Insertion, Deletion Codes With Feature-Based Embedding: A New Paradigm for Watermark Synchronization With Applications to Speech Watermarking

David J. Coumou, *Member, IEEE*, and Gaurav Sharma, *Senior Member, IEEE*

***Abstract*—A framework is proposed for synchronization in feature-based data embedding systems that is tolerant of errors in estimated features. The method combines feature-based embedding with codes capable of simultaneous synchronization and error correction, thereby allowing recovery from both desynchronization caused by feature estimation discrepancies between the embedder and receiver; and alterations in estimated symbols arising from other channel perturbations. A speech watermark is presented that constitutes a realization of the framework for 1-D signals. The speech watermark employs pitch modification for data embedding and Davey and Mackay's insertion, deletion, and substitution (IDS) codes for synchronization and error recovery. Experimental results demonstrate that the system indeed allows watermark data recovery, despite feature desynchronization. The performance of the speech watermark is optimized by estimating the channel parameters required for the IDS decoding at the receiver via the expectation-maximization algorithm. In addition, acceptable watermark power levels (i.e., the range of pitch modification that is perceptually tolerable) are determined from psychophysical tests. The proposed watermark demonstrates robustness to low-bit-rate speech coding channels (Global System for Mobile Communications at 13 kb/s and AMR at 5.1 kb/s), which have posed a serious challenge for prior speech watermarks. Thus, the watermark presented in this paper not only highlights the utility of the proposed framework but also represents a significant advance in speech watermarking. Issues in extending the proposed framework to 2-D and 3-D signals and different application scenarios are identified.**

***Index Terms*—Feature-based watermarking, insertion deletion codes, pitch watermarking, speech watermarking, watermark synchronization.**

I. Introduction

AS IN ANY communication system, multimedia watermarking methods[1] require synchronization between the transmission and the reception sides before data transfer can occur. In watermarking, however, synchronization poses a more acute challenge than in traditional communication systems because the multimedia cover signal (and not the watermark) is, in fact, the primary signal being conveyed from the source to the destination. Between watermark embedding and extraction, it is reasonable in most systems to assume that the perceptual content and quality of the multimedia signal is largely preserved. Within this constraint, however, the multimedia signal may be subject to a variety of linear and nonlinear signal-processing operations. In applications where an original is available at the receiver, registration of the received signal to the original can enable synchronization [1], [2]. For the large majority of applications where an original is not available at the receiver, we are usually faced with an effective watermark channel for which synchronization is difficult.

A number of approaches have been explored for synchronization in oblivious watermarking (see [3] and [4] for an overview/taxonomy). Methods presented in the literature can be broadly categorized into two main classes: 1) methods that embed the watermark data in multimedia signal features that are invariant to the signal-processing operations, or in regions determined by such features and 2) methods that enable synchronization through the estimation and (approximate) reversal of the geometric transformations that the multimedia signal has been subjected to after watermark embedding. Approaches in the former category include methods that use the Fourier–Melin transform space for rotation, translation, scale invariance [5], methods that embed watermarks in geometric invariants, such as image moments [6], [7], and methods that use semantically meaningful signal features, either for embedding [8] or for partitioning the signal space into regions for embedding [9]. Examples of the latter category are methods using repeated embedding of the same watermark [10], [11] or the inclusion of a transform domain pilot watermark [12] explicitly for the purpose of synchronization.

Among these techniques, the methods based on semantic features hold considerable promise since these features are directly related to the perceptual content of the multimedia signal and, therefore, conserved in benign and malicious signal-processing operations. Kutter [13] introduced this class of techniques as second-generation watermarking methods and identified three essential properties for the semantic features: 1) invariance to noise; 2) covariance to geometrical transformations; and 3)

Manuscript received April 7, 2007; revised January 30, 2008. Parts of this work were included in an ICME 2006 paper [40] and an invited presentation for CISS 2006 [4]. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ton Kalker.

D. J. Coumou is with the Electrical and Computer Engineering Department, University of Rochester, Rochester, NY 14627 USA and also with MKS Instruments Inc., Rochester, NY 14623 USA (e-mail: DavidCoumou@ieee.org).

G. Sharma is with the Electrical and Computer Engineering Department, University of Rochester, Rochester, NY 14627 USA and also with the Department of Biostatistics and Computational Biology, University of Rochester Medical Center, Rochester, NY 14642 USA (e-mail: gaurav.sharma@rochester.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2008.920728

[1]For our discussion, we consider watermarking systems to broadly include all digital data-embedding systems.

resilience against local modifications. Despite their conceptual advantages, second-generation watermarking methods have proven to be difficult to implement in practical systems [3], [14]. A primary reason for this difficulty is that robust and repeatable extraction of semantically meaningful signal features continues to be a challenging research problem in itself. In particular, benign processing or a malicious change may cause additional feature points to be detected or some existing feature points to be deleted, leading to desynchronization of the watermark channel.

In this paper, we propose a new framework for synchronization in these second-generation methods based on error-correction codes for channels with insertions and deletions [15], [16]. We demonstrate the framework using a speech watermarking system based on pitch modification previously developed within our group [8] and illustrate how it allows recovery of synchronization despite mismatches in estimated features between embedding and receiving ends. The demonstration also addresses the challenging problem of speech watermarking over low bit-rate compression channels [17], which is a useful contribution in itself.

The rest of this paper is organized as follows. In Section II, we introduce a general framework for feature-based multimedia data embedding with coding for simultaneous synchronization and error correction. Sections III–V describe a speech watermark that constitutes a realization of this framework. Section III describes a data-embedding method for speech that utilizes pitch modification. In Section IV, we provide a model for communication channels characterized by insertion, deletion, and substitution events and introduce Davey and MacKay's [15] coding methodology for reliable communication over such channels. Section V then provides a short overview of the complete speech-watermarking system and relates it to the general framework. Section VI describes the implementation of the speech-watermark and includes results of psychophysical tests performed in order to determine perceptually tolerable limits for pitch-based embedding. Experimental results for the proposed speech watermark with synchronization are presented in Section VII, where the method is also compared with a simple spread-spectrum watermark in order to illustrate that desynchronization is encountered over low-bit-rate coding channels. Section VIII presents conclusions and discusses possible extensions and future work. Algorithms used in the encoding/decoding process for the joint synchronization and error recovery are summarized in the Appendix that constitutes the final section of this paper. The performance of the decoding process is improved by using an expectation maximization algorithm in order to estimate the channel parameters. The algorithm utilized for this purpose is also included in the Appendix.

II. Feature-Based Multimedia Data Embedding With Synchronization

Fig. 1 is an overview of the multimedia data embedding framework that we propose here for the purpose of watermark synchronization. We describe the method in a general setting and present specific details for speech embedding in the following sections. The dashed block in the figure represents the

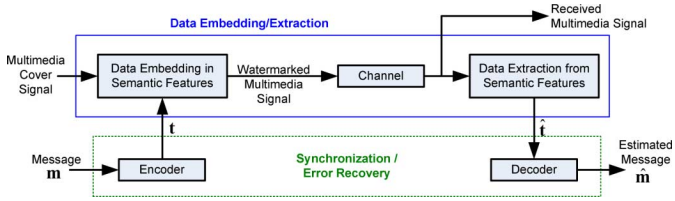

Fig. 1. Feature-based data embedding with synchronization.

basic data embedding and extraction technique, which at the transmitting end, embeds data $\mathbf{t}$ in the signal through modifications of semantic features in the multimedia signal and, at the receiving end, extracts the data $\hat{\mathbf{t}}$ through the estimation of the semantic features. Since distortions introduced in the channel (or even in the embedding process itself) may cause extracted data to differ from that at the transmitter [14], we incorporate an additional encoding/decoding step shown in the dotted block in Fig. 1 for synchronization and error recovery.

