Correspondence

Composite Signal Representation for Fast and Storage-Efficient Processing of Encrypted Signals

Tiziano Bianchi, Alessandro Piva, and Mauro Barni

***Abstract*—Signal processing tools working directly on encrypted data could provide an efficient solution to application scenarios where sensitive signals must be protected from an untrusted processing device. In this paper, we consider the data expansion required to pass from the plaintext to the encrypted representation of signals, due to the use of cryptosystems operating on very large algebraic structures. A general composite signal representation allowing us to pack together a number of signal samples and process them as a unique sample is proposed. The proposed representation permits us to speed up linear operations on encrypted signals via parallel processing and to reduce the size of the encrypted signal. A case study—1-D linear filtering—shows the merits of the proposed representation and provides some insights regarding the signal processing algorithms more suited to work on the composite representation.**

***Index Terms*—Homomorphic encryption, secure signal processing, signal processing in the encrypted domain, signal representation.**

I. Introduction

The possibility of processing encrypted signals directly in the encrypted domain (hereafter referred to as s.p.e.d., standing for signal processing in the encrypted domain) is receiving an increasing attention as a way to satisfy the security requirements stemming from applications wherein valuable or sensible signals have to be processed by a nontrusted party [1]. The list of applications that would benefit from the availability of s.p.e.d. tools is virtually endless, including access to a database containing encrypted data or signals [2], [3], database access by means of encrypted queries [4], remote processing of private data, like medical recordings or biometric signals, by nontrusted parties [5], [6], transcoding of encrypted contents [7], buyer–seller watermarking protocols [8], just to mention some.

Though apparently impossible, processing encrypted signals is indeed feasible by relying on probabilistic homomorphic encryption [9], [10] and secure multiparty computation (MPC) [11]–[13]. In this paper, we focus on techniques based on homomorphic encryption, since they constitute the basis for any practical implementation of s.p.e.d. theory. For an overview of the role of MPC in s.p.e.d. theory, readers may refer to [14].

Manuscript received April 07, 2009; accepted October 12, 2009. First published November 06, 2009; current version published February 12, 2010. This work was supported in part by the European Commission through the IST Programme under Contract 034238—SPEED and in part by the Italian Research Project (PRIN 2007): "Privacy aware processing of encrypted signals for treating sensitive information". The information in this document reflects only the author's views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Fernando Perez-Gonzalez.

T. Bianchi and A. Piva are with the Dipartimento di Elettronica e Telecomunicazioni, Università di Firenze, I-50139 Firenze, Italy (e-mail: tiziano.bianchi@unifi.it; alessandro.piva@unifi.it).

M. Barni is with the Dipartimento di Ingegneria dell'Informazione, Università di Siena, 53100 Siena, Italy (e-mail: barni@dii.unisi.it).

Digital Object Identifier 10.1109/TIFS.2009.2036230

A cryptosystem is said to be homomorphic with respect to an operation $\star$, if another operation $\circ$ exists such that, given two plaintexts m_1 and m_2, we have

$$D[E[m_1] \circ E[m_2]] = m_1 \star m_2 \tag{1}$$

where D and E indicate, respectively, the decryption and encryption operators. Additively homomorphic cryptosystems, for which $\star = +$ and $\circ = \cdot$, play a central role in s.p.e.d. theory. For such systems we have

$$D[E[m_1] \cdot E[m_2]] = m_1 + m_2 \tag{2}$$

$$D[E[m_1]^c] = c \cdot m_1 \tag{3}$$

where c is a constant factor, hence allowing the application of many basic signal processing tools directly in the encrypted domain. This is the case of linear operators like the discrete Fourier transform (DFT) [15], finite impulse response (FIR) filters, correlation, and simple operations among two or more signals like componentwise signal addition.

A problem with the use of homomorphic encryption is that signals need to be encrypted sample-wise. Suppose, for instance that a party P_1 wants to compute the projection ρ of a signal $x(n)$ on a direction $u(n)$ known by another party P_2, and assume that neither P_1 wants to disclose $x(n)$ to P_2, nor P_2 is willing to reveal $u(n)$ to P_1. If $x(n)$ is encrypted sample-wise by P_1 and sent to P_2, the encrypted projection $E[\rho]$ can be computed by P_2 without knowing the unencrypted values as $E[\rho] = \prod_{i=1}^{M} E[x(i)]^{u(i)}$ with M being the length of the sequence $x(n)$. At the end, P_2 will send $E[\rho]$ to P_1 that will obtain the result of the projection by decrypting the received value.

Samplewise encryption of signals poses some severe complexity problems since it introduces a huge expansion factor between the original signal sample and the encrypted one. To fix the ideas, let us assume that the Paillier cryptosystem is used (see [16] for more details about the Paillier cryptosystem); in this case, each encrypted sample is an element of $\mathbb{Z}_{N^2}$, i.e., the set of integer numbers modulo N^2 with N being at least 1024 bit long, that is each encrypted sample needs at least 2048 bits to be represented. By considering that plain signal samples are usually represented by a few bits (e.g., 8 bits for images or 16 bits for ECG signals [17]), we conclude that due to encryption, signals are expanded by a factor ranging from 125 to 250.

A straightforward way to avoid the samplewise encryption of signals is to pack several samples together, and then encrypt them as a unique message. For instance, this simple idea is used in [18], where R l-bit messages $m_1, \ldots, m_R$ are bundled within a single composite message x as follows:

$$x = m_1 \cdot 2^0 + m_2 \cdot 2^L + \cdots + m_R \cdot 2^{L(R-1)}. \tag{4}$$

If L is larger than l, samples will remain distinct in the composite representation; moreover, if L is sufficiently large, adding two composite messages corresponds to the addition of the single messages composing them, and multiplying the composite message by a constant factor is equivalent to multiplying each single message by the same factor. Though already outlined in the scientific literature, the limits and opportunities offered by the composite representation of signals have not been thoroughly investigated yet. Despite the apparent simplicity, in fact, the signal representation suggested in (4) presents a number of problems that need to be tackled, and offers several opportunities to be

exploited. In order to do so, in this paper, we first generalize (4), by allowing any integer value B to be used instead of 2 in the power expansion of x. In this way we gain flexibility in the choice of the number of samples that can be packed into a single message x. Second, we consider the problems introduced by the necessity of working with signed numbers. Finally, and most importantly, we show how some of the most common signal processing operations can be performed very efficiently by exploiting the intrinsic parallelism of the proposed composite representation. Specifically, we discuss the application of block-wise linear transforms and linear FIR filters to signals whose samples are represented through a generalized version of (4). In order to substantiate our discussion, the paper ends with the presentation of a case study in which a time-domain and a frequency-domain implementation of an FIR filter are compared, showing that due to the particular structure of the composite representation of signals, a time-domain implementation may be preferable to a standard frequency-domain implementation.

