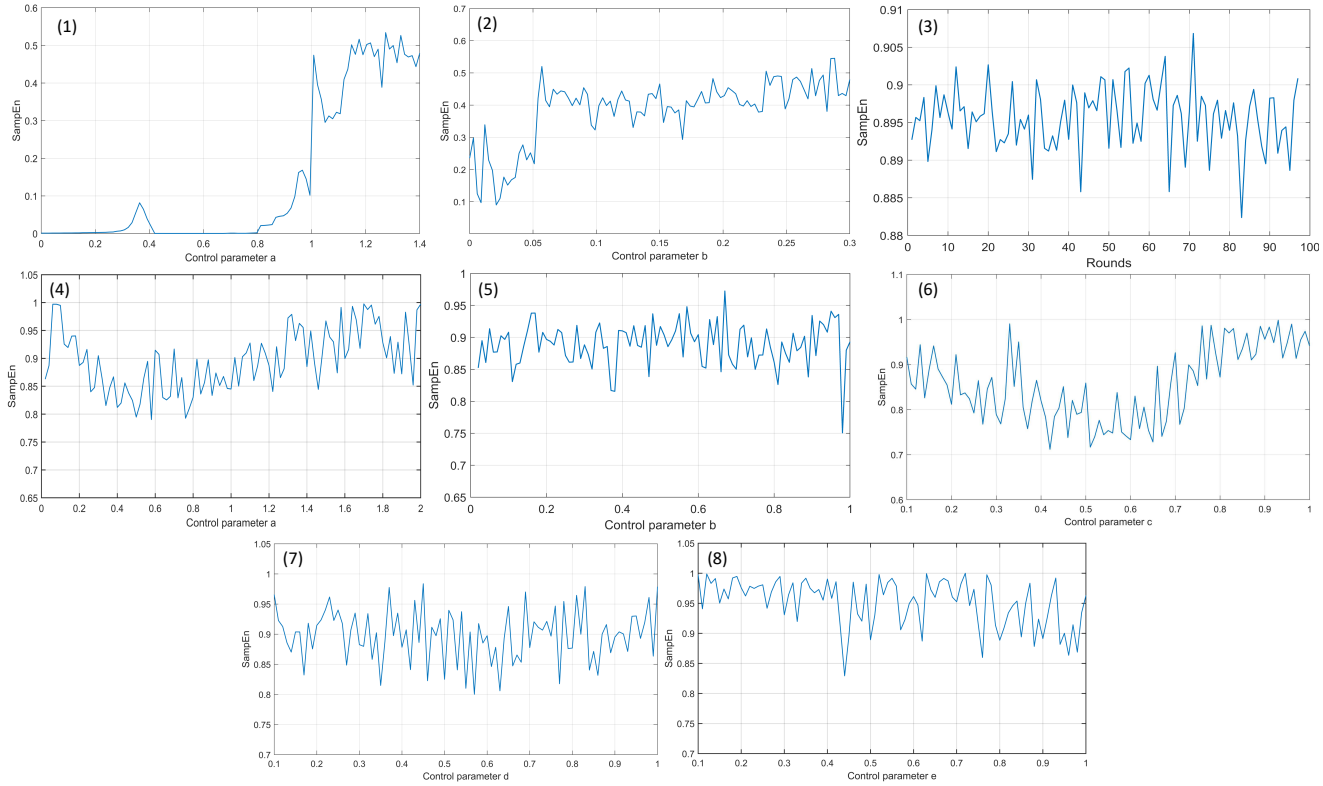

Fig. 13: The *SampEn* analysis : (1),(2) for the original Hénon map as a function of the control parameters a and b, (3) for the Matlab *rand* function versus the round number, and (4),(5),(6),(7),(8) for the improved Hénon map as a function of the control parameters a, b, c, d, and e.

tems to synchronize quickly, the appropriate interval for the control parameter b is $[> 0.05, < 0.55]$.

Because the system is implemented using fixed-point precision arithmetic, we can simply manipulate the control parameters' binary words. As a result, we proposed the control parameters generator to achieve what was previously said about the new control parameter boundaries (Fig.1). The Fig.15 depicts its internal basic scheme.

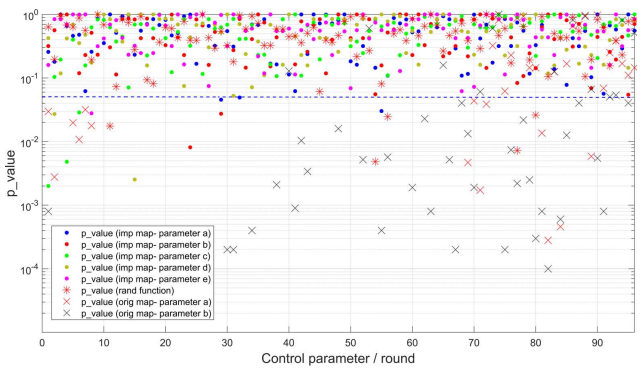

Fig. 14: The Matlab *runstest* p_{value} as a function of the control parameters (for original and improved Hénon maps) and rounds (for the Matlab *rand* function), the blue dashed line represents the limit of the 5% under which the obtained values are considered failed cases.

The Key_a (an integer of 31 bits used to construct the parameter a) is passed through a *Slice* block to extract the three MSBs (signal S_{a1}) and the remaining 28 bits (S_{a2}). These two signals are combined with a constant (of 1 bit of length) $S_{a3} =$ "1" in binary to create a new binary vector in the order : $S_{a1}\&S_{a3}\&S_{a2}$ (a new integer of 32 bits of length). Then after, the newly obtained integer is reinterpreted as $32Q31$. In other words, the 32-bit integer is converted to a fractional number with a fractional part of 31 bits of length and an integer part of 1 bit. The obtained number corresponds to the control parameter a, has the following boundaries:

- $Key_a = 2^{31}$, corresponds to $a = 1.999999999534339$ (in binary 111**1**11111111111111111111111111111).
- $Key_a = 0$, corresponds to $a = 0.125 =$ (in binary 000**1**00000000000000000000000000000).

In the binary data, the 1 represented in bold is the constant S_{a3}. It is obvious that we always have $a \in [0.125, 1.999999999534339]$ for any given value to the

Table 2: Results of the NIST statistical tests performed on both the original and improved Hénon maps.

TEST	Original Hénon map	Improved Hénon map								
		Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Rate of success
	P_{value}	P_{value}	P_{value}	P_{value}	P_{value}	P_{value}	P_{value}	P_{value}	P_{value}	
Frequency	**Failed**	0,3025	0,0183	0,0477	0,7489	0,1510	0,2766	0,2369	0,3525	100 %
Block freq (m = 128)	**Failed**	0,9995	0,0680	0.9997	0,9999	0,9406	0,9986	0,0352	0,0459	100 %
Cusum-forward	**Failed**	0,0623	0,0285	0,0615	0,9485	0,9632	0,1937	0,1189	0,2068	100 %
Cusum-reverse	**Failed**	0,1079	0,0174	0,0750	0,8661	0,7875	0,4003	0,3319	0,5870	100 %
Runs	**Failed**	0,1485	0,2370	0,5072	0,4931	0,4183	0,0782	0,1469	0,0590	100 %
Long runs of ones	**Failed**	0,2279	0,6954	0,5607	0,3391	0,0545	0,0758	0,2294	0,0384	100 %
Binary Matrix Rank	0,2258	0,5713	0,3983	0,1195	0,9567	0,6053	0,8983	0,5984	0,3003	100 %
Spectral DFT	**Failed**	0,6014	0,8846	**Failed**	0,7135	0,8846	0,9306	0,7495	0,7495	87.5 %
NonOver Temp (m=9)	**Failed**	0,5834	0,9543	0,1262	0,8574	0,4334	0,6719	0,8004	0,6729	100 %
Over Temp (m=9)	**Failed**	0,5548	0,7912	0,9434	0,2863	0,9071	0,0335	0,1189	0,1535	100 %
Universal	**Failed**	0,2130	0,9611	**Failed**	**Failed**	0,2604	0,7902	**Failed**	0,5766	62.5 %
Approx Entro (m = 10)	0,3410	0,8654	0,9199	0,9300	0,9998	0,9995	0,9984	0,9997	0,9991	100 %
Random Excur	**Failed**	0,0735	**Failed**	0.0733	0,1409	**Failed**	0,0331	0,0134	0,0105	75 %
Random Excur Var	**Failed**	0,3763	**Failed**	0.0351	0,2041	**Failed**	0,0170	0,0305	0,0433	75 %
Lin Comp (M = 500)	0,33581	0,2411	0,8507	0,0906	0,0537	0,3198	0,1482	0,6435	0,1867	100 %
Serial (m = 16, $\nabla\Psi^2_m$)	**Failed**	0,9796	**Failed**	0,7606	**Failed**	0,8373	0,3167	0,6080	0,6536	75 %