The framework that is presented is generic and requires further exploration of several aspects depending on the type of signal and the application: determination of appropriate features, selection of an embedding domain, and method that offers desired resilience, selection of suitable codes for the recovery of synchronization, and error correction. We focus our investigation on the particular problem of synchronization when feature estimates between the embedding and receiving ends may differ, which has stymied feature-based watermarking methods. For this purpose, we select a speech watermarking application that affords a significant simplification due to the 1-D nature of the signal. At the same time, speech watermarking still presents fundamental challenges due to the special structure of low-bit-rate speech coders that are based on linear predictive coding methods [18]. A unique characteristic of these techniques among multimedia compression standards is that they are based on modeling the signal source (i.e., the vocal tract apparatus, rather than the human perceptual characteristics at the receiving end [18]–[22]). The compressor analyzes the speech signal to determine appropriate model parameters which are communicated to the receiving end. The decompressor at the receiver utilizes the parameters received to synthesize an approximation to the speech signal. This process preserves the relevant signal features that constitute the model parameters but does not offer any guarantees for preservation of the signal waveform or geometry (i.e., the time axis).

Thus, in the watermarking context, low-bit-rate encoding represents a nonmalicious geometric distortion channel. Specifically, for the adaptive multirate (AMR) speech encoder [21], regions of silence may not necessarily be reconstructed with the same duration, causing desynchronization in watermarking methods relying on the signal geometry for synchronization. For this reason, low-bit-rate speech compression channels present a particularly difficult challenge for waveform and transform-domain-based embedding methods [23]. The nature of these low-bit-rate coding channels also makes them ideally suited for feature-based watermarking, where the signal features for the embedding are matched to the encoding and decoding. We develop our feature-based speech watermark considering low bit-rate encoding channels. We also consider additive noise distortions but

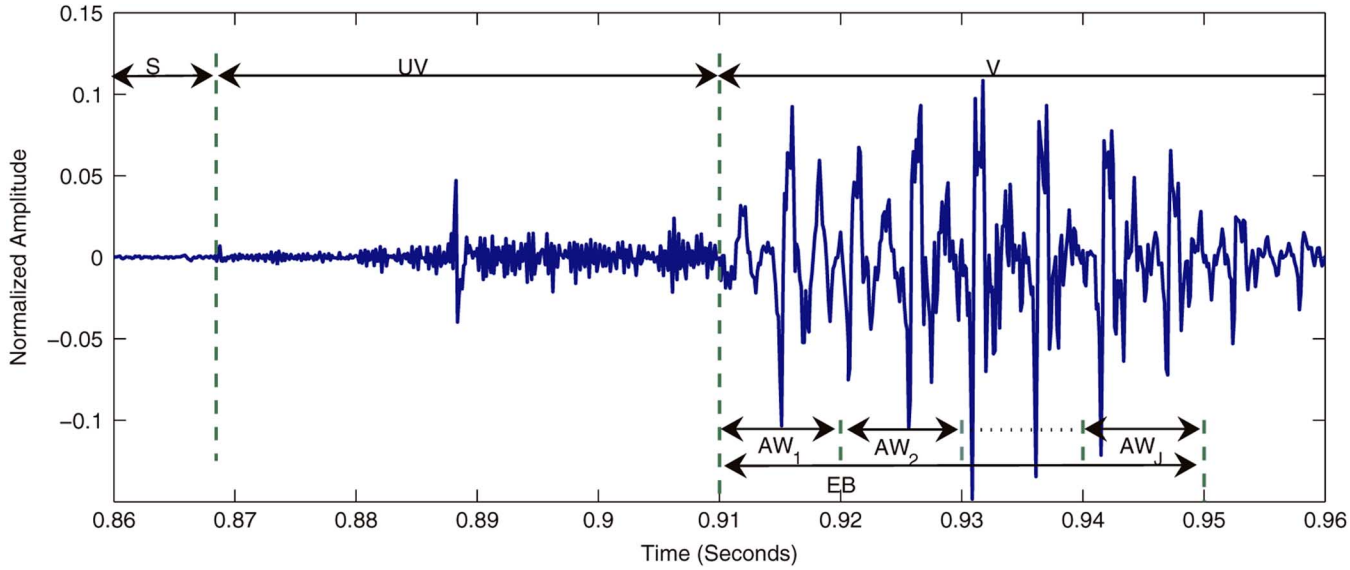

Fig. 2. Part of a speech signal illustrating partitioning for pitch-based data embedding; S: silence segment, UV: unvoiced segment, V: voiced segment, AW: analysis windows, EB: embedding block consisting of J analysis windows.

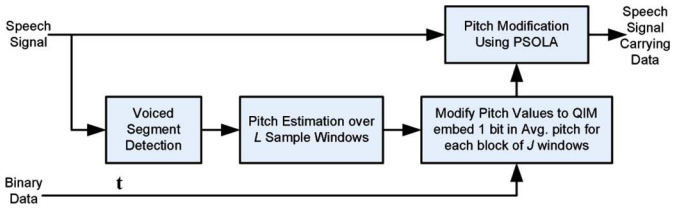

Fig. 3. Data embedding in speech by pitch modification.

do not currently address other factors (i.e., malicious geometrical distortion). Our speech watermark implementation uses pitch modification for embedding [8] and a concatenated coding system [15] for synchronization, each of which is described in the proceeding two sections.

III. Data Embedding in Speech by Pitch Modification

The pitch (i.e., fundamental period) of voiced regions of a speech signal is utilized as the "semantic" feature for data embedding [8]. This choice is motivated by the structure of most speech encoders [18]–[22] that ensure pitch information is preserved. We illustrate this by using a portion of a speech signal as shown in Fig. 2. This segment shows a initial silence segment (S), followed by an aperiodic unvoiced segment (UV), which, in turn, is followed by a voiced segment (V). The V is identified in the speech signal as the region having energy above a threshold and exhibiting periodicity. Within these voiced segments, the pitch is estimated by analyzing the speech waveform and estimating its local fundamental period over nonoverlapping analysis windows (AWs) of L samples each. An embedding block (EB) comprises several AWs.

The embedding method is schematically illustrated in Fig. 3. Data are embedded by altering the pitch period of voiced segments that have at least M contiguous windows. M is experimentally selected to avoid small isolated regions that may erroneously be classified as voiced.

Within each selected voice segment, one or more bits are embedded. A single bit is embedded by the quantization index modulation (QIM) of the average pitch value. This corresponds to the method presented in [8]. For multibit embedding, the voiced segment is partitioned into blocks of J contiguous analysis windows $(J \leq M)$ and a bit is embedded by scalar QIM of the average pitch of the corresponding block. Specifically, the average pitch for a block is computed as

$$p_{\text{avg}} = \frac{1}{J}\sum_{i=1}^{J} p_i \tag{1}$$

where $\{p_i\}_{i=1}^{J}$ are the pitch values corresponding to the analysis windows in the block.

Scalar QIM [24] is applied to the average pitch for the block

$$p'_{\text{avg}} = Q_b(p_{\text{avg}}) \tag{2}$$

where $b \in \{0,1\}$ is the embedded bit and $Q_b(u) = Q(u - b\Delta/2;\Delta) + b\Delta/2$ denotes the corresponding quantizer, where $Q(\bullet;\Delta)$ denotes the integer-scalar quantizer with scaling parameter Δ.

Modified pitch intervals for the analysis windows in the block are computed as

$$p'_i = p_i + (p'_{\text{avg}} - p_{\text{avg}}). \tag{3}$$

The corresponding pitch modifications are then incorporated in the speech waveform using the pitch synchronous overlap add (PSOLA) [25] algorithm. Note that the embedding in average pitch values over blocks of analysis windows enables embedding even when the pitch period exceeds the duration of a single window and reduces perceptibility of the changes introduced. The use of multiple embedding blocks within a voiced segment (of J analysis windows) ameliorates data capacity compared to the single-bit embedding in each voice segment.

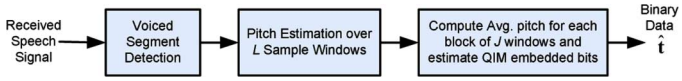

Fig. 4. Extraction of data embedded in speech by pitch modification.

