Revista Mexicana de Física
ISSN: 0035-001X
rmf@ciencias.unam.mx
Sociedad Mexicana de Física A.C.
México

Hasimoto-Beltrán, R.
Low-complexity chaotic encryption system
Revista Mexicana de Física, vol. 53, núm. 1, febrero, 2007, pp. 58-65
Sociedad Mexicana de Física A.C.
Distrito Federal, México

Low-complexity chaotic encryption system

R. Hasimoto-Beltrán
Department of Computer Sciences, Center for Research in Mathematics (CIMAT),
Jalisco s/n, Col. Mineral de Valenciana, Guanajuato, 36240 Gto, México,
Phone: +(52)-(473) 732-7155, Fax: +(52)-(473) 732-5749
e-mail: hasimoto@cimat.mx

Recibido el 7 de diciembre de 2006; aceptado el 26 de enero de 2007

Secure multimedia communication presents new challenges that are difficult to handle using by currently adopted encryption schemes (RSA—Rivest-Shamir-Adelman, AES—Advanced Encryption Standard, and IDEA—International Data Encryption algorithm) [28]. It requires the processing of huge amounts of information at speeds ranging from Kilobits/sec (Kbs) to the order of Megabits/sec (Mbs). With this in mind, we propose a secure low-complexity encryption system based on a *4-array* of independently iterated chaotic logistic maps with a new *Spatiotemporal* feedback scheme as a diffusion process. The robustness of the system to opponents' attack is enhanced by using a periodic three-level pseudo-random perturbation scheme, one at the system-key level and two at the map array level. An analysis of the proposed scheme regarding its vulnerability to attacks, statistical properties and implementation performance is presented. To the best of our knowledge we provide a simple and secure encryption system for real-time multimedia communications with the fastest software execution time reported in the literature.

Keywords: Discrete chaotic encryption; block ciphers; symmetric encryption.

La seguridad en sistemas de comunicación de multimedia (texto, audio, imagen y video) representa un reto difícil de alcanzar para los actuales estándares de cifrado (RSA-Rivest-Shamir-Adelman, AES-Advanced Encryption Standard e IDEA-International Data Encryption algorithm) [28]. Se requiere el procesamiento de grandes cantidades de información a velocidades que fluctúan entre los Kilobits/seg (Kbs) hasta los Megabits/seg (Mbs). Enfocados en este problema se propone un sistema de cifrado seguro y eficiente basado en un arreglo de *mapas logísticos* independientemente iterados junto con un sistema de retroalimentación *espacio-temporal* usado como proceso de difusión de la información. Adicionalmente se hace uso de tres niveles de perturbación para modificar el estado actual del sistema e incrementar así su robustez contra ataques de oponentes; una perturbación es a nivel de la llave del sistema y dos adicionales a nivel de los mapas caóticos. El análisis de resultados muestra excelentes propiedades estadísticas del sistema propuesto, sensibilidad a las condiciones iniciales y la más alta velocidad de ejecución reportada en la literatura para llevar a cabo comunicaciones de multimedia en tiempo real.

Descriptores: Encriptación caótica discreta; cifradores de bloque; encriptación simétrica.

PACS: 05.45.Gg/Pq/Ac

1. Introduction

Discrete Chaotic Systems (DCSs) have many of the good properties required in cryptography; the most prominent are sensitivity to parameters, sensitivity to initial conditions and unpredictable trajectories [1,14,21]. The first two properties are related to *diffusion*, and the last one to *confusion* in the cryptographic nomenclature. Confusion is intended to make the relationship between ciphertext and plaintext statistically independent, while diffusion is intended to spread out the influence of a single plaintext digit over many ciphertext digits to hide the statistical structure of the plaintext [16]. These properties have been the basis to develop secure analog and digital communication systems.

A discrete chaotic encryption system consists of a digital generator of chaos (nonlinear dynamic map), which takes an input message known as plaintext and produces an independent masked output message known as ciphertext. General purpose (data or compression independent) chaotic encryption schemes, which include the great majority in the literature, iterate a single, one-dimensional chaotic map [1,3,12,19,21,26,27]. Higher dimensional maps (2-D and 3-D) have been proposed for image data confusion and diffusion [6,8,10,11,13,27]. Confusion is performed by a chaotic permutation of pixel coordinates and diffusion by a (linear or non-linear) transformation on the gray level. Few proposals based on chaos theory have emerged for voice and video encryption [18,20,22,24] with limited speed processing, to fulfill current multimedia demands.

