Jestr

Journal of Engineering Science and Technology Review 6 (4) (2013) 90-103
Special Issue on Recent Advances in Nonlinear Circuits: Theory and Applications

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

Research Article

Secure Image Encryption Based On a Chua Chaotic Noise Generator

A. S. Andreatos[*, 1], and A. P. Leros[1, 2]

[1]*Div. of Computer Engineering and Information Science Hellenic Air Force Academy, Dekeleia Air Force Base, Dekeleia, Attica, TGA-1010, Greece*

[2]*Dpt. of Automation, School of Technological Applications, Technological Educational Institute of Chalkis, 34400, Psachna, Evia, Greece*

Received 3 August 2013; Revised 1 October 2013; Accepted 22 October 2013

Abstract

This paper presents a secure image cryptography telecom system based on a Chua's circuit chaotic noise generator. A chaotic system based on synchronised Master–Slave Chua's circuits has been used as a chaotic true random number generator (CTRNG). Chaotic systems present unpredictable and complex behaviour. This characteristic, together with the dependence on the initial conditions as well as the tolerance of the circuit components, make CTRNGs ideal for cryptography. In the proposed system, the transmitter mixes an input image with chaotic noise produced by a CTRNG. Using thresholding techniques, the chaotic signal is converted to a true random bit sequence. The receiver must be able to reproduce exactly the same chaotic noise in order to subtract it from the received signal. This becomes possible with synchronisation between the two Chua's circuits: through the use of specific techniques, the trajectory of the Slave chaotic system can be bound to that of the Master circuit producing (almost) identical behaviour. Additional blocks have been used in order to make the system highly parameterisable and robust against common attacks. The whole system is simulated in Matlab. Simulation results demonstrate satisfactory performance, as well as, robustness against cryptanalysis. The system works with both greyscale and colour jpg images.

Keywords: Image encryption, cryptography, Chua's circuit, chaotic true random number generator, Master-Slave configuration, synchronisation, secure communications, black image attack.

1. Chua's Circuit Presentation and Applications

Chua's circuit is a simple circuit consisting of five components, one of which is non-linear and presents negative conductivity [1]. Chua's circuit is a well-known very simple circuit producing chaotic behaviour. Chaotic systems for long periods of time exhibit unpredictable and complex behaviour. Their states move around some attractors and pass from one trajectory to another in unpredictable time, without following some pattern. Therefore, Chua's circuit has found many applications in physics, communication and control, mechanics, as well as, chemistry, economics, medicine, etc. [2].

Chua's circuit has also been used as a chaotic noise generator. Because of this property, it has found many applications in cryptography and steganography. Cryptography and steganography Telecom systems use two Chua's circuits, one at the transmitter and one at the receiver [3, 4], synchronised in a Master-Slave configuration [5]. In that work, the two Chua's circuits have identical topology but different component values. A continuous linear controller has been used to synchronise the two circuits. The concept is the following: The slave Chua circuit on the receiver side produces a chaotic signal which is driven by the controller to the corresponding chaotic signal produced by the Master Chua's circuit at the transmitter side [5].

The chaotic true random number generator (CTRNG) engine of the proposed system is based on a Chua's circuit. Chua's (standard) circuit is a simple non-linear circuit, producing chaotic behaviour for a specific set of component values; in particular, its behaviour is characterised by a double-scroll chaotic attractor [6, 7]. The circuit was proposed by Prof. Leon O. Chua, in order to demonstrate chaos in an actual physical model and to prove that the Lorenz double-scroll attractor is chaotic [1, 6]. Chua's circuit suits the study of chaos well, because one can precisely control its parameters and observe the results on an oscilloscope. Therefore, it has found many applications [1, 7].

In order for a dynamic system to be considered chaotic, the following conditions must be satisfied [8]:
1) the system should be topologically mixing;
2) its attractors must be dense;
3) it should be very sensitive to the initial conditions.

Chua's circuit meets all these requirements. Fig. 1 represents the standard Chua's circuit; it consists of plain passive components plus a special, non-linear device (N_R) with negative resistance g_{NR} (Fig. 1b). The latter is implemented with active components. During the three decades following its invention (1983), Chua's circuit

* E-mail address: aandreatos@gmail.com
ISSN: 1791-2377

became popular because it is easy to construct, and many people have built the circuit using off-the-shelf electronic components. In fact, the circuit can be constructed using only resistors, capacitors, inductors, diodes and op-amps [1, 7, 9].

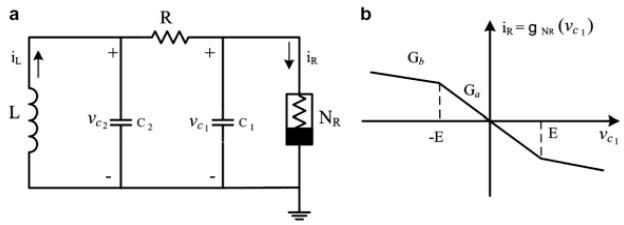


Fig. 1. (a) Standard Chua's circuit; (b) v–i characteristic of the non-linear device. Source: [9].

In Fig. 1, V_{C1} and V_{C2} denote the voltages across the capacitors C_1 and C_2 respectively, i_L is the current through the inductor L, and $g_{NR}(V_{C1})$ is the non-linear function which defines the V–i characteristic of the nonlinear device, represented by the piecewise-linear function of Fig. 1b. We have also assumed that the inductor L has a parasitic resistance R_0. By solving the above circuit we get the following differential equations (1-3). Equation (4) describes the nonlinear device, which in fact is piecewise-linear:

$$C_1 \frac{dV_{C1}}{dt} = \frac{1}{R}(V_{C2} - V_{C1}) - g_{NR}(V_{C1}) \quad (1)$$

$$C_2 \frac{dV_{C2}}{dt} = \frac{1}{R}(V_{C1} - V_{C2}) + i_L \quad (2)$$

$$L \frac{di_L}{dt} = -V_{C2} - R_0 i_L \quad (3)$$

$$g_{NR}(V_{C1}) = G_b V_{C1} + \frac{1}{2}(G_a - G_b)(|V_{C1} + E| - |V_{C1} - E|) \quad (4)$$

V_{C1}, V_{C2} and i_L are the state variables of the chaotic system; in our code they are represented by three discrete-time variables: $X_1(t)$, $X_2(t)$ and $X_3(t)$. For t=0 we have the initial conditions. The continuous differential equations of $X_1(t)$, $X_2(t)$ and $X_3(t)$ which result from equations 1-3 are solved using the ode45 Matlab toolbox.

Chua's circuit response is very sensitive to the initial conditions, values and tolerances of its components. The circuit produces completely different output waveforms for changes in the initial conditions or variations in the component values [4, 7, 9]. The parameter values that have been used here are shown in Tab. 1:

Table 1. Parameter values of the Chua's circuit.

C_1	C_2	L	R_0	R	G_a	G_b	E
10 nF	100 nF	22 mH	22 Ω	1720 Ω	-0.7 mS	-0.5 mS	1.8 V

In the current work, the initial conditions of the Master and Slave Chua circuits are as follows: $\{V_{C1} = 0\ V, V_{C2} = 1\ V, i_L = 0\ A\}$ and $\{V_{C1}' = 0\ V, V_{C2}' = 1.1\ V, i_L' = 0\ A\}$ respectively [5].

Fig. 2 shows the double-scroll chaotic attractor poduced by our Chua's circuit. As we can see, it meets the second aforementioned requirement.

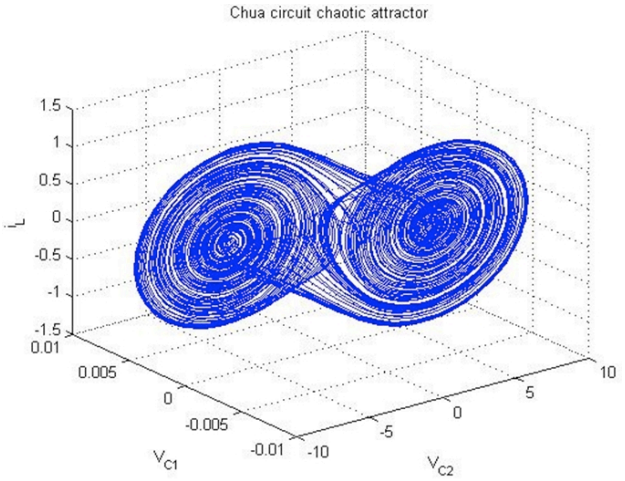


Fig. 2. Double-scroll chaotic attractor of Chua's circuit.