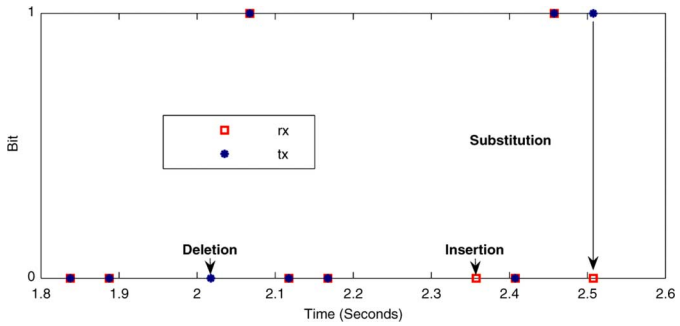

Fig. 5. IDS events in pitch data embedding/extraction.

At the receiver (shown in Fig. 4), the speech waveform is analyzed to detect voiced segments, and pitch values are estimated for nonoverlapping analysis windows of L samples each. In a process mirroring the embedding operation, the average pitch values are computed over blocks of J contiguous analysis windows. For each block, an estimated value of the embedded bit is computed as the index 0/1 of the quantizer $\{Q_b(\ )\}_{b=0}^{1}$ that has a reconstruction value closest to the average pitch. This provides an estimate of the embedded data.

Since the method embeds data only over voiced segments, it is immune against processing and shortening/lengthening of the silence regions, which may occur in low-bit-rate speech coding. Furthermore, a *new embedding block begins at the start of each embeddable voiced segment*. Hence, the start locations of the voiced segment implicitly synchronize the time windows for the embedding and extraction of different bits. This is analogous to carrier synchronization within a communication system [26]. Once this "carrier synchronization" is accomplished, synchronization at the symbol level is the remaining requirement for data communication. In this respect, one challenge for the data embedding by pitch modification is that estimates of voiced segments at the receiver may differ from those at the embedder [2] [8]. Multiple voiced segments at the embedder may coalesce into a single voiced segment at the receiver, or vice-versa. In addition, relatively small voiced segments may be detected at one end and not the other. In general, these types of mismatches result in insertion, deletion, and substitution (IDS) errors in the estimates of the embedded data. Insertion/deletion events are particularly insidious since they cause a loss of synchronization and cannot be corrected using conventional error-correction codes.

An example that illustrates IDS events in the recovery of pitch-based data embedding is shown in Fig. 5, where a time window is shown along with the embedded bits (* symbols) and extracted bits (□ symbols). From the plot, we can see that synchronism is not maintained between the embedded and extracted bits. Time locations with overlapping star and square symbols correspond to instances where embedded and extracted bits match, locations where both are present but do not match correspond to *substitution* events, instances where a square symbol occurs without a corresponding star symbol represent locations where a spurious bit is *inserted* in the received stream, and stars without corresponding squares represent a *deletion* of the corresponding transmitted bit. In Fig. 5, we see one insertion, one deletion, and one substitution event as indicated.

To address this problem, we next incorporate concatenated coding techniques [15] that allow us to synchronize and recover data over IDS channels.

IV. Synchronization Over IDS Channels

To recover from insertion/deletion events, we adopt a concatenated coding scheme developed by Davey and MacKay [15] that utilizes an outer q-ary low density parity check (LDPC) code and an inner sparse code combined with a synchronization marker vector. We first present an intuitive overview of the method and then present details of our implementation.

Fig. 6 illustrates the method schematically. We begin by considering the synchronization marker vector $\mathbf{w}$, which is a fixed (preferably pseudorandom) binary vector of length Nn that is independent of the message data $\mathbf{m}$, and known to the transmitter and receiver. It forms the data embedded at the transmitter when no (watermark) message is to be communicated. In the absence of any substitutions, knowledge of this marker vector allows the receiver to estimate insertion/deletion events and, thus, regain synchronization (with some uncertainty).

Message data to be communicated is "piggy-backed" onto the marker vector. This is accomplished by mapping the message to a unique *sparse binary vector* via a codebook, where a sparse vector is a vector that has a small number of 1's in relation to its length. The sparse vector is then incorporated in the synchronization marker prior to embedding as intentional (sparse) bit inversions at the locations of 1's in the sparse vector. Conceptually,[3] once the receiver synchronizes, since the synchronization marker vector is known to the receiver, bit inversions in the marker vector can be determined. If the channel does not introduce any substitution errors, these bit inversions indicate the locations of the 1's from the sparse vector and, therefore, allow recovery of the sparse vector and thereby the message. In the presence of additional channel-induced substitutions, the estimates of the sparse vector are uncertain. This uncertainty is resolved by the outer q-ary LDPC code. The q-ary codes offer a couple of benefits over binary codes. First, suitably designed q-ary codes with $q \geq 4$ offer performance improvements over binary codes [27]–[29] even for channels without insertions/deletions. Second, specifically for the case of IDS channels, the q-ary codes allow improved rates [15], [29] (as described at the end of this section).

A. Encoder (Inner and Outer)

For simplicity, in the following discussion, we consider the transmission of a single message block in the setup of Fig. 6. The

[2]As remarked earlier, these types of errors are encountered in almost all feature-based data embedding methods.

[3]This description is not strictly correct since the estimated synchronization has some ambiguities (as can be readily argued to be the case for any marker vector-based synchronization method). However, provided that the IDS events are reasonably infrequent, the outer LDPC code is able to compensate for the ambiguities in synchronization and the errors introduced by the channel.

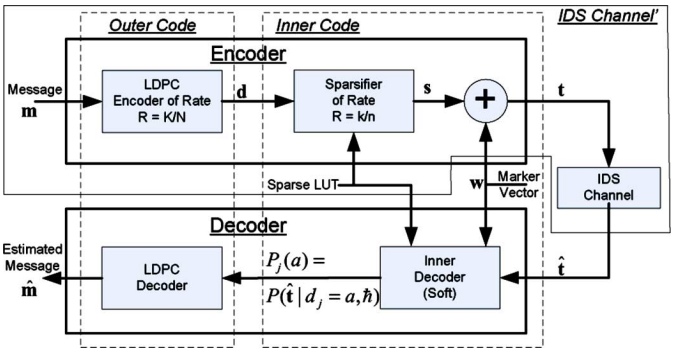

Fig. 6. Coding for IDS channels.

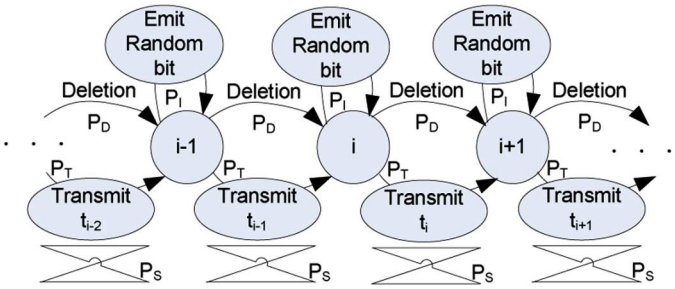

Fig. 7. IDS channel hidden Markov model.

watermark message data $\mathbf{m}$ is a block of Kq-ary symbols (with $q = 2^k$ for some k). The message $\mathbf{m}$ is encoded (in systematic form) using a rate K/Nq-ary LDPC code to obtain codeword $\mathbf{d}$, which is a block of Nq-ary symbols. The LDPC code is specified by a sparse $(N - K) \times N$ parity check matrix $\mathbf{H}$ with entries selected from $\mathrm{GF}(q)$ (i.e., the Galois field with $q = 2^k$ elements). The rate k/n sparsifier maps each q-ary symbol into an n-bit sparse vector using a lookup table (LUT) containing $q = 2^k$ entries of sparse n-bit vectors. Thus, corresponding to the codeword $\mathbf{d}$, there are *(Nn)* bits that form the sparse message vector $\mathbf{s}$ that is added to the marker vector $\mathbf{w}$ (of the same length). The overall rate of the concatenated system is *(Kk)/(Nn)* message bits per bit communicated over the IDS channel (i.e., per embedded bit).