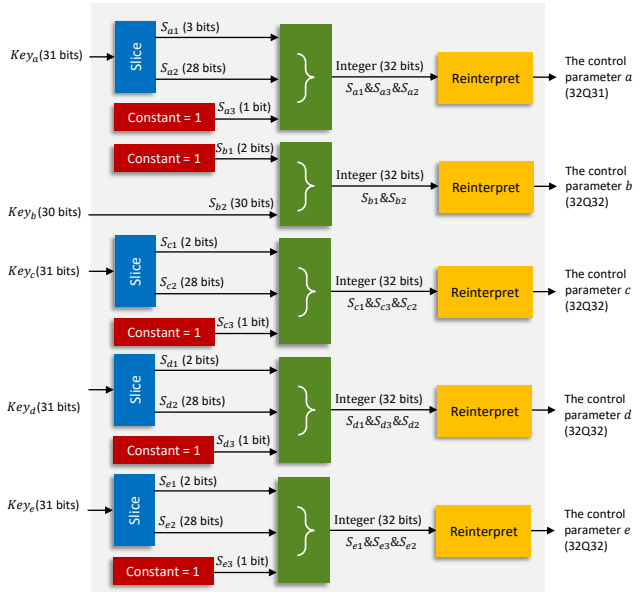

Fig. 15: The internal basic scheme of the control parameters generator.

Key_a, and this exactly what we want since $0.125 > 0.05$.

The Key_b (the key constructing the parameter b), which is an integer (S_{b2}) of 30 bits of length, is combined with a constant of two bits of length (S_{b1} = "01" in binary) to form a new binary vector in this order: S_{b1}&S_{b2} (a new integer of 32 bits of length). The new obtained integer is then reinterpreted as $32Q32$, a fractional number of 32 bits for the fractional part and 0 bits for the integer part. The obtained number represents the control parameter b that has the following boundaries:

- $Key_b = 2^{30}$, corresponds to $b = 0.499999999767169$ (in binary **01**11111111111111111111111111111111).
- $Key_b = 0$, corresponds to $b = 0.25$ (in binary **01**0000 00000000000000000000000000).

The 01 represented in bold represents the constant S_{b1}. It is obvious that we always have $b \in [0.25, 0.499999999767169]$ for any given value to the Key_b; and this exactly what we want since $0.25 > 0.05$ and $0.499999999767169 < 0.55$.

The process is the same for the keys Key_c, Key_d, and Key_e (keys that construct the control parameters c, d, and e), each key is coded in 31 bits and passed through a *Slice* block to extract the two upper bits (S_{c1}, S_{d1}, and S_{e1}), and the remaining 28 bits (S_{c2}, S_{d2}, and S_{e2}) Each of these two signals pairs are combined with constants (each constant = "1" in binary) of 1 bit of length (S_{c3}, S_{d3}, and S_{e3}) to form new binary vectors in these orders: S_{c1}&S_{c3}&S_{c2}, S_{d1}&S_{d3}&S_{d2}, and S_{e1}&S_{e3}&S_{e2} (new integers of 32 bits of length). The newly obtained integers are then converted to a fractional precision numbers of $32Q32$ (32 bits for the fractional part and 0 bit for the integer part). The following are the boundaries of each new control parameter (c, d, and e):

- $Key_c, Key_d, Key_e = 2^{31}$, corresponds to $c, d, e = 0.999999999767169$ (in binary 11**1**11111111111111111 1111111111111).

– $Key_c, Key_d, Key_e = 0$, corresponds to $c, d, e = 0.125$ (in binary 00**1**00000000000000000000000000000).

The 1 represented in bold is the constant that added to each key. It is obvious that we always have $c, d, e \in [0.125, 0.999999999767169]$ for any given values to these keys, and this exactly what we want since $0.125 > 0.05$.

"Key strength" refers to the encryption system's sensitivity to minor changes in the key.Generally, the smallest amount of change is flipping one bit in the key.A good encryption system should provide highly sensitive keys. This property is also linked to the confusion property explained before, which means that the relationship between the key and the corresponding ciphertext is not in a simple way to know. Knowing full or partial information about the ciphertext shouldn't reveal any information about the used key, even if the structure of the encryption system is well known.

When an intruder tries to recover the plaintext using a key that is extremely close to the original key used for encryption (difference of one bit), no partial or complete information about the plaintext should be given. Two identical plaintexts encrypted with two very close keys, on the other hand, should produce completely different ciphertexts. This concept is linked to the modern notion of the "*avalanche effect*". Flipping one bit in the key must change each output bit with a probability of 50% (strict avalanche criterion).

Fig.16 presents the the encryption and decryption processes of an RGB image. The results show the difference between two ciphertexts encrypted with very close keys (a difference of one bit between each tried key and its corresponding real key). We can see that the two obtained ciphertexts are completely different in each case, despite the fact that the difference in each key is only one bit (LSB). This confirms that the PCBSC has highly sensitive keys.

There is also another important issue related to the key; the equality of strength. This property (as mentioned previously) characterizes the Avalanche Effect. The aim of this test is to evaluate if the keys are equally strong or not. In a good encryption scheme, the BER between the plaintext and the recovered one should be zero for the real keys and around 0.5 for the other keys whenever we are close to or far from the real keys.

The results of the BER evolution as a function of keys pairs are shown in Fig.17. The difference in BER between the plaintext and the recovered text is computed using a one-bit change at each step. According to the obtained results, the PCBSC has a strong key, and the evolution of the BER is flat around 0.5, except for the real keys used for encryption, where the

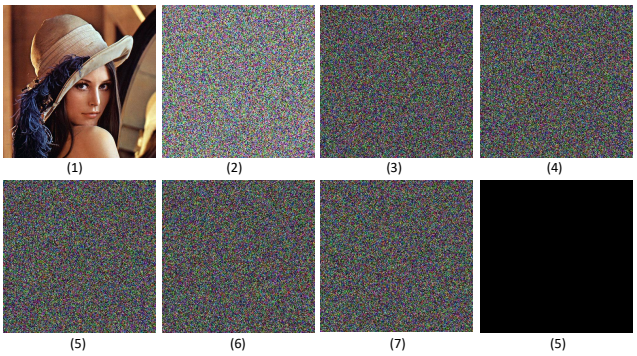

Fig. 16: Key sensitivity evaluation : (1) the original image, (2) the encrypted image, (3) the difference between (2) and an encrypted image with one bit flipped in key_a (LSB), (4) the difference between (2) and an encrypted image with one bit flipped in key_b (LSB), (5) the difference between (2) and an encrypted image with one bit flipped in key_c (LSB), (6) the difference between (2) and an encrypted image with one bit flipped in key_d (LSB), (7) the difference between (2) and an encrypted image using the real keys.

BER is equal to zero. We can confidently state that the *avalanche effect* has been guaranteed.

5.1.2 The encryption block sub-key

As previously stated, the encryption block shown in Fig.3 adds another level of complexity to the PCBSC, hence increasing its security. Given that this part is about the system's key, it's reasonable to evaluate the impact of the sub-keys that control this block. Each of these sub-keys ($key_{a(0)}$, $key_{b(0)}$, $key_{c(0)}$,$key_{d(0)}$, and $key_{e(0)}$) represents the LSB of the corresponding system's key (key_a, key_b, key_c,key_d, and key_e). Because of the control parameter's high sensitivity (evaluation has been performed), any change to these sub-keys results in a different state.

