Chaotic Encryption Scheme for Real-Time Digital Video*

Shujun Li[a], Xuan Zheng[b], Xuanqin Mou[a] and Yuanlong Cai[a]

[a]Institute of Image Processing, School of Electronics & Information Engineering
Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China
[b]Department of Electrical Engineering, Polytechnic University, Brooklyn, NY 11201, USA

ABSTRACT

In this paper, we propose a novel video encryption scheme based on multiple digital chaotic systems, which is called CVES (Chaotic Video Encryption Scheme). CVES is independent of any video compression algorithms, and can provide high security for real-time digital video with fast encryption speed, and can be simply realized both by hardware and software. What's more, CVES can be extended to support random retrieval of cipher-video with considerable maximal time-out; the extended CVES is called RRS-CVES (Random-Retrieval-Supported CVES). Essentially speaking, CVES is a universal fast encryption system and can be easily extended to other real-time applications. In CVES, 2^n chaotic maps are used to generate pseudo-random signal to mask the video, and to make pseudo-random permutation of the masked video. Another single chaotic map is employed to initialize and control the above 2^n chaotic maps. Detailed discussions are given to estimate the performance of CVES/RRS-CVES, respectively from the viewpoints of speed, security, realization and experiments.

Keywords: Chaotic Video Encryption Scheme (CVES), Real-Time, Random Retrieval, RRS-CVES, Piecewise Linear Chaotic Map (PLCM)

1. INTRODUCTION

In the digital world nowadays, the security of digital images/videos becomes more and more important since the communications of digital products over network occur more and more frequently. In addition, special and reliable security in storage and transmission of digital images/videos is needed in many digital applications, such as pay-TV, confidential video conferencing and medical imaging systems, etc. Generally speaking, the well-developed modern cryptography should be the perfect solution to this task. As we know, many perfect ciphers have been established and applied widely since 1970s, such as DES, IDEA and RSA [1]. But most conventional ciphers cannot be directly used to encrypt digital video in real-time systems because their encryption speed is not fast enough, especially when they are realized by software. In addition, the existence of different compression algorithms in digital video systems makes it more complicated to incorporate the encryption part into the whole system. Thus, to protect the content of digital images/videos, some specific encryption systems are needed.

Recently, many different video encryption schemes have been proposed [2–12]. Most of them are joint compression-encryption methods, which are specially designed to provide reliable security for MPEG video stream [4,5,7–12]. From the works of [13–15], some video encryption schemes are known to be not secure enough from strict cryptographic viewpoint. Actually, there still exist trade-offs between the security and the encryption speed in many video encryption systems [13].

In this paper, we propose a novel video encryption schemes based on multiple chaotic systems, which is called Chaotic Video Encryption Scheme (CVES). It can provide high security with rather fast encryption speed, and can be realized simply by both hardware and software. CVES is independent of any video compression algorithms so it will not be limited by the format of encrypted video, which is one important merit of CVES compared with other video encryption systems. In addition, CVES can be extended to support random retrieval of encrypted video with considerable maximal time-out. Essentially speaking, CVES is a UNIVERSAL fast encryption system with high security, so it can be easily extended to other real-time applications.

* This paper has been published in *Proceedings of SPIE*, vol. 4666, pp. 149-160, *Real-Time Imaging VI*, edited by Nasser Kehtarnavaz, March 2002, SPIE–The International Society for Optical Engineering.

Further author information: Shujun Li (corresponding author) – personal web site: `http://www.hooklee.com`.

This paper is organized as follows. In Sect. 2, we firstly give a brief survey of chaotic cryptography nowadays and the realization problem of digital chaotic systems. CVES and its extended version supporting random retrieval RRS-CVES (Random-Retrieval-Supported CVES) are detailedly described in Sect. 3. The performance of CVES/RRS-CVES is estimated in Sect. 4, respectively from the viewpoints of the speed, security, realization and experiments. The last section is the conclusion of this paper.

2. CHAOTIC CRYPTOGRAPHY

2.1. Chaos vs. Cryptography

Chaos theory is established since 1970s from many different research areas, such as physics, mathematics, biology and chemistry, etc. [16]. The most well-known characteristics of chaos are the so-called "butterfly-effect" (the sensitivity to the initial conditions), and the pseudo-randomness generated by deterministic equations. Many researchers have pointed out that there exists tight relationship between chaos and cryptography [17–20]. Many fundamental characteristics of chaos, such as the mixing property and the sensitivity to initial conditions, can be connected with "confusion" and "diffusion" property in good ciphers. Considering chaos theory has developed well in recent decades, chaos may become a new rich source of new ciphers.

Interestingly, the idea of chaotic cryptography can be traced back to Shanon's classic paper [21] titled "Communication Theory of Secrecy Systems" published in 1949. Of course, he couldn't use the unborn word "chaos"; he just mentioned that **well-mixing** transformations used in a good secrecy systems can be constructed by the basic **stretch-and-fold** mechanism, which really implies "chaos" (consider the baker map) [16]. In fact, any conventional cipher can be regraded as a chaotic or pseudo-chaotic system, since confusion and diffusion mean deterministic disorder – chaos [1, 16, 17].

2.2. State-of-the-Art of Chaotic Cryptography

The first scientific paper about chaotic cryptography [22] was published in 1989, in which the author suggested a novel stream cipher based on one-dimensional chaotic map. In the next year, the chaos synchronization technique was firstly reported and the secure communications via chaos synchronization was presented [23]. From then on, chaotic cryptography has developed from different areas, chiefly physics, electrical and electronics engineering, computer science, and applied mathematics. Many digital chaotic ciphers [18–20, 24–31] and analog chaotic secure communication approaches [19] have been proposed; the cryptanalytic works also have been developed to estimate the security of the proposed chaotic ciphers [18, 19, 26, 31–38]. It has been known that many proposed chaotic cryptosystems can be broken by some cryptanalytic methods, such as most analog chaotic secure communication approaches [36, 37] and some digital chaotic ciphers [19, 31, 32, 34, 35].

For digital chaotic ciphers, the following problems should be considered carefully. **1) What about the encryption speed?** Compared with the conventional ciphers, most chaotic ciphers have much slower encryption speed. There are some different reasons to cause such a problem: multiple iterations for encrypting one plain-block, the use of floating-point arithmetic and complicated chaotic maps. **2) What chaotic map should be used?** Most chaotic ciphers must use specific chaotic maps to ensure the security, which will limit their wider applications. It is desired that a chaotic cipher can work well with a large number of chaotic maps. **3) How to realize the chaotic ciphers, by hardware or software?** A good chaotic cipher should be realized easily by both hardware and software with low cost. In [27], we gave detailed discussions of these problems.

2.3. How to Make Fast Chaotic Ciphers without Loss of Security?

There are two general ways to design digital chaotic ciphers: 1) generating pseudo-random key-stream using chaotic systems to encrypt the plaintext; 2) using the plaintext and/or secret key as the initial conditions and/or control parameters, iterating/inverse-iterating chaotic systems for n times to obtain the ciphertext. The first way corresponds to the stream ciphers and the second to the block ciphers. Investigate currently known digital chaotic ciphers, we can find the following three facts: 1) Most chaotic block ciphers require to iterate the employed chaotic systems for **many** times to make the ciphertext independent of the plaintext, which will markedly reduce the encryption speed. 2) Most chaotic stream ciphers employ one **single** chaotic system to generate pseudo-random numbers to mask the plaintext, which may weaken the capability to potential attacks.