DCSs have a relevant drawback not found in their continuous counterpart; the number of iterations for a chaotic state to repeat itself (known as cycling length) is finite. This means that DCSs are short-cycled, with their largest theoretical cycle length $CL = 2^L$ states, where L is the bit precision of the machine [4]. In practice, $\mathrm{CL} \ll 2^L$ for almost every chaotic trajectory with a maximal length of $O(2^{\varepsilon L})$, $0<\varepsilon<1$[17]. In spite of this, sensitivity to parameter and initial conditions is maintained [23]. Current research in DCSs has focused on increasing the cycle length using *Perturbance-based* schemes, which transform stable chaotic cycles into non-stable ones. Tao and Ruli, 1998 [23], proved that periodic perturbations increase the cycle length of chaotic systems by $\sigma\Delta(2^L - 1) \gg 2^L$, where σ is a positive integer and Δ is the perturbation period. They obtained a lower bound of $\Delta(2^L - 1) \gg 2^L$, which considerably improves the maximum cycle length with respect to the

unperturbed case [$O(2^{\varepsilon L})$]. Ideally, perturbation magnitudes should be obtained by a uniform pseudo-random number generator to improve the dynamical properties of digital chaotic systems [4,5,23].

In this work, we propose a novel symmetric encryption system based on a *4-array* of independently iterated chaotic maps, a three-level periodic perturbation, and a two-mode feedback called *spatiotemporal feedback*. The perturbation scheme changes the current system condition by modifying the system-key (third-level perturbation) and the trajectory of the chaotic maps (first and second level perturbations) to increase the system's security against statistical and differential attacks [15]. The system key is periodically modified using a pseudo-random number generator called RANROT [9], while every map trajectory is modified using the system's output itself (ciphertext) rather than a predefined perturbation equation as currently used in the literature [4,5,23]. Since chaotic maps are iterated independently, ciphertext inter-dependency is created by adding spatiotemporal feedback to current ciphertext value. Spatiotemporal feedback takes into account the evolution of a single chaotic map and the cross-evolution of all maps, making the system extremely sensitive to plaintext and system-key. Our final goal is to develop a simple, secure, and compression-independent encryption system for current real-time multimedia demands.

The rest of the paper is organized as follows. In the next section we discuss the properties of the chosen chaotic map and describe the proposed encryption/decryption system. In Sec. 3 experimental results and security of the proposed systems are analyzed. Conclusions are presented in Sec. 4.

2. Chaotic encryption scheme

2.1. Logistic map

An important step in discrete chaotic encryption is the selection of the map. For its mathematical simplicity and good chaotic properties, our selection is the well-known logistic map represented by:

$$X_n = \lambda X_{n-1}(1 - X_{n-1}), \quad \lambda \in [1, 4], \quad X \in [0, 1], \quad (1)$$

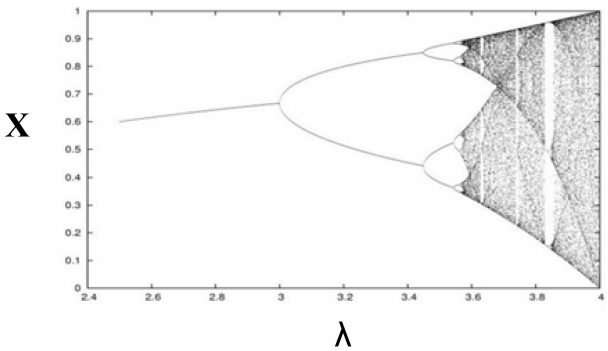

FIGURE 1. Bifurcation diagram for the logistic map.

with a corresponding bifurcation diagram depicted in Fig. 1. As λ increases from 1 to 4, the map experiences a period doubling to chaos [25]. In particular for $\lambda \geq 3.5699$ (known as an accumulation point), it shows chaotic behavior; however, there are many periodic windows (with all kinds of periods) that appear abruptly. A very well-known and prominent period-3 window appears at$\lambda = 1 + \sqrt{8} = 3.8284$. Short period windows of the logistic map should be avoided during the encryption process, because they reveal statistical information useful for attackers to break into the system. This problem can be alleviated by perturbing the cycling chaotic signal with period T every Δ iterations for $\Delta \leq T$ [23]. The perturbation will drive the signal away from its cycle after i number of iterations, where i depends directly on the perturbation magnitude [21]. Therefore, the new period in the perturbed cycle is now$\sigma \Delta T$, where in the average case $T = \sqrt{2^L}$ [7].

2.2. Proposed encryption scheme

We propose a symmetric block-based cipher described by the following components: 1) chaotic system-key generation; 2) an array of chaotic logistic maps with spatiotemporal feedback; and 3) a three-level perturbation process. Each component works as follows.

1. *System-key Generation*

   An initial seed is first created for the generation of a B bits ($\geq$ *128*) system-key (K) using RANROT [9]. As part of the encryption system's security, K is constantly modified using both *fixed* and *forced* updates. Fixed-key update is part of the three-level perturbation scheme in which K is replaced periodically after a random number of iterations using RANROT. Forced-key update on the other hand is used as a resynchronization process between cipher and decipher in the case of data errors during transmission (or when security is compromised). When errors occur, decipher cannot continue doing its job (due to dependencies between current and previous ciphertexts), and issues a resynchronization signal to the cipher to initialize the system with a new K. The resynchronization process is allowed at any time of the encryption/decryption process.