The findings of the evaluation of the sensitivity of the sub-keys are shown in Fig.18, where it is obvious that the encryption block is extremely sensitive to the sub-key. Thus, the encryption block adds an extra layer of protection to the PCBSC.

5.1.3 The Key space

The key space is the possible (valid) keys given to an cryptographic algorithm. A symmetric cryptographic algorithm (such as in our case) should provide a large enough key space to avoid brute-force attacks. A 128-bit symmetric key is computationally secure against brute-force attack [54], thus, in our case, the sum of the whole

key space is 31 bits (Key_a) + 30 bits (Key_b) + 31 bits (Key_c) + 31 bits (Key_d) + 31 bits (Key_d) =154 bits. Therefore, the key space is 154, i.e., there $2^{154} = 2.2836 \times 10^{46}$ possible key.

5.2 Image statistical analysis

Images have some special characteristics compared to other data types, and the relationship between pixels and grayscale levels can reveal information to an intruder. Hence, a good cryptographic algorithm should disperse any statistical information about the original image into the encrypted one. The following statistical analysis has been performed on the PCBSC to evaluate its immunity against statistical attacks:

5.2.1 Image histogram

An image histogram represents the number of pixels for each grayscale level in the image. This analysis has been performed on the PCBSC. The Fig.19 presents the histograms of the original image and the encrypted one. We can see clearly that the encrypted image has a histogram that is quite uniform compared to the original one. This indicates that the statistical properties of the original image are completely dispersed in the encrypted image.

5.2.2 Correlation between adjacent pixels

For an ordinary image with meaningful visual perception, the correlation between adjacent pixels is always high as their pixel values are close to each other [55]. A good cryptographic algorithm produces an encrypted image with a very low correlation between adjacent pixels. To perform this test on the original image and the encrypted one, we randomly selected 1500 pixels from each image over the three directions (horizontal, vertical, and diagonal). For each direction, we computed the correlation coefficient for each pixel pair using the following formulas:

$$\gamma_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{7}$$

Where $D(x)$ and $D(y)$ represent respectively the variance of x and y, $cov(x,y)$ represent the covariance between x and y. We have also :

$$cov(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y)) \text{ and}$$

$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2$ with the mathematical expectation of x $E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i$, $D(y)$ can be computed with the same manner.

The correlation between adjacent pixels' analysis results are shown in Fig.20. For both the original and encrypted images, the figure depicts the correlation distribution between two horizontally adjacent pixel pairs. The original image has a high correlation; the greyscale levels are concentrated around the graph ($x = y$), whereas the encrypted image has a low correlation and the greyscale levels are dispersed across the whole 2D plan.

Table.3 contains numerical data for the correlation coefficients across the three dimensions for both the original and the encrypted images.

5.2.3 Information entropy

Entropy is another efficient metric that assesses the randomness of a particular encrypted image. It represents the average information generated from all pixels. The entropy H is defined as [55] :

$$H = \sum_{n=0}^{L-1} \frac{P_n}{log_2(P_n)} \qquad (bits/message) \tag{8}$$

Where P_n is the probability of occurrence of the given gray level, and L is the maximum of the gray level, each pixle in a grayscale image is coded in 8 bits, resulting in $L = 2^8$.

Images that provide a uniformed distribution of the gray level have good randomness quality and thus result in high information entropy (H approaching the value of 8). The results of the information entropy analysis performed on the PCBSC are given in Table.4.

The analysis is carried out on both the original Lina RGB image and its correponding encrypted image. The achieved information entropy results are quite good for the encrypted image. The obtained entropy values are very close to the value of 8 for each color component. This is a clear indication of the PCBSC's efficiency and reliability.

5.3 The differential attack analysis

The differential attack's goal is to determine how a small change in the system's input affects the output. It's a chosen plaintext attack of some sort. *Biham* and *Shamir* were the first to propose it [57], then *G.Chen* later proposed a more efficient modified version [58]. The diffusion property in an encryption system can also be characterised using the diffirantial attack.

In order to carry out this attack on the PCBSC, we must encrypt two plain images that are identical