B. IDS Channel Model

The IDS channel is assumed to follow a hidden Markov model (HMM), as shown in Fig. 7 [15], [16]. The states $\ldots, i-1, i, i+1, \ldots$ represent the (hidden) states of the model, where state i represents the situation where we are done with[4] the $(i-1)$th bit $t_{(i-1)}$ at the transmitter and poised to transmit the ith bit t_i. Consider the channel in state i. One of three events may occur starting from this state: 1) with probability P_I, a random bit is inserted in the received stream and the channel returns to state i; 2) with probability P_T, the ith bit t_i is transmitted over the channel and the channel moves to state $(i+1)$; and 3) with probability P_D the ith bit t_i is deleted and the channel moves to state $(i+1)$. When transmission occurs, the corresponding bit is communicated to the receiver over a binary symmetric channel with crossover probability P_S. A substitution (error) occurs when a bit is transmitted but received in error. The probabilities P_I, P_T, P_D, and P_S constitute the parameters for the HMM, which we will collectively denote as $\hbar'$. Note that we use two versions of the model corresponding to the blocks labeled IDS channel and IDS channel' in Fig. 6. For the latter, the substitution probability is increased suitably to account for the additional substitutions caused by the message insertion.

[4]This is either through a transmission (which may be correct or in error) or through a deletion event.

C. Inner Decoder

The soft inner decoder uses the HMM for the channel, to efficiently compute symbol-by-symbol likelihood probabilities $P_j(a) = P(\hat{\mathbf{t}}|d_j = a, \hbar)$ for $1 \leq j \leq N$, where $\hbar = (\hbar', \mathbf{w})$ represents the known information at the receiver. Note that since the symbols comprising $\mathbf{d}$ are, in fact, q-ary, $P_j(a)$ is a probability mass function (pmf) over all the q possible values of a. These pmfs form the (soft) inputs to the outer LDPC iterative decoder. The computations in the inner decoder are performed using a forward–backward procedure [30] for HMM corresponding to the IDS channel' followed by a combination step for the HMM for IDS channel [15] (see Fig. 6). Details of these may be found in [15] and a brief summary of the equations is included in the Appendix.

Note that as an alternative to this process, a Viterbi algorithm could be utilized to determine a maximum-likelihood sequence of transitions corresponding to the received vector. However, the process is suboptimal and superior performance is obtained from the forward–backward algorithm for HMM state estimation [15].

D. Outer Decoder

The symbol-by-symbol probability-mass-function vectors $\{P_j(a)\}_{a \in \mathrm{GF}(q)}; j = 1, \ldots N$ obtained from the (soft) inner decoder are the inputs for the outer q-ary LDPC decoder. The LDPC decoder is a probabilistic iterative decoder that uses the sum-product algorithm [31] to estimate marginal posterior probabilities $P(d_j|\hat{\mathbf{t}}, \mathbf{H})$ for the codeword symbols $\{d_j\}_{j=1}^{N}$. Each iteration uses message passing on a graph for the code (determined by $\mathbf{H}$) to update estimates of these probabilities. Upon completion of an iteration, tentative values for these symbols are computed by picking the q-ary value x_j for which the marginal probability estimate $P(d_j|\hat{\mathbf{t}}, \mathbf{H})$ is maximum. If the vector of estimated symbols $\mathbf{x} = [x_1, \ldots x_N]$ satisfies the LDPC parity check condition $\mathbf{Hx} = \mathbf{0}$, the decoding terminates and the message $\mathbf{m}$ is determined as the last K symbols of $\mathbf{x}$. If the maximum number of iterations are exceeded without a valid parity check, a decoder failure occurs. The equations associated with the outer decoder are summarized in the Appendix.

E. Observations/Comments

One can note that there are a couple of benefits from the use of q-ary codes for our application as opposed to binary codes. First, insertion/deletion events introduce uncertainty around the locations where they occur. Using groupings of k binary symbols into a q-ary symbol allow the grouping of these uncertain regions into q-ary symbols and reduces the number of symbols over which the uncertainty is distributed, thereby offering improved performance. This advantage of

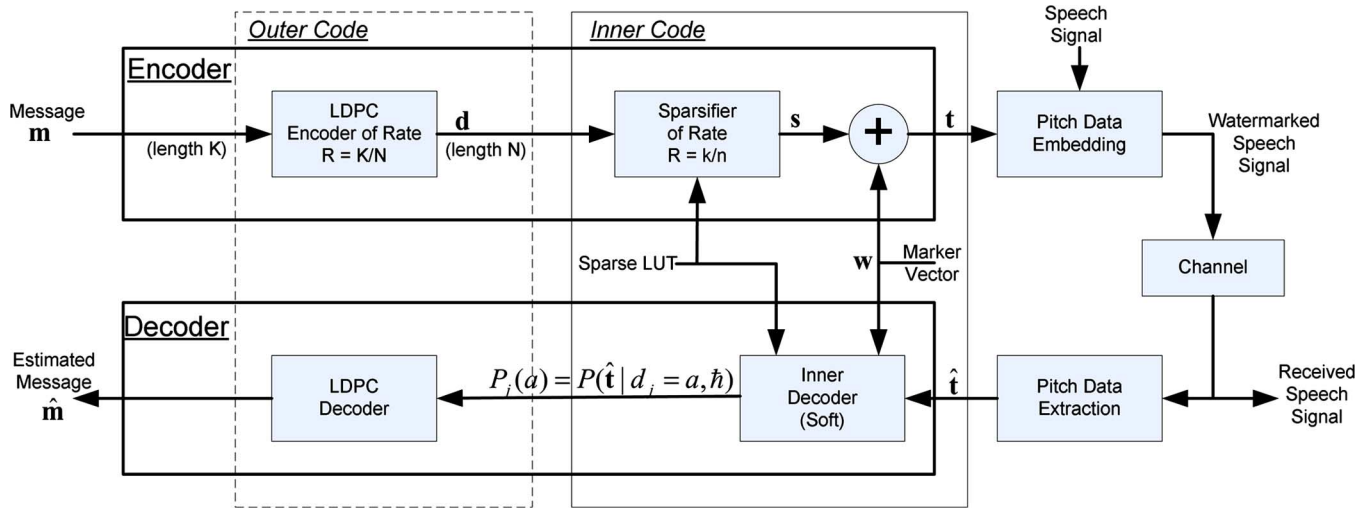

Fig. 8. Pitch-based speech watermark with synchronization.

q-ary codes is similar to the advantage that they offer in correcting burst errors, commonly exploited in Reed–Solomon codes [32]. Second, increasing the value of n to the point in which the entropy per bit does not increase [15] is desirable in order to design a more effective sparsifier and to obtain better estimates of the symbol-by-symbol likelihood probabilities $P_j(a)$. However, increasing n reduces the overall information rate *(Kk)/(Nn)*. Using the q-ary code allows us to compensate for this by increasing k in comparison to a binary code (for which $k = 1$).

V. Pitch Data Embedding in Speech With Synchronization

The block diagram in Fig. 8 depicts the complete system showing both the speech data embedding and the concatenated coding system for recovering from IDS errors. Except for the channel, the individual elements of the system have been previously described. For our system, we consider a nonmalicious operating environment in which the channel can consist of low-bit-rate voice coders. Since these codecs are based on source models for speech, the pitch based-embedding is particularly appropriate—this was the original motivation for the selection of pitch as a parameter for embedding [8].

VI. Implementation

We implemented the proposed system using the PRAAT toolbox [33] for the pitch manipulation operations for analysis and embedding and MATLAB for the inner and outer encoding and decoding processes. The channel operations corresponding to various compressors were performed using separately available speech codecs.

A. Perceptually Tolerable Limits for Pitch-Based Embedding

A psychophysical test was performed with 32 listeners in order to evaluate the discriminability of watermark embedding and an acceptable range of QIM step sizes for embedding. In a paired comparison experiment, a segment of the original speech signal and the watermarked version of the segment were presented to a listener who was then asked to determine which of the two versions, if any, could be identified as modified. The experiment was repeated for QIM step sizes ranging from 10 to 30 Hz. The presentation of the original and watermarked version was randomized for each trial and for each observer, the order in which the different watermarked versions were presented was randomly permuted. As a function of the QIM step size, the fraction of observers who were able to correctly identify the watermarked version is shown in Fig. 9. From the figure, one can see that less than 50% of the listeners were able to correctly identify the watermarked version for QIM step sizes under 15 Hz. QIM step sizes of less than 15 Hz were therefore deemed acceptable for the embedding.