2. *Encryption System*

   Once K is generated, it is divided into 8 equal parts (Fig. 2) in order to initialize the maps' variables and parameters of the 4-array logistic map as follows:

   $$X_{i,0} = K(2i - 1), \quad \lambda_i = 3.73364 + [K(2i)/2^{B/8} + K(2i)/10^{h_8} + (a \oplus b)/2^{B/16}]/10, \quad i \in \{1, 2, 3, 4, \} \quad (2)$$

   where $X_{i,0}$ and λ_i are thei^{th} map variable initial condition and parameter, respectively,

with $0.2 \leq X_{i,0} \leq 0.8$ (except $X_{i,0} \approx 0.5$) and $3.73364 \leq \lambda_i \leq 3.9987$, h_8 is the number of digits in the largest decimal number represented by $B/8$ bits ($K(2i)/10^{h_8} = 0.(2^{B/8})$), $a \oplus b$ term is the exclusive-OR (XOR) of the most and least significant bits of $K(2i)$ respectively having both equal size bit representation of $B/16$. $X_{i,0}$ and K are de-correlated by iterating$X_{i,0}$, $1 \leq i \leq 4$ an *RT* random number of times over all maps:

$$\begin{aligned} For \quad & i \in \{1,2,3,4\} \\ & \gamma = X_{i,0} \\ & repeat \quad RT \quad times \\ & \qquad \gamma = \gamma.\lambda_1.(1-\gamma) \\ & \qquad \gamma = \gamma.\lambda_2.(1-\gamma) \\ & \qquad \gamma = \gamma.\lambda_3.(1-\gamma) \\ & \qquad \gamma = \gamma.\lambda_4.(1-\gamma) \\ & end - repeat \\ & \qquad X_{i,0} = \gamma \\ & end - for \end{aligned} \tag{3}$$

Even a one-bit change in K will generate a completely different map orbit, which in turn generates different ciphertexts (note that $X_{i,0}$ is influenced by all system maps). *RT* loop does not affect the system performance, since it is executed when the cryptosystem is restarted or after the fixed-ke$y$ update.

Once $X_{i,j}$ and $\lambda_i$values have been obtained, the $4 - array$ of logistic maps can be written as:

$$\begin{aligned} X_{i,j} &= \lambda_i.X_{i,j-1}.[1 - X_{i,j-1}], \\ i \in \{1,2,3,4\}, \quad & j \in \{1,2,3,...\}, \\ X \in [0,1], \quad & \lambda \in (3.7,4] \end{aligned} \tag{4}$$

where i and$j$ represent the map and state indexes respectively. For a fixed state j, four map variables $(X_{i,j}, i \in \{1,2,3,4\})$ are obtained to encrypt their corresponding plaintext of size$B/4$ using the following equation:

$$\begin{aligned} C_{i,k} = & ([P_k + X'_{i,k}] \bmod 2^{B/4}) \oplus X'_{i,k} \\ & \oplus ([X'_{i+1,k} + X'_{i+2,k}] \bmod 2^{B/4}) \\ & \oplus ([C_{i-1,k} + C_{i,k-1}] \bmod 2^{B/4}), \\ & i \in \{1,2,3,4\}, k = (j + i - 1), \end{aligned} \tag{5}$$

where k is the cipher iteration index ($k = 4j$ at the end of state j), X' is the corresponding integer representation of X using $B/4$ bits, P_k is the k^{th} plaintext input, $C_{i-1,k}$ is the previous cyphertext output ($i - 1$) of the current iteration (k^{th}), and $C_{i,k-1}$ is the previous cyphertext output of the same i^{th} map, but from the $k - 1$ iteration. A total of B bits (the size of the system-key) are encrypted per state iteration ($B/4$ encrypted bits per map). $C_{i-1,k}$ and $C_{i,k-1}$ represent the temporal and spatial feedback respectively. The initial spatial feedback $C_{0,k}$ takes in the last ciphertext output of the previous iteration ($C_{4,k-1}$) to spread the system changes on to future ciphertexts and all four logistic map variables (see next subsection). To increase the encryption system's security, ciphertext output $C_{i,k}$ is masked using all four map variables:

$$\begin{aligned} C^p_{i,k} &= C_{i,k} + X'_T, \\ X'_T &= X'_{1,k} \oplus X'_{2,k} \oplus X'_{3,k} \oplus X'_{4,k} \end{aligned} \tag{6}$$

Therefore, decipher cannot use $C^p_{i,k}$directly to find its corresponding plaintext, it needs to knowX'_T.

3. *Three-Level Perturbation*

In order to increase the cycle length of the logistic maps and hence the system's security, a three-level periodic perturbation scheme is proposed. The first two perturbation levels are related to the system variables and the third one is related to the system-key. At the first perturbation level, the trajectory of every map is slightly modified to increase its cycle length [5,23]; at the second perturbation level, the current system variable is randomly changed, creating a totally new trajectory for the system; and at the third perturbation level, the system-key value is renewed using RANROT. Third-level perturbation represents a reset operation, since the entire encryption/decryption system parameters are completely modified (system-key, map' variable and parameter).