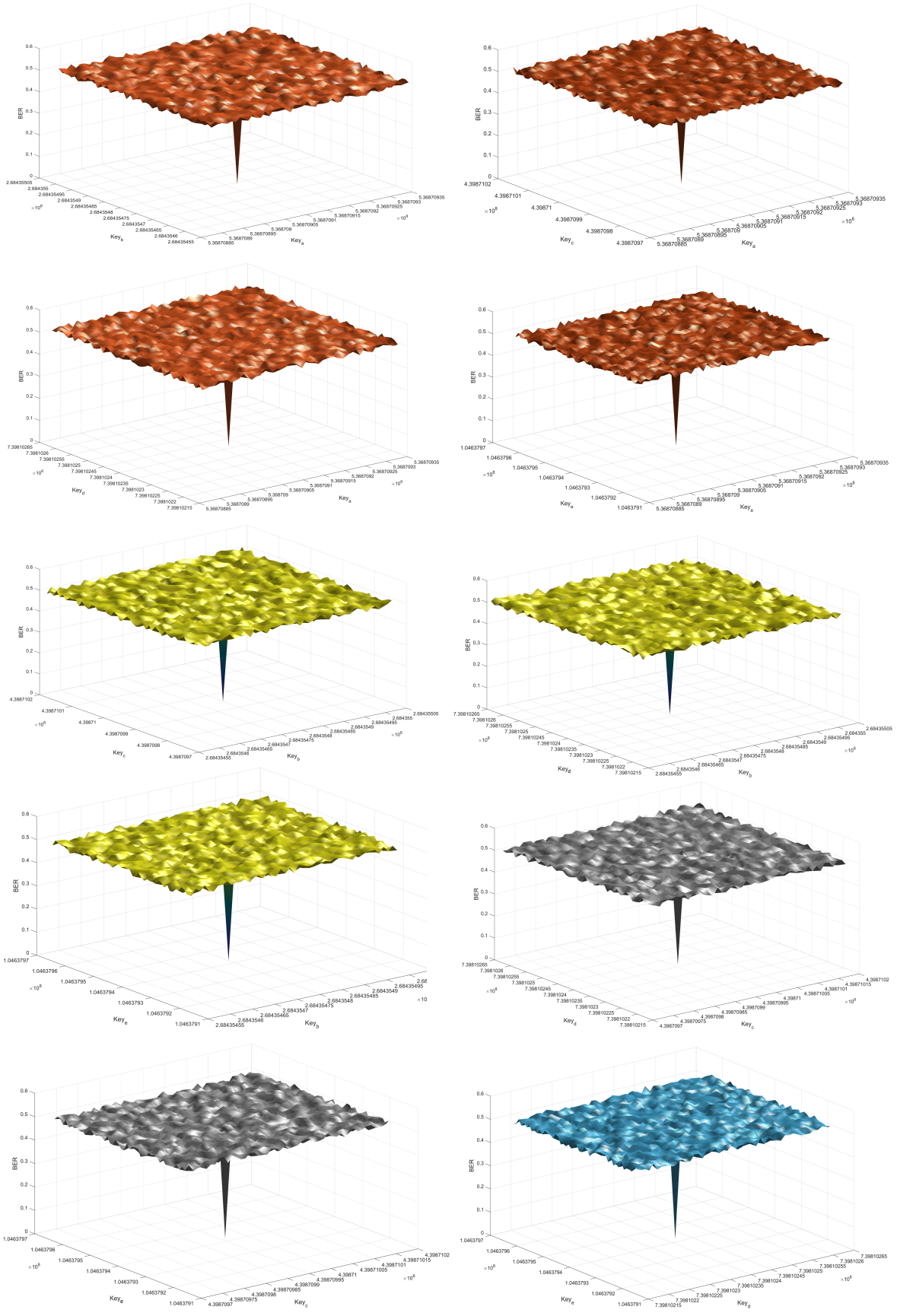

Fig. 17: BER evolution as a function of key pairs

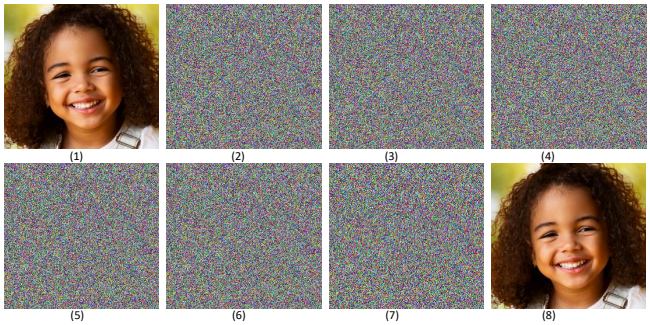

Fig. 18: Evaluation of the sensitivity to the sub-keys controlling the encryption block: (1) the original image, (2) the encrypted image, (3) the recovered image using a wrong $key_{a(0)}$, (4) the recovered image using the a $key_{b(0)}$, (5) the recovered image using a wrong $key_{c(0)}$, (6) the recovered image using a wrong $key_{d(0)}$, (7) the recovered image using a wrong $key_{e(0)}$, and (8) the recovered image using the real sub-keys.

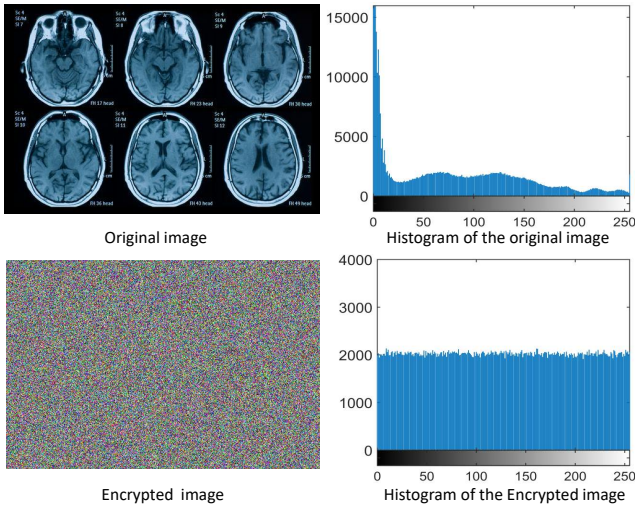

Fig. 19: Image histogram analysis.

except for the first pixel (i.e., they differ only in the first pixel). The differences between the corresponding encrypted images, denoted C_1 and C_2, are then determined. The PCBSC is immune to these types of attacks if the difference between C_1 and C_2 is significant.

According to [58], two measures are used to carry out the differential attack: the Number of Pixels Change Rate ($NPCR$) and the Unified Average Changing Intensity ($UACI$). For a good encryption system, the $NPCR$ should be greater than 99%, whereas the $UACI$ should be greater than 33%, the $NPCR$ and $UACI$ are respectively given by:

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \tag{9}$$

$$UACI = \frac{1}{W \times H}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\% \tag{10}$$

Where :

$$D(i,j) = \begin{cases} 0 & if \quad C_1(i,j) = C_2(i,j) \\ 1 & otherwise. \end{cases} \tag{11}$$

W and H represent respectively, the width and the height of the image.

Differential attack analysis has been carried out on the PCBSC. We have randomly chosen the position of one pixel in the grayscale Lena original image and changed it. Then the original lena image and the midified one (by changing the chosen pixel) are encrypted and decrypted for several rounds. The corresponding encrypted images for each round are used to compute the $NPCR$ and $UACI$. The obtained results are presented in Fig.21.

According to the obtained results, we find that the $NPCR$ varies from 0.996022 % to 0.998432 %, whilst the $UACI$ from 33.381326 % to 33.561924 % . Thus, the results show that the PCBSC is immune to differential attack and that the diffusion property is guaranteed.

6 Performance under noisy channel

This section aims to evaluate the PCBSC under noisy channel conditions, i.e., the evaluation of the proposed synchronization circuit. As previously stated, the proposed synchronization circuit is used to prevent the system from the effects of noise as much as possible. The existence of noise is an important concern, especially considering the fact that chaotic systems are are extremely sensitive to their initial conditions.

Traditional synchronization methods (driving the response system continuously) are substantially more vulnerable to channel noise. In the case of the proposed synchronization circuit, the synchronization is done only for specific moments of time. For a good comparison, the evaluation is done as follows: We use two configurations, the first containing the PCBSC (emitter and receiver) with the traditional (continuous driving

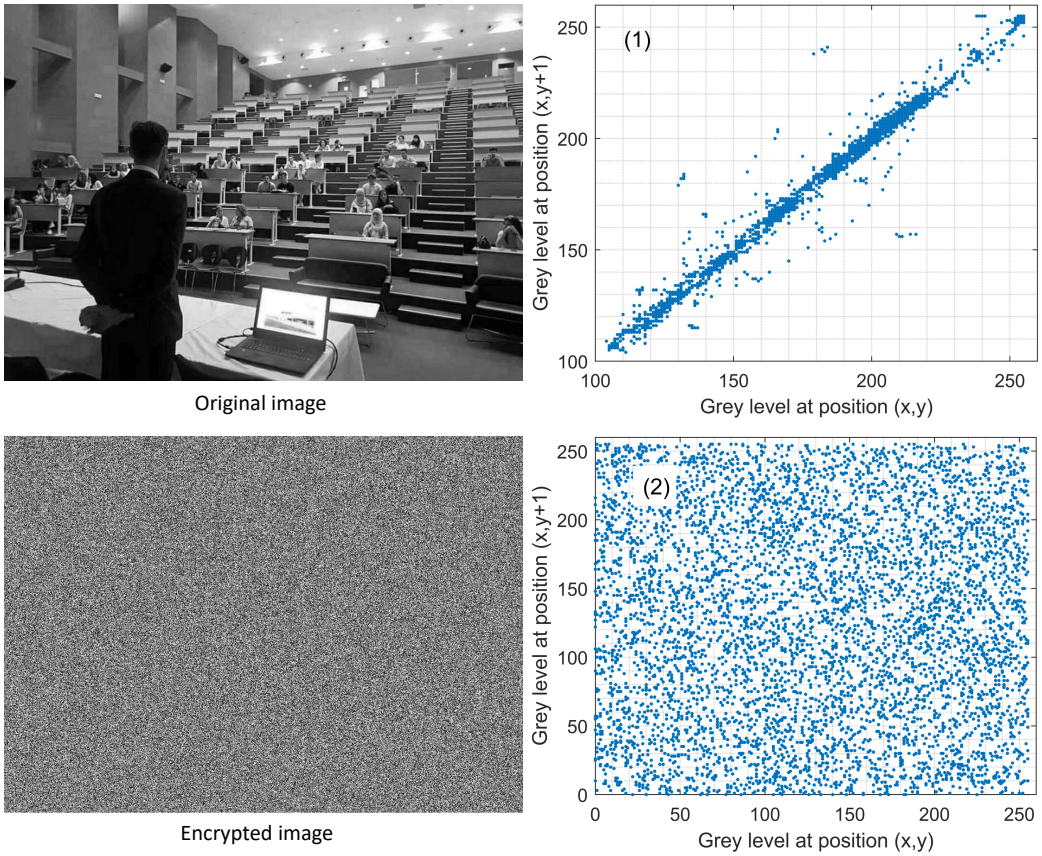

Fig. 20: The correlation between adjacent pixels' analysis results: (1) for the original image, and (2) for the encrypted one.

Table 3: Correlation coefficients values of two adjacent pixels.