The rest of this paper is organized as follows. In Section II, a brief review of the Paillier cryptosystem is given. In Section III, the proposed composite representation is introduced and discussed. Section IV considers the application of block-wise transforms, and linear (FIR) filters to composite signals. In Section V, the case study regarding 1-D linear filtering is presented. Finally, some conclusions are drawn in Section VI.

II. Probabilistic Homomorphic Encryption: The Paillier Cryptosystem

As we said, a homomorphic cryptosystem allows us to carry out some basic algebraic operations on encrypted data by translating them into corresponding operations in the plaintext domain. The concept of privacy homomorphism was first introduced by Rivest *et al.* [9], who defined privacy homomorphisms as encryption functions which permit encrypted data to be operated on without preliminary decryption of the operands.

According to the correspondence between the operation in the ciphertext domain and the operation in the plaintext domain, a cryptosystem can be additively homomorphic or multiplicatively homomorphic: in this paper, we are interested in the former. Additively homomorphic cryptosystems allow us, in fact, to perform additions, subtractions, and multiplications with a known (nonencrypted) factor in the encrypted domain.

Another crucial concept for the s.p.e.d. framework is probabilistic encryption. As a matter of fact, many of the most popular cryptosystems are deterministic, that is, given an encryption key, and given a plaintext, the ciphertext is univocally determined. The main drawback of these schemes for s.p.e.d. applications is that it is easy for an attacker to detect if the same plaintext message is encrypted twice: indeed, since usually signal samples assume only a limited range of values, an attacker will be easily able to decrypt the ciphertexts, or at least to derive meaningful information about them. In [10], the concept of probabilistic or semantically secure cryptosystem has been proposed. In such schemes, the encryption function E is a function of both the secret message m and a random parameter r that is changed at any new encryption. Specifically, two subsequent encryptions of the same message m result in two different encrypted messages $c_1 = E(m, r_1)$ and $c_2 = E(m, r_2)$. Of course, for a correct behavior, the scheme has to be designed in such a way that $D(c_1) = D(c_2) = m$, that is the decryption phase is deterministic, and does not depend on the random parameter r.

Encryption schemes that satisfy both the homomorphic and probabilistic properties detailed above do exist. One of the most known homomorphic and probabilistic schemes is the one presented by Paillier in [16], and later modified by Damgård and Jurik in [19]. Both Paillier and Damgård and Jurik cryptosystems are additively homomorphic.

More extensive processing would be allowed by the availability of a fully (or algebraically) homomorphic encryption scheme, that is a scheme that is additive and multiplicative homomorphic. For instance, a fully homomorphic cryptosystem would allow to compute several useful statistics, like the energy of a signal, without requiring interactive protocols. Very recently, a fully homomorphic scheme has been proposed in [20], thus showing that secure and fully homomorphic encryption is indeed possible; however, the complexity of the system proposed in [20] is by far too complex for any practical application.

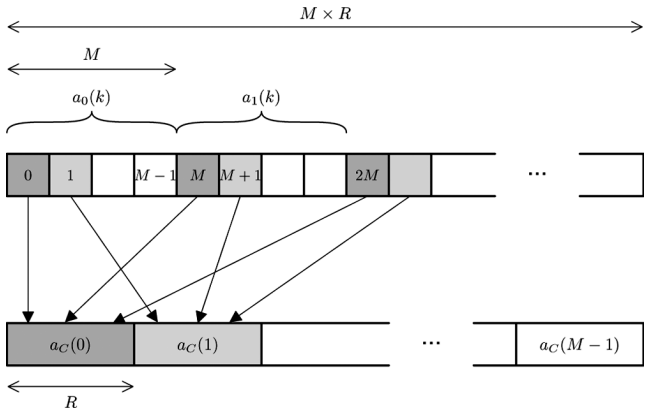


Fig. 1. Graphical representation of an M-polyphase composite representation of order R. The values inside the small boxes indicate the indexes of the samples of $a(n)$. Identically shaded boxes indicate values belonging to the same composite word.

III. Composite Representation of Signals

In this section, we generalize the composite representation expressed by (4), and discuss the conditions that allow us to recover the original samples starting from the composite representation. We also consider the problems associated to the transformation from the sample-wise to the composite domain and vice versa.

Let us consider an integer valued signal $a(n) \in \mathbb{Z}$, satisfying $|a(n)| \leq Q$, where Q is a positive integer. Given a pair of positive integers B, R, we define the *composite* representation of $a(n)$ of order R and base B as

$$a_C(k) = \sum_{i=0}^{R-1} a_i(k) B^i, \quad k = 0, 1, \ldots, M-1 \tag{5}$$

where $a_i(k)$, $i = 0, 1, \ldots, R-1$ indicate R disjoint subsequences of the signal $a(n)$.

The kth element of the composite signal $a_C(k)$, represents a word where we can pack R samples of the original signal, chosen by partitioning the original signal samples $a(n)$ into M sets of R samples each.

Several choices can be made to partition the original signal. In the following, we will consider the choice $a_i(k) = a(iM + k)$. In this case, each composite word will contain R samples which are spaced M samples apart in the original sequence, i.e., belonging to one of the Mth order polyphase components of signal $a(n)$: this representation will be referred to as M-polyphase composite representation (M-PCR). A graphical interpretation of M-PCR is provided in Fig. 1. As it will be shown in the following, such a representation allows several common processing tasks to be performed on composite signals.