The first-level perturbation is expressed as:

$$X^p_{i,j} = X_{i,j} + \frac{1.1 + C_{4,j}(i)}{10^{h_{16}}} \quad i \in \{1,2,3,4\} \tag{7}$$

where $C_{4,j}(i)$ is the i^{th} element of the spatial feedback $C_{4,j}$ at the current state j (Fig. 3). We post-process $X^p_{i,j}$ so that its first digit after the decimal point stays the same as in$X_{i,j}$; therefore, $abs(X^p_{i,j} - X_{i,j}) < 10^{-1}$. The reason for using $C_{4,j}$ to perturb the encryption system can be seen by considering a differential attack, where the attacker can choose a pair of plaintexts (with a predefined distance) and get their corresponding ciphertext outputs. If the cipher is not robust, the attacker may find a relationship between plaintext and ciphertext and consequently find the system-key (for more information on differential attack see Ref. 14). To prevent this from happening in our proposed system, every single plaintext change is exacerbated by disturbing not only future ciphertext outputs through spatiotemporal feedback, but also the

maps' variables through the first-level perturbation [Eq. (7)]. The combined effects (feedback and perturbation) generate totally different trajectories for any pair of plaintexts when iterated by the system. Another reason for using $C_{4,j}$ is because of its uniform distribution (as discussed in Sec. 3), a basic requirement for perturbation schemes [23]. The second-level perturbation replaces each map variable by a new value occurring as a result of cross-iterating the map variable through all maps [similar process as in Eq.(3)]. For thei^{th} map in state j, its new system variable is obtained by:

$$\begin{aligned} For \quad & i \in \{1,2,3,4\} \\ & \gamma = X_{i,j} \\ & For \quad k \in \{i, [i \bmod 4]+1, [(i+1) \bmod 4]+1, [(i+2) \bmod 4]+1\} \\ & \gamma = \gamma.\lambda_k.(1-\gamma) \\ & end-for \\ & X^P_{i,j} = \gamma \\ end-for & \end{aligned} \tag{8}$$

That is, new system variables are influenced by all maps. The third level perturbation replaces current system-key using RANROT every random number of iterations. Every time the system-key is updated, the new key is sent to decipher in order to update system maps variables and parameters. The cycle of the perturbations represented by $PT_i, 1 \leq i \leq 3$,can be randomly selected to increase the system-key space in case of a brute force attack (the opponent tries every possible system-key combination until the right one is found). We define the perturbation cycles as follows: PT_1 = *[(a_random_number) modulus 10] + 30*, PT_2=n_1 .PT_1 + $PT_1/2$ and PT_3 = n_2 .PT_2 + $PT_1/4$ for n_1 and n_2 positive integers greater than one. The simpler it is the more frequent the perturbation. The value of PT_1 is related to the sensitivity of the logistic map to a magnitude change of $1/2^{B/8}$ in the initial condition. For $B = 128$ bits, the minimum magnitude change of two system variables is $\sim 10^{-5}$, which requires about 30 map iterations for their trajectories to diverge chaotically [21]. This result is important for the cipher in order to produce different trajectories when input values differ in the least significant bits. Since the second-level perturbation is more severe than the first-level (the system variable is completely replaced, allowing the system to find its way out of short cycles immediately), we defined PT_2 to be a multiple of PT_1 plus a de-synchronization term to avoid the application of PT_1 and PT_2 at the same time. The same scheme applies to third level perturbation (system-key), where PT_3can be any number times PT_2 plus a de-synchronization term ($PT_1/4$).

2.3. Decryption scheme

The corresponding decryption system resembles to a certain extent the encryption system except for Eqs. (6) and (5),

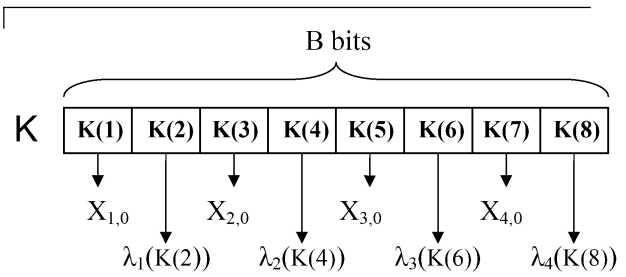

FIGURE 2. System-key (K) partition ($B/8$) for the creation of map variables ($X_{i,0}$) and corresponding parameters (λ_i).

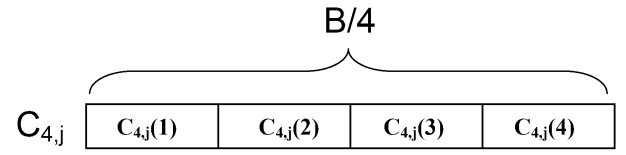

FIGURE 3. Ciphertext of the fourth map at state j (C_{4j}) used for perturbing the array of logistic maps. $C_{4j}(i)$ size is $B/16$ bits.