Direction	Original image			Encrypted image		
	Red	Green	Blue	Red	Green	Blue
Horizontal	0.977423	0.969979	0.950527	-0.001270	0.000932	0.002259
Vertical	0.981018	0.979189	0.963465	-0.000671	-0.000539	0.001540
Diagonal	0.961647	0.956636	0.935327	-0.001006	-0.003029	-0.000119

of the response system) synchronization method; the second containing the PCBSC, this time using the proposed synchronization circuit. In each case, the driving signal is passed through a QPSK modulator and then a noisy channel with additive white Gaussian noise (AWGN). The received signal is demodulated, and the synchronization takes place.

As described previously, the proposed synchronization circuit includes a counter that serves as a timer for the response system to be disconnected from the driver signal and act independently. The longer the response system operates independently, the more protected it is from channel noise.

Table 4: Information entropy analysis results.

Color component	Original image	Encrypted image
Red	7.567987	7.999225
Green	7.102690	7.999229
Blue	6.829355	7.999354

To evaluate the performance of the proposed synchronisation scheme, we compute the BER between the original Lina image and the recovered one as a function of the SNR (Signal to Noise Ratio). The PCBSC is evaluated in two scenarios: one with the proposed synchronisation circuit and the other without. In the first scenario, the BER is computed for various internal counter sizes (m). This is to show how the time it takes for the response system to operate independently affects the system's performance when running on a noisy channel.

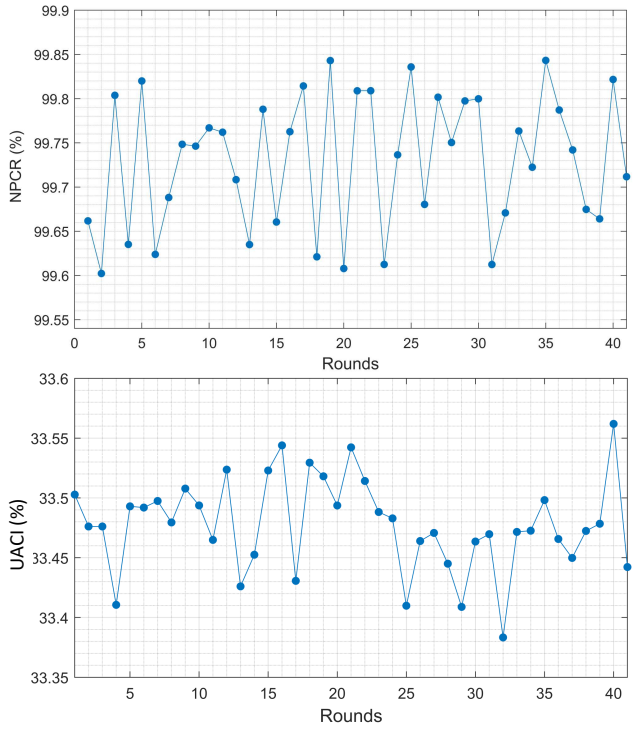

Fig. 21: NPCR and UACI evolution as a function of encryption/decryption rounds for the PCBSC.

Fig.22 depicts the final result. This result clearly reveals that the PCBSC is more efficient against channel noise when the proposed sychnronization circuit is used. On the other hand, the longer the response system is allowed to run independently, the more protected it is against channel noise.

Using $m = 16$ bits as an example, the BER between the original image and the recovered one is 0 at an SNR of 3.5 dB, whereas the BER for the PCBSC without the proposed synchronization circuit is 0.354. Fig.23 and Fig.24 present the recovered images as a function of SNR for both configurations, it is obvious that the proposed synchronization circuit outperforms the usual synchronization method in terms of immunity against channel noise.

7 Real-time wireless transmission with an FPGA-based implementation of the PCBSC

The real-time FPGA-based implementation of the PCBSC over a wireless connection is discussed in this section. We used two Basys 3 boards, including a Xilinx Artix-7 FPGA (XC7A35T series) with two 2.4 GHz radio modules based on the Nordic Semiconductor nRF24L01+ chip.

The Vitis Model Composer tool was used to design the PCBSC so that we could proceed to the synthesis and hardware implementation phases as quickly as possible. Model Composer is a model-based design tool that enables rapid design exploration within the MathWorks Simulink® environment and accelerates the path to production for Xilinx® programmable devices through automatic code generation [59].

An RGB image is encrypted and recovered using the FPGA-based PCBCS. The real-time evaluation using the wireless connection of the PCBSC is done as follows (Fig.25):

- Because the VGA port on the Basys 3 board only supports 4 bits of depth for each color component, an RGB image with 24 bits of depth (8 bits for each color component) is transformed to an RGB image with 12 bits of depth (4 bits for each color component).
- The transformed image is then saved on the FPGA chip's BROM (Block Read Only Memory).
- The original image (P) saved on the BROM is then encrypted using the PCBSC (12 LSBs of the y_k chaotic signal are used to encrypt it). The original image (P) is also transmitted to the screen along with the encrypted image (E) across the VGA interface for display.
- The encrypted image and the driver signal ($S = x_k^2$) are combined into a single signal, which is then divided into bytes blocks (B). The UART component (Universal Asynchronous Receiver-Transmitter) then serializes the incoming bytes (B) and sends them serially to the microcontroller across the RS-232 interface.
- The microcontroller, which has the role of controlling the nRF24L01+ module, receives the incoming serial data (SR) and sends it again to the nRF24L01+ module over the SPI interface.
- The received data is wirelessly sent to the receiver PCBSC across the nRF24L01+ module. The data is subsequently received by the nRF24L01+ receiving module, which serially delivers it to the microcontroller and then to the FPGA-based receiver PCBSC through the RS-232 port.
- The received serial data is then rearranged to form the driver signal and the encrypted image. The driver signal is sent to the proposed synchronization circuit and the encrypted image to the receiver encryption block. When the synchronization takes place, the original image is recovered and sent to the VGA component in conjection with the encrypted image to display them on the screen.

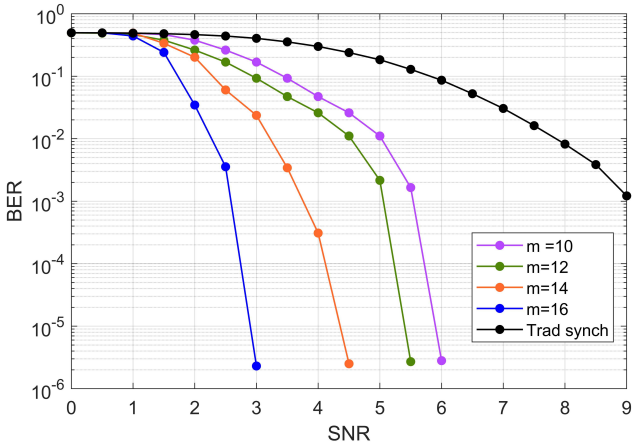

Fig. 22: The evolution of the BER between the original image and the recovered one as a function of SNR, the PCBSC using the propsed sychnronization circuit with different counter sizes (m), and the PCBSC with the traditional sychnronization method.

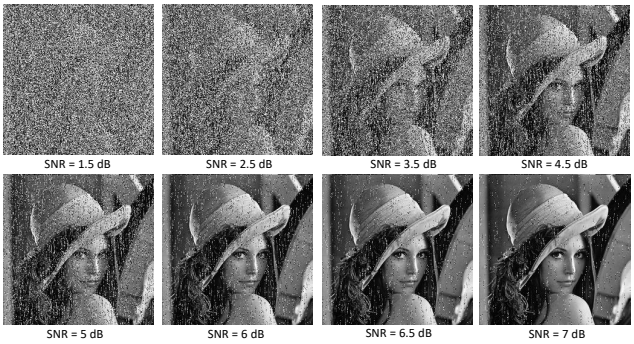

Fig. 23: The recovered images for different values of SNR (PCBSC without the proposed synchronization circuit)

Fig. 24: The recovered images for different SNR values (PCBSC with the proposed synchronization circuit)

The Fig.26 presents the FPGA-based hardware configuration of the PCBSC, in which we can see the Basys 3 board implementing the emitter PCBSC and the other

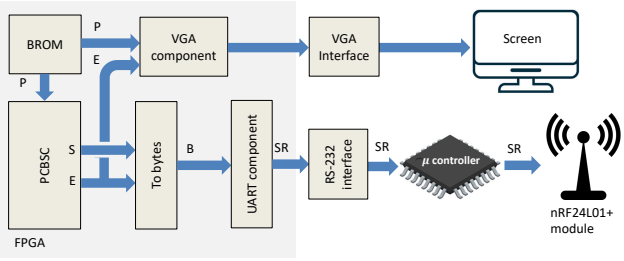

Fig. 25: The basic scheme of the FPGA-based implementation of the PCBSC for image encryption with wireless connection (emitter side).

implementing the receiver system. The first screen displays the original image and the encrypted one, whereas the second screen displays the received encrypted image and the recovered one. It is clear that the original image is well recovered.