We have discussed the second fact in [27] and suggest using multiple chaotic systems instead of one single chaotic system in chaotic stream cipher. 3) Generally, chaotic stream ciphers run much faster than chaotic block ciphers.

Apparently, it is not very easy to design a fast chaotic cipher without loss of security. This paper suggests a novel solution to this problem – CVES, which is a product cipher of a chaotic stream sub-cipher and a chaotic block sub-cipher. Multiple chaotic systems are used to avoid the potential insecurity of the stream sub-cipher, and an entirely novel encryption scheme is used to avoid the defect of the block sub-cipher. The high security is ensured by the product of the stream sub-cipher and the block sub-cipher. In addition, to further promote the encryption speed of CVES, the following basic principles are also adopted.

1) Using fixed-point arithmetic instead of floating-point arithmetic. As we know, the fixed-point arithmetic runs much faster than the floating-point arithmetic. When the chaotic systems are realized in digital computer with finite precision, the use of fixed-point arithmetic will be very useful to the encryption speed. Furthermore, the fixed-point arithmetic is also helpful to simplify the hardware realization (i.e., reduce the total cost) and improve the portability between different platforms or hardware structures.

2) Using the simplest chaotic systems. More complicated chaotic systems are usually suggested being used to ensure the security of established chaotic ciphers. But the use of complicated chaotic systems will lower the encryption speed twofold: i) the more complicated the chaotic maps, the more time the chaotic iterations will consume; ii) many complicated chaotic systems must run with floating-point arithmetic, which makes the iterations further slower. As we know, the simplest chaotic systems are logistic map and piecewise linear chaotic maps (PLCM). Only one multiplication/division and several additions/subtractions (comparisons) are needed for one iteration of the above chaotic maps. Actually, piecewise linear chaotic maps have been widely used to design digital chaotic ciphers [24–27, 30, 31]. In this paper, piecewise linear chaotic maps are used in CVES. In Sect. 3.4.2, some knowledge about PLCM is given.

3) Using larger encryption unit. For block ciphers, larger encryption unit means larger block size; and for stream ciphers, it means larger size of single key in the generated key-stream. It is obvious that the encryption speed has positive relationship with the size of encryption unit. Of course the size of encryption unit cannot be too large, which will make the secret key too long and the encryption system unfeasible.

2.4. Realization of Digital Chaotic Systems

When chaotic systems are used to construct digital ciphers, one important issue must be carefully considered, which is about the dynamical degradation of **digital** chaotic systems realized in finite precision. It has been found that the dynamical properties of digital chaotic systems are far different from theoretical ones [30, 32, 33, 38–41]. The related problems include short cycle-length, non-ideal distribution and correlation, etc. Up till now, there is not yet an established theory to measure such degradation and direct us how to improve it.

Assume the finite precision is L (bits), there are the following reasons to cause this problem: 1) All values represented with finite precision are binary decimals formulated as $a/2^L(a = 0 \sim 2^L - 1)$. Since the Lebesgue measure of all the binary decimals is zero, they cannot represent the exact dynamics of the chaotic systems defined on real interval. 2) There are only 2^L values to represent the chaotic orbits. So the cycle length of any chaotic orbit will be not larger than 2^L, and almost every one will be much smaller than 2^L. 3) Some quantization errors will be introduced when the digital chaotic systems are iterated, which makes the dynamics of digital chaotic systems badly depart from the theoretical one.

In order to overcome the degradation, several engineering remedies have been proposed: using higher finite precision [32,33], perturbation-based algorithm by pseudo-random number [30,40], and cascading multiple chaotic systems [41]. In this paper, the perturbation-based algorithm proposed in [30] is employed to realize digital chaotic systems. Here, we give a brief description to the perturbation-based algorithm.

A simple pseudo-random number generator (PRNG) is employed to make a small perturbing signal $pt(i)$. The l lowest bits of the chaotic orbits are perturbed by $pt(i)$ with perturbing interval Δ. The perturbation function can be XOR or mod function. The perturbing PRNG can be freely selected in different realizations, such as the maximal length LFSR (m-LFSR) in hardware and the linear congruential generators in software. In order to maintain dynamics of the chaotic systems, $pt(i)$ should be much smaller than the perturbed chaotic orbit, i.e.,

$l \ll n$. But it cannot be too small, or such perturbation cannot provide enough improvement on the dynamical properties of digital chaotic systems.

The perturbation-based algorithm can efficiently improve the cryptographic properties of the digital chaotic systems. Firstly, the cycle length of the chaotic orbit can be controlled by the cycle length of the perturbing PRNG. Assume the cycle length of the PRNG is T, the cycle length of the chaotic orbit can be easily deduced to be $\sigma \cdot \Delta \cdot T$, where σ is a positive integer [30]. If we use one m-LFSR as the perturbing PRNG, whose degree is the finite precision L, the cycle length will be $\sigma \cdot \Delta \cdot (2^L - 1) \gg 2^L$. Secondly, the perturbing signal smoothens the invariant density function slightly, which will not influence the security of the digital chaotic ciphers. Thirdly, the small perturbation can drive the chaotic orbit into a more complicated world because chaos is sensitive to initial conditions. The combination of digital chaos and perturbing signal will make both chaos-theory-based and conventional cryptanalysis much more difficult.

3. CHAOTIC VIDEO ENCRYPTION SCHEME – CVES

The Chaotic Video Encryption Scheme (CVES) is shown in Fig. 1. The plain-video is encrypted *cluster* by *cluster*, where a *cluster* can be one or more video frames. In fact, we can also consider the video stream as a continuous bit-stream without any video format and take fixed-size bits as a *plain-cluster*. Obviously, CVES can be easily extended to other real-time secure applications.