This paper is organised as follows: in subsection 1.2 there is a brief literature review; section 2 presents the proposed system; section 3 discusses its security features; finally, we conclude in section 4.

1.1. Literature review

In the literature there is a number of previous works closely related to ours.

Yalçın et al. [10] have demonstrated a true random bit generation from a double-scroll attractor, based on de-skewing techniques. In this way the extraction of an unbiased, truly random bit sequence out of a Chua's circuit was made possible.

Volos et al. [11] have presented an image encryption process based on the chaotic true random bit generation technique of Yalçın et al., using a proprietary circuit instead of Chua's circuit. That system was The proposed scheme for encryption and decryption of greyscale images was based on the XOR function [12] and it was simulated in Matlab.

Andreatos and Leros [13] have presented a colour image cryptosystem based on a Chua Circuit Chaotic Noise Generator, using addition (instead of XOR) to mix the signal (image) with chaotic noise, simulated in Matlab.

Leros and Andreatos [5] have also demonstrated a Steganography Telecom System based on a Chua's Circuit Chaotic Noise Generator, simulated in Simulink. Again, addition instead of XOR was used to mix the signal with chaotic noise.

Volos [4] presented a greyscale image encryption system based on the interaction between two mutually coupled Chua's circuits based on the XOR function. Volos (2013) presented also a greyscale image encryption system based on a chaotic true random bit generator. However, this generator is based on the coexistence of two different synchronisation phenomena (chaotic synchronisation and inverse π-lag synchronisation) which are observed in the case of two mutually coupled nonlinear circuits.

In this paper we present an image cryptosystem based on a Chua Circuit Chaotic Noise Generator, using the XOR function. A modified version of the True Random Bit Generation engine proposed by Yalçın et al. [10], using a single threshold, has been used. To make the system more

robust, a pre-processing stage plus two more parametric XOR stages have been added. The proposed system was simulated in Matlab. Simulation results showed robustness and resistance against common attacks. The system works with both greyscale and colour images.

It has been proven that under specific conditions coupled chaotic systems can be broken [14]. Therefore, in this work we adopted the straightforward method for generating true random bit sequences out of continuous-time chaotic systems, introduced by Yalçın et al. [10]. In particular, a chaotic signal is sampled using proper thresholds to produce unbiased bits. This method has also been used successfully by Volos et al. [11]. The same principle has also been used by Volos [4].

Statistical tests performed by Yalçın et al. [10] and Volos et al. [11] confirm the success and security of this technique. In our simulation the CTRNG produced 411995 zeroes (50.02325 %) and 411560 ones (49.97675 %).

2. System Description

A simplified block diagram of the proposed system is shown in Fig. 3. The system consists of the Transmitter and the Receiver. Next, system operation will be presented.

Generally speaking, the Transmitter gets an uncompressed image for encryption and transmission; this image is somehow mixed with chaotic noise N(t) produced by the Master Chua's circuit. The details will be given below. The encrypted image is then transmitted through the (insecure) channel. At the Receiver an identical chaotic noise N(t) must be generated in order to be removed from the received signal (using the inverse process) and give back the original image. For this reason, a similar (Slave) Chua's circuit is needed at the Receiver. In order for the Slave Chua circuit to produce the exactly the same chaotic noise, the two Chua's circuits must be closely synchronised.

During the last two decades, the chaotic synchronisation problem has received a tremendous interest [15-23]. Therefore, the state of the Master Chua's circuit must be sent to the Receiver in real time. A continuous linear controller is being used to synchronise the two Chua circuits [5]. For the synchronisation between Master and Slave Chua's circuits, Pyragas' continuous control method has been used [24]. This method was chosen because it was relatively easy to implement. A simplified block diagram of the synchronisation circuit is shown in Fig. 4 [5].

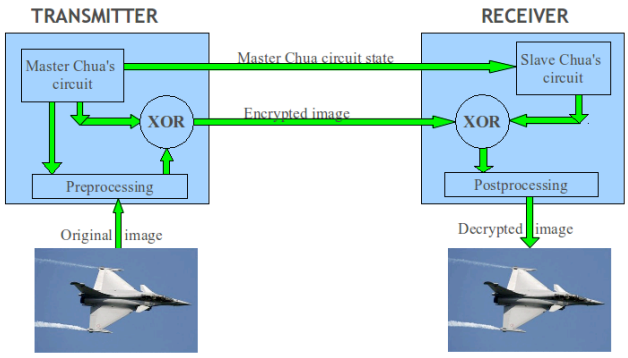


Fig. 3. Simplified block diagram of the proposed system.

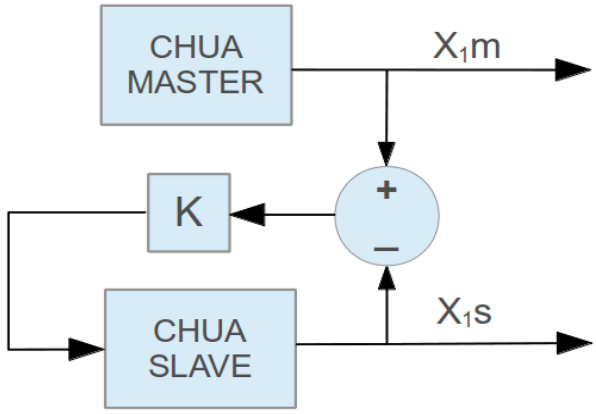


Fig. 4. Master-Slave Synchronisation circuit block diagram.

As we can see in Fig. 4, the difference of X_1 variables of Master (X_1m) and Slave (X_1s) circuits is driving the linear controller which then regulates the Slave circuit in order to eliminate the difference [5].

In order to enhance the security of our system, three stages of processing have been used (Fig. 5), described below.

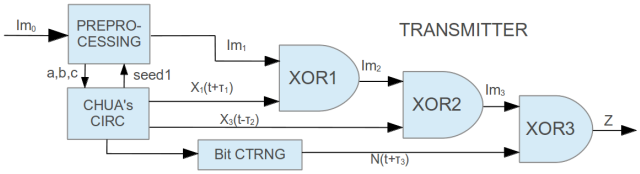


Fig. 5. Detailed block diagram of the encryption unit of the Transmitter.

2.1. Pre-processing of the clear image

The purpose of this stage is to alter the characteristic of the clear (input) image and make the system parametric and robust. It is based on permutation, a commonly used technique in modern encryption schemes due to its speed [12, 25:ch.8, 26:ch.8, 27]. In our system, pre-processing includes the following stages:

a) Reversing the order of pixels in selected rows and columns; in general, the order number of rows to be changed may be produced by a random number function of the Chua's circuit, but in the current implementation we have just applied it on even rows and columns.

b) Reordering (permuting) the rows or/and columns on the output of the previous stage. This might equally well be the result of another random number function fed by the Chua's circuit (seed1) and/or other factors such as date functions, nonces[1] and one-time pads, to assure the freshness of the cipher images and cancel playback attacks [25-27]. This feature is very important in Pseudo-Random Number Generator (PRNG) implementations. In the current simulation we have just used a software PRNG for simplicity.

c) More operations, such as change of specific pixel values, may also be applied at this stage in order to increase robustness. In general, the user would be able to select from a predefined set of pre-processing options, unknown to intruders. The selected option will be communicated through a code which will make part of the key. Let's call the output

1. A nonce is a number that will be used only once in a secret communication using cryptography.

of this stage Im_1. The pre-processing unit also extracts the dimensions of the input image (a, b, c) and feeds them to the CTRNG engine. Where: a is the image height or number of rows, b is the image width or number of columns and c equals 1 for greyscale images or 3 for colour images.

2.2. First stage of processing

The Chua's circuit produces a string of length 2 or 3 orders of magnitude longer than the image length (a*b). In order to make the system more robust, we have implemented in this stage the well-known method of Initialisation Vectors (IV) used in Cipher Block Chaining [25:ch.8]: the transmitter generates a matrix of random numbers [0-255] of the input image dimensions; this is XORed by the input image (pre-processing), pixel-by-pixel, before the main cryptography process. To increase robustness, an independent TRNG source could be used (instead of X_1); for instance, a second Chua's circuit.

Variables τ_1, τ_2 and τ_3 are random numbers produced by a random number generator fed by the Chua's circuit (seed1) and/or other factors such as date functions, to make the system multi-parametric, that is, to increase key length.

In the current implementation, a subset of consecutive values from the Chua variable X_1, of length equal to the image length, is randomly selected ($X_1(t+\tau_1)$). This string is then converted to pixel integers (in the range 0-255) and XORed pixel-by-pixel with Im_1 to give Im_2.