7.1 Hardware-based performance and implementation cost

The FPGA resource utilization report is a useful tool for determining the performance of the designed system as well as the cost of implementation. The resource utilization report provided by Vivado is shown in Fig.27. The figure shows the resources available on the XC7A35T FPGA chip, as well as the number of resources required by the PCBSC (LUTs, Flip Flops, DSPs, and Inputs/Outputs). It is clear that the PCBSC requires minimal hardware resources; it requires just 0.32% percent of LUTs, 0.22% percent of Flip Flops, 15% of DSP slices, and occupies only 16.04% of the inputs/outputs. As a result, the PCBSC is both simple and inexpensive to implement.

8 The PCBSC versus other proposals

As previously stated, what distinguishes our work is that it addresses all of the challenges facing the use of chaos in cryptography, as opposed to many other works that only handle some of them and turn a blind eye to the others. This section summarizes the results of comparison of the PCBSC with other proposals from many sides. The PCBSC performance is only compared with a some randomly picked works in this context due to their diversity and huge numbers. The key aspects of comparison are summarized in Table.5. ($\times$) denotes that the authors did not address this issue or that the test was not conducted, whereas ($\surd$) denotes that this issue is addressed.

Table 5: The PCBSC versus other proposals, (*) result obtained using single precision arithmetic

Proposal	Differential attack (%)		Correlation coefficients			Entropy	Key space	Synchro-nization	Dynamical degradation
	NPCR	UACI	H	V	D				
[60]	99.8093	33.4805	-0.0021	0.0203	0.0081	7.99920	$2^{128(*)}$	×	√
[61]	99.7055	33.5106	-0.0032	-0.0007	-0.00018	×	2^{256}	×	×
[62]	×	×	0.0061	0.0018	-0.0024	7.99860	1.1579×10^{105}	×	×
[63]	99.6061	33.4150	-0.0016	-0.0010	-0.0015	×	2^{149}	×	×
PCBSC	99.8432	33.3813	0.0014	-0.0006	-0.0013	7.99927	2^{154}	√	√

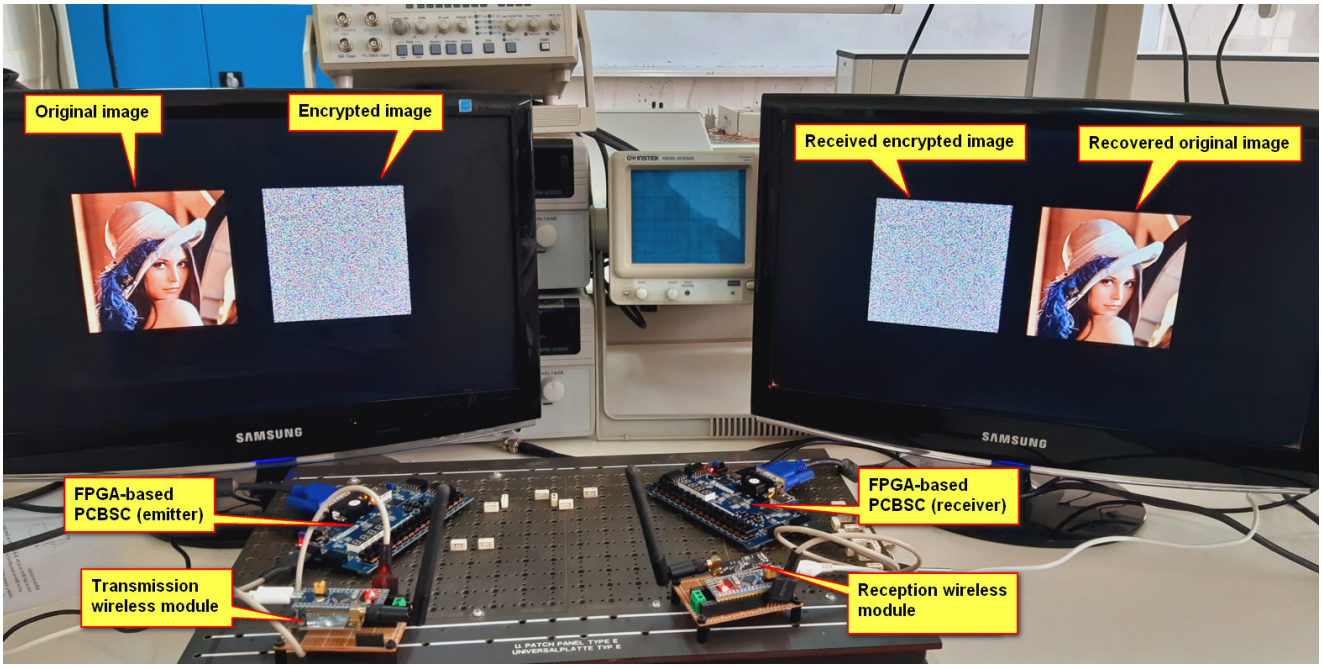

Fig. 26: FPGA-based hardware configuration of the PCBSC (Emitter and Receiver).

Resource	Utilization	Available	Utilization %
LUT	104	32600	0.32
FF	141	65200	0.22
DSP	18	120	15.00
IO	17	106	16.04

Fig. 27: FPGA resources utilization report.

Generally, we can deduce from this table that the PCBSC provides the best results in terms of the differential attack results. The obtained results in terms of the correlation coefficients between adjacent pixels of the encrypted image show that the PCBSC provides the best results, especially over both horizontal and vertical directions.

In comparison to the other proposals, the obtained entropy value indicates that the PCBSC produces encrypted images with a high level of complexity.

The PCBSC has a key of 2^{154} of space, which is larger than the proposals in [60] and [63]. Larger key spaces, such as in [61] and [62], are obtained due to the high arithmetic precision (64 bits) used. However, employing high arithmetic precision in addition to the complicated structures of these proposed encryption schemes has a direct impact on the implementation cost.

Another significant issue concerning the key is that, despite the fact that the keys are large enough, the authors in [60], [61], and [62] have ignored an important issue, namely the weak keys. That is, because the keys are created from the initial conditions and the control parameters, the authors did not indicate the intervals where the control parameters lead to weak keys in order to avoid them, but instead used the entire intervals. This is a major problem that undermines the security of the proposed encryption schemes.

Concerning synchronization and dynamical degradations in digitized chaotic systems, none of these proposals have mentioned this important point, with the exception of the authors in [60], who used a simple technique to enhance the dynamical properties, but their

proposed technique is only applicable to the chaotic system's output, leaving the main properties of the original chaotic system unchanged (including the intervals of the control parameters leading to no chaotic behavior).

9 Conclusion

Much effort has been expended in this work to develop a robust and efficient chaos-based stream cipher. Indeed, despite the large number of recently proposed chaos-based encryption systems, none of them has addressed all of the challenges associated with chaos-based cryptography to our knowledge. Some proposals address certain problems while ignoring others. In the case of our PCBSC, practically all of these issues have been addressed in a single cryptosystem.

The dynamical degradations inherent in digitized chaotic systems are minimized by employing an effective self-perturbation circuit that enhances the statistical properties of the digitized chaotic map, extends the intervals of its control parameters to exhibit chaotic behavior, and significantly increases its cycle length. This was proven by the good results obtained from the statistical and mathematical evaluation tools used for this purpose, which demonstrated that the statistical and randomness properties of the original chaotic map had been greatly enhanced.