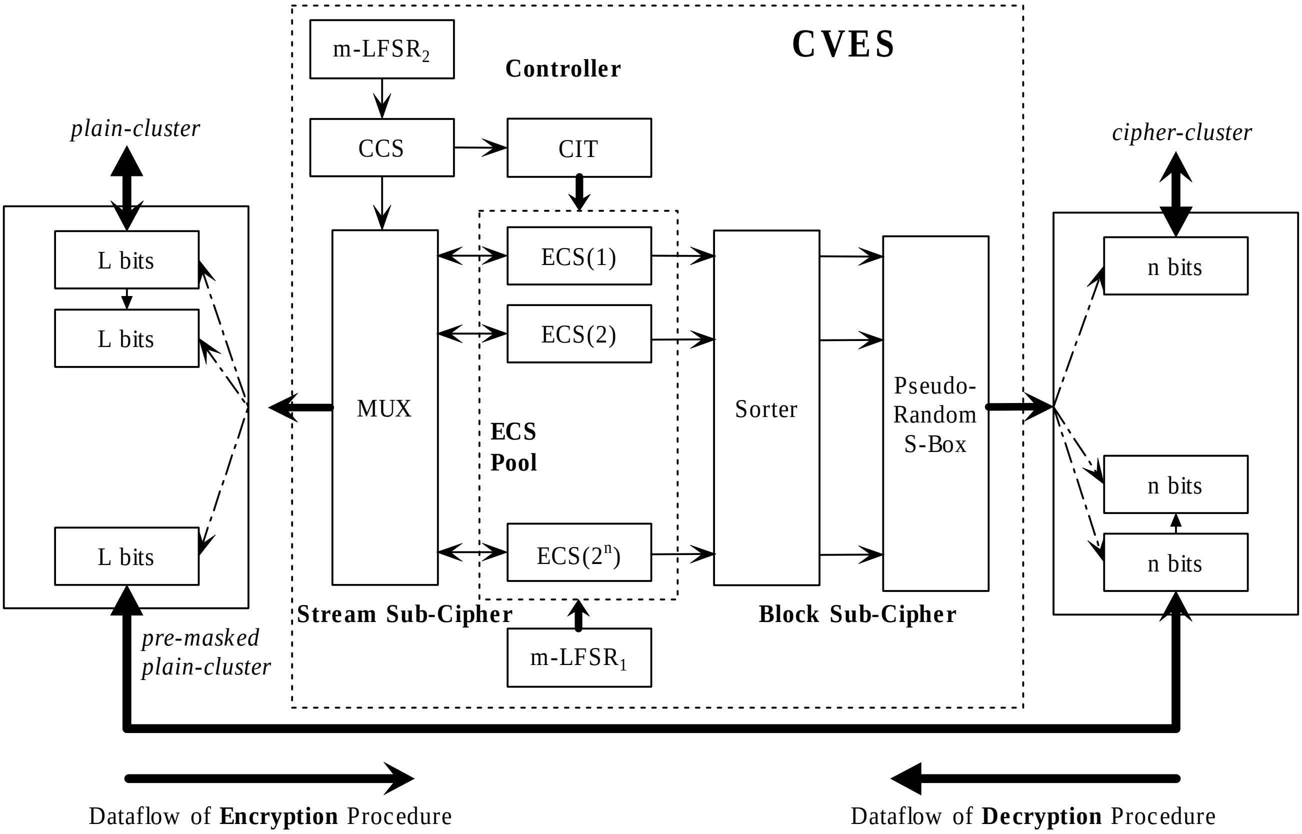


Figure 1. Encryption and Decryption Procedure of CVES

3.1. Components

Before describe the encryption/decryption procedure of CVES, we firstly introduce the components of CVES.

1) **ECS Pool**: 2^n digital chaotic systems, which are called Encryption Chaotic Systems (ECS) and denoted by ECS(1) $\sim$ ECS(2^n), compose the kernel part of CVES – ECS Pool. All 2^n ECS-es are based on a same one-dimensional chaotic maps $F_e(x_e, p_e)$ defined on $I = [0, 1]$, with different control parameters $p_e(1) \sim p_e(2^n)$. All ECS-es are realized in finite computing precision L (bits) with perturbation-based algorithm, and one maximal length LFSR m-LFSR$_1$ is used as the perturbing PRNG. The degree of m-LFSR$_1$ is L_1, and the perturbing intervals of the 2^n ECS-es are $\Delta_e(1) \sim \Delta_e(2^n)$. The current states of the 2^n ECS-es $x_e(1) \sim x_e(2^n)$ are stored in 2^n L-bit memory units.

2) **CCS**: A single digital chaotic systems is used to control the initialization and the chaotic iterations of the 2^n ECS-es. It is called Control Chaotic System (CCS). CCS is also based on a one-dimensional chaotic map $F_c(x_c, p_c)$ defined on $I = [0, 1]$, which can be different from F_e. CCS is also realized in finite precision L (bits) with perturbation-based algorithm, and another maximal length LFSR m-LFSR$_2$ is used as the perturbing PRNG. The degree of m-LFSR$_2$ is L_2, and the perturbing interval of CCS is Δ_c.

3) **CIT**: A Control Information Table (CIT) is used to store the required information in CVES. In regards to the information stored in CIT, please see Sect. 3.2 and 3.3. The CCS and CIT compose the controller part.

4) **Stream Sub-Cipher**: A $2^n \times 1$ MUX controlled by CCS is employed to select an ECS to generate a L-bit chaotic key, which is used to XOR the *plain-cluster* L-bit block by L-bit block. The *plain-cluster* encrypted by the stream sub-cipher is called *pre-masked plain-cluster*.

5) **Block Sub-Cipher**: A $2^n \times 2^n$ L-bit sorter and 2^n n-bit memory units compose a Pseudo-Random S-Box Generator (PRSBG). The generated pseudo-random $n \times n$ S-Box is used to substitute the *pre-masked plain-cluster* n-bit block by n-bit block. Here please note that the PRSBG at encryption end and the one at decryption end should be inverse.

3.2. Encryption/Decryption Procedure

Based on the introduction to the components of CVES, we can describe the encryption procedure as follow. Here, we consider the $x_e(1) \sim x_e(2^n)$, $p_e(1) \sim p_e(2^n)$ and x_c, p_c as L-bit binary integers, not the binary decimals in $[0, 1]$ (under L-bit fixed-point arithmetic), to simplify the description.

Secret key: $K = \{x_c, p_c\}$, the key space is 2^{2L}.

Initialization: **a)** Iterate CCS for 2^n times to obtain 2^n pseudo-random initial conditions $x_{e0}(1) \sim x_{e0}(2^n)$ for all ECS-es. The 2^n initial conditions are stored in CIT. **b)** Iterate CCS for 2^n times again to obtain 2^n pseudo-random control parameters $p_{e0}(1) \sim p_{e0}(2^n)$ for all ECS-es. If there are at least two control parameters are same or any control parameter equals $0^\dagger$, discard all 2^n control parameters and re-initialize the ECS-es. The 2^n control parameters are also stored in CIT. **c)** Sort the 2^n initial conditions $x_{e0}(1) \sim x_{e0}(2^n)$ to generate a pseudo-random rank sequence $l(1) \sim l(2^n)$. The sequence is used to initialize the perturbing intervals as follows: $\Delta_e(i) = P_r(l(i))$, where $P_r(i)$ denotes the i^{th} prime number larger than 2. The 2^n primes numbers $P_r(1) \sim P_r(2^n)$ are pre-calculated and stored in CIT. Then iterate CCS for one or more times to obtain the pseudo-random perturbing interval Δ_c, which should be smaller than 2^n, and also a prime number. **d)** Iterate each ECS(i) for $\eta > \lceil \lambda(i) \rceil$ times, where $\lambda(i)$ is the Lyaponov exponent of ECS(i).