Next, Im_2 is XORed pixel-by-pixel with another string of length a*b selected from a random place of the Chua variable X_3, $X_3(t-\tau_2)$, to produce Im_3. If the circuit has memory, the string $X_3(\tau_2-t)$ could be used instead (at the expense of additional computing time), since it was observed that it produces better results and increases security. This could also come from the second Chua's circuit.

2.3. Second stage of processing

During this stage the output of the previous stage, i.e. Im_3, is XORed pixel-by-pixel with a substring of the Chua Noise, $N(t+\tau_3)$, of length equal to the image length, randomly selected (τ_3 is a random number). The main difference from the previous stage is that instead of using a chaotic variable (such as X_1), a true random bit sequence N(t) is used.

The Chua Noise is produced by an engine (Bit CTRNG in Fig. 5) similar to that presented in the bibliography [10, 11] which produces an unbiased, random bit sequence out of biased input sequences. In fact, histograms of the chaotic variables X_1, X_2, X_3 are not uniform but rather biased; this is also evident from the attractor (Fig. 2). Therefore it was necessary to apply post-processing on a function of the chaotic signals, to eliminate the bias at the output of the bit generator, using thresholding [10, 11]. The result of this stage is the output of the Transmitter, Z (Fig. 6b) to be fed into the channel for transmission.

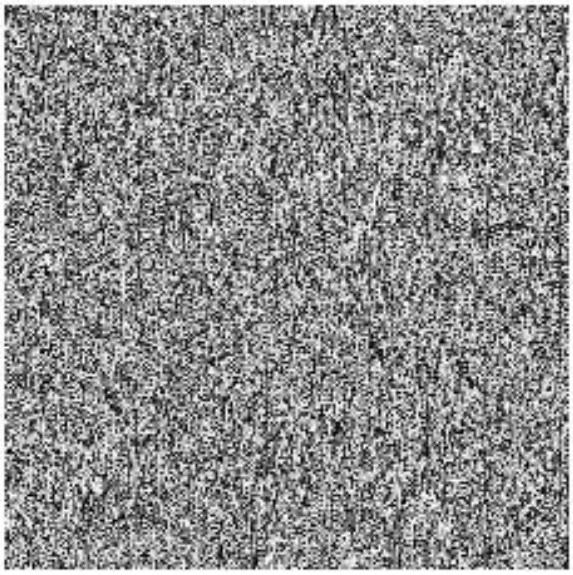
Obviously, the inverse processing takes place at the Receiver, with reverse order, i.e., decryption and post-processing.

2.4. Results

Next, some results will be presented to demonstrate the operation of our system. Standard test images will be used in order to facilitate the understanding of the results.

Fig. 6 shows the encrypted image for greyscale input image 'Lena.jpg' (225x225 pixels) while Fig. 7 demonstrates another example of greyscale image processing using test image 'baboon.jpg' (100x100 pixels).

(a) Original image 'Lena'



(b) Encrypted image 'Lena'

(c) Decrypted image 'Lena'

Fig. 6. Original, encrypted and decrypted greyscale image 'Lena'.

(a) Original image 'baboon'

(b) Encrypted image 'baboon'

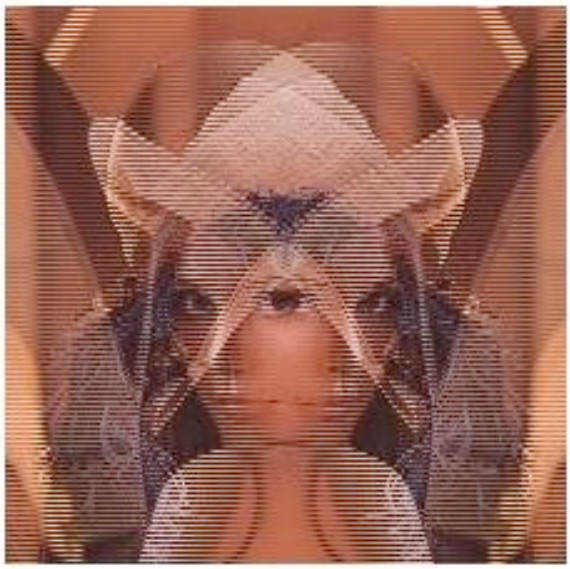
(c) Decrypted image 'baboon'

Fig. 7. Original, encrypted and decrypted image 'baboon'.

Colour images are treated as three-dimensional matrices where the third dimension represents the colour component (R, G, B) and takes the values 1, 2, 3.

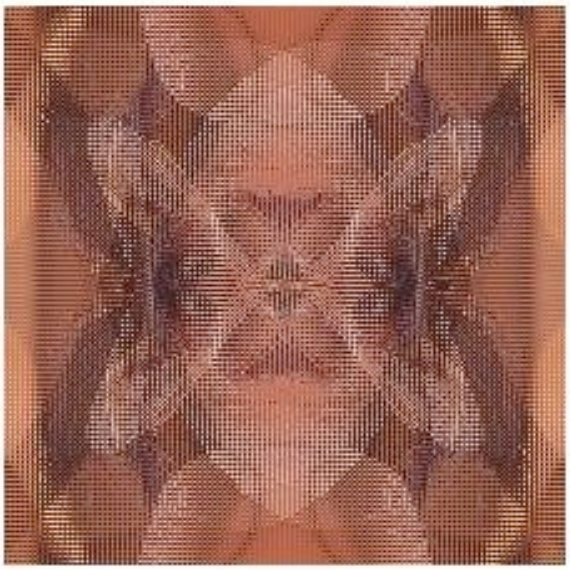
Fig.8 demonstrates the result of the various stages of pre-processing and encryption of the colour image 'Lena_c' at the Transmitter (225x225 pixels).

(a) Input image

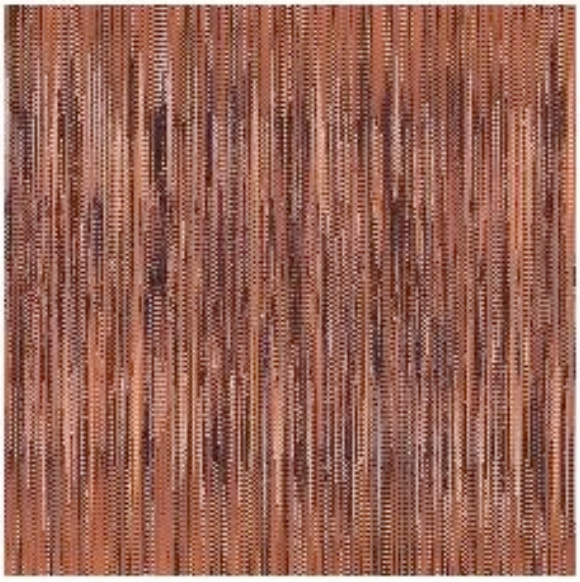


(b) After inverting even rows

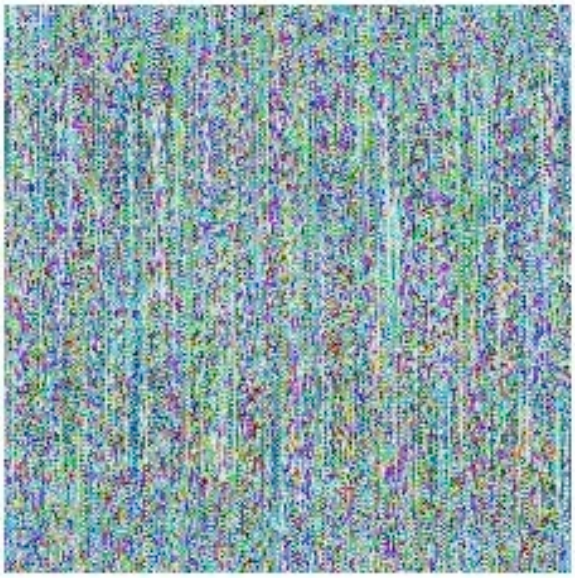
(continued)