While the composite representation may seem a trivial one, its use for the parallel processing of an encrypted signal is not straightforward, especially if we want to represent and process negative values. To do so, we must first establish some properties. These are given by the following theorem:

Theorem 1: Let us assume that

$$|a(n)| < Q \quad \forall n \tag{6}$$

$$B > 2Q \tag{7}$$

$$B^R \leq N \tag{8}$$

where N is a positive integer, and let $a_C(k)$ be defined as in (5). Then, the following holds:

$$0 \leq a_C(k) + \omega_Q < N \tag{9}$$

where $\omega_Q = Q\sum_{i=0}^{R-1} B^i = Q(B^R - 1)/(B-1)$. Moreover, the original samples can be obtained from the composite representation as

$$a_i(k) = \left\{\left[(a_C(k) + \omega_Q) \div B^i\right] \mathrm{mod} B\right\} - Q. \tag{10}$$

Proof: Let us express

$$a_C(k) + \omega_Q = \sum_{j=0}^{R-1} [a_j(k) + Q]\, B^j. \tag{11}$$

Thanks to (6) and (7), we have $0 \leq a_j(k) + Q \leq 2Q \leq B - 1$. Hence, $a_C(k) + \omega_Q$ can be considered as a positive base-B integer whose digits are given by $a_j(k) + Q$. Moreover, since $a_C(k) + \omega_Q$ has R digits, it is bounded by

$$a_C(k) + \omega_Q \leq \sum_{j=0}^{R-1} (B-1)B^j = B^R - 1 < N \tag{12}$$

where the last inequality comes from (8).

As to the second part of the theorem, for each i we have

$$a_C(k) + \omega_Q = B^i \sum_{j=i}^{R-1} [a_j(k) + Q] B^{j-i} + \sum_{j=0}^{i-1} [a_j(k) + Q] B^j. \tag{13}$$

Thanks to the properties of $a_j(k) + Q$, we have $\sum_{j=0}^{i-1} [a_j(k) + Q] B^j \leq B^i - 1$. Hence

$$\begin{aligned} &[a_C(k) + \omega_Q] \div B^i \\ &\quad = \sum_{j=i}^{R-1} [a_j(k) + Q] B^{j-i} \\ &\quad = B \sum_{j=i+1}^{R-1} [a_j(k) + Q] B^{j-i-1} + a_i(k) + Q \end{aligned}$$

from which (10) follows hence completing the proof.

A. Packing and Unpacking Operations

Let us now analyze how it is possible to go from the samplewise representation to the composite one and back both when the plain signal and when the encrypted signal is available. We assume that the signal belongs to a party P_1 who owns the decryption key, and it is processed by a second, nontrusted, party P_2.

When working on plain data, the analysis carried out so far ensures that given the original signal samples $a(n)$, it is possible to compute the composite representation according to (5), and vice versa that the original signal values can be correctly computed from the composite representation according to (10).

When dealing with encrypted data, the first part of the previous theorem demonstrates, first of all, that the composite representation can be safely encrypted by using a homomorphic cryptosystem defined on modulo N arithmetic: in fact, as long as the hypotheses of the theorem hold, the composite data $a_C(k)$ takes no more than N distinct values, so the values of the composite signal can be represented modulo N without loss of information.

Concerning the security of the composite signal encryption, if we work with a semantically secure cryptosystem, the security is automatically achieved. The Paillier cryptosystem can be proved to be semantically secure under the assumption that deciding N-residuosity classes in $\mathbb{Z}^*_{N^2}$ is hard [16]. In [21], it is also shown that the Paillier cryptosystem hides the $\log_2 N - b$ least significant bits of the plaintext if one assumes that computing N-residuosity classes remains hard even when we are told that the size of the plaintext is less than 2^b. In order to apply the result in [21] to the security of the composite representation, one has to avoid encoding the signal samples in the b most significant bits of the plaintext.

Let us now consider the case where the original signal samples $a(n)$ have been encrypted samplewise by P_1 by using an additive (semantically secure) homomorphic cryptosystem; the encryption of the composite representation can be performed directly in the encrypted domain by P_2 through (5) and by exploiting the homomorphic properties of the cryptosystem.

Going from the composite to the samplewise representation, however, is not possible in the encrypted domain by means of homomorphic computations only, since such a conversion requires rounding and division. Then unpacking has to be carried out by the data owner P_1, or performed by means of a properly designed interactive protocol[1] involving P_1 and P_2.

IV. Processing Encrypted Composite Signals

In addition to allowing huge memory saving, in many cases of practical interest, the composite representation permits the parallel processing of several signal samples thus leading to computational savings as well. In this section, we illustrate the above possibility by focusing on two cases of great practical interest: block-wise linear transforms and linear filtering.

A. Block-Wise Linear Transforms

Given a finite length signal having size W, a linear transform can be defined by the following relationship:

$$y(r) = \sum_{n=0}^{W-1} T(n,r)a(n), \quad r = 0, 1, \ldots, W-1 \tag{14}$$

where $T(n,r)$ are a set of coefficients defining the particular transform.

In the following, we will assume that the transform coefficients have been quantized according to some rule, i.e., $T(n,r) \in \mathbb{Z}$, and that $|T(n,r)| < Q_T$.

If the transform is implemented according to (14), the transformed signal can be bounded as $|y(r)| \leq WQ_TQ$. However, several practical transforms can be factorized by relying on the properties of $T(n,r)$'s (see, for example, the DFT). Such factorizations usually lead to a faster implementation, permitting us to compute the whole transform as a series of smaller and very simple elementary transforms linked together by suitable scaling factors. Depending on the quantization of the intermediate steps, the final bound on the transformed signal is usually larger than that obtained for the direct implementation. Hence, in the following we will assume $|y(r)| \leq Q_S$, where Q_S is an upper bound that should be computed according to the particular implementation of the transform.

[1]When B is a power of two, the unpacking could be performed using the protocol in [22]. However, since the above protocol has to extract every bit of the composite representation, it leads to an inefficient solution.

Usually, a transform is applied to the whole signal. However, when the size of a signal is not specified *a priori* or it is very large, it is customary to partition the signal into adjacent blocks having some predetermined size and apply the transform to each block separately. This is the case, for example, of audio and image coding.

In our case, the signal $a(n)$ is first partitioned into adjacent blocks; then each block is transformed separately as

$$u_i(r) = \sum_{k=0}^{M-1} T(k,r)a(iM+k), \quad r = 0,1,\ldots,M-1 \tag{15}$$

where i indicates the ith block being transformed. Since the same processing is applied to all the blocks, it is suitable for a parallel implementation relying on the composite representation of $a(n)$.

To be specific, we define the equivalent parallel blockwise transform as

$$u_C(r) = \sum_{k=0}^{M-1} T(k,r)a_C(k), \quad r = 0,1,\ldots,M-1. \tag{16}$$

where $a_C(k)$ is the M-PCR of $a(n)$. In the following, we will assume that the length of $a(n)$ is a multiple of RM, so that the length of the corresponding M-PCR is a multiple of M.