Encryption Procedure: One *plain-cluster* is firstly encrypted by the stream sub-cipher, then by the block sub-cipher. We respectively depict how the two sub-ciphers work. **Stream Sub-Cipher**: The stream sub-cipher encrypts the *plain-cluster* L-bit block by L-bit block. Assume the L-bit plain-block and the L-bit cipher-block are b_p and b_c respectively. The encryption procedure can be denoted as follows: run CCS once, get $I_n = (x_c \bmod 2^n) + 1$, then iterate ECS($I_n$) once; $b_c = b_p \oplus x_e(I_n)$. If the last block has $L' < L$ bits, just encrypt it with the highest L' bits of $x_e(I_n)$. Note that only the selected ECS is iterated once for the encryption of one plain-block, which is very useful to promote the encryption speed and enhance the security of the stream sub-cipher. The encryption procedure goes until the *plain-cluster* exhausts, and then the *pre-masked plain-cluster* is sent to the block sub-cipher for further encryption. **Block Sub-Cipher**: The block sub-cipher is a simple

†If L is not too larger than n, the probability of this event may be rather large to make the initializations slow. For example, when $n = 8, L = 16$, the probability is about 0.4. So it is desired that $L \gg n$, which can ensure the probability is near 0 (when $n = 8, L = 24$, the probability is only about 0.002).

substitution cipher with time-variant $n \times n$ S-Box pseudo-randomly controlled by the 2^n ECS-es. The S-Box is generated as follows: after the last *plain-cluster* is encrypted and before the current *plain-cluster* is encrypted by the stream sub-cipher, sort the 2^n current states of all ECS-es, then the indexes of the sorted states and the indexes of the original states compose the S-Box for encryption. Use the generated S-Box to substitute the *plain-cluster* pre-masked by the stream sub-cipher n-bit block by n-bit block[‡]. After the encryption of the current *plain-cluster* is complete, the stream sub-cipher continues to encrypt the next *plain-cluster*. The encryption procedure goes until the plain-video exhausts.

Decryption Procedure: Decryption is the inverse of the encryption (see Fig. 1). The *cipher-cluster* is firstly decrypt by the block sub-cipher, where the S-Box is the inversion of the one for encryption. Then the *pre-decrypted cipher-cluster* is decrypted by the stream sub-cipher.

3.3. Modified CVES Supporting Random Retrieval – RRS-CVES

In the above CVES, random retrieval of cipher-video cannot be supported, since the chaotic orbits of CCS and all ECS-es cannot be predicted only from the position of a *cipher-cluster* in the whole cipher-video. To decrypt a *cipher-cluster*, we must decrypt all *cipher-clusters* before it. That is to say, the original CVES can only support sequent retrieval, not random retrieval. Fortunately, we can make some modifications on the original CVES to add this function. The modified CVES is called Random-Retrieval-Supported CVES (RRS-CVES).

Initialization: Besides the initialization operations a)∼d) in original CVES, the following three operations are added. **a′) Generating Reset Information**: Run the CCS for $2+2^n$ times to generate two L-bit pseudo-random numbers p_+, x_+ and 2^n m-bit pseudo-random numbers $\tau_e(1) \sim \tau_e(2^n)$[§], which are also stored in CIT. Here, $\tau_e(i)(i = 1 \sim 2^n)$ should satisfy the following requirements: $\gcd(\tau_e(i), 2) = 1$ and $\tau_e(i) \geq \tau_{min}$, where τ_{min} should not be very small. In regards to the selections of m and τ_{min}, we will give some details in Sect. 3.4. The $2+2^n$ extra pseudo-random numbers are used to reset the 2^n ECS-es. **b′) Generating Sequence of Chaotic Iterations**: Sort the 2^n pre-defined control parameters $p_{e0}(1) \sim p_{e0}(2^n)$ to generate a rank sequence $r_e(1) \sim r_e(2^n)$, where $r_e(i) = 1 \sim 2^n$. The sequence is stored in CIT and will be used to control the chaotic iterations of the 2^n ECS-es. **c′) Initializing Iteration Counters**: 2^n L-bit memory units $C_1(1) \sim C_1(2^n)$ are used to store the iteration numbers of the 2^n ECS-es. Another 2^n L-bit memory units $C_2(1) \sim C_2(2^n)$ are used to store the reset numbers of the 2^n ECS-es. Set the $2 \cdot 2^n$ L-bit memory units to zeros.

To sum up, for RRS-CVES, there are the following predefined data stored in CIT: 1) *Initial Conditions* – $x_{e0}(1) \sim x_{e0}(2^n)$; 2) *Control Parameters* – $p_{e0}(1) \sim p_{e0}(2^n)$; 3) *Perturbing Intervals* – $\Delta_e(1) \sim \Delta_e(2^n)$; 4) *Prime Numbers List* – $P_r(1) \sim P_r(2^n)$; 5) *Reset Information* – $\tau_e(1) \sim \tau_e(2^n)$ and p_+, x_+; 6) *Rank Sequence for Chaotic Iterations* – $r_e(1) \sim r_e(2^n)$; 7) *Iteration/Reset Counters* – $C_1(1) \sim C_1(2^n)$ and $C_2(1) \sim C_2(2^n)$. In original CVES, only the first four ones are required.

Encryption Procedure: In RRS-CVES, the stream sub-cipher is modified with reset mechanism, but the block sub-cipher is untouched at all. In RRS-CVES, $r_e(1) \sim r_e(2^n)$ is used to select an ECS to encrypt the current plain-block (for the i^{th} plain-block, select $ECS(r_e(i \bmod 2^n))$ as the current ECS), instead of iterating CCS in original CVES. For any ECS(i), after it runs once, increase its iteration counter by 1: $C_1(i) + +$. If $C_1(i) \bmod \tau_e(i) = 0$, **reset** ECS($i$) as follows: $x_{e0}(i) = (x_{e0}(i) + x_+) \bmod 2^L$, $x_e(i) = x_{e0}(i)$, and $C_1(i) = 0, C_2(i) + +$. If $C_2(i) \bmod \tau_e(i) = 0$, **reset** ECS($i$) as follows: $p_{e0}(i) = (p_{e0}(i) + p_+) \bmod 2^L$, $p_e(i) = p_{e0}(i)$ and $C_1(i) = C_2(i) = 0$. **Decryption Procedure**: Make the same modifications like encryption procedure.

From the encryption procedure of RRS-CVES, we can see the following fact. Consider the cipher-video as a L-bit data-stream, if we know the position of one *cipher-cluster* in the L-bit stream, it is possible to reconstruct the corresponding states of all ECS-es and then decrypt the *cipher-cluster*, within considerable maximal time-out. Assume the position of the *cipher-cluster* is I_L, i.e., the total number of L-bit cipher-blocks before the *cipher-cluster* is I_L. We can reconstruct all 2^n ECS-es as follows:

[‡] The size of any plain-cluster should be divided exactly by n, otherwise some synchronization marks must be added and the cipher-video must have some specific format. When $n = 8$, it is rather easy to satisfy this requirement.

[§] An m-bit number x' can be obtained from L-bit chaotic states x as follows: $x' = x \bmod 2^m$ or $x' = x >> (L - m)$.

1) $I_{ECS} = (I_L \bmod 2^n) + 1$, $I'_L = I_L/2^n$;
2) $i = 1 \sim 2^n$: $I_{c1}(i) = I'_L/\tau_e(i)$, $I'_{c1}(i) = I'_L \bmod \tau_e(i)$, $I_{c2}(i) = I_{c1}/\tau_e(i)$;
3) $i = 1 \sim 2^n$: $x_e(i) = (x_{e0}(i) + I_{c1}(i) \cdot x_+) \bmod 2^L$, $p_e(i) = (p_{e0}(i) + I_{c2}(i) \cdot p_+) \bmod 2^L$;
4) $i = 1 \sim I_{ECS}$: Run ECS(i) for $I_{c1}(i)' + 1$ times, and $i = I_{ECS} + 1 \sim 2^n$: Run ECS(i) for $I_{c1}(i)'$ times;
5) Decrypt the cipher-cluster as normal procedure.