where the decipher solves for $C_{i,k}$ and P_k respectively. For successful deciphering, the cipher creates an initial seed and sends it to the decipher (encrypted using the Advanced Encryption Standard—AES for example) in order for both systems to recreate independently the same system-key, map' variables and parameters, RT, and perturbation values along the entire encryption/decryption process. The seed can be either a predefined value (password, keyword, etc) or calculated on the fly by the cipher (by reading the computer clock, communication identifier, etc.). The cipher computes the ciphertext corresponding to the current plaintext input $C^P_{i,k}$=$F(P_k)$ [see Eqs. (5) and (6)], and the decipher upon receiving $C^P_{i,k}$ performs the inverse operation $P_K = F^{-1}(C^P_{i,k})$ represented by:

$$C_{l,k} = C^P_{l,k} - X'_T;$$

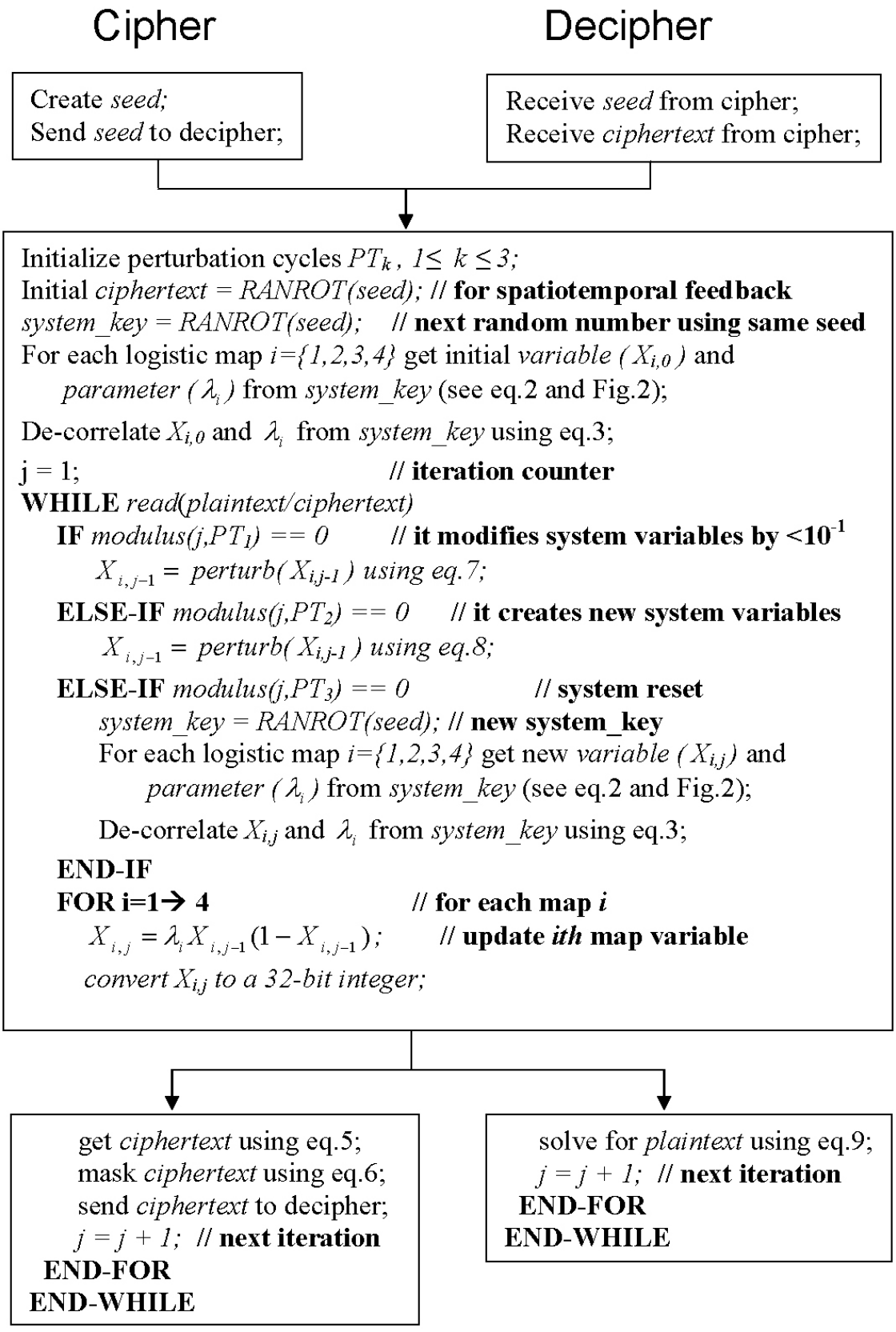

FIGURE 4. Pseudocode of the encryption/decryption system.

TABLE I. File type and size used in the encryption process.