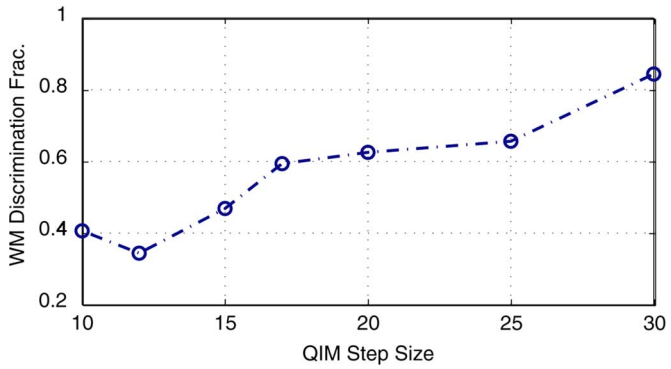

Fig. 9. Watermark discriminability (fraction of listeners correctly identifying watermarked version) as a function of QIM step size.

B. IDS Coding

A $q = 16$-ary LDPC code with rate $1/4$ was utilized as the outer code. The code was obtained by generating an irregular q-ary parity check matrix $\mathbf{H}$ based on Davey and Mackay's constructions [29], [37]. The parity check matrix was designed for a column weight of 2.4 (empirically shown by Davey to be near optimum for $q = 16$ [29]) and rows of the matrix were assigned q-ary symbol values from the heuristically optimized sets made

available by Mackay [37]. A generator matrix for systematic encoding was obtained using Gaussian elimination.

For the sparse LUT, we generated $q = 2^k$ vectors of length n with the lowest possible density of 1's and ordered them sequentially to represent the $q = 2^k$ possible values for a codeword symbol. The marker vector $\mathbf{w}$ was generated using a pseudo-random number generator whose seed served as a shared key between the transmitter and receiver. The mean density of sparse vectors was obtained from the sparse LUT and made available to the inner decoder for the forward–backward passes. The inner decoder used the forward–backward procedure for HMMs to estimate the posterior probabilities and the outer LDPC decoder used iterative probabilistic decoding. A brief summary of these steps is provided in the Appendix.

C. Channel Parameter Estimation

The HMM parameters for the effective IDS channel were estimated using the Baum–Welch re-estimation procedure [30]. The re-estimation equations are also summarized in the Appendix. The method was initialized using parameter values obtained by a sample run of the pitch-based embedding and extraction process that was manually aligned to provide synchronization, thereby allowing empirical estimation of the probabilistic parameters. The corresponding initial parameter values were $P_I = 0.04, P_D = 0.04$, and $P_S = 0.07$. The overall system performance was found to be not unduly sensitive to the channel parameter values. In particular, we demonstrate in the following section that the use of these initial values, without the Baum–Welch re-estimation, causes only a minor degradation in performance.

VII. Experimental Results

In order to evaluate the performance of our proposed speech watermark, we used sample speech files from audio books and various Internet sources [34], [35] and from a database provided by the NSA for the testing of speech compression algorithms [20], [21]. The sample speech files consist of continuous sentences read by male/female speakers and sampled at 16 kHz with 16 b/sample, which corresponds to a data rate of 256 kb/s.

In order to test the system, random message vectors of $q = 16$-ary message symbols were generated. These were arranged in blocks of $K = 25$ and encoded as LDPC code vectors of length $N = 100$. The length of the sparse vectors was chosen as $n = 10$; resulting in an overall coding rate of 0.10. The binary data obtained from the sparsifier was embedded into the speech signal by QIM of the average pitch over $J = 5$ windows of 10 ms each using a quantization step Δ that ranged between 6–15 Hz (the impact of the embedding was perceptually tolerable over this range of step-sizes as indicated by the results of the psychophysical tests in the preceding section).

The communication channel was variously chosen as follows.

1) None (i.e., the speech waveform was unchanged between embedding and extraction).
2) Global System for Mobile Communications coder, version 06.10 (GSM-06.10) at 13 kb/s. This codec is commonly used in today's second-generation (2G) cellular networks that comply with GSM standard [20].

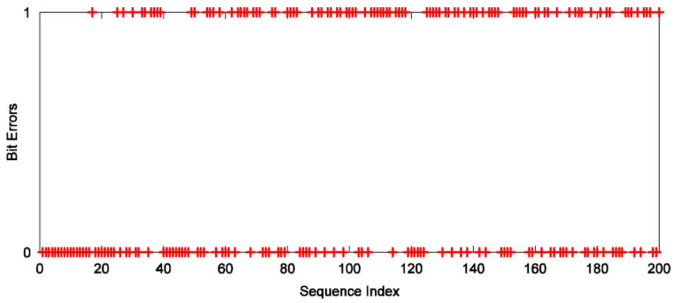

Fig. 10. Differences between inserted and extracted bits in the absence of synchronization.

3) Adaptive multirate coder (AMR) at 5.1 kb/s. This codec has been standardized for third-generation cellular networks (3GPP standard) [21].

A. Sample Run Results

We first present results for a sample run of one block through the system. The purpose of these results is to illustrate the ability of the method to regain synchronization despite synchronization loss for the underlying pitch-based embedding. Monte Carlo results that illustrate the statistical behavior of the technique for different parameter values are deferred to the next section. A QIM step size of $\Delta = 15$ Hz is used throughout this subsection.

Fig. 10 illustrates the differences between inserted bits $\mathbf{t}$ in the speech waveform and extracted bits $\hat{\mathbf{t}}$ where the status of the first 200 of 1000 embedded bits are indicated as "+" symbols at 0 along the y axis and indicate locations where the embedded and extracted bits match and those at 1 indicate locations where they differ. As can be seen in the initial segment, there is reasonable agreement between the symbols but beyond that, the agreement between the bits is no better than random. This is primarily due to a loss of synchronization between the embedded and extracted bitstreams. Once synchronization is lost, independent bits embedded at different locations are, in fact, being compared, which match with probability half.

Table I shows a comparison for a typical successful run across the different "channels" that we enumerated earlier. The columns in the table list the initial error count, the number of errors after the decoding, and the computation requirements in terms of the number of LDPC iterations, as well as the computation times spent by our (unoptimized) decoder in the inner and outer coders for the concatenated synchronization code. From Table I, we can note that in all cases, the loss of synchronization initially produces a rather high apparent bit-error rate but the proposed method is able to recover synchronization and correct errors to correctly recover the embedded data. The decoding consumes most of the computation time in the experiments. The computation times for the inner and outer decoder are listed in Table I. The numbers in the table illustrate the fact that the inner decoder has a rather high computational burden[5] (which is expected given the nonlinearity of the inner code)

[5]Our MATLAB-based implementation is quite inefficient for the inherently serial computations required in this process and it is possible that the process could be considerably improved with an alternate implementation.

TABLE I
COMPARISON OF ERROR-CORRECTION PERFORMANCE AND DECODER EXECUTION TIMES OVER DIFFERENT "CHANNELS"

Channel (Compression)	Bit Errors w/o Synchronization	Errors after Synchronization	LDPC Decoder Iterations	Inner Decoder Execution Time	LDPC Decoder Execution Time
None	518	0	5	177.8 s	3.7 s
AMR 5.1 kbps	492	0	7	188.5 s	3.9 s
GSM 13 kbps	472	0	7	181.4 s	4.6 s

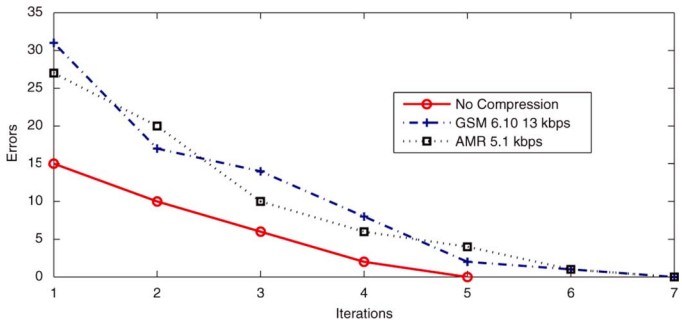

Fig. 11. LDPC iteration count versus the number of errors for the outer decoder.

and that this constitutes the major computational load for the proposed technique.