The PCBSC has an efficient control parameters generator that generates appropriate parameters, resulting in strong secret keys with a large space (2^{154}). The PCBSC also provides an efficient encryption scheme, which, in addition to the system's keys, can ensure confusion and diffusion properties. The security analysis showed the efficiency of the proposed encryption scheme in addition to the strength of the keys.

The PCBSC takes into account the effect of transmission channel noise on synchronization performance. It provides an efficient synchronization circuit that minimizes the effects of channel noise on the system as much as possible. The synchronization circuit evaluation results showed that the performance of the PCBSC under noisy channels is significantly improved when compared to a configuration employing a traditional synchronization approach.

The PCBSC takes into account the effect of transmission channel noise on the synchronization's performance. It incorporates an effective synchronization circuit that minimizes the effects of channel noise on the systems. The synchronization circuit's evaluation results indicate that its performance under noisy channels is significantly improved when compared to a configuration employing a conventional synchronization approach.

The PCBSC has been evaluated in real-time using an FPGA with a wireless transmission link in which an RGB image was encrypted and recovered in . According to the resources utilization report, the PCBSC has a simple structure and its implementation cost is low.

In general, when designing the PCBSC, we considered almost all of the challenges associated with the use of chaos in cryptography, including the influence of digitization on the dynamical features of chaotic systems, security, synchronization under noisy channels, complexity, and performance. This is a feature that is seldom present in a single design in the most recently proposed chaos-based encryption schemes.

Declarations

- Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.
- The authors declare that they have no conflict of interest.

References

1. Alvarez, Gonzalo, and Shujun Li. "Some basic cryptographic requirements for chaos-based crypto-systems". International journal of bifurcation and chaos 16.08 (2006): 2129-2151, doi: 10.1142/S0218127406015970.
2. Lawande, Q. V., B. R. Ivan, and S. D. Dhodapkar. "Chaos based cryptography: a new approach to secure communications". BARC newsletter 258.258 (2005).
3. Pecora, Louis M., and Thomas L. Carroll. "Synchronization in chaotic systems". Physical review letters 64.8 (1990): 821, doi: 10.1103/PhysRevLett.64.821.
4. Cuomo, Kevin M., Alan V. Oppenheim, and Steven H. Strogatz. "Synchronization of Lorenz-based chaotic circuits with applications to communications". IEEE Transactions on circuits and systems II: Analog and digital signal processing 40.10 (1993): 626-633, doi: 10.1109/82.246163.
5. Pecora, Louis M., et al. "Fundamentals of synchronization in chaotic systems, concepts, and applications". Chaos: An Interdisciplinary Journal of Nonlinear Science 7.4 (1997): 520-543, doi: 10.1063/1.166278.
6. Wu, Chai Wah, and Leon O. Chua. "A simple way to synchronize chaotic systems with applications to secure communication systems". International Journal of Bifurcation and Chaos 3.06 (1993): 1619-1627, doi: 10.1142/S0218127493001288.
7. Cuomo, Kevin M., and Alan V. Oppenheim. "Circuit implementation of synchronized chaos with applications to communications". Physical review letters 71.1 (1993): 65, doi: 10.1103/PhysRevLett.71.65.
8. Cuomo, Kevin M., Alan V. Oppenheim, and Steven H. Isabelle. "Spread spectrum modulation and signal masking using synchronized chaotic systems". Massachusetts Institute of Technology. Research Laboratory of Electronics) ; 570.(1992), http://hdl.handle.net/1721.1/4182.

9. Dedieu, Herve, Michael Peter Kennedy, and Martin Hasler. "Chaos shift keying: modulation and demodulation of a chaotic carrier using self-synchronizing Chua's circuits". IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 40.10 (1993): 634-642, doi: 10.1109/82.246164.
10. Parlitz, Ulrich, et al. "Transmission of digital signals by chaotic synchronization". International Journal of Bifurcation and Chaos 2.04 (1992): 973-977, doi: 10.1142/S0218127492000562.
11. Yang, Tao, and Leon O. Chua. "Secure communication via chaotic parameter modulation". IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 43.9 (1996): 817-819, doi: 10.1109/81.536758.
12. Li, Shujun, Gonzalo Alvarez, and Guanrong Chen. "Breaking a chaos-based secure communication scheme designed by an improved modulation method." Chaos, Solitons & Fractals 25.1 (2005): 109-120, doi: 10.1016/j.chaos.2004.09.077.
13. Alvarez, Gonzalo, et al. "Breaking two secure communication systems based on chaotic masking." IEEE Transactions on Circuits and Systems II: Express Briefs 51.10 (2004): 505-506, doi: 10.1109/TCSII.2004.836047.
14. Alvarez, Gonzalo, and Shujun Li. "Breaking network security based on synchronized chaos." Computer Communications 27.16 (2004): 1679-1681, doi: 10.1016/j.comcom.2004.05.007.
15. JinFeng, Hu, and Guo JingBo. "Breaking a chaotic secure communication scheme." Chaos: An interdisciplinary journal of nonlinear science 18.1 (2008): 013121, doi: 10.1063/1.2885388.
16. Yang, Tao. "Recovery of digital signals from chaotic switching." International journal of circuit theory and applications 23.6 (1995): 611-615, doi: 10.1002/cta.4490230607.
17. Philip, Ninan Sajeeth, and K. Babu Joseph. "Chaos for stream cipher." arXiv preprint (2001): cs/0102012v1.
18. Kohda, Tohru, and Akio Tsuneda. "Chaotic bit sequences for stream cipher cryptography and their correlation functions." Chaotic Circuits for Communication. Vol. 2612. International Society for Optics and Photonics, 1995, doi: 10.1117/12.227907.
19. Li, Ping, et al. "A stream cipher based on a spatiotemporal chaotic system." Chaos, Solitons & Fractals 32.5 (2007): 1867-1876, doi: 10.3182/20060628-3-FR-3903.00061.
20. Lian, Shiguo, et al. "A chaotic stream cipher and the usage in video protection." Chaos, Solitons & Fractals 34.3 (2007): 851-859, doi: 10.1016/j.chaos.2006.03.120.
21. Merah, Lahcene, Adda Ali-Pacha, and Naima Hadj-Said. "Real-time based on synchronized chaotic systems." Nonlinear Dynamics 82.1 (2015): 877-890, doi: 10.1007/s11071-015-2202-2.
22. Xu, Hui, Xiaojun Tong, and Xianwen Meng. "An efficient chaos pseudo-random number generator applied to video encryption." Optik 127.20 (2016): 9305-9319, doi: 10.1016/j.ijleo.2016.07.024.
23. Taha, Mohammed Abu, et al. "Design and efficient implementation of a chaos-based stream cipher." International Journal of Internet Technology and Secured Transactions 7.2 (2017): 89-114, doi: 10.1504/IJITST.2017.087131.
24. Merah, Lahcene, Ali-Pacha Adda, and Hadj-said Naima. "Enhanced chaos-based pseudo random numbers generator." 2018 International Conference on Applied Smart Systems (ICASS). IEEE, 2018, doi: 10.1109/ICASS.2018.8652079.
25. Ayubi, Peyman, Saeed Setayeshi, and Amir Masoud Rahmani. "Deterministic chaos game: a new fractal based pseudo-random number generator and its cryptographic application." Journal of Information Security and Applications 52 (2020): 102472, doi: 10.1016/j.jisa.2020.102472.
26. Tang, Guoping, Xiaofeng Liao, and Yong Chen. "A novel method for designing S-boxes based on chaotic maps." Chaos, Solitons & Fractals 23.2 (2005): 413-419, doi: 10.1016/j.chaos.2004.04.023.
27. Khan, Majid, et al. "An efficient method for the construction of block cipher with multi-chaotic systems." Nonlinear Dynamics 71.3 (2013): 489-492, doi: 10.1007/s11071-012-0675-9.
28. Yi, Longteng, et al. "A novel block encryption algorithm based on chaotic S-box for wireless sensor network." IEEE Access 7 (2019): 53079-53090, 10.1109/ACCESS.2019.2911395.
29. Çavuşoğlu, Ünal, et al. "A novel approach for strong S-Box generation algorithm design based on chaotic scaled Zhongtang system." Nonlinear dynamics 87.2 (2017): 1081-1094.
30. Farah, Tarek, Rhouma Rhouma, and Safya Belghith. "A novel method for designing S-box based on chaotic map and teaching–learning-based optimization." Nonlinear dynamics 88.2 (2017): 1059-1074, doi: 10.1016/j.physleta.2012.01.009.
31. Hussain, Iqtadar, et al. "Construction of s-box based on chaotic map and algebraic structures." Symmetry 11.3 (2019): 351, doi: 10.3390/sym11030351.
32. Wang, Xingyuan, et al. "Chaotic encryption algorithm based on alternant of stream cipher and block cipher." Nonlinear Dynamics 63.4 (2011): 587-597, doi: 10.1007/s11071-010-9821-4.
33. Ren, Haijun, et al. "A novel method for one-way hash function construction based on spatiotemporal chaos." Chaos, Solitons & Fractals 42.4 (2009): 2014-2022, doi: 10.1016/j.chaos.2009.03.168.
34. Kanso, Ali, Hamdi Yahyaoui, and Mohammed Almulla. "Keyed hash function based on a chaotic map." Information Sciences 186.1 (2012): 249-264, doi: 10.1016/j.ins.2011.09.008.
35. Ahmad, Musheer, et al. "A simple secure hash function scheme using multiple chaotic maps." 3D Research 8.2 (2017): 13, doi: 10.1109/TCSII.2005.848992.
36. Todorova, Mihaela, et al. "SHAH: Hash function based on irregularly decimated chaotic map." arXiv preprint arXiv:1808.01956 (2018), doi: 10.24425/123546.
37. Yoon, Ji Won, and Hyoungshick Kim. "An image encryption scheme with a pseudorandom permutation based on chaotic maps." Communications in Nonlinear Science and Numerical Simulation 15.12 (2010): 3998-4006, doi: 10.1016/j.cnsns.2010.01.041.
38. Fu, Chong, et al. "An efficient and secure medical image protection scheme based on chaotic maps." Computers in biology and medicine 43.8 (2013): 1000-1010, doi: 10.1016/j.compbiomed.2013.05.005.
39. Sravanthi, Dasari, et al. "Simple permutation and diffusion operation based image encryption using various one-dimensional chaotic maps: a comparative analysis on security." Advances in Data and Information Sciences. Springer, Singapore, 2020. 81-96, doi: 10.1007/978-981-15-0694-9_9.
40. S. Li, G. Chen, and X. Mou, "On the dynamical degradation of digital piecewise linear chaotic maps," Int. J. Bifurcation Chaos, vol. 15, no. 10, pp. 3119–3151, Oct. 2005, doi: 10.1142/S0218127405014052.