Proposition 2: If $B > 2Q_S$, then $u_i(r), i = 0,1,\ldots,R-1$, can be exactly computed from the modulo N representation of $u_C(r)$.

Proof: Let us consider the following equalities:

$$\begin{aligned} u_C(r) &= \sum_{k=0}^{M-1} T(k,r) \sum_{i=0}^{R-1} a(iM+k)B^i \\ &= \sum_{i=0}^{R-1} \left[\sum_{k=0}^{M-1} T(k,r)a(iM+k) \right] B^i \\ &= \sum_{i=0}^{R-1} u_i(r)B^i. \end{aligned} \tag{17}$$

Then, it suffices to note that $|u_i(r)| \le Q_S$ and replace Q with Q_S in the proof of Theorem 1.

B. Convolution and Linear Filtering

The output of a linear filter having impulse response $h(n)$ when the input is the sequence $a(n)$ is given by the convolution of two sequences, defined as

$$y(n) = \sum_{r=-\infty}^{\infty} h(r)a(n-r). \tag{18}$$

In practical applications, the convolution algorithm is useful when the impulse response of the filter is finite. In our application, we consider an FIR filter of length L and we will assume that the input sequence $a(n)$ has length $P = RM$. Hence, we will consider the finite convolution

$$u(n) = \sum_{r=0}^{L-1} h(r)a(n-r), \quad n = 0,1,\ldots,P+L-2 \tag{19}$$

where we assume $h(r) \in \mathbb{Z}$, and $a(n) = 0$ for $n \ge P$. Due to the properties of $a(n)$, we can bound the output as $|u(n)| \le Q \sum_{r=0}^{L-1} |h(r)| = Q_F$. The proposed block convolution is defined as

$$u_C(k) = \sum_{r=0}^{L-1} h(r)\tilde{a}_C(k-r), \quad k = 0,1,\ldots,M+L-2 \tag{20}$$

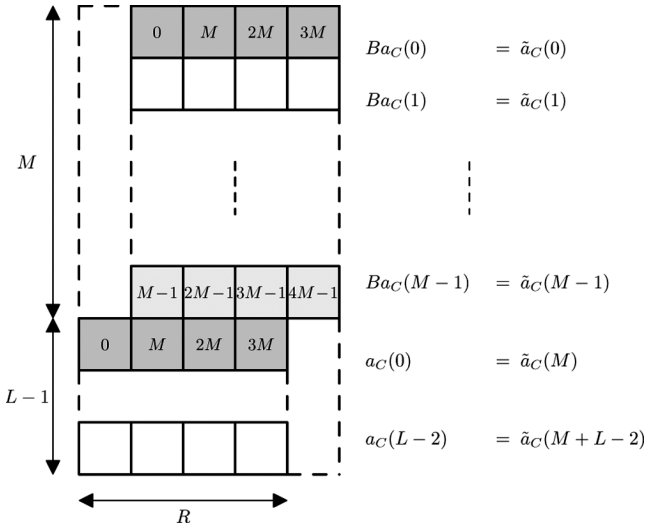


Fig. 2. Graphical representation of $\tilde{a}_C(k)$ as in (21). The values inside the boxes indicate the index of the samples within the composite representation: for $0 \le k < M$, each sample of $a_C(k)$ is shifted one position to the right, whereas for $M \le k < M+L-1$ it is a copy of $a_C(k-M)$.

where we define

$$\tilde{a}_C(k) = \begin{cases} Ba_C(k), & 0 \le k < M \\ a_C(k-M), & M \le k < M+L-1 \\ 0, & \text{elsewhere} \end{cases} \tag{21}$$

and $a_C(k)$ is the M-PCR of $a(n)$. A graphical representation of the disposition of the elements of $a(n)$ within $\tilde{a}_C(k)$ is given in Fig. 2. As it can be seen, the particular concatenation of the composite words (rows) properly extends each of the R parallel subsequences (columns) at the boundaries.

Proposition 3: If $B > 2Q_F$ and $B^{R+1} \le N$, then $u(n)$ can be exactly computed from the modulo N representation of $u_C(k), L-1 \le k < M+L-1$, as

$$u(iM+k) = \left\{ \left[(u_C(k) + \omega_{Q_F}) \div B^{i+1} \right] \bmod B \right\} - Q_F \tag{22}$$

where $\omega_{Q_F} = Q_F \sum_{i=0}^{R} B^i$.

Proof: Let us consider $u_C(k)$. For $L-1 \le k < M$, we have

$$\begin{aligned} u_C(k) &= \sum_{s=k-L+1}^{k} h(k-s)Ba_C(s) \\ &= \sum_{s=k-L+1}^{k} h(k-s) \sum_{j=0}^{R-1} a(jM+s)B^{j+1}. \end{aligned} \tag{23}$$

By letting $i = j+1$ and $r = k-s$, we obtain

$$\begin{aligned} u_C(k) &= \sum_{s=k-L+1}^{k} h(k-s) \sum_{i=1}^{R} a(iM-M+s)B^i \\ &= \sum_{i=1}^{R} \left[\sum_{r=0}^{L-1} h(r)a(iM-M+k-r) \right] B^i \\ &= \sum_{i=1}^{R} u(iM-M+k)B^i \end{aligned} \tag{24}$$

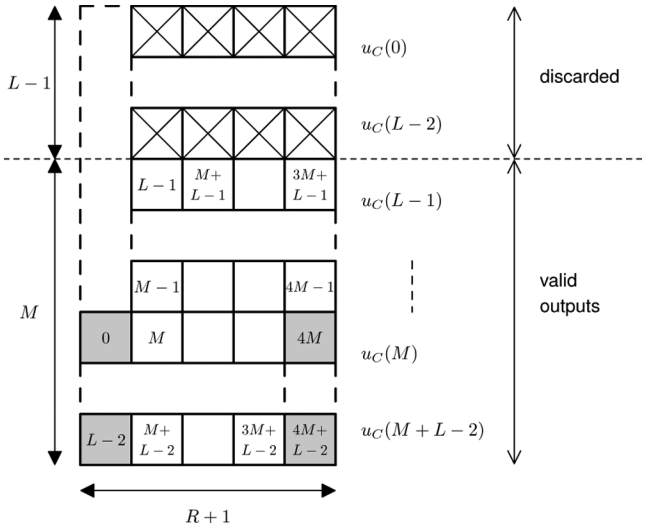


Fig. 3. Graphical representation of $u_C(k)$. The values inside the boxes indicate the index of the samples within the composite representation. Blank boxes indicate valid convolution output values. Shaded boxes indicate convolution outputs at the boundaries. Crossed boxes indicate values that have to be discarded.