We can see some pre-computation is used to reconstruct the current states of the 2^n ECS-es. Thus, the maximal time-out for random retrieval will be determined by the $\sum_{i=1}^{I_{ECS}}(I'_{c1}(i) + 1) + \sum_{i=I_{ECS}+1}^{2^n} I'_{c1}(i)$ times chaotic iterations in step 4). Assume the consuming time for one chaotic iteration is τ_0, the maximal time-out τ will satisfy $2^n \cdot \tau_{min} \leq \tau/\tau_0 \leq 2^{n+m}$. In the Sect. 4.1, we will further discuss this problem.

3.4. Configure CVES and RRS-CVES

3.4.1. Configuration Parameters

To optimize CVES and RRS-CVES in practical applications, some parameters should be carefully configured.

The most important system parameters are L and n. 1) L: Since the key space is 2^{2L}, L should be large enough to provide high security. In addition, to simplify the realization of CVES in digital computers, $L = 32$ or 64 is suggested. 2) n: Apparently, the realization complexity of CVES/RRS-CVES has positive exponential relation with n ($O(2^n \cdot L)$ bits memory is needed). Thus, n cannot be too large, and we suggest $n = 8$.

It has been known that the perturbing parameters of the 2^n ECS-es and CCS are very useful to improve the degradation of digital chaotic maps. In [30], the authors stated that the perturbing intervals can be very large, such as 10^6 when $L = 40$. But we argued that they cannot be too large from strict cryptographic consideration[¶]. When $n = 8$, the maximal perturbing interval is 1021 (256^{th} prime number larger than 2), which is acceptable.

The size of one *cluster* is another important parameter of CVES/RRS-CVES. Although the size needn't be fixed, the fixed-size *cluster* is useful to simplify the realization and the performance estimation. Assume a *cluster* contains P_{max} L-bit blocks. We will conclude that P_{max} can be used to adjust the encryption speed. Generally speaking, the larger P_{max} is, the faster the encryption speed will be. Further details will be given in Sect. 4.1. If the *cluster* size is variant, the average size $\overline{P}_{max}$ can be used to estimate the encryption speed.

For RRS-CVES, m and τ_{min} are used to control the reset operations of the 2^n ECS-es. Generally, we suggest $m \leq n$ and $\tau_{min} \geq 2^{n/2}$. Then the maximal time-out will satisfy $2^{3n/2} \leq \tau/\tau_0 \leq 2^{2n}$. Since n is not too large, such a maximal time-out can be acceptable in most real-time applications (see Sect. 4.1 for more details).

3.4.2. Selection of the Chaotic Maps

As we know [38, 42], the piecewise linear chaotic maps (PLCM) have perfect dynamical properties, i.e., cryptographic properties. In addition, the most complicated components required for the digital PLCM-s are digital dividers, which are easily realized using fixed-point arithmetic both by hardware and software. Therefore, PLCM-s are suggest being used to construct the ECS pool and CCS, which will optimize both the encryption speed and the realization. Of course, for digital PLCM-s, some special problems should be considered carefully to avoid possible insecurity (see [38] for more details), but these problems are essentially avoided in CVES.

In CVES/RRS-CVES, we suggest the following PLCM used in [27, 31]:

$$F(x,p) = \begin{cases} x/p, & x \in [0,p) \\ (x-p)/(\frac{1}{2}-p), & x \in [p, \frac{1}{2}] \\ F(1-x,p), & x \in (\frac{1}{2}, 1] \end{cases}, \tag{1}$$

where $0 < p < \frac{1}{2}$. The above map is one of the simplest PLCM satisfying the following properties: 1) It is ergodic, mixing and exact; 2) It has uniform invariant density function $f(x) = 1$; 3) The auto-correlation function of the chaotic orbit $\tau(n) = \delta(n)$, where $\tau(n) = \frac{1}{\sigma}\lim_{N\to\infty}\frac{1}{N}\sum_{i=0}^{N-1}(x_i - \bar{x})(x_{i+n} - \bar{x})$ ($\bar{x}, \sigma$ are the mean value and the variance of x respectively). Please see [43] for the exact definitions of related notions: ergodicity, mixing, exactness, and the invariant density function.

[¶] Consider the following fact: even when L is large enough, there always exist some chaotic orbits leading to short cycle length. An extreme example is the digital tent map, any orbits from $a/2L$ will lead to zero after at most L iterations.

4. PERFORMANCE ESTIMATION

4.1. Speed

We can estimate the encryption speed based on the speed of the two sub-ciphers. Generally speaking, the hardware system of CVES/RRS-CVES will run faster than the software system, considering the parallel mechanism can be used in hardware realization. Without loss of generality, assume all ECS-es and CCS are realized by Eq. (1), and the *cluster* size is fixed: $P_{max} \cdot L$ bits.

Hardware realization: Generally speaking, one L-bit fixed-point division consumes L clock cycles, then one chaotic iteration approximately consumes L clock cycles. Consider the multiple pipelining techniques can be used here, the stream sub-cipher encrypts one L-bit plain-block per L clock cycles. Assume the time consuming by the sorter is τ_s (clock cycles), for the most time-consuming sorter, $\tau_s = 2^n \cdot (2^n - 1)$; and for the optimized sorter using the quick sorting algorithm, $\tau_s = n \cdot 2^n$. In addition, the block sub-cipher encrypts one n-bit per one clock cycles. To sum up, for one *plain-cluster*, the total consuming clock cycles is $L \cdot P_{max} + \tau_s + P_{max} \cdot L/n$, where τ_s denotes the time consuming by the sorter. If the basic clock frequency is f_b MHz, the final speed of CVES will be $f_b \Big/ \left(1 + \frac{1}{n} + \frac{\tau_s}{L \cdot P_{max}}\right)$ Mbps. Apparently, P_{max} can adjust the encryption speed. When $L = 32, n = 8$, $\tau_s = n \cdot 2^n$ and $P_{max} = n \cdot 2^n = 2048$, the encryption speed is $\frac{32}{37} f_b$ Mbps. Such a speed is faster than many fast conventional ciphers. Of course, the estimated speed here is just a theoretical value, and the actual speed will be somewhat different from the exact hardware design.

From the above discussion, we can see the actual encryption speed is chiefly determined by the stream sub-cipher when $P_{max} \geq \tau_s/L$, and by the sorter in the block sub-cipher if $P_{max} < \tau_s/L$. For most applications of CVES/RRS-CVES, we suggest $P_{max} = n \cdot 2^n$. When $L \geq 32, n = 8$, P_{max} will be larger than τ_s/L (even for the most time-consuming sorter, $\tau_s/L = 2^n \cdot (2^n - 1)/L < n \cdot 2^n$). Hence, in most conditions, the sorter can be realized chiefly from the consideration of simplifying the hardware scale, not promoting the sorting speed.