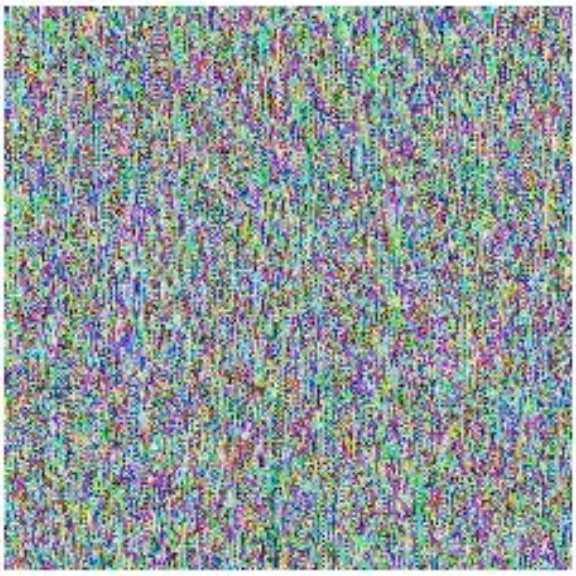
(c) After inverting even columns



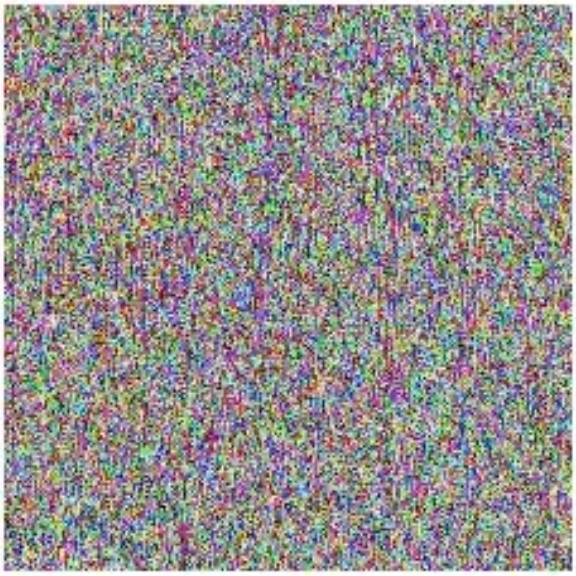
(d) After permuting columns



(e) After XOR1



(f) After XOR2

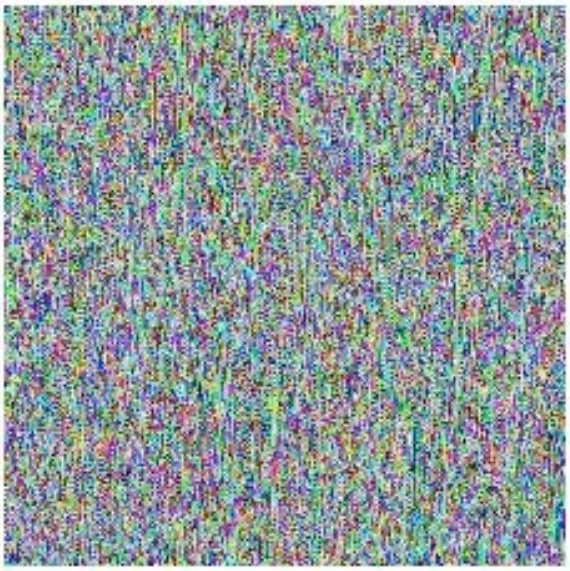


(g) After XOR3

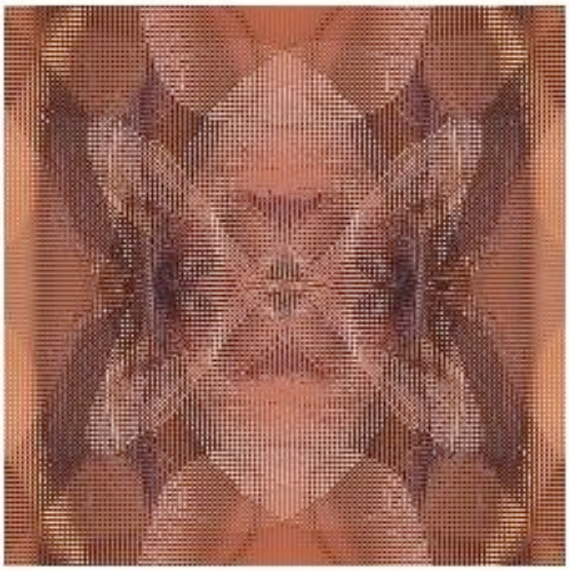
(h) Transmission in channel

Fig. 8. Output of the various stages of pre-processing and encryption at the Transmitter.

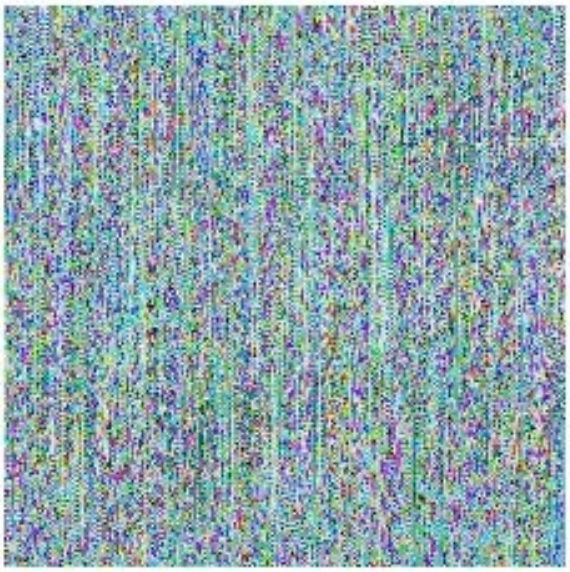
Fig. 9 demonstrates the result of the various stages of decryption and post-processing of the image 'Lena_c' through the various stages of the Receiver.



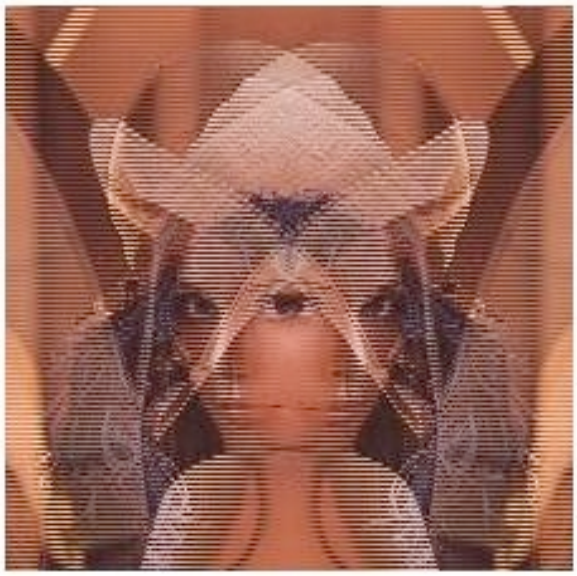
(a) After XOR3



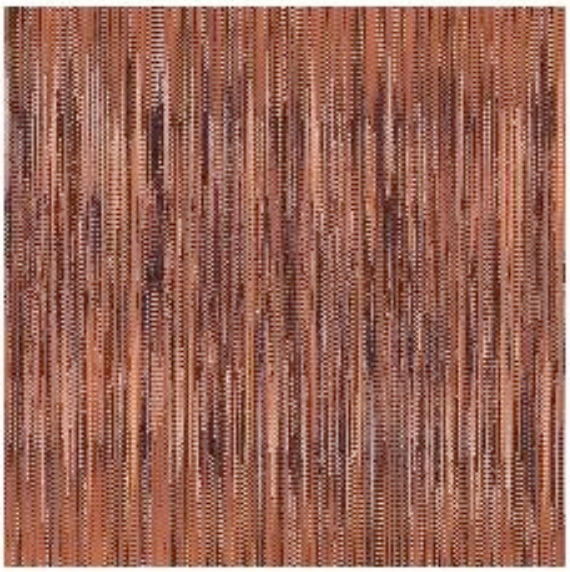
(d) After unpermuting columns



(b) After XOR2



(e) After uninverting even columns



(c) After XOR1

(f) After uninverting even rows = decrypted image

Fig. 9. Output of the various stages of decryption and post-processing at the Receiver.

It is clear that the signal preserves the dimensions of the input image (a*b) throughout the whole processing.

2.5. Key space

The key of the Stegosystem is the exact values of the following set of parameters:

a) The values of the Chua's circuit components (C_1, C_2, L, R_0, R) and parameters (G_a, G_b, E). If we assign 11 bits to each parameter, we get 9*11= 99 bits. We could have used (slightly) different component values at the Slave Chua's circuit, to increase the key length.

b) For the initial conditions of the Master and Slave Chua's circuits, i.e. 6 values * 11 bit each, we need 66 bits.

c) The parameters of the pre-processing unit. For the selection of a pre-processing algorithm of the first stage out of a set of 16 or more predefined options we need at least 4-5 bits. For the selection of a permutation algorithm out of a set of 16 or more predefined options we need at least 4-5 bits. In PRNG implementations we might also need some more bits for the seed of the PRNG permutation algorithm, say 10 bits. For τ_1, τ_2 and τ_3: 3 values * 11 bit each = 33 bits.

Thus, totally we need at least: 99 + 66 + 10 + 33 = 208 bits.