whereas for $M \leq k < M+L-1$, we have

$$\begin{aligned} u_C(k) &= \sum_{s=k-L+1}^{M-1} h(k-s) B a_C(s) \\ &\quad + \sum_{s=M}^{k} h(k-s) a_C(s-M) \\ &= \sum_{s=k-L+1}^{M-1} h(k-s) \sum_{j=0}^{R-1} a(jM+s) B^{j+1} \\ &\quad + \sum_{s=M}^{k} h(k-s) \sum_{i=0}^{R-1} a(iM-M+s) B^{i}. \end{aligned} \tag{25}$$

By letting $i = j+1$, we obtain (26), shown at the bottom of the page. We have $|\sum_{s=M}^{k} h(k-s)a(-M+s)| < Q_F$ and $|u_C(k)| \leq Q_F$. Hence, the demonstration follows from the proof of Theorem 1 by replacing Q with Q_F and R with $R+1$.

Fig. 3 shows the organization of the output values within $u_C(k)$. Note that the first $L-1$ composite words must be discarded since they are affected by boundary effects.

1) Overlap-Add and Overlap-Save Implementations: The overlap-add and overlap-save methods are well-known techniques for efficiently implementing convolutions of long signals through the fast Fourier transform (FFT) [23]. For the sake of brevity, we will consider only the overlap-add method: similar results can be derived in a straightforward manner for the overlap-save method. Let us consider the signal $a(n)$ and an FIR filter $h(n)$ of length L. The overlap-add method partitions the input signal into nonoverlapping blocks of M samples and appends $L-1$ zeros to each block. The lth input block is then obtained as

$$a_l(n) = \begin{cases} a(lM+n), & 0 \leq n < M \\ 0, & M \leq n < M+L-1. \end{cases} \tag{27}$$

Usually, M is chosen so that $K = M+L-1$ is a power of two. Thanks to the properties of the FFT, the circular convolution between $a_l(n)$ and $h(n)$ can be obtained as

$$y_l(n) = \mathcal{F}_K^{-1}\{\mathcal{F}_K\{a_l(n)\} \cdot \mathcal{F}_K\{h(n)\}\} \tag{28}$$

where $\mathcal{F}_K\{\}$ denotes the K-point FFT of a sequence.

Finally, the linear convolution between $a(n)$ and $h(n)$ is obtained by adding the last $L-1$ samples of each block to the first $L-1$ samples of the following block, that is

$$y(lM+n) = \begin{cases} y_l(n) + y_{l-1}(n+M), & 0 \leq n < L-1 \\ y_l(n), & L-1 \leq n < M. \end{cases} \tag{29}$$

The overlap-add technique can be easily adapted to the proposed composite representation, by using the previously described block-wise version of the FFT. Let us consider $a_C(k)$, the M-PCR of $a(n)$. A parallel overlap-add implementation can be obtained by considering the following blocks of composite words:

$$\tilde{a}_C(k) = \begin{cases} a_C(k), & 0 \leq k < M \\ 0, & M \leq k < M+L-1. \end{cases} \tag{30}$$

Note that the above structure is equivalent to R consecutive blocks extended with a tail of $L-1$ zeros, since a composite word having zero value contains all zero samples. Hence, we can compute R circular convolutions in parallel by applying the following processing:

$$\tilde{y}_C(k) = \mathcal{BF}_K^{-1}\{\mathcal{BF}_K\{\tilde{a}_C(k)\} \cdot \mathcal{F}_K\{h(n)\}\} \tag{31}$$

where $\mathcal{BF}_K$ denotes a block-wise K-point FFT. Finally, we can obtain the M-PCR representation of the linear convolution by using a parallel overlap-add step, which is implemented as follows:

$$y_C(k) = \begin{cases} \tilde{y}_C(k) + B\tilde{y}_C(k+M), & 0 \leq k < L-1 \\ \tilde{y}_C(k), & L-1 \leq k < M. \end{cases} \tag{32}$$

A graphical representation of the data structures involved in the parallel overlap-add method is given in Fig. 4.

Note that in this case the composite representation has order $R+1$, due to the multiplication by B. Hence, similar to the block convolution case, the proposed scheme can be safely used with modulo N arithmetic if $B^{R+1} \leq N$.

$$\begin{aligned} u_C(k) &= \sum_{s=k-L+1}^{M-1} h(k-s) \sum_{i=1}^{R} a(iM-M+s) B^i + \sum_{s=M}^{k} h(k-s) \sum_{i=0}^{R-1} a(iM-M+s) B^i \\ &= \sum_{s=M}^{k} h(k-s) a(-M+s) + \sum_{i=1}^{R} \left[\sum_{s=k-L+1}^{k} h(k-s) a(iM-M+s) \right] B^i \\ &= \sum_{s=M}^{k} h(k-s) a(-M+s) + \sum_{i=1}^{R} u(iM-M+k) B^i \end{aligned} \tag{26}$$

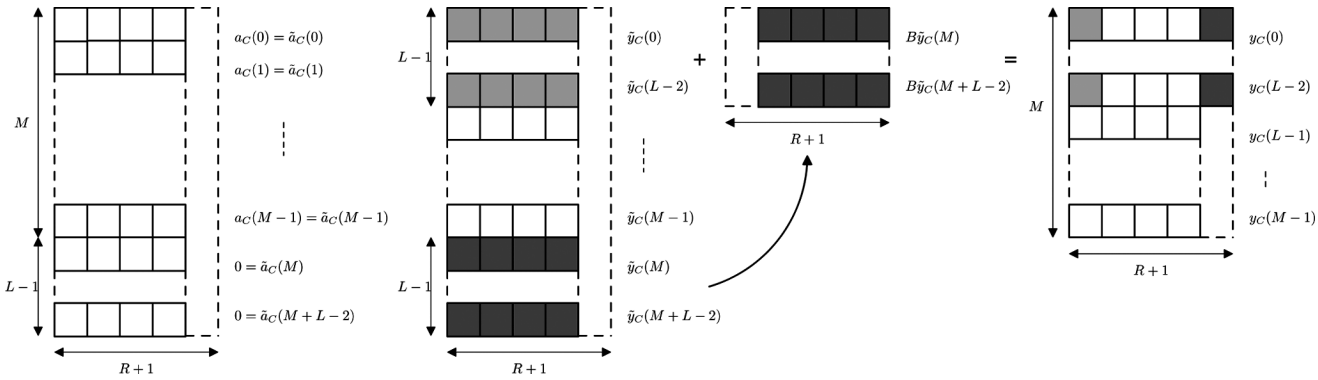


Fig. 4. Graphical representation of parallel overlap-add data structures. From left to right: input block $\tilde{a}_C(k)$; intermediate circular convolution $\tilde{y}_C(k)$; final linear convolution $y_C(k)$. Shaded boxes indicate convolution outputs at the boundaries.