File Type	Size (Kbytes)
Text	1.07
Audio (.wav)	91.8
Image (lenna)	256
Movie (pres-clinton-final-days.mov)	16281.6

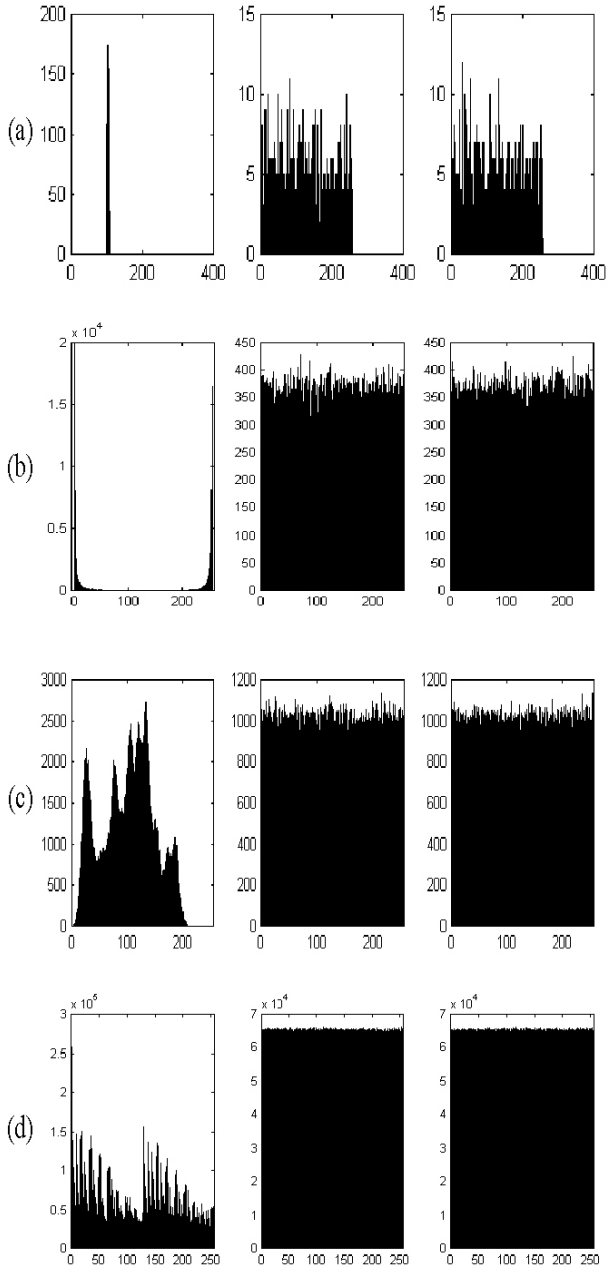

FIGURE 5. Histogram of plaintext (left column) and corresponding ciphertext for two different system-keys (center and right columns). Plaintext corresponds to data shown in Table I: (a) text file, (b) audio file, (c) image file, and (d) movie file.

$$P_k = [C_{l,k} \oplus X\prime_{i,k} \oplus ([C_{i-1,k} + C_{i,k-1}] \bmod 2^{B/4}) \oplus ([X'_{i,k+1} + X_{i,k+2}] \bmod 2^{B/4}) + 2^{B/4} - X'_{i,k}] \bmod 2^{B/4},$$
$$i \in \{1,2,3,4\}, \quad k = (j + i - 1) \tag{9}$$

Figure 4 shows the pseudocode of both cipher and decipher systems.

3. Security analysis and experimental results

Our proposed scheme is flexible regarding the system-key size and number of chaotic maps used for the encryption process; however, there must be some congruency between their corresponding bit sizes. In general, B (size of K) can be a multiple of m bits (B_m) for$m \in \{8, 16, 32\}$, and the number of chaotic maps can be at least B_m/m (one map per$m$ bits of K) and at the most $B_m/8$. A recommendation is not to use more than 32 bits of K for the generation of $X_{i,0}$ andλ_i, $i \in \{1, 2, 3, \ldots\}$ (16 bits for each value), since a change in the least significant bit of K may be imperceptible (unless 1^{st} and 2^{nd} level perturbation frequencies are increased).

Our proposed scheme has been applied to different data files (see Table I) with different statistical properties (see original histograms in Fig. 5). For the experiment we use the following setting: B_{32} =128 bits (creating four logistic maps), initial spatiotemporal feedback is selected randomly using RANROT, *RT = 20*, $PT_1 = 35$ iterations, $PT_2 = n_1$ PT_1, and *PT3*= $n_2 PT_2$, for $n_1 = n_2 = 5$ (these two variables could have also been computed randomly to increase system space search). Recall that the actual value of PT_1 represents the number of iterations needed to capture system-key magnitude changes of 10^{-5}.

3.1. Security analysis

As previously mentioned, a good cryptosystem must have the following properties [2,7,10]:

- Sensitivity to system-key: for two keys (or plaintexts) with the slightest difference, no distinguishable difference between the corresponding ciphertext can be found by any known statistical analysis.
- Sensitivity to plaintext: flipping one bit in the plaintext should create a completely different ciphertext.
- Statistical independency: the cipher text should be statistically indistinguishable from the output of a truly random function, and should be statistically the same for all keys.