Software (PRNG) implementations will need some more bits for algorithm selection and seed.

The remaining bits (to reach the key length of 256) may be used to add some extra redundancy (e.g. nonces, time/date functions, even a check sum), or for future extensions, in order to satisfy the relevant cryptographic principle [26: par. 8.1.5]. A key length of 256 bits is considered as adequate today [12].

Should a second Chua's circuit be used to increase robustness, the key length would increase to 512 bits.

3. Security Analysis

3.1. Common attacks considered

The most common attacks considered in Cryptanalysis are [12:p.18, 25:p.693, 27:p.24]:

cipher image only attack,
known image attack and
chosen image attack.

3.1.1. Cipher image-only attack

In some cases, a potential intruder (traditionally called "Trudy") may have access only to the intercepted cipher image, with no certain information about the contents of the plain image. Statistical analysis can help in a cipher image-only attack.

3.1.2. Known plain image attack

If the intruder somehow knew for sure that some specific pixel pattern appeared in the cipher image, then she could have determined the (plain image, cipher image) pairings. When an intruder knows some of the (plain image, cipher image) pairs, we refer to this as a known plain image attack on the encryption scheme.

3.1.3. Chosen plain image attack

In a chosen plain image attack, the intruder is able to choose the plain (input) image and obtain its corresponding cipher image. For example, for simple text encryption algorithms, if Trudy could get Alice (traditionally, the user at the transmitter) to send the message "The quick brown fox jumps over the lazy dog", she could completely break the encryption scheme. Sophisticated encryption techniques should not be broken by chosen-plain image attacks.

3.2. Resistance to common attacks

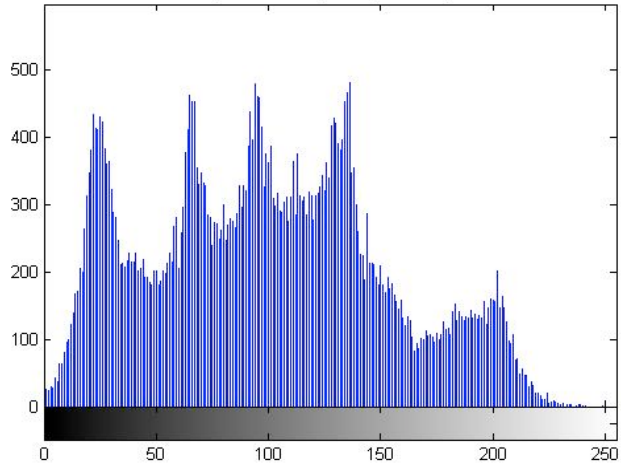
In this paragraph the resistance of the proposed system to common attacks will be examined.

3.2.1. Resistance to cipher image-only attacks

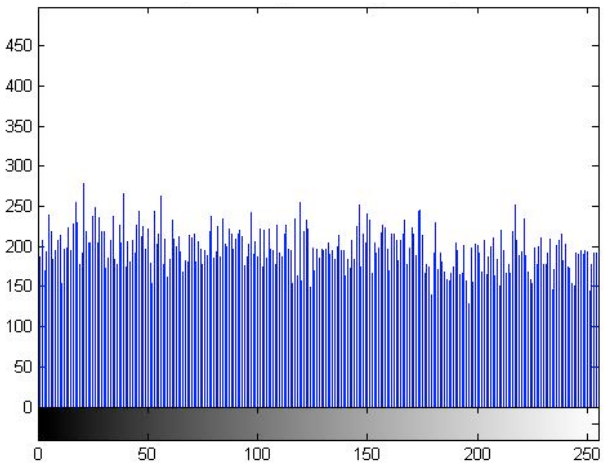
The proposed system uses an engine similar to those presented in the bibliography [4, 10]. Indeed, a binary sequence produced by a Chua TRNG is used to encrypt the plain image via the XOR operator. It has been proven [4, 10] that this scheme presents resistance to statistical and brute-force attacks. Some preliminary results are presented below.

Histogram analysis

The histograms of two plain images and the corresponding encrypted images are shown below. Fig. 10 shows the encrypted image for the typical greyscale test image 'Lena.jpg' (225x225 pixels).



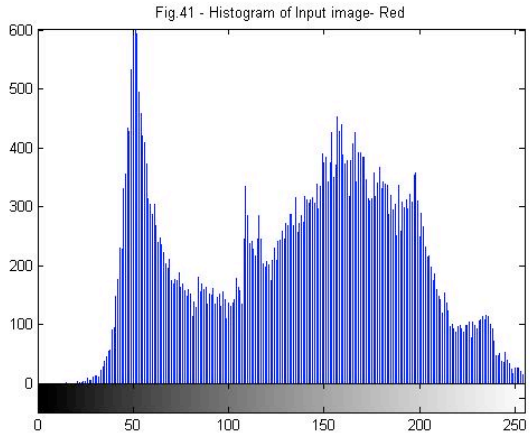
(a) Histogram of plain greyscale image 'Lena'



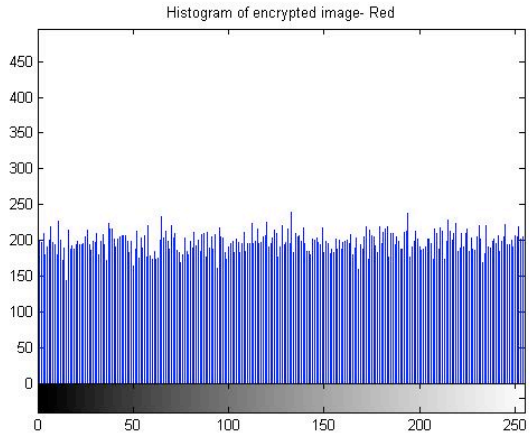
(b) Histogram of encrypted image 'Lena'

Fig. 10. Histograms of greyscale plain image and corresponding encrypted image 'Lena'.

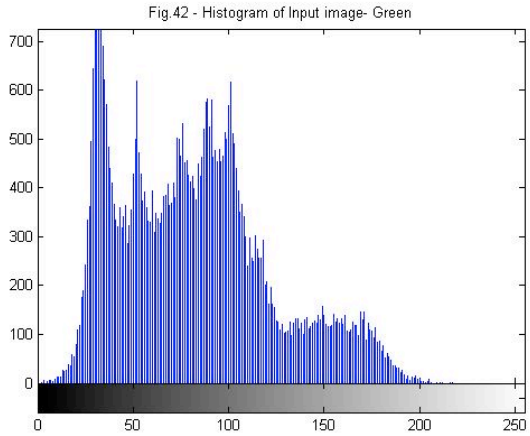
Figure 11 demonstrates the histograms of the original and encrypted images 'Lena_c' for each of the three colour components.



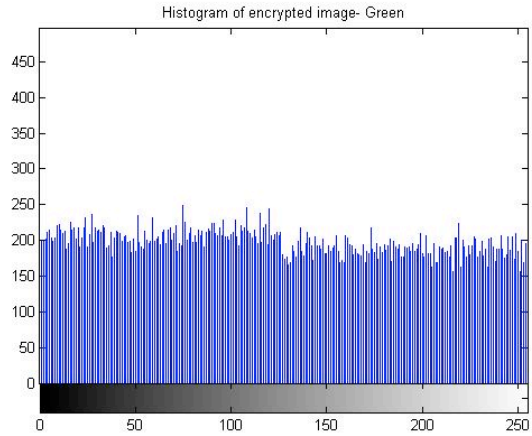
(a) Histogram of plain image 'Lena_c' – Red



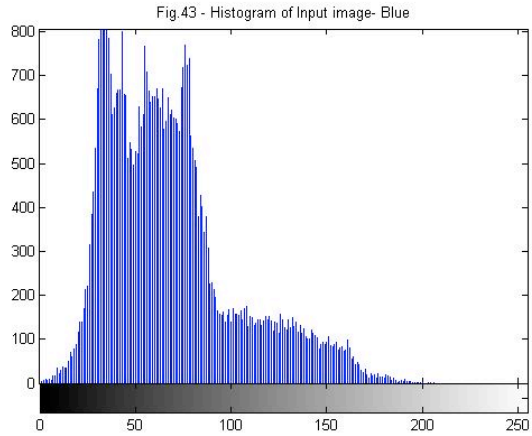
(b) Histogram of encrypted image 'Lena_c' – Red



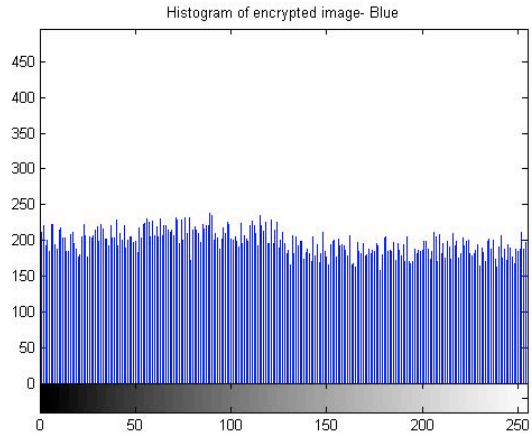
(c) Histogram of plain image 'Lena_c' – Green



(d) Histogram of encrypted image 'Lena_c' – Green



(e) Histogram of plain image 'Lena_c' – Blue



(f) Histogram of encrypted image 'Lena_c' – Blue

Fig. 11. Histograms of the plain (a, c, e) and encrypted (b, c, f) colour image 'Lena_c'.