C. Practical Considerations

The proposed composite representation permits us to reduce both the computational complexity and the bandwidth usage in s.p.e.d. applications. With reference to blockwise transforms, the composite representation allows us to process R samples in parallel by using a single s.p.e.d. operation. As to the block convolution, by computing a single output of (23) we obtain R output values of the original convolution. Therefore, the complexity of a block operation is reduced by a factor R with respect to that of the corresponding operation implemented sample-by-sample. Also the bandwidth usage is reduced by the same factor, since in all cases we pack R signal samples into a single ciphertext.

It is important to note that the constraints deriving from different applications usually lead to different choices of R. Let us consider an estimate of the number of samples that can be safely packed into a single word. A safe implementation requires at least $B = 2Q_Z + 1$, where Q_Z is a bound on the output of the computation. Since we must have $B < \sqrt[R]{N}$, this leads to

$$R_{\max} = \left\lfloor \frac{\log_2 N}{\log_2(2Q_Z + 1)} \right\rfloor \approx \left\lfloor \frac{\lfloor \log_2 N \rfloor}{\log_2 Q_Z + 1} \right\rfloor = R_{U,Z}. \quad (33)$$

Moreover, the block convolution imposes a smaller B with respect to the other applications: since $R + 1$ samples must be packed in a single composite word, we have $R_{\max}^{conv} = R_{U,Z} - 1$.

V. Linear Filtering of 1-D Signals

In this section, we describe a case study showing the impact that the composite signal representation has on linear (FIR) filtering of 1-D signals. The choice of FIR filtering as a case study has a twofold motivation. First of all, linear filtering is one of the most common processing tools applied to 1-D signals. Second, as we have shown in Section IV, the application of an FIR filter to a composite signal is not straightforward, and the choice of the most efficient filter implementation depends on several factors, such as the order R of the composite representation, the length of the impulse response of the filter, and the required accuracy.

In the considered scenario, P_1 owns the signal to be filtered and the decryption key, while P_2 owns the coefficients of the filter. At the end of the processing, P_2 will have the encrypted and filtered signal in composite form. As outlined in Section III-A, at this point, we have two possibilities: 1) the encrypted and composite result is sent to P_1, who can decrypt the composite data and extract the samples from the composite representation, whose parameters B and R, as well as the final Q_Z, are assumed to be public; 2) the encrypted and composite results is the input of another secure protocol that requires a samplewise representation and the intermediate plaintext samples cannot be disclosed, so that we need a suitable protocol to extract the encrypted samples from the encrypted composite representation. In the proposed example, we will concentrate on the filtering of the encrypted signal samples only. The analysis of a suitable unpacking protocol is left for future research.

To be specific, the proposed case study concerns the comparison between a time-domain and a frequency-domain implementation of an FIR filter when the inputs are represented as composite words. For the time-domain implementation, we consider the parallel implementation of convolution described in Section IV-B, while for the frequency-domain implementation, we will consider the parallel implementation of the overlap-add method depicted in Section IV-BI. An M-PCR signal representation is assumed in both cases.

In traditional frequency-domain implementations, a careful choice of the FFT size usually yields a significative complexity reduction with respect to a time-domain implementation. However, when using the composite representation for an implementation in the encrypted domain, we must take into account that the FFT, due to its multistage structure, will cause an expansion of the magnitude of the output values and, consequently, a reduction of the number of samples that can be packed together. Hence, it is possible that for some choices of the parameters, a parallel time-domain implementation will be more convenient than a frequency-domain one.

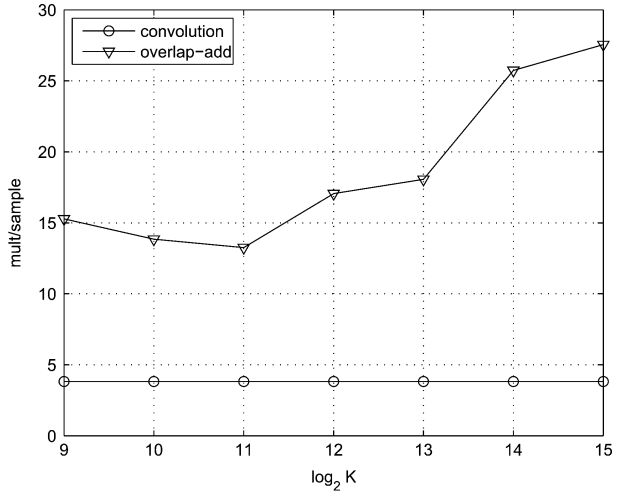
The complexity of the different implementations will be evaluated in terms of the number of multiplications required. In fact, in s.p.e.d. applications based on homomorphic encryption, multiplications are usually implemented through modular exponentiations, whose cost is much higher than that of other modular operations.

The *block* convolution requires L multiplications for each composite word, where L is the filter length. If each composite word contains $R_{U,\mathrm{CONV}}$ output samples, this leads to a complexity of

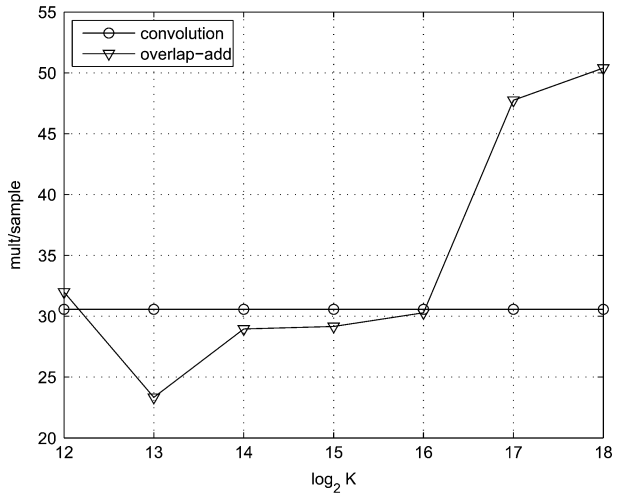
$$\mathcal{C}_{\mathrm{CONV}} = \frac{L}{R_{U,\mathrm{CONV}}} \frac{\mathrm{mult}}{\mathrm{sample}}. \quad (34)$$

