

# A Permutation-Based Correlation-Preserving Encryption Method for Digital Videos

Daniel Socek<sup>1</sup>, Hari Kalva<sup>1</sup>, Spyros S. Magliveras<sup>2</sup>,  
Oge Marques<sup>1</sup>, Dubravko Čulibrk<sup>1</sup>, and Borko Furht<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering, and  
<sup>2</sup> Department of Mathematical Sciences,  
Florida Atlantic University, Boca Raton FL 33431, USA

**Abstract.** In this work we present and analyze a conceptually novel encryption method for digital videos. This type of encryption has several advantages over currently available video encryption algorithms. We analyze both security and performance aspects of our method, and show that the method is efficient and secure from a cryptographic point of view. Even though the approach is currently feasible only for a selected class of video sequences and video codecs, the method is promising and future investigations might reveal its broader applicability.

## 1 Introduction

In most digital video security architectures, video encryption plays a key role in ensuring the confidentiality of the video transfer. However, conventional, general-purpose encryption algorithms (such as AES) are not suitable for video encryption in many digital video applications [10, 5, 3, 7], mainly since these algorithms do not conform to various video application requirements that we discuss next. In order to overcome this problem, a significant number of video encryption algorithms specifically designed for digital videos have been proposed (e.g. [8, 15, 10, 4, 3]).

In general, applying a well-established, general-purpose, symmetric-key encryption algorithm to a video sequence is a good idea from a security point of view. However, there are applications with requirements that are not supported with conventional encryption methods. Thus, encryption algorithms specifically designed to support these requirements are desirable. These aspects include the following: (1) *level of security and perception*, (2) *format-compliance*, (3) *degree of bitstream expansion*, and (4) *error-tolerance*.

To identify an optimal level of security, we have to carefully compare the cost of the multimedia information to be protected versus the cost of the protection itself. Light-weight encryption, often called *degradation*, may be sufficient for distributing multimedia content of low value. Often, degradation intentionally preserves some perceptual information with visual quality that is unacceptable for entertainment purposes. This type of encryption is referred to as *perceptual encryption*. If the video contains sensitive industrial, governmental or military

information, then the cryptographic strength must be substantial and no perceptual information should be preserved after encryption.

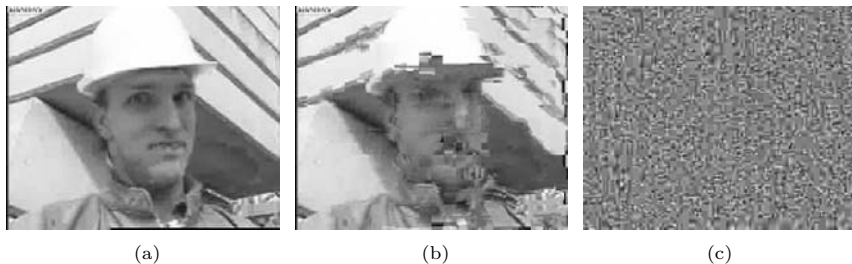
In many applications it is desired that the encryption algorithm preserves the video compression format. In other words, after encrypting the encoded video, ordinary decoders can still decode it without crashing. This is a quite important aspect in digital video broadcasting applications where an encrypted video is broadcast to all system users. This property of an encryption algorithm is often called *format-compliance* (also called *transparency*, *transcodability* or *syntax-awareness*). When feeding the decoder with the format-compliant encrypted data the produced output appears either perceivable but somewhat distorted or non-perceivable and random, depending on the type of encryption (see Fig. 1).

It is required in many applications that the encryption transformation preserves the size of a bitstream. This is known as the *constant bitrate* requirement. However, more often than not, it is simply preferred that the output produced by an encryption-equipped encoder and the output produced by an ordinary encoder have similar sizes (a near-constant bitrate). A near-constant bitrate is likely to occur when a block cipher is used for encryption, since in that case the encrypted output is always a multiple of the blocksize.

Finally, for many multimedia systems *error-tolerance*, or *error-resilience*, is usually of high importance. Advanced video coding systems (e.g. H.264) have their own error correcting mechanisms. Hence, a video encryption algorithm that preserves these mechanisms is favorable for video systems with noisy channels.

For the most part, modern cryptography is designed for a generic bitstream, and as such, it disregards the aforementioned properties of a digital video and the requirements of a typical digital video application. In the next section, we present an overview of the video encryption algorithms proposed in the past mainly to overcome some of these application-related problems.

The rest of this paper is organized as follows. Section 2 provides a brief overview of existing video encryption algorithms, and Section 3 introduces our proposed method. In Section 4 we present a security analysis of our method, while in Section 5 we provide its performance analysis with some experimental results. Finally, Section 6 holds our conclusions and suggestions for further research.