We will question the security of our scheme by using these three properties (not in the same order). Figure 5 shows the histograms of plaintext and corresponding ciperhtext of data files described in Table I. For each plaintext, we use two

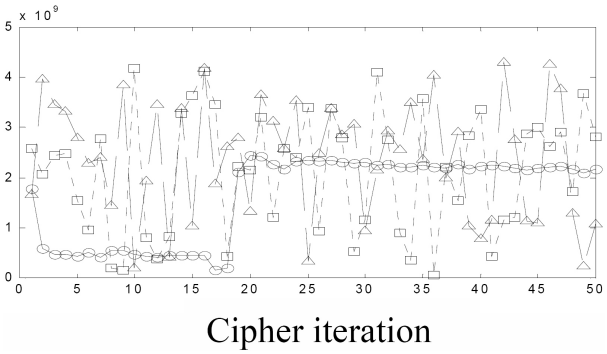

FIGURE 6. Sensitivity to system-key changes. Plaintext (circled continuous line) encrypted with two slightly different system-keys (least significant bit changed).

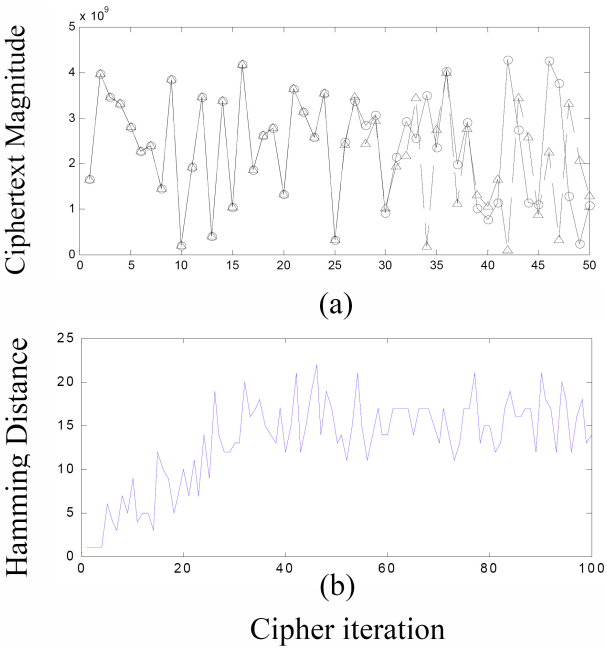

FIGURE 7. Sensitivity to system-key changes without perturbation scheme. (a) Ciphertexts of a pair of chosen plantexts with the least significant bit changed; (b) Hamming distance of corresponding ciphertexts.

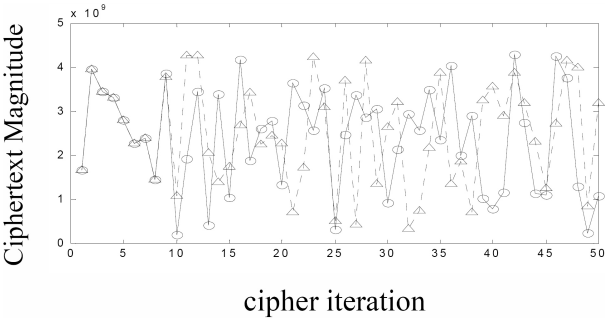

FIGURE 8. Same as in Fig. 7a with first level perturbation at 9^{th} cipher iteration (dashed line).

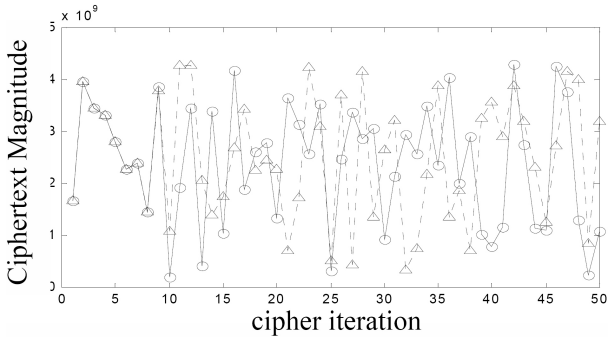

FIGURE 9. Same as in Fig. 8 with second level perturbation.

randomly chosen keys in order to prove statistical independence from the scheme. In all cases, the ciphertext histogram is uniform and independent of the plaintext histogram and system-key. As an average over all data files, 99.6% of the total bytes and 50% of the total bits were changed during the encryption process, providing the best protection against attacks. The scheme response to a slight change in the system-key (flipping the least significant bit) is shown in Fig. 6. Because of the de-correlation process between the system-key and maps' variables and parameters [Eq. (3)], the ciphertext output diverges from practically the first iteration. Let us now analyze the effect in the difference (in the least significant bit) of a pair of plaintexts on the ciphertext output sequence without perturbation (this is known as differential attack). Figure 7a shows that apparently the scheme reacts slowly until it reaches cipher iteration 30; but if we look at the Hamming distance between the ciphertexts (original and modified) shown in Fig. 7b, we see an immediate reaction from the first iteration, with only one bit changed initially. After ten iterations, approximately eight of the bits are different; at ciphertext iteration 30 and thereafter, the Hamming distance fluctuates around 15 bits meaning that the two ciphertexts outputs diverge chaotically. The number of iterations for the system to reach *HD=15* bits can be reduced if plaintext is iterated chaotically before the encryption. This can considerably affect the performance of the scheme; instead, we use map perturbation together with spatiotemporal feedback (see Sec. 2) as a better alternative. Figures 8 and 9 depict the perturbed case of Fig. 7a for first and second level perturbation, respectively [see Eqs. (7) and (8)] at ciphertext 9^{th}. In both cases, the reaction of the cryptosystem is immediate, influencing future ciphertexts output values. Since perturbation modifies map variables, the rest of the cipher trajectory is completely different from the unperturbed case.