As we can see, the histograms of the encrypted images are almost uniformly distributed hence similar, even though the histograms of the plain images differ significantly. This testifies the robustness of the proposed encryption system over statistical attacks. In general, the proposed system always produces an almost flat histogram at the output, cancelling this type of attack.

Autocorrelation analysis

Normal (meaningful) images present strong autocorrelation[2] between adjacent pixels, hence rows and columns. An image encryption system should drastically eliminate input image autocorrelation by producing an encrypted image with low autocorrelation at the output (close to 0).

To compare the autocorrelation of a plain and encrypted image we have calculated the correlation coefficient γ of each pair of pixels horizontally, vertically, diagonally and anti-diagonally by using the following formulae:

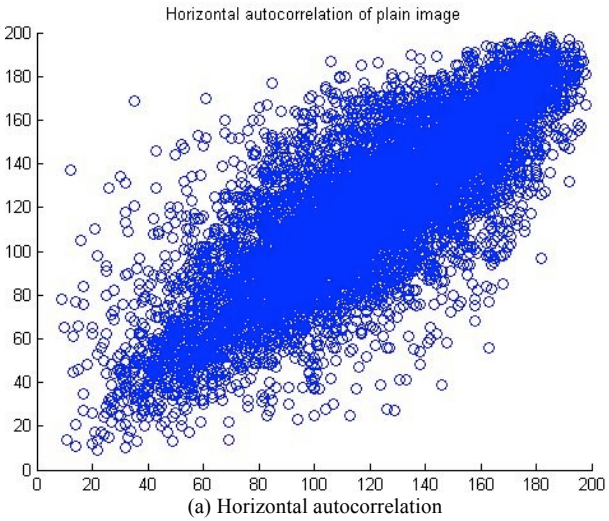
$$\gamma(x,y)=\frac{\mathrm{cov}(x,y)}{\sigma_x\sigma_y} \tag{5}$$

where:

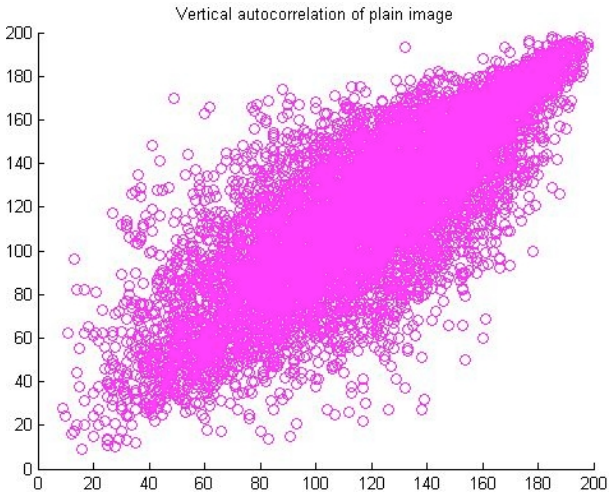
$$\mathrm{cov}(x,y)=\frac{1}{N}\sum_{i=1}^{N}\left[x_i-E(x)\right]\left[y_i-E(y)\right] \tag{6}$$

In eq. (6) "E" is the expected value operator and σ_x^2 represents the variance of variable x. The values of γ(x,y) lie in the range [−1, 1], with 1 indicating perfect correlation, −1 indicating perfect anti-correlation and 0 indicating no correlation. The evaluation of these formulae in MATLAB was realised by the use of built-in functions [28].

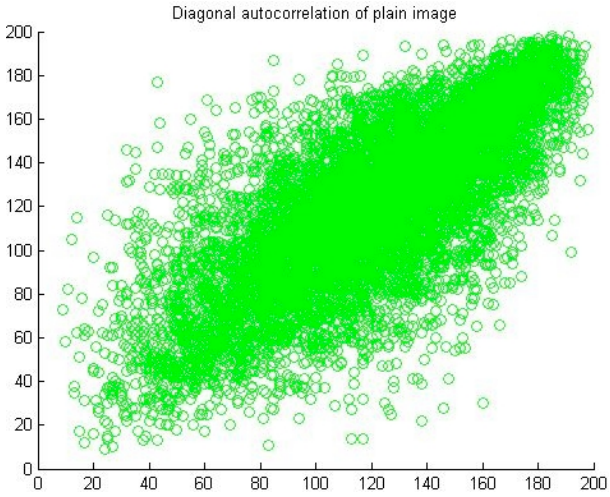
Fig. 12 shows the autocorrelation diagrams of the plain image 'baboon.jpg'; Fig. 13 shows the autocorrelation diagrams of the encrypted image. Similar results are obtained for a number of test plain images, even for all-black and all-white images.



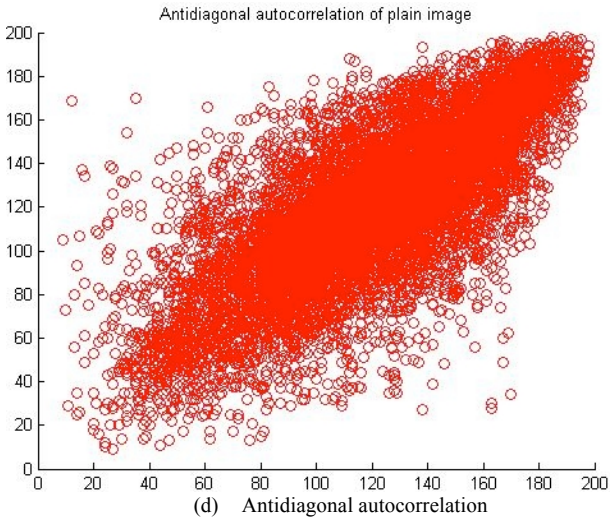
(a) Horizontal autocorrelation



(b) Vertical autocorrelation



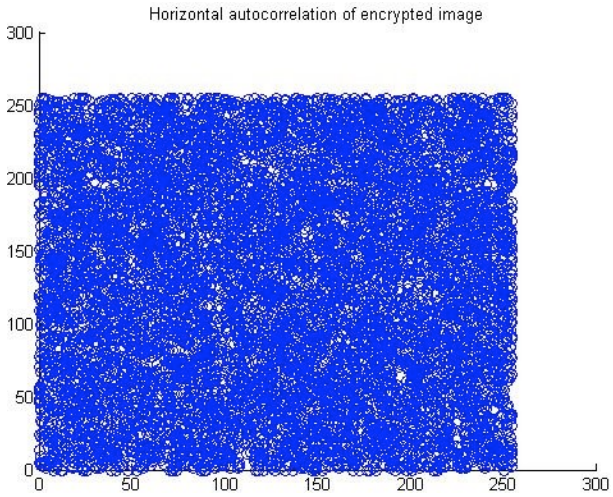
(c) Diagonal autocorrelation



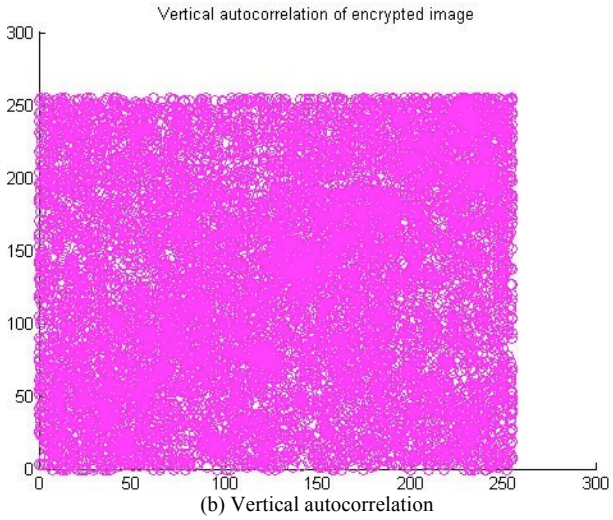
(d) Antidiagonal autocorrelation

Fig. 12. Autocorrelation diagrams of the plain greyscale image 'baboon'.