**Fig. 1.** Decoded video produced by an ordinary decoder for (a) video without encryption, (b) video encrypted with format-compliant perceptual encryption, and (c) video encrypted with high-security format-compliant encryption.

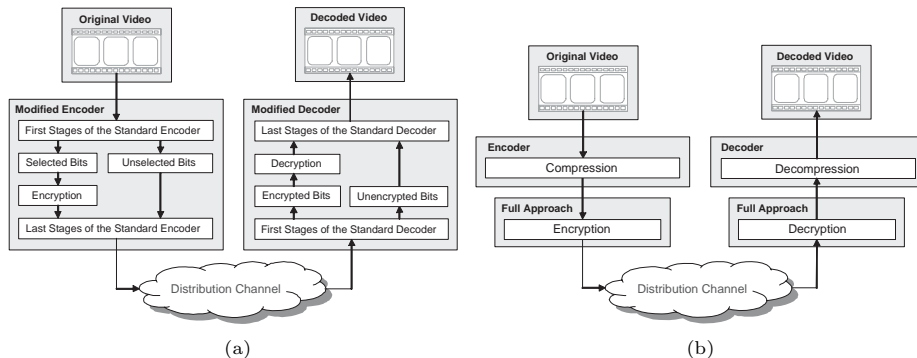
## 2 Overview of Video Encryption Algorithms

In general, there are two basic research methodologies regarding the encryption of digital videos: *selective encryption* approaches and *full encryption* approaches. While full encryption approaches are designed to encrypt the entire video bitstream (raw or compressed), selective encryption approaches perform encryption only on certain, carefully selected parts of the video bitstream. Most of the proposed methods belong to the category of selective encryption approaches since, in many instances, encrypting the entire bitstream introduces an expensive performance overhead.

The idea of selective video encryption was introduced independently by Meyer and Gadegast (SECMPEG) [8], and by Maples and Spanos (Aegis) [15]. Since then, a significant number of selective encryption proposals appeared in the scientific literature. A good reference for selective image and video encryption methods along with some security analysis is given in [3, 7].

As Fig. 2a depicts, most selective video encryption algorithms require a modification of the codec, which is a drawback for systems with pre-installed hardware codecs. Another common problem with selective encryption approaches is the lack of standard methods for proper security evaluation. Many selective encryption approaches were broken shortly after they were proposed because scientists found ways to exploit the information from the remaining, unencrypted bits. For example, Agi and Gong [1] showed weaknesses in SECMPEG [8] and Aegis [15] a year later. Also, a year after Tang proposed an image/video encryption method [16] based on permuting the zig-zag reordering after a DCT transformation, Qiao and Nahrstedt [11] discovered serious weaknesses against attacks derived from statistical analysis of the unencrypted DCT coefficients. Seidel et al. [12] show some weaknesses in the video encryption algorithms by Shi et al. soon after the original proposal [13]. The true correlation between encrypted and unencrypted bits is often hard to properly estimate in terms of required security levels. Hence, the selective encryption approaches are much easier to design and evaluate against the performance aspects, than to properly evaluate in terms of security.

Full encryption approaches essentially encrypt the entire multimedia data. The full encryption approach is not to be confused with the so-called *naïve approach*, which is itself in the category of full encryption. By a naïve approach one understands a video encryption method that uses a conventional, general-purpose modern cryptosystem to encrypt the entire video bitstream. A “conventional cryptosystem” refers to a modern symmetric-key cryptosystem that is either one of the encryption standards, or a well-established general-purpose cryptosystem that was designed and evaluated by reputable cryptographic experts, companies, or agencies. Full approaches that are not considered naïve are mostly systems that are specifically designed to encrypt a specific multimedia type of data. These approaches, which we refer to as *alternative full approaches*, take into consideration the performance and security aspects needed for an effective multimedia encryption.



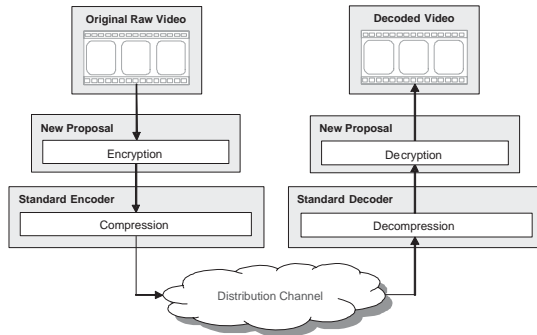
**Fig. 2.** The usual architectures for: (a) selective video encryption where both encoder and decoder must be modified since the encryption/decryption is performed during the compression/decompression stages, and (b) full (and naïve) video encryption where the entire compressed video bitstream is encrypted using a standard cryptosystem (e.g. AES) or an alternative fast cryptosystem (e.g. chaos-based).