Software realization: Software realization will be much slower than hardware realization since there is no parallel mechanism in software realization. It can be approximately evaluated that the speed of software realization will be several times slower than the hardware realization. An experimental system is designed with Microsoft® Visual C++ to test the actual speed under Microsoft® Windows™ platform. The final speed is about 1/10 of the CPU frequency. For example, on a PC with a 667MHz Pentium®III CPU, the speed is about 60Mbps; on a PC with a 223MHz Celeron® CPU, the speed is about 20Mbps. Such a speed is rather high for a software cipher. In the experimental system, we find that the stream sub-cipher, whose kernel is the digital PLCM-s, plays crucial role on the final speed.

Finally, let us discuss the time consuming on the initialization and the time-out problem of RRS-CVES.

For the initialization of CVES, the most time-consuming procedure is about $(2 + \eta) \cdot 2^n$ chaotic iterations and a sorting procedure of 2^n data, which means $(2 + \eta) \cdot 2^n \cdot L + n \cdot 2^n$ clock cycles. Assume $\eta = 4 > \lceil \lambda \rceil = 2$, the consuming time will be about $(6L + n) \cdot 2^n$ clock cycles. When $L = 32, n = 8$, the time is $51,200$ clock cycles. Similarly, for RRS-CVES, the consuming time of chief procedure can be calculated to be about $(3 + \eta) \cdot 2^n \cdot L + 2 \cdot n \cdot 2^n = ((3 + \eta) \cdot L + 2n) \cdot 2^n$ clock cycles. Assume $\eta = 4$ and $L = 32, n = 8$, the time will be $61,400$ clock cycles. Obviously, the initialization will not consume too much time.

In regards to the maximal time-out for random retrieval of RRS-CVES, we have pointed out that $2^{3n/2} \leq \tau/\tau_0 \leq 2^{2n}$ in Sect. 3.4.1. When $L = 32, n = 8$, we can get $2^{17} \leq \tau \leq 2^{21}$ clock cycles (consider τ_0 equals to L clock cycles). If $f_b \geq 200$ MHz, the maximal time-out will be controlled within 10ms.

4.2. Security

4.2.1. Essentially Features to Avoid Potential Attacks

In CVES/RRS-CVES, there are three essential features to ensure the high security. 1) The stream sub-cipher is made of 2^n asymptotically independent chaotic maps (ECS Pool), and the sequence of chaotic iterations is controlled pseudo-randomly by another independent chaotic map (CCS). The above two facts make the statistical cryptanalysis much more difficult. 2) For different *cluster*, the entirely different S-Box is pseudo-randomly determined by the current 2^n states of ECS pool, which makes CVES/RRS-CVES similar to a one-time-a-pad cryptosystem. 3) The product of the stream sub-cipher and the block sub-cipher makes the known-plaintext and chosen-plaintext attack impossible. Extremely, even when the 2^n states of ECS pool are all known (then the related S-Box is also known), it is impossible to derive the secret key $K = \{x_c, p_c\}$ since the key is separated from the 2^n states by previous chaotic iterations of 2^n ECS-es. Please recall the initialization procedure, $\eta \geq \lceil \lambda \rceil$ pre-iterations are required, which is used to avoid the above attack if the 2^n chaotic states of the first *cluster* are known. Actually, in practice, such a attack is rather difficult since the current chaotic states cannot be obtained from a pair of *plain-cluster* and *cipher-cluster*[‖].

There is one "severe defect" in CVES: the different initial conditions may generate entirely identical orbits, which is the natural result of the fact that all chaotic maps are multi-to-one function. For example, the piecewise linear chaotic map $F(x)$ denoted by Eq. (1) is 4-to-1 map. $\forall y \in (0, 1)$, there will be 4 values (the pre-images of y) satisfying $f(x) = y$. Therefore, the chaotic orbits from the 4 initial conditions are identical. Such a defect may be used in brute-force attacks to lessen attack complexity: when p_c is searched exhaustively, one can only search 1/4 values of x_c, not all the possible values, then the attack complexity will be only 1/4 of ideal one, and the key entropy will decrease by 2 bits. Fortunately, for digital chaotic maps realized with perturbation-based algorithm, this problem is not friendly at all to illegal eavesdroppers, although the above result is absolutely true in real number field. When the chaotic systems are realized with pseudo-random perturbation, the chaotic orbits are improved to satisfy approximate uniform distribution, i.e., 2^n different chaotic inputs generates 2^n different chaotic outputs. Of course, it is impossible to essentially avoid this problem, so the key entropy of CVES will be a little less than $2L$, not exact $2L$.

4.2.2. Cryptographic Properties of Ciphertext

As a good cipher, CVES have the following basic cryptographic properties.

1) Balance: Since the chaotic orbits of the 2^n ECS-es have uniform distribution function, then the *plain-clusters* pre-masked by the stream sub-cipher will also have uniform distribution function. Consider the block sub-cipher subsequently substitutes the *pre-masked plain-clusters*, which cannot change the uniform distribution because the substitution operation with S-Box is a surjective map. Consequently, the cipher-video will be balanced.

2) Avalanche Property with Respect to Secret Key: If the secret key $K = \{x_c, p_c\}$ changes only one bit, then the initial conditions or control parameters of ECS-es will change much because CCS' sensitivity to initial conditions and control parameters. The initial conditions and/or control parameters change a little, the ciphertext will change much, which implies the avalanche property of ciphertext.

4.2.3. Cycle Length of the Stream Sub-Cipher

In CVES/RRS-CVES, both the stream sub-cipher and the block sub-cipher are based on the digital orbits of the 2^n ECS-es and CCS. Consider the current 2^n states of ECS pool as a 2^n-dimensional vector (here we call it Chaotic Vector) , the cycle length of this vector will be a crucial factor to measure the security of the whole system. The cycle length should be large enough to avoid repeated encryption pattern.

From [30], we can easily get the cycle length of CCS: $T_c = \sigma_c \cdot \Delta_c \cdot (2^L - 1)$, and the cycle length of 2^n ECS-es: $T_e(i) = \sigma_e(i) \cdot \Delta_e(i) \cdot (2^L - 1)(i = 1 \sim 2^n)$, where $\{\sigma_e(i)\}_{i=1}^{2^n}, \sigma_c$ are positive integers. Although it is difficult to measure the exact cycle length of Chaotic Vector, we can derive its order:

$$\mathrm{lcm}\left(\sigma_e(1), \cdots, \sigma_e(2^n), \sigma_c\right) \cdot \left(2^{L_1} - 1\right) \cdot \left(2^{L_2} - 1\right) \cdot \mathrm{lcm}\left(\prod\nolimits_{i=1}^{2^n} \Delta_e(i), \Delta_c\right). \tag{2}$$

When $n = 8$, $\mathrm{lcm}\left(\prod_{i=1}^{2^n} \Delta_e(i), \Delta_c\right) > 2^{2297}$. Such a length is **HUGE** enough for any secure applications.

[‖] Consider the *cluster* size cannot be too large, it will be difficult to reveal the pseudo-random S-Box and the current chaotic states of ECS pool via any statistical cyrptanalytic methods.