We also examined the behavior of the iterative decoding for the outer LDPC decoder for the experimental runs of Table I. The results are shown in Fig. 11 where the number of symbol errors as a function of the LDPC iteration count is shown for each case. From the results, we can see in the absence of compression the number of errors rapidly falls and correct decoding is achieved in less than seven iterations in the example presented.

B. Monte Carlo Simulation Results

Next, we present Monte Carlo (MC) simulation results for more extensive experiments, again, using the three previously cited speech compression channels and an additive white Gaussian noise (AWGN) channel. For this purpose, the sample speech segments (containing female and male speech from diverse sources) were concatenated to produce a speech signal of approximately 2 h in length. Four runs were performed over the resulting signal for each channel with different realizations of the marker vector $\mathbf{w}$, producing a total 200 simulation runs for each channel.[6] The results from these experiments are summarized by determining, for each choice of experimental settings, the percentage of simulation runs for which the embedded data was successfully recovered.

Fig. 12 illustrates the impact of varying the QIM quantizer step-size Δ for the three compression channels considered. In general, an increase in the QIM step size also increases the embedding distortion. Though, as discussed in Section VI, the embedding distortion is almost imperceptible for QIM step sizes less than the 15 Hz maximum that we consider in this investigation. Results are provided for two channel parameter estimation scenarios. Fig. 12(a) shows the results obtained using the static set of initial channel parameters indicated in Section VI and Fig. 12(b) shows the results obtained when the channel parameters are re-estimated using the Baum–Welch algorithm. The results obtained with channel parameter estimation offer a modest improvement over the static parameters for higher QIM step sizes and perform slightly worse for the lower QIM step sizes because the channel degradation also degrades the estimates obtained. Apart from these minor differences, the results in both figures follow common trends: As can be expected, increasing values of Δ provide increased robustness and thereby a higher success percentage. For a quantizer step size of $\Delta = 15$ Hz, data were successfully recovered in more than 95% of the simulations for all three channels. Observe that these three channels present varying degrees of difficulty for watermark recovery. The only source of errors for the channel corresponding to no compression are the differences in estimated features between the embedder and the receiver caused by the change in the signal from the embedding process itself. These become progressively infrequent as the QIM step size Δ is increased and for $\Delta = 15$ Hz, the data are successfully recovered. Both the GSM and AMR channels introduce very significant distortions,[7] causing additional errors that the watermark system must overcome. The extremely low-bit-rate AMR channel is the most challenging.

The performance of the watermark over an AWGN channel is shown in Fig. 13 for QIM step sizes Δ of 10, 12, and 15 Hz. The abscissa of the plot indicates the AWGN signal-to-noise power ratio (SNR) and the ordinate indicates the percentage of simulation runs for which data were successfully recovered. In informal experiments, an AWGN SNR below 27 dB produced a clearly audible distortion and around 20 dB resulted in objectionable audio quality. Once again, two channel parameter estimation scenarios are considered. Fig. 13(a) shows the results obtained using the static set of initial channel parameters indicated in Section VI and Fig. 13(b) shows the results obtained when the channel parameters are re-estimated using the Baum–Welch algorithm. In this case, a clear improvement can be seen with the channel parameter estimation though the static parameters also offering reasonable performance. From the graph in Fig. 13(b), one can see that over the range of perceptually acceptable AWGN attacks, the method performs quite well with only minor degradation in comparison with the noiseless channel case, which was included in Fig. 12(b).

C. Spread-Spectrum Watermark Comparison

In order to illustrate the challenge posed by speech watermarking (for low-rate compression), we also evaluated an oblivious spread-spectrum watermarking method that is conceptually similar to that proposed by Cheng [17]. Our scheme is de-

[6]The time for the simulations with our experimental code and the manual nature of the interaction with PRAAT and the speech codecs did not readily allow larger Monte Carlo experiments.

[7]GSM and AMR coding resulted in SNR values of approximately 3.9 dB and 2.0 dB, respectively.

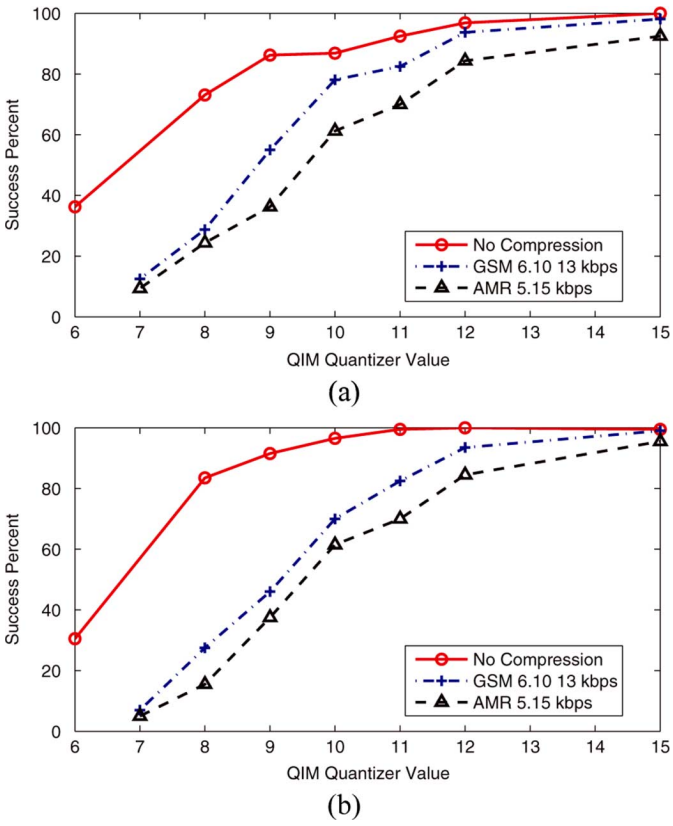

Fig. 12. Monte Carlo simulation results over speech compression channels. (a) Without channel parameter estimation. (b) With channel parameter estimation.

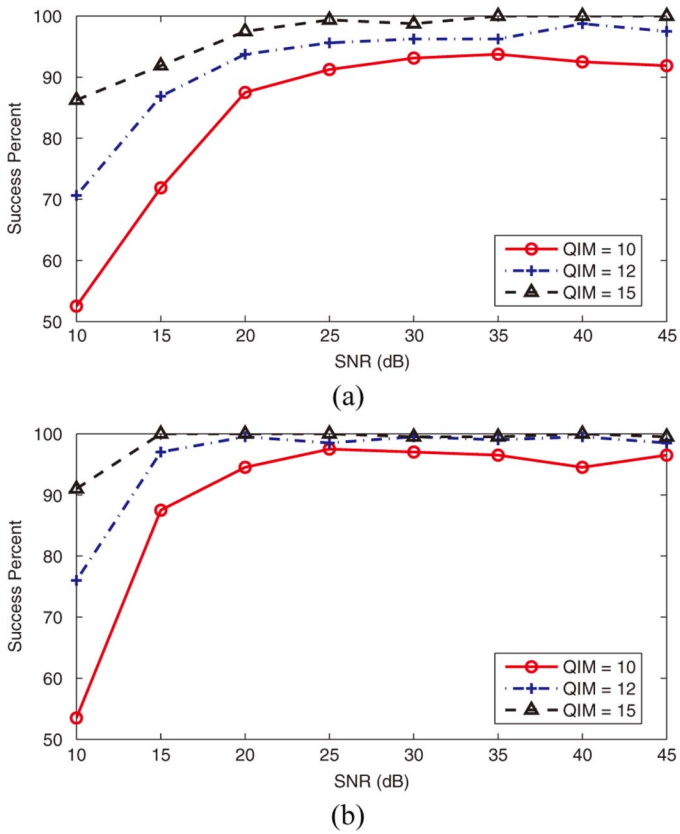

Fig. 13. Results from Monte Carlo simulations over an AWGN channel. (a) Without channel parameter estimation. (b) With channel parameter estimation.

picted in Fig. 14, where the normalized correlation value forms the output. A threshold detector converts the value to a positive/negative detection response. By varying the threshold value and conducting MC experiments, we can obtain receiver operating curves (ROCs) that plot the estimated detection and false alarm probabilities against each other.