A notable path that the research community took for the full alternative video encryption approaches was the use of fast chaotic maps to achieve encryption. Chaos-based methods are apparently promising due to their fast performance. Although many chaotic encryption approaches were shown to be insecure, there are chaotic encryption algorithms that, up to date, remain unbroken (e.g. [4]). An excellent overview of these approaches, along with their comparative security analysis is presented in [5] and [3]. There are a few recently proposed fast, hardware-friendly full encryption methods that are based on a class of neural networks [2]. However, these methods were later shown to be less secure than originally anticipated [14]. Yi *et al.* proposed a new fast encryption algorithm for multimedia (FEA-M), which bases the security on the complexity of solving nonlinear Boolean equations [17]. The scheme was shown insecure against several different attacks [18, 9].

In addition to questionable security, the full alternative encryption approaches are not application-friendly when applied after compression, and the application requirements such as format-compliance or error-resilience are not supported. Fig. 2b shows the typical structural design of full encryption approaches, where encryption is performed after compression and where no modification to the codec is needed.

### 3 Our Model

Strong encryption is a process that produces randomized data. On the other hand, compression efficiency is directly dependent on the presence of source data redundancy. The more the data is correlated, the better the compression, and vice versa. One may ask the following important question: Is it possible to design an encryption mechanism of reasonable security that preserves, or perhaps



**Fig. 3.** The architecture of the proposed algorithm: The encryption and decryption are performed outside the current video system. This results in no modification to the codec and fully compliant, application-friendly video outputs.

even increases the compressibility (correlation) of data? To our knowledge, no such system has ever been proposed in the published scientific literature. Such system is here referred to as the *Correlation-Preserving Encryption* (CPE). CPE for digital videos encoded with spatial-only coding is indeed possible to achieve with permutation-based transformations. In this section we present the details of our CPE approach. The architecture of this approach is depicted in Fig. 3.

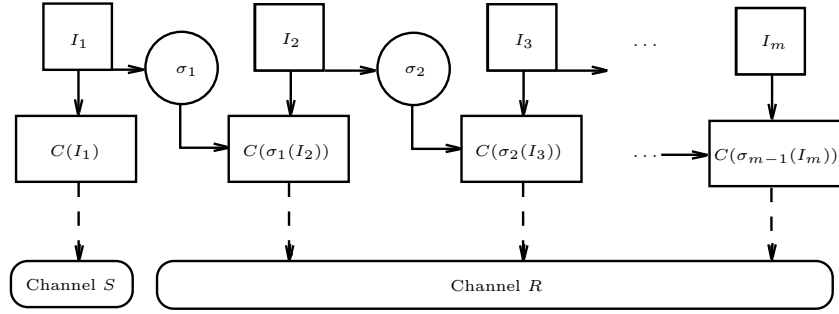
Let  $\mathcal{V}$  be a video sequence consisting of  $m$  frames denoted by  $I_1, I_2, \dots, I_m$ . For our model we assume that all frames in  $\mathcal{V}$  are part of a single scene with a relatively low movement, captured with a static camera, so that the differences between adjacent frames are relatively small. Furthermore, we assume that each frame has a dimension of  $w \times h$  and up to  $2^n$  different pixel values (colors), such that  $w \times h$  is much larger than  $2^n$ . By the pigeonhole principle this condition favors higher frequencies of the pixel values within a frame. Finally, let  $\sigma_i$  denote a sorting permutation of  $I_i$ , and  $\sigma_i(I_i)$  the image with sorted pixels from  $I_i$ .

We describe two versions of our algorithm. The first one is designed for lossless spatial-only codecs, such as Animated GIF (A-GIF), Motion PNG (M-PNG), or Motion Lossless JPEG (M-JLS), while the second one is targeted for lossy spatial-only codecs, such as Motion JPEG (M-JPEG).

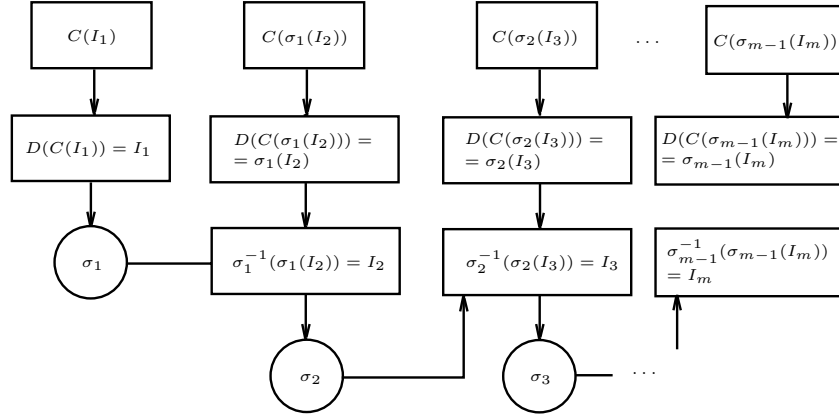
Suppose Alice wishes to securely transmit a video sequence  $\mathcal{V} = I_1, I_2, \dots, I_m$  to Bob. We assume that if Alice would like to transmit  $\mathcal{V}$  to Bob non-securely, she would normally use video compression algorithm  $C$  (the encoder) and decompression algorithm  $D$  (the decoder). She opens two channels with Bob, the regular, non-secure multimedia distribution channel  $R$ , and the secure channel  $S$  where transmission data is encrypted using some standard method (e.g. AES-based protocol).