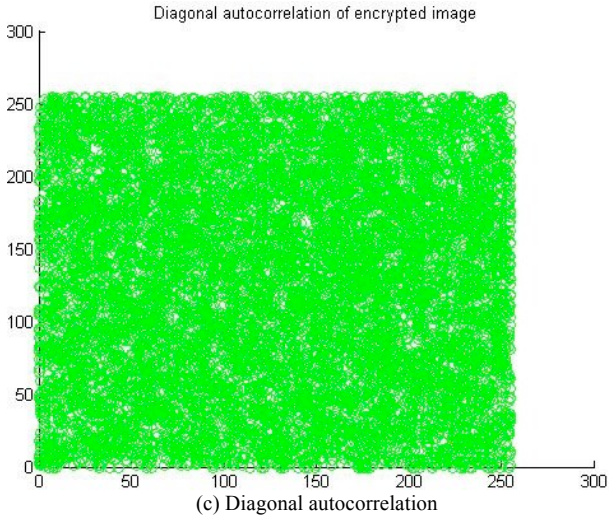
2. The term "autocorrelation" (instead of correlation) has been used because it refers to the similarity of a signal (image) with itself .



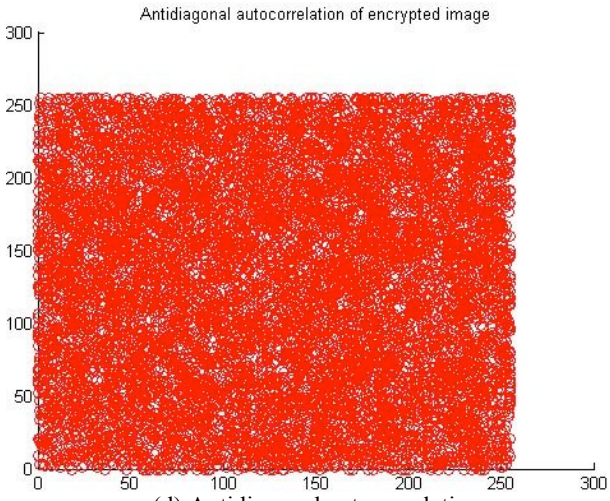
(a) Horizontal autocorrelation



(b) Vertical autocorrelation



(c) Diagonal autocorrelation



(d) Antidiagonal autocorrelation

Fig. 13. Autocorrelation diagrams of the encrypted image 'baboon'.

Thus, the autocorrelation of the original image has been cancelled in the encrypted image; thus, the proposed system resists statistical attacks based on autocorrelation analysis.

The result is also confirmed by the correlation coefficients of two adjacent pixels, as shown in the following Tables 2 and 3.

Table 2. Autocorrelation coefficients of two adjacent pixels in the plain and encrypted image 'Lena'.

Autocorrelation type	Plain image Lena	Encrypted image Lena
Horizontal	0.9205	0.0008
Vertical	0.9589	0.0530
Diagonal	0.8948	0.0033
Antidiagonal	0.9185	-0.0047

Table 3. Autocorrelation coefficients of two adjacent pixels in the plain and encrypted image 'baboon'.

Autocorrelation type	Plain image baboon	Encrypted image baboon
Horizontal	0.8136	0.0009
Vertical	0.8177	0.0981
Diagonal	0.7579	-0.0004
Antidiagonal	0.7664	0.0111

From Tables 2 and 3 we can see that the proposed encryption system drastically reduces the autocorrelation of input images, achieving good quality encryption.

3.2.2. Resistance to known plain image attacks

In general, CTRNG systems implemented in hardware are secure since even tiny variations in the component values, due to environmental changes, aging or other factors, will produce different chaotic curves in the long term. However, this is not true for software implementations, unless extra measures (such as the use of nonces or one-time pads) are taken.

If the intruder somehow knew for sure that some specific pixel pattern appeared in the cipher image, then she could have determined some (plain image, cipher image) pairs, to drastically reduce cryptanalysis time. This would be fairly

easy in simple systems using reversible operators such as XOR and addition, because these do not change the pixel positions but only their values.

So if the intruder knew that the plain image contained specific patterns at a specific region (e.g. black and white shapes like the chess-board), then she would try to guess the encryption scheme looking at (attacking) the specific region. However, the pre-processing stage used by the proposed system alters pixel places, making it robust against known plain image attacks.

Moreover, in the proposed system, the transmitter generates a matrix of random numbers [0-255] of the input image dimensions; this is XORed by the input image (1st processing), pixel-by-pixel, before the main cryptography process. Due to the two random numbers employed here (namely τ_1 and τ_2), different substrings of the Chua variable X_1 will be used each time; in this way, known image attack based on almost identical images is cancelled because the first XOR processing will produce totally different intermediate (hence, also final) results.

3.2.3. Resistance to chosen plain image attacks

Chosen image attacks using images with specific patterns will be cancelled since input images will be altered by the pre-processor to something unknown to the attacker.

Let's consider an example of a chosen image attack. Two special input images will be used: all-white image and all-black image. In both cases pre-processing has no effect; however, 1st and 2nd processing units using XOR will produce an output with almost uniform histogram; hence altering the sharp histogram of the chosen input image.

A simple but smart chosen image attack which breaks pseudorandom (software) chaotic systems or even hardware chaotic systems under specific conditions (e.g. resets) with deterministic or quasi-deterministic output is the following. In such systems, the cipher image C_i for a specific input image S_i will be the same (or, almost the same). In such simple systems, the cipher image C_i of an input image Si will be computed as:

$$C_i = N_i \text{ XOR } S_i \quad (7)$$

where N_i is the corresponding RND sequence (for instance, the Chua Noise) of the size of the image.

In this case, the following chosen image attack will break the cryptosystem:

1) An intruder records an arbitrary number of cipher images of size a*b.
2) The intruder submits a black image S_0 (all pixels are 0) of size a*b and records the cipher image C_0.
3) Next the intruder performs the operation C_0 XOR C_i to reveal any image S_i!

This is because:

$$C_0 = N_0 \text{ XOR } S_0 = N_0 \quad (8)$$

In this way, the intruder gets access to the pure output of the PRNG. Note that this holds also for systems using other operations instead of XOR used in chaotic cryptosystems (e.g., addition).

This attack (which we have called "**the black image attack**") is not possible with our system since the output is not a single XOR function of the input and the RNG but a rather complex one, as shown in Fig. 5. With a uniform input image (white or black), pre-processing has no effect; hence, $Im_1 = Im_0$. Next, $Im_2 = Im_0$ XOR $X_1(t+\tau_1)$. If Im_0 = black (all 0's) then $Im_2 = X_1(t+\tau_1)$. However,

$$Im_3 = Im_2 \text{ XOR } X_3(t+\tau_2) = X_1(t+\tau_1) \text{ XOR } X_0(t+\tau_2) \quad (9)$$

Next,

$$Z = Im_3 \text{ XOR } N(t+\tau_3) = X_1(t+\tau_1) \text{ XOR } X_3(t+\tau_2) \text{ XOR } N(t+\tau_3) \quad (10)$$

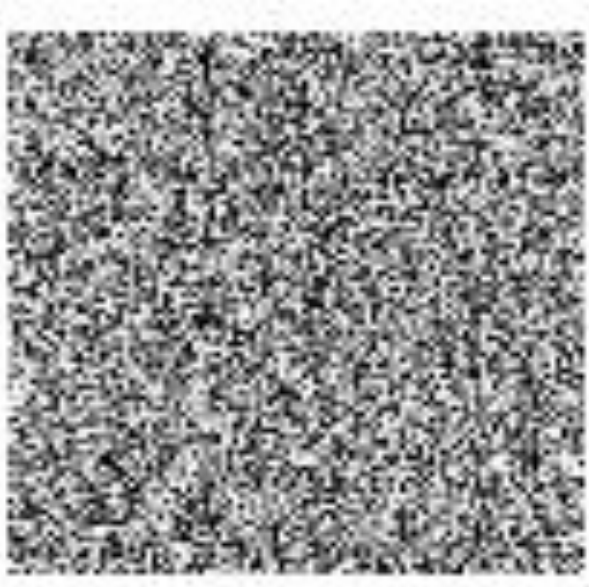
which does not reveal the system.
In general,

$$Z = F(Im_0) \text{ XOR } X_1(t+\tau_1) \text{ XOR } X_3(t+\tau_2) \text{ XOR } N(t+\tau_3) \quad (11)$$

where $F(Im_0)= Im_1$ represents the pre-processing function.