For our experiments, we generate 2000 sample pseudo-random sequences as the spread-spectrum "watermark" vectors (this roughly matches the number of spread-spectrum watermarks to our uncoded data embedding rate). The random sequences were scaled by a factor α and added to the speech in order to embed the watermark, where α was chosen as the smallest value such that the resulting embedding was barely audible (0.07% of the signal dynamic range in our case). We performed 2500 simulations over our concatenated speech signal in order to obtain the MC results.

In the absence of any compression, the ROC curve is a perfect inverted L. The results for the two compression channels are shown in Fig. 15. We find moderate success for the GSM compression channel but for AMR compression, the ROC curve is very close to a straight diagonal line, which is the worst possible performance and matches the performance of a random detector that does not use the input signal at all.

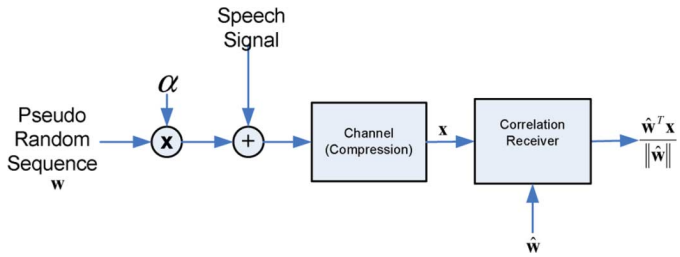

Fig. 14. Alternative spread-spectrum speech watermark.

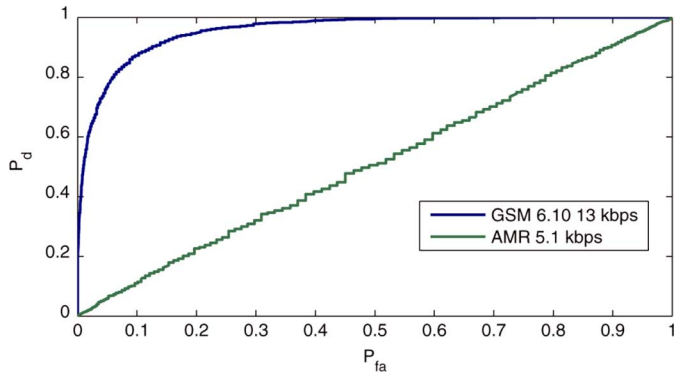

Fig. 15. ROC for a spread-spectrum speech watermark over GSM and AMR channels.

VIII. Conclusion and Discussion

This paper introduced a new paradigm for synchronization in multimedia watermarking that combines feature-based embedding with error-correction codes capable of correcting insertion, deletion, and substitution errors. We presented a speech watermark as an instantiation of the paradigm. Low-bit-rate

encoding methods motivated the feature-based embedding in this application and the 1-D nature of the signal offered suitable simplification for realization of a practical watermarking scheme. The experimental results for our speech watermark illustrate that the framework allows recovery of embedded data under common scenarios where some mismatches in the features detected at the transmitting and receiving ends are inevitable. The speech watermark is robust to low-bit-rate speech coders that are commonly used in speech communication applications. Since these encoders have been debilitating for common watermarking methods that presume synchronization, the work presented here represents an advance in speech watermarking in addition to illustrating the utility of the proposed framework.

This paper is a first step offering a promising new approach for jointly addressing synchronization and error correction in feature-based multimedia data embedding. The framework proposed here was demonstrated in a speech-watermark suitable for operation over low-bit-rate encoding channels, which, although nonmalicious, pose very significant desynchronization challenges. The positive results obtained in this difficult scenario are rather encouraging but several issues must be addressed in order to apply the methodology in broader feature-based watermarking scenarios. Specifically, fundamental advances in the error-correction coding methodology are required to provide meaningful extensions for 2-D and 3-D data (e.g., images and video).

Irrespective of signal dimensionality, some further explorations are also of interest, particularly for addressing robust embedding scenarios as opposed to the semifragile application considered in our work. In this regard, our embedding method based on pitch modification is not robust against time-axis scaling attacks (that are the equivalent to valumetric scaling attacks for QIM methods) and alternate (local) methods of embedding would therefore be of interest. Additional work is also required on the security of the scheme. A potential security weakness of feature-based embedding methods is that an adversary may also attempt to detect and alter significant features in an attempt to defeat the watermark [41]. An additional security weakness arises in our implementation due to the fact that the QIM embedding presented does not employ any dithering (synchronization would be a prerequisite in order to use dithering). A malicious attacker may attempt to estimate quantizer levels and deliberately disrupt the embedded signal. We note that the marker sequence is partly analogous to a dither signal. An interesting direction for further investigation therefore would be to explore whether a "soft marker signal," which is not constrained to be binary, could be utilized to serve the simultaneous purposes of dithering and synchronization.

APPENDIX

The IDS correction code is based on the work of Davey and MacKay [15], [29] though the specific codes and parameters were selected in view of our watermarking application. This Appendix provides a compendium of the elements that were not described in the main text of our paper: The HMM-based inner decoder, the Baum–Welch procedure for the re-estimation of the channel model parameters, and the outer LDPC code.

A. Hidden Markov Model-Based Inner Decoder

The inner decoder computes the symbol-by-symbol likelihood probability mass functions $\{P_j(a)\}_{a\in \mathrm{GF}(q)}$ for $j = 1,\dots N$ from the extracted data $\hat{\mathbf{t}}$ at the watermark receiver. In this process, it utilizes the channel model and the model parameters. We assume that the channel labeled IDS channel in Fig. 6 has parameters $\hbar' := (P_I, P_T, P_D, P_S)$ that correspond, respectively, to insertion, transmission, deletion, and substitution probabilities. If we consider the channel labeled IDS channel' in Fig. 6 with input as the marker vector $\mathbf{w}$ and output as the extracted data $\hat{\mathbf{t}}$ at the receiver, the insertion, transmission, and deletion probabilities for this channel are the same viz, P_I, P_T, P_D, respectively, whereas due to the additional substitutions introduced by the message data, the probability of substitution changes to $P_f = P_S(1-f) + f(1-P_S)$, where f denotes the mean density of the sparse LUT. For practical implementation, we assume that the maximum number of consecutive insertions allowed in the model of Fig. 7 is limited to I_m. After I_m consecutive insertions, the channel does not allow any additional insertions and undergoes a deletion with probability P_D or transmits the current bit with probability $\bar{P}_T = (1-P_D)$. Next, we define the drift ψ_i at position i as the number of insertions minus the number of deletions encountered before the channel enters state i and forward probabilities $F_i(\kappa) := P(\hat{t}_1,\dots,\hat{t}_{i-1+\kappa}, \psi_i = \kappa|\hbar)$ and the backward probabilities $B_i(\kappa) := P(\hat{t}_{i+\kappa},\dots|\psi_i = \kappa, \hbar)$, for emission of the leading and trailing ends of the received sequence (under indicated conditionings). These are readily calculated using the HMM forward–backward recursions

$$F_i(\kappa) := \sum_{\eta=\kappa-I_m}^{\kappa+1} F_{i-1}(\eta)\mathrm{T}^{i-1}_{\eta\kappa}(\hat{t}_{i-1+\eta},\dots,\hat{t}_{i-1+\kappa})$$

$$B_i(\kappa) := \sum_{\eta=\kappa-1}^{\kappa+I_m} B_{i+1}(\eta)\mathrm{T}^{i}_{\kappa\eta}(\hat{t}_{i+\kappa},\dots,\hat{t}_{i+\eta})$$