**The Lossless Case.** The following is the outline of the proposed algorithm for lossless codecs:

1. Given a video sequence  $\mathcal{V} = I_1, \dots, I_m$ , Alice computes  $\sigma_1$ .



**Fig. 4.** Diagram of our encryption algorithm for lossless spatial-only video codecs.

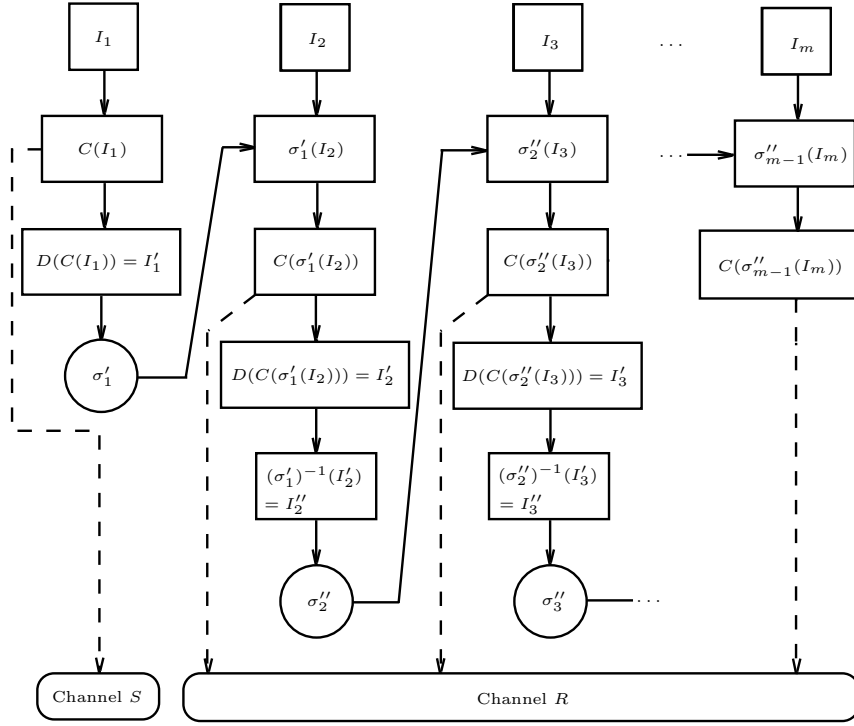


**Fig. 5.** Diagram of our decryption algorithm for lossless spatial-only video codecs.

2. Alice calculates  $C(I_1)$  and transmits it through channel  $S$ . This is the secret part (the key) of the algorithm.
3. For each next frame  $I_i$ ,  $i = 2, \dots, m$ , Alice does the following:
  - (a) She computes the frame  $\sigma_{i-1}(I_i)$  and the permutation  $\sigma_i$ ;
  - (b) Alice then applies the standard encoder to the frame  $\sigma_{i-1}(I_i)$  and transmits the encoded frame  $C(\sigma_{i-1}(I_i))$  to Bob via regular, non-secure multimedia channel  $R$ .

At the other end, Bob performs the following decryption algorithm in order to recover the original video sequence  $\mathcal{V}$ :

1. Bob decodes  $C(I_1)$  into  $I_1$  and from it calculates a sorting permutation  $\sigma_1$ .
2. For each received frame  $C(\sigma_{i-1}(I_i))$ ,  $i = 2, \dots, m$ , Bob does the following:
  - (a) Decodes  $C(\sigma_{i-1}(I_i))$  into  $\sigma_{i-1}(I_i)$  and calculates  $I_i = \sigma_{i-1}^{-1}(\sigma_{i-1}(I_i))$  where  $\sigma_{i-1}^{-1}$  is an inverse permutation of  $\sigma_{i-1}$ ;
  - (b) Calculates a sorting permutation  $\sigma_i$  from  $I_i$ .