As to the parallel overlap-add, it requires a K-point FFT, a K-point inverse FFT (IFFT), and K complex multiplications for a block of $MR_{U,\mathrm{OLA}}$ output samples, where $K = M + L - 1$ and $R_{U,\mathrm{OLA}}$ is the number of samples that can be packed in a single word. The complexity of s.p.e.d. FFT and IFFT is the same and has been derived in [15] both for radix-2 and radix-4 algorithms. We assume that the complexity of a complex multiplication is equal to four real multiplications. When using the radix-2 algorithm, the complexity of the frequency-domain implementation can be seen to be

$$\mathcal{C}_{\mathrm{OLA}} = \frac{6K \log_2 K - 8K}{MR_{U,\mathrm{OLA}}} \frac{\mathrm{mult}}{\mathrm{sample}}. \quad (35)$$



(a)



(b)

Fig. 5. Complexity of block convolution versus parallel overlap-add. 8-bit input values ($Q = 2^7$) and 8-bit implementation ($Q_T = 2^7$) have been assumed. The upper bound on convolution outputs has been assumed $Q_F = 2^{14}$. We have assumed $\lfloor \log_2 N \rfloor = 1023$. (a) $L = 256$; (b) $L = 2048$.

In order to compare the two implementations, we consider the filtering of an 8-bit signal, i.e., $Q = 2^7$. An 8-bit quantization of the filter coefficients is assumed, i.e., $Q_T = 2^7$. The same precision is assumed for quantizing the FFT coefficients. Finally, we assume that $\sum_{n=0}^{L-1} |h(n)| < 2^7$, so that the upper bound on the convolution output is $Q_F = 2^{14}$. As to the upper bound on the output of the overlap-add implementation, we can exploit the results in [15]. The upper bound on the output of a radix-2 FFT/IFFT can be expressed as

$$Q_{R2} = Q_1 Q_T^{\nu-2} + 4\epsilon_1 \left(2Q_T + \frac{1}{\sqrt{2}}\right)^{\nu-2} + \sum_{l=0}^{\nu-3} \frac{2^{\nu-1-l}}{\sqrt{2}} Q_1 Q_T^{\nu-3-l} \left(2Q_T + \frac{1}{\sqrt{2}}\right)^{l} \quad (36)$$

where $\nu = \log_2 K$, Q_1 is the scale factor multiplying the input values, and ϵ_1 is an upper bound on the quantization error. The final upper bound can be found by recursively using the aforementioned result. Once the corresponding upper bounds have been found, they can be inserted into (33) to find both $R_{U,\mathrm{CONV}}$ and $R_{U,\mathrm{OLA}}$.

TABLE I
NUMBER OF SAMPLES R THAT CAN BE PACKED INTO A SINGLE WORD WHEN USING A BLOCK CONVOLUTION AND A PARALLEL OVERLAP-ADD IMPLEMENTATION WITH A K-POINT FFT. 8-bit INPUT VALUES ($Q = 2^7$) AND 8-bit IMPLEMENTATION ($Q_T = 2^7$) HAVE BEEN ASSUMED. THE UPPER BOUND ON CONVOLUTION OUTPUTS HAS BEEN ASSUMED $Q_F = 2^{14}$. WE HAVE ASSUMED $\lfloor \log_2 N \rfloor = 1023$

	convolution	overlap-add						
K	-	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}
R	68	7	6	6	5	5	4	4

The complexities of the two implementations are compared in Fig. 5, considering two filter lengths, $L = 256$ and $L = 2048$. As it can be seen, with the shorter filter the block time-domain implementation is more convenient than the frequency-domain one, irrespective of the size of the FFT. In order to have some gain from the overlap-add implementation, we have to consider quite a long filter. Such a behavior can be explained by looking at Table I, in which the value of R for different implementations and different FFT sizes is shown. It is evident that the use of the s.p.e.d. FFT greatly reduces the number of samples that can be packed in a single encrypted word, so the computational saving of the overlap-add method in terms of number of multiplications is counterbalanced by the reduced efficiency of the composite representation.

VI. CONCLUSIONS

In this paper, we have analyzed the possibility of reducing the expansion factor required in s.p.e.d. applications based on homomorphic encryption by packing together several signal samples into a unique composite word. To do so, we provided a general framework extending an idea put forward in [18] and derived precise conditions that permit us to process the underlying signal by operating directly on the composite words, thus achieving a significant gain from a computational complexity perspective. The advantage brought by the composite representation is, therefore, twofold: on the one hand, it permits us to speed up computation via parallel processing; on the other hand, it reduces the size of the encrypted signals, having beneficial effects on both storage and bandwidth usage.

We illustrated several ways of exploiting the composite representation to perform s.p.e.d. operations, like linear transforms and convolutions. The proposed implementations scale the computational complexity by a factor R, where R is the number of samples that are safely packed into a single word. The encrypted signal size is reduced by the same factor.

A case study has been proposed, where we investigated the use of the composite representation for linear filtering of 1-D sequences. The results show that the composite representation can be a viable solution to the problems of both data expansion and increased complexity arising from the processing of encrypted data. Interestingly, when resorting to the composite representation it can be more convenient to use a simpler processing algorithm—like filtering through convolution—since they allow packing together more samples than the corresponding fast versions.

Throughout the paper, we have focused on 1-D signals; however, all the arguments can be easily extended to 2-D or multidimensional signals.

An open problem that is left for future research is the development of an efficient protocol that permits us to pass from the composite to the sample-wise representation without the parties involved in the protocol sharing any secret information. Existing schemes, in fact, are either computationally inefficient or can only be applied to the particular case of $B = 2$ (see [22] for an example in this sense).

REFERENCES

[1] A. Piva and S. Katzenbeisser, "Signal processing in the encrypted domain," *EURASIP J. Inf. Security*, vol. 2007, p. 1, 2007, Article 82790.

[2] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proc. 2000 ACM SIGMOD Int. Conf. Management of Data (SIGMOD'00)*, New York, 2000, vol. 29(2), pp. 439–450.