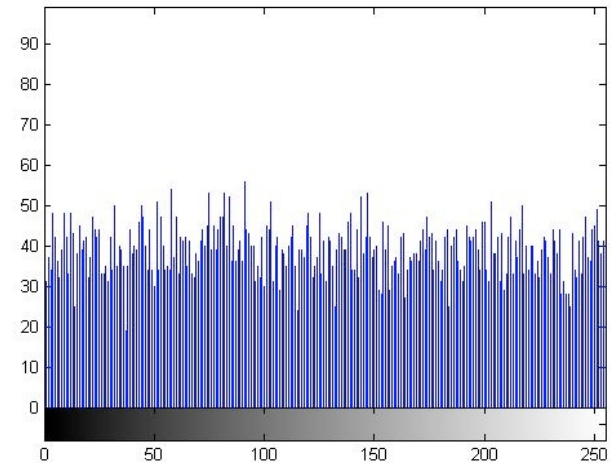
Figure 14 demonstrates results which verify the resistance to the black image attack, for a 100x100 image.

(a) Input image

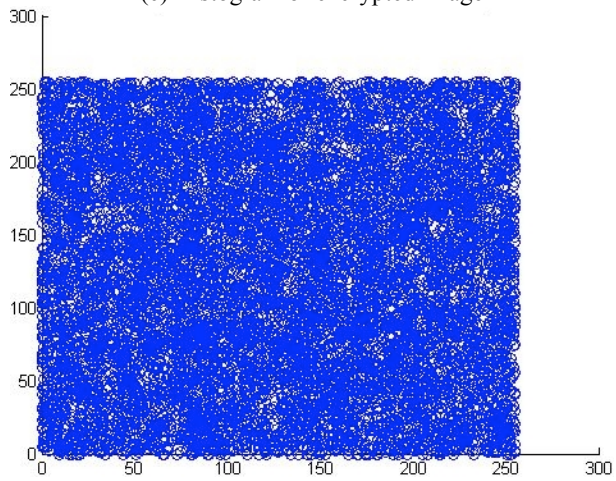


(b) Encrypted image

(continued)



(c) Histogram of encrypted image



(d) Horizontal autocorrelation of encrypted image

Fig. 14. Resistance to the black image attack.

If we need to make the system even more secure, we can introduce a second RNG, independent from the Chua's CTRNG; this could be another Chua's circuit or just a PRNG. This would feed the second input of XOR1 (instead of $X_1(t+\tau_1)$). Pre-processing could also be made more complex, e.g., altering pixel values.

4. Conclusion

In this paper we have presented an image encryption system based on Chua's circuit, with enhanced security. This characteristic is achieved by the special architecture of the encryption unit, which includes a pre-processing unit using permutation techniques, as well as, three XOR operations using the chaotic signals X_1, X_3 and a true random binary sequence produced from the chaotic signals using thresholding. The system works with uncompressed images, both greyscale and colour.

To enhance system security, parameterisation and key length, three integer variables have been introduced: τ_1, τ_2 and τ_3. By means of these variables the freshness of the cipher images may be assured in order to avoid playback attacks. Another possible application is the use of one-time pads [26:ch.8]. Although these features may not be needed in hardware implementations, (where they can be even set to constants), they are rather necessary supplements to PRNG (software) implementations.

The proposed system may be easily implemented in hardware or software. Even Chua's circuit can be implemented as a low-cost CMOS chip [10]. This leads to a feasible, fast and low-cost implementation [10]. The non-reproducible, unpredictable output which is invariant of the input image guarantees a secure and robust image communication. Software (PRNG) implementations will still be robust due to the additional building blocks and parameters used (seed1, τ_1, τ_2 and τ_3).

We have also examined the robustness of our system against the common attacks, namely cipher image only attack, known image attack and chosen image attack.

In conclusion, the proposed system presents the following features:

- Secure, robust against known attacks;
- Easy to implement, efficient, high speed;
- Portable; Low-cost;
- Multiparametric.

The extra parameters introduced in the system architecture (seed1, τ_1, τ_2 and τ_3) enable the implementation of secure PRNGs. Therefore, the proposed system is a good candidate for cryptographic applications where the above features play a crucial role, such as, in warfare.

References

1. L. O. Chua, IEICE Trans. Fundamentals **E77-A**(11), 1811 (1994).
2. B. R. Andrievskii and A. L. Fradkov, Automation and Remote Control **65**(4), 505 (2004).
3. T. M. Hoang, International Journal of Electrical and Electronics Engineering **4(3)**, 240 (2010).
4. C. K. Volos, JAMB **3**(1), 123 (2013).
5. A. Leros and A. Andreatos, Chaotic Modeling and Simulation Journal, **2013**(1), 1998 (2013). http://www.cmsim.eu/papers_pdf/january_2013_papers/21_CMSIM_2013_Leros_Andreatos_1_199-208.pdf
6. T. Matsumoto, IEEE Transactions On Circuits and Systems **CAS-31**(12), 1055 (1984).
7. N. N. Grigoropoulos, Chaotic Systems, analysis and applications of Chua's circuit, Diploma Thesis. Supervisor: Prof. A. P. Leros, Technological Institute of Chalkis, School of Technological Applications, 34400 Psachna, Evia, Greece (2009; in Greek).
8. B. Hasselblatt and A. Katok, A First Course in Dynamics: With a Panorama of Recent Developments. University Press, Cambridge (2003).
9. L. Gámez-Guzmán, C. Cruz-Hernández, R.M. López-Gutiérrez, and E.E. García-Guerrero, Commun. Nonlinear Sci. Numer. Simulat. **14**, pp. 2765–2775 (2009).
10. M. E. Yalçın, J. A. K. Suykens, and J.Vandewalle, IEEE Transactions on Circuits and Systems —I: Regular Papers, **51**, 7, pp. 1395-1404 (2004).
11. Ch. K. Volos, I. M. Kyprianidis, and I. N. Stouboulos, Image Encryption Process Based on a Chaotic True Random Bit Generator, In Proc. 16th IEEE International Conference on Digital Signal Processing (DSP 2009), Santorini, Greece, pp. 1091-1094 (2009).
12. B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd edition, John Wiley & Sons (1996).
13. A. S. Andreatos and A. P. Leros, Image Cryptosystem based on a Chua Circuit Chaotic Noise Generator, Presented at the Cryptography and its applications in the Armed Forces conf., Hellenic Army Academy, Vari, Greece (2012). http://www.sse.gr/files/Epistimoniko_Synedrio_6_4_12.pdf.

Available from: http://t-h.wikispaces.com/file/view/Andreatos-Leros-SSE.pdf.
14. G. Álvarez, F. Montoya, M. Romera, and G. Pastor, IEEE Transactions on Circuits and Systems - II: Express Briefs **51(10)** (2004).
15. L. M. Pecora and T. L. Carroll, Physical Review Letters **64**(8), pp. 821-824 (1990).
16. T. Kapitaniak, Controlling Chaos Theoretical and Practical Methods in non-linear Dynamics. Elsevier Ltd. (1996).
17. L.M. Pecora, T. L. Carroll, G. A. Johnson, and D. J. Mar, Fundamentals of synchronization in chaotic systems, concepts, and applications. American Institute of Physics (1997).
18. S. Boccaletti, C. Grebogi, Y.-C. Lai, H. Mancini, and D. Maza, The Control of Chaos Theory and Applications. Physics Reports, Elsevier (2000).
19. A. Pikovsky, Synchronization, A universal concept in nonlinear sciences. New York: Cambridge University Press (2001).
20. J. Gonzalez, Synchronization and Control of Chaos, An Introduction for Scientists and Engineers. Covent Garden, London: Imperial College Press (2004).
21. A. L. Fradkov and R. J. Evans, Control of chaos: Methods and applications in engineering, Annual Reviews in Control, Elsevier B. V. (2005).
22. F. L. Lewis, Chaos in Automated Control. Boca Raton, FL: Taylor & Francis Group (2006).
23. E. Schöll, Handbook of Chaos Control, 2nd Ed. Weinheim: Verlag GmbH & Co. (2008).
24. K. Pyragas, Physics Letters A (1992).
25. J. F. Kurose and K. W. Ross, Computer Networking: A Top-Down Approach, 5th edition. Pearson Education (2010).
26. A. Tanenbaum, Computer Networks, 4th Edition. Prentice-Hall (2003).
27. W. Stallings, Cryptography and Network Security: Principles and Practice, 5th edition, Pearson Education (2010).
28. How to find Correlation of an image. http://stackoverflow.com/questions/3689840/how-to-find-correlation-of-an-image. Edited Sept. 12 (2010).