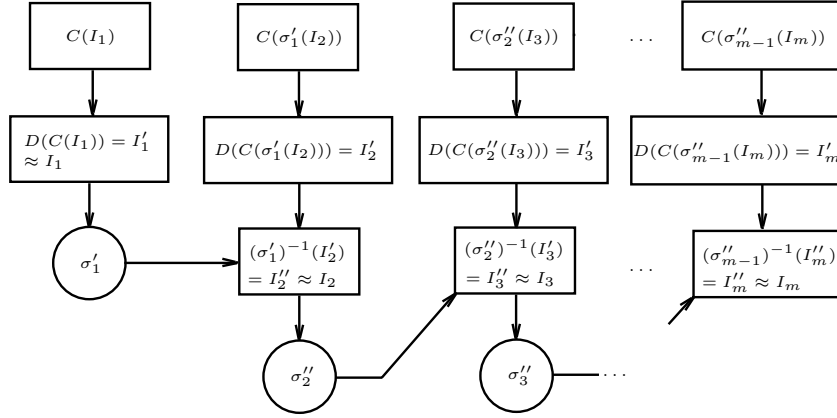
**Fig. 6.** Diagram of our encryption algorithm for lossy spatial-only video codecs.

**The Lossy Case.** The proposed algorithm for lossy codecs is described as follows:

1. Given a video sequence  $\mathcal{V} = I_1, \dots, I_m$ , Alice first computes  $C(I_1)$  and then  $I'_1 = D(C(I_1))$  from which she calculates a sorting permutation  $\sigma'_1$ .
2. Alice sends  $C(I_1)$  via secure channel  $S$  to Bob.
3. She applies  $\sigma'_1$  to  $I_2$ , computes  $C(\sigma'_1(I_2))$ , and sends it via  $R$  to Bob.
4. Next, she computes  $I'_2 = D(C(\sigma'_1(I_2)))$  and then  $I''_2 = (\sigma'_1)^{-1}(I'_2)$  from which she calculates a sorting permutation  $\sigma''_2$ .
5. For each next frame  $I_i$ ,  $i = 3, \dots, m$ , Alice does the following:
  - (a) Applies  $\sigma''_{i-1}$  to  $I_i$ , computes  $C(\sigma''_{i-1}(I_i))$ , and sends it to Bob via  $R$ ;
  - (b) Computes  $I'_i = D(C(\sigma''_{i-1}(I_i)))$ ;
  - (c) Applies  $(\sigma''_{i-1})^{-1}$  to get  $I''_i = (\sigma''_{i-1})^{-1}(I'_i)$ ;
  - (d) Calculates a sorting permutation  $\sigma''_i$ .

At the receiver's side, Bob performs the following decryption algorithm to recover the original video sequence  $\mathcal{V}$ :

1. Bob calculates  $D(C(I_1)) = I'_1 \approx I_1$  calculates a sorting permutation  $\sigma'_1$ .



**Fig. 7.** Diagram of our decryption algorithm for lossy spatial-only video codecs.

2. From  $C(\sigma'_1(I_2))$  he computes  $I'_2 = D(C(\sigma'_1(I_2)))$ .
3. Bob approximates  $I_2 \approx I''_2 = (\sigma'_1)^{-1}(I'_2)$ .
4. He then recovers a sorting permutation  $\sigma''_2$  of  $I''_2$ .
5. For each received frame  $C(\sigma''_{i-1}(I_i))$ ,  $i = 3, \dots, m$ , Bob does the following:
  - (a) Decodes  $C(\sigma''_{i-1}(I_i))$  into  $I'_i = D(C(\sigma''_{i-1}(I_i)))$ ;
  - (b) Approximates  $I_i \approx I''_i = (\sigma''_{i-1})^{-1}(I'_i)$ ;
  - (c) If  $i < m$  he calculates a sorting permutation  $\sigma''_i$  of  $I''_i$ .

## 4 Security Analysis of Our Model

**Key Space.** In order to grasp the true complexity of an exhaustive search of the key space related to our method, one should answer the following question: Given a color histogram of an  $w \times h$  image  $I$ , how many different images can be formed out of the histogram color values? Note that  $I$  is just one of these images. We use some known group theoretic properties to provide the answer to this question.

Let  $S_n$  denote the symmetric group of permutations on  $n$  points. Thus, in spirit of our previous notation,  $\sigma_I \in S_{w \times h}$ . Furthermore, if  $\mathbb{Z}_n^m$  denotes the  $m$ -dimensional set of integers modulo  $n$ , then  $I \in \mathbb{Z}_{2^n}^{w \times h}$ . Next we review the definitions of a permutation group action, an orbit, and a stabilizer.