[3] B. Pinkas, "Cryptographic techniques for privacy-preserving data mining," *SIGKDD Explor. Newsl.*, vol. 4, no. 2, pp. 12–19, 2002.

[4] J. Shashank, P. Kowshik, K. Srinathan, and C. Jawahar, "Private content based image retrieval," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2008, pp. 1–8.

[5] J. F. Canny, "Collaborative filtering with privacy," in *Proc. IEEE Symp. Security and Privacy*, 2002, pp. 45–57.

[6] W. Du and M. J. Atallah, "Privacy-preserving cooperative scientific computations," in *Proc. 14th IEEE Computer Security Foundations Workshop*, Nova Scotia, Canada, Jun. 11–13, 2001, pp. 273–282.

[7] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Trans. Signal Process.*, vol. 52, no. 10, pp. 2992–3006, Oct. 2004.

[8] N. Memon and P. Wong, "A buyer-seller watermarking protocol," *IEEE Trans. Image Process.*, vol. 10, no. 4, pp. 643–649, Apr. 2001.

[9] R. Rivest, L. Adleman, and M. Dertouzos, "On data banks and privacy homomorphisms," in *Foundations of Secure Computation*, R. A. DeMillo, Ed. *et al.* New York: Academic, 1978, pp. 169–179.

[10] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[11] A. C. Yao, "Protocols for secure computations," in *Proc. 23rd IEEE Symp. Foundations of Computer Science*, Chicago, IL, Nov. 1982, pp. 160–164.

[12] O. Goldreich, S. Micali, and A. Wigderson, "How to play ANY mental game," in *Proc. 19th Annu. ACM Conf. Theory of Computing*, New York, 1987, pp. 218–229.

[13] R. Cramer, I. Damgård, and J. B. Nielsen, "Multiparty computation from threshold homomorphic encryption," *Lecture Notes in Computer Science*, vol. 2045, pp. 280–299, 2001.

[14] Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven, and M. Barni, "Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing," *EURASIP J. Inf. Security*, vol. 2007, p. 20, 2007, Article 78943.

[15] T. Bianchi, A. Piva, and M. Barni, "On the implementation of the discrete Fourier transform in the encrypted domain," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 1, pp. 86–97, Mar. 2009.

[16] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Lecture Notes in Computer Science*. New York: Springer-Verlag, 1999, vol. 1592, pp. 223–238.

[17] M. Chen, O. Boric-Lubecke, and V. M. Lubecke, "$0.5-\mu$m CMOS implementation of analog heart-rate extraction with a robust peak detector," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 4, pp. 690–698, Apr. 2008.

[18] J.-R. T. Pastoriza, S. Katzenbeisser, M. Celik, and A. Lemma, "A secure multidimensional point inclusion protocol," in *ACM Multimedia and Security Workshop (MMSEC'07)*, New York, 2007, pp. 109–120.

[19] I. Damgård and M. Jurik, "A generalisation, a simplification and some applications of Paillier's probabilistic public-key system," *Public Key Cryptography*, pp. 119–136, 2001.

[20] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Theory of Computing (STOC'09)*, New York, 2009, pp. 169–178.

[21] D. Catalano, R. Gennaro, and N. Howgrave-Graham, "The bit security of Paillier's encryption scheme and its applications," in *Proc. Int. Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT'01)*, London, U.K., 2001, pp. 229–243.

[22] B. Schoenmakers and P. Tuyls, "Efficient binary conversion for Paillier encrypted values," in *Advances in Cryptology (EUROCRYPT 2006)*. Berlin, Heidelberg, Germany: Springer, 2006, pp. 522–537.

[23] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. : Prentice-Hall International Inc., 1975.

Reversible Image Watermarking Using Interpolation Technique

Lixin Luo, Zhenyong Chen, Ming Chen, Xiao Zeng, and Zhang Xiong

***Abstract*—Watermarking embeds information into a digital signal like audio, image, or video. Reversible image watermarking can restore the original image without any distortion after the hidden data is extracted. In this paper, we present a novel reversible watermarking scheme using an interpolation technique, which can embed a large amount of covert data into images with imperceptible modification. Different from previous watermarking schemes, we utilize the interpolation-error, the difference between interpolation value and corresponding pixel value, to embed bit "1" or "0" by expanding it additively or leaving it unchanged. Due to the slight modification of pixels, high image quality is preserved. Experimental results also demonstrate that the proposed scheme can provide greater payload capacity and higher image fidelity compared with other state-of-the-art schemes.**

***Index Terms*—Additive interpolation-error expansion, data hiding, interpolation-error, reversible watermarking.**

I. INTRODUCTION

Digital watermarking is a kind of data hiding technology. Its basic idea is to embed covert information into a digital signal, like digital audio, image, or video, to trace ownership or protect privacy. Among different kinds of digital watermarking schemes, reversible watermarking has become a research hotspot recently. Compared with traditional watermarking, it can restore the original cover media through the watermark extracting process; thus, reversible watermarking is very useful, especially in applications dictating high fidelity of multimedia content, such as military aerial intelligence gathering, medical records, and management of multimedia information.

Since the earliest reversible watermarking scheme was invented by Barton [1] in 1997, dozens of reversible watermarking methods have been reported in the literature and classified into three categories by Feng *et al.* [2]: reversible watermarking using data compression, reversible watermarking using difference expansion (DE), and reversible watermarking using histogram operation. In these categories, the first kind has complex computation and limited capacity, whereas the latter two are better in both of two criteria.

DE, also known as a kind of integer wavelet transform, was first proposed by Tian [3]. By expanding the difference between the two neighboring pixels of pixel pairs, Tian explored the redundancy in digital images to achieve a high-capacity and low-distortion reversible watermarking. Later on, Alattar [4] extended Tian's scheme by a generalized DE method which hid several bits in the DE of vectors of adjacent pixels. Then, Kim *et al.* [5] proposed a novel scheme devoted to reduce the size of the location map. Furthermore, Lin *et al.* [6] proposed another DE scheme, where the location map was removed completely.

Manuscript received July 16, 2009; accepted October 09, 2009. First published November 06, 2009; current version published February 12, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ton Kalker.

The authors are with the School of Computer Science and Engineering, Beihang University, Beijing 100191, China (e-mail: lixinluo@cse.buaa.end.cn; chzhyong@buaa.edu.cn; mingchen@cse.buaa.edu.cn; zengxiao29@gmail.com; xiongz@buaa.edu.cn).

Digital Object Identifier 10.1109/TIFS.2009.2035975