4.2.4. Pseudo-Random S-Boxes of the Block Sub-Cipher

In this sub-subsection, we discuss the statistical properties of S-Boxes pseudo-randomly generated by the chaotic states of ECS pool. Because the 2^n ECS-es have the same invariant density function $f(x) = 1$ on the same interval $I = [0,1]$, the generated S-Boxes by sorting the 2^n chaotic states can be depicted as the *rank statistics* of 2^n random variables with identical and independent distribution functions. Let $R(1), R(2), \cdots, R(2^n)$ denote the rank statistics, then the following fact is true: for any permutation $\{i(1), i(2), \cdots, i(2^n)\}$ on $\{1, \cdots, 2^n\}$, $P\{R(1) = i(1), R(2) = i(2), \cdots, R(2^n) = i(2^n)\} = \frac{1}{2^n!}$, i.e., the rank statistics is equiprobable and symmetric [44]. It is very useful to construct ciphers with perfect cryptographic properties. Of course, there will be some weak S-Boxes in all $2^n!$ possible ones, but the number will be much smaller than the strong ones. What's more, the product of stream sub-cipher and block sub-cipher makes the detection of weak S-Boxes difficult. Under the worse condition, if one weak S-Box is broken, only the related *plain-cluster* will be influenced, all other *plain-clusters* with different S-Boxes will still keep secure.

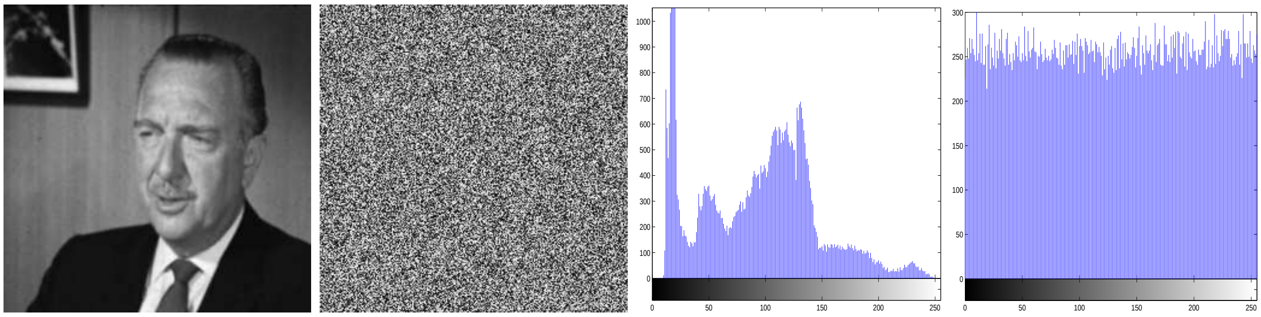
4.3. Realization Complexity

Because generally L and n can be divided exactly by 8, the software realization of CVES/RRS-CVES will be very simple, since 8-bit byte is supported well by almost all programming languages under different platforms. Therefore, we focus on the realization complexity by hardware in this subsection.

The most important hardware devices are one L-bit digital dividers to iterate the digital chaotic systems and a $2^n \times 2^n$ sorter. Other devices include: two m-LFSR-s, and some memory units to store the CIT, current 2^n chaotic states and the generated S-Box. For CVES, the CIT needs $4 \cdot 2^n$ L-bit memory units and the S-Box needs 2^n n-bit memory units. For RRS-CVES, the CIT needs $8 \cdot 2^n$ L-bit memory units and the S-Box still needs 2^n n-bit memory units. When $L = 64$, $m = 8$, the total number of memory units of CVES is about $4 \cdot 2^n \cdot L + n \cdot 2^n = 67,584$ bits $= 8,448$ bytes. For RRS-CVES, the total number is $9 \cdot 2^n \cdot L + n \cdot 2^n = 149,504$ bits $= 18,688$ bytes. In addition, a data buffer whose size equals to the *cluster* size may be also needed to facilitate the substitution of the block sub-cipher after the encryption made by the stream sub-cipher.

In CVES/RRS-CVES, the sorter is the most complicated device. As we have mentioned in Sect. 4.1, how to reduce the realization complexity and cost is the chief consideration in the realization of the sorter, since the final speed is not chiefly determined by the sorter (the cluster size is generally larger than $n \cdot 2^n / L$).

4.4. Experiments



a) One Plain-Frame

b) Cipher-Frame Encrypted by CVES

c) Histogram of the Plain-Frame

d) Histogram of the Cipher-Frame

Figure 2. Uncompressed Digital Video Encrypted with CVES

For a uncompressed digital video, we test the practical performance of CVES. In Fig. 2, we give the comparison of one plain-frame and the cipher-frame. We can see the plain-image is encrypted to a cipher-image with uniform histogram, which implies the perfect cryptographic properties of CVES.

5. CONCLUSION

In this paper, we propose a new encryption scheme (Chaotic Video Encryption Scheme – CVES) for real-time digital video based on multiple digital chaotic maps, which can overcome the problem between the encryption speed and high security existing in other known video encryption systems. CVES is a product cipher that contains a stream sub-cipher and a block sub-cipher. CVES can be extended to RRS-CVES, an enhanced version supporting random retrieval of cipher-video with considerable maximal time-out. Detailed analyses have shown that CVES/RRS-CVES has fast speed and high security, and can be realized easily by both hardware and software. In the future, we will investigate the further issues about the security and realization of CVES/RRS-CVES and try to complete the standard realization packages with VLSI (hardware) and C++ language (software).