Whenever  $X$  is a set or group then  $|X|$  denotes the size of  $X$ . If  $|X| = n$ , then  $S_n$  acts on  $X$  by permuting the elements of  $X$ . In this group action, if  $x \in X$  then an *orbit* of  $x$ , denoted by  $\mathcal{O}(x)$ , is a subset of  $X$  defined by:

$$\mathcal{O}(x) = \{\pi(x) : \pi \in S_n\} \subset X,$$

while the *stabilizer* of  $x$ , denoted by  $\mathcal{S}(x)$ , is the subgroup of  $S_n$  defined by:

$$\mathcal{S}(x) = \{\pi \in S_n : \pi(x) = x\} \leq S_n.$$

Notice that if  $I$  is a  $w \times h$  image with pixel values from  $\mathbb{Z}_{2^n}$ , then the number of different images that can be formed by permuting the pixels in  $I$  is exactly the size of the orbit of  $I$  (i.e.,  $|\mathcal{O}(I)|$ ) under the group action of  $S_{w \times h}$  acting on  $\mathbb{Z}_{2^n}^{w \times h}$ , which would answer our previously posted question. The following well-known theorem from an elementary group theory helps determine this number: If  $G$  is a group acting on set  $X$ , and  $x \in X$ , then  $|\mathcal{O}(x)| = |G|/|\mathcal{S}(x)|$ . In our case,  $|G| = |S_{w \times h}| = (w \times h)!$ . Therefore, if we can determine  $|\mathcal{S}(I)|$ , then we know  $|\mathcal{O}(I)|$ .

Let  $U(I) \subset \mathbb{Z}_{2^n}$  denote the set of unique values of  $I$ . Furthermore, if  $u \in U(I)$  then let  $N(u)$  denote the number of times pixel value  $u$  appears in  $I$ . Then the following theorem holds:

**Theorem 1.** *If  $G = S_{w \times h}$  is a permutation group action on image  $I \in \mathbb{Z}_{2^n}^{w \times h}$ , and  $U(I) = \{u_1, \dots, u_k\}$ , then the size of the stabilizer of  $I$  is*

$$|\mathcal{S}(I)| = \prod_{i=1}^k N(u_i)!$$

*Proof.* Let  $u \in U(I)$ . Then, there are  $N(u)$  pixels in the image  $I$  that have the value of  $u$ . Thus, there are  $N(u)!$  ways of permuting these values among themselves without changing  $I$ . Doing this for all elements of  $U(I)$  results in a total number of  $N(u_1)! \times \dots \times N(u_k)!$  ways of permuting  $I$  without changing it. This is exactly the size of the stabilizer of  $I$ .

Using the result of Theorem 1 we have that the number of different images that can be formed out of pixels from  $I$ , where  $U(I) = \{u_1, \dots, u_k\}$ , is

$$|\mathcal{O}(I)| = \frac{(w \times h)!}{\prod_{i=1}^k N(u_i)!}$$

Thus, there are exactly  $(w \times h)! / \prod_{i=1}^k N(u_i)!$  permutations in  $S_{w \times h}$  that permute  $I$  differently. These permutations are the effective key space of our method. Therefore, the size of the key space depends on the color histogram of the encrypted frame. As one can see, this number is extremely large when considering natural images.

**Known/Chosen-Plaintext and Chosen-Ciphertext Attacks.** In general, permutation-only video encryption is considered weak against known/chosen-plaintext attack, and the chosen-ciphertext attack [6]. However, all of the previously proposed permutation-only video encryption methods rely on generating the secret permutation using a secret key. Under this scenario, all of the aforementioned attacks are trying to recover the secret key (or a part of it) that was used for the current or future encryptions. Our method for video encryption does not rely on such principle, and there is no secret key upon which a permutation is generated. Namely, our method rely on the sorting permutation for the previous frame, and thus, a key is directly dependant of the plaintext. Under

the chosen-plaintext attack, the adversary can easily compute the sorting permutation for the chosen frame, but this gives no information about the sorting permutations for the unknown frames. Similarly, under the chosen-ciphertext attack, the adversary is able to recover the unsorting permutation for the chosen encrypted (sorted) frame, but this gives no information regarding other unknown ciphertexts. In fact, all frames with the same histogram encrypt into the same ciphertext, so when the adversary encounters the same ciphertext again, the previous unsorting permutation may give a completely inaccurate plaintext. A limited known-plaintext attack is applicable to our method, because the adversary can recover all frames that follow the known frame until the scene changes and key frame is updated. However, as mentioned earlier, our system is such that it is not possible to re-use the secret key for future scenes or sequences, since there is no key that is independent of the plaintext.

**Known Weaknesses.** If one frame of a video scene is known, all of the frames that follow within that scene are then computable by the adversary. In addition, if the adversary possesses the information on the possible videos to be encrypted, he or she may be able to recognize which video sequence is being transmitted from Alice to Bob by observing the publicly given histograms of frames. These two attacks are unavoidable in the proposed scheme, and the scheme should not be used where these attacks could be of interest to the adversary.