41. Flores-Vergara, A., García-Guerrero, E. E., Inzunza-González, E., López-Bonilla, O. R., Rodríguez-Orozco, E., Cárdenas-Valdez, J. R., & Tlelo-Cuautle, E. (2019). "Implementing a chaotic cryptosystem in a 64-bit embedded system by using multiple-precision arithmetic." Nonlinear Dynamics, 96(1), 497-516, doi: 10.1007/s11071-019-04802-3.
42. Wu, Qiujie, et al. "Research on cascading high-dimensional isomorphic chaotic maps." Cognitive Neurodynamics 15.1 (2020): 157-167, doi: 10.1007/s11571-020-09583-9.
43. L. Merah, A. Ali-Pacha, N. Hadj-Said, and M. Belkacem, "New and efficient method for extending cycle length of digital chaotic systems." Iranian J. Sci. Technol., Trans. Electr. Eng., vol. 43, no. S1, pp. 259–268, Jul. 2019, doi: 10.1007/s40998-018-0122-0.
44. Pei, Chao, et al. "Trade-off of security and performance of lightweight block ciphers in Industrial Wireless Sensor Networks." EURASIP Journal on Wireless Communications and Networking 2018.1 (2018): 1-18, doi: 10.1186/s13638-018-1121-6.
45. Shannon, Claude E. "A mathematical theory of cryptography." Mathematical Theory of Cryptography (1945).
46. Merah, L., Lorenz, P., & Adda, A. P. (2021). A New and Efficient Scheme for Improving the Digitized Chaotic Systems From Dynamical Degradation. IEEE Access, 9, 88997-89008, doi: 10.1109/ACCESS.2021.3089913.
47. Alawida, Moatsum, Azman Samsudin, and Je Sen Teh. "Digital cosine chaotic map for cryptographic applications." IEEE Access 7 (2019): 150609-150622, doi: 10.1109/ACCESS.2019.2947561.
48. Pincus, Steven M. "Approximate entropy as a measure of system complexity." Proceedings of the National Academy of Sciences 88.6 (1991): 2297-2301, doi: 10.1073/pnas.88.6.2297.
49. Delgado-Bonal, Alfonso, and Alexander Marshak. "Approximate entropy and sample entropy: A comprehensive tutorial." Entropy 21.6 (2019): 541, doi: 10.3390/e21060541.
50. Henry, Miguel. "Permutation Entropy." Aptech, Data Analytics Blog (2020). Available : https://www.aptech.com/blog/permutation-entropy.
51. Henry, Miguel, and George Judge. "Permutation entropy and information recovery in nonlinear dynamic economic time series." Econometrics 7.1 (2019): 10, doi: 10.3390/econometrics7010010
52. Richman, Joshua S., and J. Randall Moorman. "Physiological time-series analysis using approximate entropy and sample entropy." American Journal of Physiology-Heart and Circulatory Physiology (2000), doi :10.1152/ajpheart.2000.278.6.h2039
53. Andrew L. Rukhin, et al. "A statistical test suite for random and pseudorandom number generators for cryptographic applications.", NIST SP-800-22 Rev 1a, url : http://www.nist.gov/manuscript-publication-search.cfm?pub_id=151222
54. "How secure is AES against brute force attacks?". EE Times. Retrieved 01/13/2022, url : https://www.eetimes.com/how-secure-is-aes-against-brute-force-attacks/.
55. Zhang, Li-bo, et al. "Cryptanalysis and improvement of an efficient and secure medical image protection scheme." Mathematical Problems in Engineering 2015 (2015), doi : 10.1155/2015/913476
56. Annadurai, S. Fundamentals of digital image processing. Pearson Education India, 2007.
57. Biham, Eli, and Adi Shamir. "Differential cryptanalysis of DES-like cryptosystems." Journal of CRYPTOLOGY 4.1 (1991): 3-72, doi : 10.1007/BF00630563
58. Chen, Guanrong, Yaobin Mao, and Charles K. Chui. "A symmetric image encryption scheme based on 3D chaotic cat maps." Chaos, Solitons & Fractals 21.3 (2004): 749-761, doi : 10.1016/j.chaos.2003.12.022
59. Xilinx INC, "Model Composer and System Generator User Guide" UG1483 (v2020.2) November 18, 2020.
60. García-Guerrero, E. E., et al. "Randomness improvement of chaotic maps for image encryption in a wireless communication scheme using PIC-microcontroller via Zigbee channels." Chaos, Solitons & Fractals 133 (2020): 109646, doi : 10.1016/j.chaos.2020.109646
61. Hua, Zhongyun, Yicong Zhou, and Hejiao Huang. "Cosine-transform-based chaotic system for image encryption." Information Sciences 480 (2019): 403-419, doi : 10.1016/j.ins.2018.12.048
62. Chai, Xiuli, et al. "An efficient chaos-based image compression and encryption scheme using block compressive sensing and elementary cellular automata." Neural Computing and Applications 32.9 (2020): 4961-4988, doi : 10.1007/s00521-018-3913-3
63. Zhu, Liya, et al. "A novel image encryption scheme based on nonuniform sampling in block compressive sensing." IEEE Access 7 (2019): 22161-22174,10.1109/ACCESS.2019.2897721