REFERENCES

1. B. Schneier, *Applied Cryptography – Protocols, algorithms, and souce code in C*, John Wiley & Sons, Inc., New York, second ed., 1996.
2. A. Şaman Tosun, "Lightweight security mechanisms for wireless video transmission," in *Proc. Int. Conf. Information Technology*, pp. 157–161, 2001.
3. H. Cheng and X. Li, "Partial encryption of compressed images and videos," *IEEE Trans. Signal Processing* **48**(8), pp. 2439–2451, 2000.
4. S. U. Shin, K. S. Sim, and K. H. Rhee, "A secrecy scheme for MPEG video data using the joint of compression and encryption," in *Proc. Int. Information Security Workshop*, *Lecture Notes in Computer Science* **1729**, pp. 191–201, Springer-Verlag, (Berlin), 1999.
5. L. Qiao and K. Nahrstedt, "A new algorithm for MPEG video encryption," in *Proc. Int. Conf. Imaging Science, Systems, and Technology*, pp. 21–29, 1997.
6. X. Wu and P. W. Moo, "Joint image/video compression and encryption via high-order conditional entropy coding of wavelet coefficients," in *Proc. Int. Conf. Multimedia Computing and Systems*, pp. 908–912, 1999.
7. C. Shi, S.-Y. Wang, and B. Bhargava, "MPEG video encryption in real-time using secret key cryptography," in *Proc. Int. Conf. Parallel and Distributed Processing Techniques and Applications*, 1999.
8. C. Shi and B. Bhargava, "A fast MPEG video encryption algorithm," in *Proc. ACM Multimedia 98*, pp. 81–88, 1998.
9. L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently," in *Proc. ACM Multimedia 96*, pp. 219–230, 1996.
10. I. Agi and L. Gong, "An empirical study of secure MPEG video transmissions," in *Proc. Internet Society Sym. Network and Distributed Systems Security*, pp. 137–144, 1996.
11. Y. Li, Z. Chen, S.-M. Tan, and R. H. Campbell, "Security enhanced MPEG player," in *Proc. Int. Workshop Multimedia Software Development*, pp. 169–175, 1996.
12. G. A. Spanos and T. B. Maples, "Performance study of a selective encryption scheme for the security of networked, real-time video," in *Proc. Int. Conf. Computer Communications and Networks*, pp. 2–10, 1995.
13. L. Qiao and K. Nahrstedt, "Comparison of MPEG encryption algorithms," *Computers & Graphics* **22**(4), pp. 437–448, 1998.
14. L. Qiao and K. Nahrstedt, "Is MPEG encryption by using random list instead of Zig-Zag order secure?," in *Proc. Int. Sym. Consumer Electronics*, pp. 226–229, 1997.
15. T. Uehara and R. Safavi-Naini, "Chosen DCT coefficients attack on MPEG encryption schemes," in *Proc. IEEE Pacific Rim Conf. Multimedia*, pp. 316–319, 2000.
16. H. Bai-Lin, *Starting with Parabolas: An Introduction to Chaotic Dynamics* (In Chinese), Shanghai Scientific and Technological Education Publishing House, Shanghai, China, 1993.
17. R. Brown and L. O. Chua, "Clarifying chaos: Examples and counterexamples," *Int. J. Bifurcation and Chaos* **6**(2), pp. 219–249, 1996.
18. L. Kocarev, T. S. Goce Jakimoski, and U. Parlitz, "From chaotic maps to encryption schemes," in *Proc. IEEE Int. Symposium Circuits and Systems* **4**, pp. 514–517, 1998.
19. G. Álvarez, F. Monotoya, G. Pastor, and M. Romera, "Chaotic cryptosystems," in *Proc. IEEE Int. Carnahan Conf. Security Technology*, pp. 332–338, 1999.

20. J. Fridrich, “Symmetric ciphers based on two-dimensional chaotic maps,” *Int. J. Bifurcation and Chaos* **8**(6), pp. 1259–1284, 1998.
21. C. E. Shanon, “Communication theory of secrecy systems,” *Bell Sys. Tech. J.* **28**(4), pp. 656–715, 1949.
22. R. Matthews, “On the derivation of a ‘chaotic’ encryption algorithm,” *Cryptologia* **XIII**(1), pp. 29–42, 1989.
23. L. M. Pecora and T. L. Carroll, “Synchronization in chaotic systems,” *Physical Review Letters* **64**(8), pp. 821–824, 1990.
24. T. Habutsu, Y. Nishio, I. Sasase, and S. Mori, “A secret key cryptosystem by iterating a chaotic map,” in *Advances in Cryptology - EuroCrypt’91*, *Lecture Notes in Computer Science* **0547**, pp. 127–140, Spinger-Verlag, (Berlin), 1991.
25. E. Alvarez, A. Fernández, P. García, J. Jiménez, and A. Marcano, “New approach to chaotic encryption,” *Physics Letters A* **263**, pp. 373–375, 1999.
26. S. Li, X. Mou, and Y. Cai, “Improving security of a chaotic encryption approach,” *Physics Letters A* **290**(3-4), pp. 127–133, 2001.
27. L. Shujun, M. Xuanqin, and C. Yuanlong, “Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography,” in *Progress in Cryptology - INDOCRYPT 2001*, *Lecture Notes in Computer Science* **2247**, pp. 316–329, Springer-Verlag, (Berlin), 2001.
28. J. Scharinger, “Fast encryption of image data using chaotic kolmogrov flows,” *J. Electronic Imaging* **7**(2), pp. 318–325, 1998.
29. M. Miyamoto, K. Tanaka, and T. Sugimura, “Truncated baker transformation and its extension to image encryption,” in *Proc. SPIE* **3814**, pp. 13–25, 1999.
30. S. Tao, W. Ruili, and Y. Yixun, “Perturbance-based algorithm to expand cycle length of chaotic key stream,” *Electronics Letters* **34**(9), pp. 873–874, 1998.
31. H. Zhou and X.-T. Ling, “Problems with the chaotic inverse system encryption approach,” *IEEE Trans. Circuits and Systems I* **44**(3), pp. 268–271, 1997.
32. D. D. Wheeler, “Problems with chaotic cryptosystems,” *Cryptologia* **XIII**(3), pp. 243–250, 1989.
33. D. D. Wheeler and R. Matthews, “Supercomputer investigations of a chaotic encryption algorithm,” *Cryptologia* **XV**(2), pp. 140–151, 1991.
34. E. Biham, “Cryptoanalysis of the chaotic-map cryptosystem suggested at EuroCrypt’91,” in *Advances in Cryptology - EuroCrypt’91*, *Lecture Notes in Computer Science* **0547**, pp. 532–534, Spinger-Verlag, (Berlin), 1991.
35. G. Alvarez, F. Montoya, M. Romera, and G. Pastor, “Cryptanalysis of a chaotic encryption system,” *Physics Letters A* **276**, pp. 191–196, 2000.
36. K. M. Short, “Signal extraction from chaotic communications,” *Int. J. Bifurcation and Chaos* **7**(7), pp. 1579–1597, 1997.
37. M. J. Ogorzatek and H. Dedieu, “Some tools for attacking secure communication systems employing chaotic carriers,” in *Proc. IEEE Int. Symposium Circuits and Systems 1998*, **4**, pp. 522–525, IEEE, 1998.
38. S. Li, Q. Li, W. Li, X. Mou, and Y. Cai, “Statistical properties of digital piecewise linear chaotic maps and their roles in cryptography and pseudo-random coding,” in *Cryptography and Coding - 8th IMA Int. Conf. Proc.*, *Lecture Notes in Computer Science* **2260**, pp. 205–221, Springer-Verlag, (Berlin), 2001.
39. J. Palmore and C. Herring, “Computer arithmetic, chaos and fractals,” *Physica D* **42**, pp. 99–110, 1990.
40. Z. Hong and L. Xieting, “Realizing finite precision chaotic systems via perturbation of m-sequences,” *Acta Eletronica Sinica* (In Chinese) **25**(7), pp. 95–97, 1997.
41. G. Heidari-Bateni and C. D. McGillem, “A chaotic direct-sequence spread-spectrum communication system,” *IEEE Trans. Communications* **42**(2/3/4), pp. 1524–1527, 1994.
42. A. Baranovsky and D. Daems, “Design of one-dimensional chaotic maps with prescribed statistical properties,” *Int. J. Bifurcation and Chaos* **5**(6), pp. 1585–1598, 1995.
43. A. Lasota and M. C. Mackey, *Chaos, Fractals, and Noise - Stochastic Aspects of Dynamics*, Springer-Verlag, New York, second ed., 1997.
44. Department of Applied Mathematics, Tsinghua University, China, *The Handbook of Modern Applied Mathematics*, vol. Probability Theory and Stochastic Process, Tsinghua University Press, Beijing, China, 2000.