## 5 Performance Analysis and Experimental Results

To evaluate the performance of our method in terms of compression, we run experiments on several fairly static greyscale sequences in CIF and QCIF formats.

**Table 1.** Performance of our method for lossless spatial-only codecs.

Sequence	Codec	Codec w/o Encryption		Codec w/ Encryption	
		Size[MB]	Avg.Frm.[KB]	Size[MB]	Avg.Frm.[KB]
<b>Akiyo</b> CIF, 300 frms	A-GIF	21.53	73.51	9.24	31.54
	M-PNG	19.09	65.15	7.96	27.17
	M-JLS	10.70	36.53	7.39	25.23
<b>Mother and Daughter</b> CIF, 300 frms	A-GIF	21.68	73.99	15.93	54.38
	M-PNG	19.23	65.64	14.97	51.11
	M-JLS	11.31	38.62	12.63	43.12
<b>Monitor Hall</b> CIF, 300 frms	A-GIF	24.88	84.91	18.50	63.14
	M-PNG	21.67	73.97	17.55	59.91
	M-JLS	13.13	44.83	14.62	49.89
<b>Grandma</b> QCIF, 100 frms	A-GIF	2.14	21.91	1.45	14.84
	M-PNG	1.90	19.46	1.30	13.34
	M-JLS	1.18	12.13	0.89	9.09
<b>Claire</b> QCIF, 100 frms	A-GIF	1.52	15.59	1.17	11.98
	M-PNG	1.35	13.86	1.02	10.48
	M-JLS	0.75	7.68	0.71	7.27
<b>Miss America</b> QCIF, 100 frms	A-GIF	1.54	15.80	1.33	13.57
	M-PNG	1.44	14.79	1.27	13.05
	M-JLS	0.81	8.34	0.91	9.37

**Table 2.** Performance of our method for lossy spatial-only codecs.

Sequence	$Q$	M-JPEG w/o Encryption			M-JPEG w/ Encryption		
		Size[MB]	Avg.Frm.[KB]	PSNR	Size[MB]	Avg.Frm.[KB]	PSNR
<b>Akiyo</b> CIF, 300 frms	90	3.96	15.93	45.29	3.71	12.72	41.63
	70	2.05	8.24	40.39	1.92	6.59	36.13
	50	1.55	6.24	38.20	1.37	4.70	33.93
<b>Mother and Daughter</b> CIF, 300 frms	90	4.21	16.96	45.25	4.98	17.07	39.97
	70	2.24	9.04	40.91	2.48	8.48	34.68
	50	1.70	6.84	38.88	1.72	5.91	32.70
<b>Monitor Hall</b> CIF, 300 frms	90	5.55	22.35	43.21	6.05	20.71	38.89
	70	3.02	12.15	38.12	2.93	10.03	33.49
	50	2.25	9.05	35.95	2.01	6.89	31.40
<b>Grandma</b> QCIF, 100 frms	90	0.57	5.94	41.04	0.35	3.58	41.55
	70	0.32	3.28	36.47	0.19	1.93	36.35
	50	0.24	2.48	34.81	0.14	1.44	34.06
<b>Claire</b> QCIF, 100 frms	90	0.41	4.22	45.19	0.31	3.19	42.23
	70	0.25	2.62	39.86	0.17	1.80	36.60
	50	0.20	2.08	37.51	0.13	1.36	34.36
<b>Miss America</b> QCIF, 100 frms	90	0.35	3.64	45.63	0.36	3.69	41.55
	70	0.19	1.98	41.52	0.19	1.98	35.97
	50	0.15	1.56	39.71	0.14	1.46	33.83

As Tables 1 and 2 show, our method preserves, and in many instances even improves the compression performance of the original codec without encryption. Table 2 also compares the loss of quality (in terms of PSNR) when our method is applied to the lossy M-JPEG with quality parameter  $Q$  that controls the quantization level in M-JPEG.  $Q$  ranges from 0 (the worst quality) to 100 (the best quality). A modest loss of quality occurs by performing the proposed encryption with lossy codecs.

Finally, we note that the computational complexity of the proposed method is very low at the decoder side for both lossless and lossy codecs, since the only computation that has to be performed involves the calculation of a sorting permutation. A standard sorting algorithm with complexity of  $\mathcal{O}(N \log N)$  can be used to calculate a sorting permutation of the given frame. Inverting and/or applying a permutation is equivalent to a table lookup.

## 6 Conclusions and Future Research

In this work we proposed a novel video encryption algorithm designed for both lossless and lossy low-motion spatial-only video codecs. The algorithm preserves (or even improves) the redundancies of the source data, and thus it can be performed before compression at the encoder side, and after decompression at the decoder side, which is a unique feature. In effect, the algorithm produces fully application-friendly output, and requires no modification to the codec modules. We also presented both security and performance analysis of our method, and showed that the algorithm is computationally efficient and resistant to typical cryptanalytic attacks. Future directions should include the investigation of extending this principle to achieve efficiency in exploiting the temporal redundancies as well, in order to achieve the applicability to more advanced video codecs such as H.26x and MPEG-x.