If the opponent chooses brute force attack, he will need to search for at least $2^{128} \approx 3.4\times10^{38}$ key possibilities in our current setting. In addition, there are four more random numbers with 5-bit representation each, *RT*, P_1, P_2, *and* P_3; so brute force attack will need to consider a total space analysis of $(2^{128}).(2^{20})$. Recall that the system-key size is very flexible and can be a multiple of 8, 16, or 32 bits depending on the desired level of security by a particular application.

Finally, a C-language implementation of the cipher system on a 940Mhz Pentium®-III, with 190Mb of memory under the Red Hat Linux operating system version 2.4.20-28.9, shows an average speed of 220Mbs (Megabits/sec), which is much faster than any other scheme reported in the literature. This reported speed is fast enough for real-time multimedia communications.

4. Conclusions

We have proposed a simple and robust symmetric block-cipher cryptosystem based on a 4-array of chaotic logistic maps, a spatiotemporal feedback, and a three-level perturbation scheme. We have shown the perfect statistical properties of the system, as well as its sensitivity to initial conditions. The system is scalable in the sense that it allows the number of maps and system-key size to increase by multiples of 8, 16 or 32 bits to add security to the system (making brute force attacks impossible). Finally, a software implementation of the system shows excellent performance to fulfill current multimedia application demands, such as real-time audio and video communications.

1. G. Álvarez, F. Montoya, G. Pastor, and M. Romera, *Proc. IEEE Int. Carnahan Conf. Security Tech.* (1999) 332.
2. G. Álvarez and S. Li, *Phys. Lett. A* **240** (1998) 50.
3. M.S. Baptista, *Phys. Lett. A* **240** (1998) 50.
4. P.M. Binder and R.V. Jensen, *Phys. Rev. A* **34** (1986) 4460.
5. J. Cernak, *Phys. Lett. A* **214** (1996) 151.
6. G. Chen, Y. Mao, and C.K. Chui, *Chaos Solit. & Fract*. **21** (2004) 749.
7. F. Dachselt and W. Schuwartz, *IEEE Trans. Circ. Syst.-I* **48** (2001) 1498.
8. S. Deng, L. Zhang, and D. Xiao, *Lecture Notes in Computer Science* **3497** (2005) 868.
9. A. Fog, *Chaotic random number generators with random cycle lengths*. http://www.agner.org/random/theory/chaosran.pdf.
10. J. Fridrich, *Int. J. Bifurcation Chaos* **8** (1998) 1259.
11. Z. Guan, F. Huang, and W. Guan, *Phys. Lett. A* **346** (2005) 153.
12. T. Habutsu, Y. Nishio, I. Sasase, and S. Mori, *Advances in Cryptology (EUROCRYPT'91)* (Springer-Verlag, 1991) p. 127.
13. Z. Han, W.X. Feng, L.Z. Hui, L.D. Hai, and L.Y. Chou, *Proc. IEEE Int. Conf. Robotics Intelligent Systems and Signal Processing*, (2003) 778.
14. G. Jakimoski and L. Kocarev, *IEEE Trans. Circ. Syst.-I* **48** (2001) 163.
15. G. Jakimoski and L. Kocarev, *Phys. Lett. A* **291**(2001) 381.
16. L. Kocarev, *IEEE Circ. Syst. Mag.* **1** (2001) 6.
17. S. Li, X. Mou, Z. Ji , J. Zhang, and Y. Cai, *Physics Communications* **153** (2003) 52.
18. S. Lian, J. Sun , Z. Wang, and Y. Dai, 8^{th} *Int. Conf. Control, Automation, Robotics and Vision* (2004) 126.
19. R.F. Machado, M.S. Baptista, and C. Grebogi, *Chaos, Solitons and Fractals* **21** (2004) 1265.
20. T. Paraskeve, N. Klimis, and K. Stefanos, *ACM Multimedia'04*.
21. N.K. Pareek, V. Patidar, and K.K. Sud, *Phys. Lett. A* **309** (2003) 75.
22. K.M. Roskin and J.B. Casper, *From Chaos to cryptography* (2002); krish@cats.ucsc.edu.
23. S. Tao, W. Ruli, and Y. Yixun, *Electronics Letters* **34** (1998) 873.
24. K.W. Tang and W. Tang, *IEEE Inter. Conf. Industrial tech. (ICIT)* (2005) 571.
25. W. Wang, Z. Liu, and B. Hu, *Phys. Rev. Lett.* **84** (2000) 2610.
26. K. Wong K, *Phys. Lett. A* **298** (2002) 238.
27. J. Yen and J. Guo, *IEEE Int. Symp. Circ. Syst. (ISCAS-2000)* **IV** (2000) 49.
28. M. Yang, N. Bourbakis, and S. Li, *IEEE Potentials* (2004) 28.