## References

1. I. Agi and L. Gong, "An Empirical Study of Secure MPEG Video Transmission", in Proc. IEEE Symposium on Network and Distributed System Security (SNDSS '96), San Diego, CA, February 22-23, IEEE Computer Society, pp. 137-144, 1996.
2. C.-Ku. Chan, C.-Kw. Chan, L.-P. Lee, and L.M. Cheng, "Encryption System Based On Neural Network", Communications and Multimedia Security Issues of the New Century, Kluwer Academic Publishers, Boston, MA, pp. 117-122, 2001.
3. B. Furht, E.A. Muharemagic, and D. Socek, "Multimedia Security: Encryption and Watermarking", vol. 28 of Multimedia Systems and Applications, Springer, 2005.
4. S. Li, X. Zheng, X. Mou, and Y. Cai, "Chaotic Encryption Scheme for Real-Time Digital Video", In Real-Time Imaging VI, Proc. SPIE, vol. 4666, pp. 149-160, 2002.
5. S. Li, G. Chen, and X. Zheng, "Multimedia Security Handbook", eds. B. Furht and D. Kirovski, vol. 4 of Internet and Communications Series, ch. 4 "Chaos-Based Encryption for Digital Images and Videos", CRC Press, 2004.
6. S. Li, C. Li, G. Chen, and N.G. Bourbakis, "A General Cryptanalysis of Permutation-Only Multimedia Encryption Algorithms", IACR's Cryptology ePrint Archive: Report 2004/374, 2004.
7. X. Liu and A.M. Eskicioglu, "Selective Encryption of Multimedia Content in Distribution Networks: Challenges and New Directions", IASTED International Conference on Communications, Internet and Information Technology (CIIT 2003), Scottsdale, AZ, November 17-19, 2003.
8. J. Meyer and F. Gadegast, "Security Mechanisms for Multimedia Data with the Example MPEG-1 Video", Project Description of SEC MPEG, Technical University of Berlin, Germany, May 1995, <http://www.gadegast.de/frank/doc/secmeng.pdf>.
9. M.J. Mihaljević and R. Kohno, "Cryptanalysis of Fast Encryption Algorithm for Multimedia FEA-M", IEEE Trans. Comm. Letters, vol. 6, no. 9, pp. 382-384, 2002.
10. L. Qiao and K. Nahrstedt, "A New Algorithm for MPEG Video Encryption", in Proc. First International Conference on Imaging Science, Systems and Technology (CISST '97), Las Vegas, NV, June 30-July 3, pp. 21-29, 1997.
11. L. Qiao, K. Nahrstedt, and M.-C. Tam, "Is MPEG Encryption by Using Random List Instead of Zigzag Order Secure?", in Proc. IEEE Intl. Symposium on Consumer Electronics, Singapore, Dec. 2-4, IEEE Comp. Soc., pp. 226-229, 1997.
12. T.E. Seidel, D. Socek, and M. Sramka, "Cryptanalysis of Video Encryption Algorithms", Tatra Mountains Mathematical Publications, vol 29, pp. 1-9, 2004.
13. Bhargava, B., Shi, C., Wang, S.-Y.: MPEG Video Encryption Algorithms. Multimedia Tools and Applications, Kluwer Academic Publishers, 24(1):57-79, 2004.
14. D. Socek and D. Culibrk, "On the Security of a Clipped Hopfield Neural Network Cryptosystem", in Proc. of the ACM Multimedia and Security Workshop (ACM-MM-Sec'05), New York City, NY, August 1-2, pp. 71-75, 2005.
15. G.A. Spanos and T.B. Maples, "Security for Real-Time MPEG Compressed Video in Distributed Multimedia Applications", in Proc. IEEE 15th Intl. Phoenix Conf. on Computers and Communications, Scottsdale, AZ, March 27-29, pp. 72-78, 1996.
16. L. Tang, "Methods for Encrypting and Decrypting MPEG Video Data Efficiently", in Proc. Fourth ACM International Multimedia Conference, Boston, MA, November 18-22, pp. 219-230, 1996.
17. X. Yi, C.H. Tan, C.K. Siew, M.R. Syed, "Fast Encryption for Multimedia", IEEE Trans. Consumer Eletronics 47 (1), pp. 101-107, 2001.
18. A.M. Youssef and S.E. Tavares, "Comments on the Security of Fast Encryption Algorithm for Multimedia (FEA-M)", IEEE Trans. Consumer Eletronics 49 (1), pp. 168-170, 2003.