where $\mathrm{T}^i_{\eta\kappa}(\boldsymbol{\sigma})$ denotes the conditional probability, conditioned on $\psi_{(i-1)} = \eta$, that $\psi_i = \kappa$ and the binary sequence $\boldsymbol{\sigma}$ of length $(\kappa-\eta)+1$ is emitted by the channel from the time the channel enters state i to the time the channel enters state $(i+1)$. The conditional probability $\mathrm{T}^i_{\eta\kappa}(\boldsymbol{\sigma})$ can be expressed in the form $\mathrm{T}^i_{\eta\kappa}(\boldsymbol{\sigma}) = \alpha_{\eta\kappa} + \beta_{\eta\kappa}\zeta^i_{\eta\kappa}(\boldsymbol{\sigma})$ where

$$\alpha_{\eta\kappa} = \begin{cases} \frac{P_I^{\kappa-\eta+1}P_D}{2^{\kappa-\eta+1}} & : -1 :\leq \kappa-\eta < I_m \\ 0 & : (\kappa-\eta) < -1, (\kappa-\eta) \geq I_m \end{cases}$$

$$\beta_{\eta\kappa} = \begin{cases} \frac{P_I^{\kappa-\eta}P_T}{2^{\kappa-\eta}} & : 0 :\leq \kappa-\eta < I_m \\ \frac{P_I^{I_m}\bar{P}_T}{2^{\kappa-\eta}} & : (\kappa-\eta) = I_m \\ 0 & : (\kappa-\eta) \leq -1, (\kappa-\eta) > I_m \end{cases}$$

$$\zeta^i_{\eta\kappa}(\boldsymbol{\sigma}) = \begin{cases} 1-P_f & : \sigma_{\kappa-\eta+1} = w_i \\ P_f & : \sigma_{\kappa-\eta+1} = (w_i \oplus 1). \end{cases}$$

Note that $\boldsymbol{\sigma}$ is a binary string of length $(\kappa-\eta+1)$, so that $\sigma_{\kappa-\eta+1}$ is the last element of $\boldsymbol{\sigma}$ (which would be the transmitted bit, if indeed a transmission occurs).

Upon completion of the forward–backward pass, the symbol-by-symbol likelihood probabilities for q-ary symbols

at the input of the sparsifier can be computed by combining the results from the bitwise forward–backward pass as[8]

$$P_j(a) = P(\hat{\mathbf{t}}|d_j = a, \hbar) \approx \sum_u \sum_v F_{j_-}(u) P(\mathbf{r}^0, \psi_{j_-} = u|\psi_{j_+} = v, d_j = a, \hbar) B_{j_+}(v)$$

where $j_- = jn, j_+ = (j+1)n, u, v$ represents the (postulated) drift at the start of the jth and $(j+1)$th symbols, respectively; $\mathbf{r}^0$ represents the bits emitted by the channel between these positions (i.e., $\mathbf{r}^0 = [\hat{t}_u, \ldots, \hat{t}_v]$), and the probability term $P(\mathbf{r}^0, \psi_{j_-} = u|\psi_{j_+} = v, d_j = a, \hbar)$ is interpreted readily from the notation. This latter probability can be efficiently computed by defining a forward probability $F_i'(\kappa) := P(\hat{t}_u, \ldots, \hat{t}_{u+i-1+\kappa}, \psi_{j_-} = u|\psi_i = \kappa, d_j = a, \hbar)$ and noting that $P(\mathbf{r}^0, \psi_{j_-} = u|\psi_{j_+} = v, d_j = a, \hbar) = F_i'(v)$, which is obtained using an additional forward pass

$$F_i'(\kappa) := \sum_{\eta=\kappa-I}^{\kappa+1} F_{i-1}'(\eta) T_{\eta\kappa}^{0,i-1} \times (\hat{t}_{i-1+\eta}, \ldots, \hat{t}_{i-1+\kappa})$$

where $T_{\eta\kappa}^{0,i}(\ )$ is defined as before with P_f replaced by P_s in the expressions.

B. Baum–Welch Re-Estimation Equations

The HMM parameters representative of the channel conditions can be estimated using the iterative Baum–Welch re-estimation procedure [30]. In terms of the forward and backward probabilities, the re-estimation equations are shown at the top of the next page, where $\hat{V}$ denotes the estimate for the parameter V and $\underline{D} = \sum_{\forall i} \sum_{\forall \eta} F_i(\eta) B_i(\eta)$.

C. Outer Q-Ary LDPC Code

Technical details for LDPC encoding/decoding may be found in relevant references on the topic [27]–[29], [31], [36], [38], [39]. A brief summary is provided here for completeness.

The q-ary LDPC code is specified by a $(N-K) \times N$ sparse parity check matrix $\mathbf{H}$ with nonzero entries in GF (q), having rank $M := N - K$. The outer encoder (Figs. 6 and 8) encodes blocks of Kq-ary symbols into corresponding codewords with Nq-ary symbols each. Codewords are N vectors, satisfying the parity check constraint $\mathbf{Hx} = \mathbf{0}, \mathbf{x} \in \mathrm{GF}^{\mathrm{N}}(q)$. An $N \times K$ generator matrix for the code in systematic form, computed from $\mathbf{H}$ forms the encoder [27], [28], [31], [39]. Codewords are obtained by multiplying message vectors in $\mathrm{GF}^K(q)$ by the generator matrix and include the message $\mathbf{m}$ as the last k symbols.

The decoder takes, as inputs, symbol-by-symbol likelihood probabilities $\{P_j(a)\}_{a \in \mathrm{GF}(q)}$ for $j = 1, \ldots N$ and estimates marginal (pseudo) posterior probabilities $Q_j^a = P(d_j = a|\hat{\mathbf{t}}, \mathbf{H})$. The term Q_j^a represents the probability that the jth received symbol is $d_j \in \mathrm{GF}(q)$ conditioned on the events that at the transmitting end, the data were encoded using the parity check matrix $\mathbf{H}$ and $\hat{\mathbf{t}}$ that is received from the IDS channel. This is accomplished by the standard soft in, soft out iterative decoding algorithm for q-ary LDPC codes summarized in Fig. 16.

[8]The two-step process utilizing a bitwise forward–backward pass followed by a forward pass for each symbol represents an approximation that ignores correlations introduced by the sparsifier except for the specific symbol under consideration.

1. Initialization:
   $\Omega \leftarrow 0$; $Q_{ij}^a \leftarrow P(d_j), \forall d_j \in GF(q)$
2. Horizontal Pass: $\forall i \ni H_{ij} \neq 0$, $\forall j \ni H_{ij} \neq 0$, and $\forall a \in GF(q)$ assign
   $$R_{ij}^a \leftarrow \sum_{u_l} \chi_{ij}\left(a, \{u_l : l \in L_{ij}\}\right) \prod_{l \in L_{ij}} Q_{il}^{u_l}$$
   where
   $$\chi_{ij}\left(a, \{u_l : l \in \bar{L}_{ij}\}\right) = \begin{cases} 1 & \text{if } H_{ij}a + \sum_{l \in L_{ij}} H_{il}u_l = 0 \\ 0 & \text{otherwise} \end{cases}$$
   and $\bar{L}_{ij} = \{l : H_{il} \neq 0, l \neq j\}$.
   Note that $\chi_{ij}\left(a, \{u_l : l \in \bar{L}_{ij}\}\right)$ selects the values for symbols other than j that satisfy the parity check condition specified by the i^{th} row of $\mathbf{H}$ when the j^{th} symbol is equal to a. The computations of $\{R_{ij}^a\}_{a \in GF(q)}$ can be performed efficiently in parallel using a fast Fourier transform (FFT) [36], specifically for the typical case of $q=2^k$ this is a k-dimensional two-point FFT.
3. Vertical Pass: $\forall i \ni H_{ij} \neq 0$, $\forall j \ni H_{ij} \neq 0$, and $\forall a \in GF(q)$ update
   $$Q_{ij}^a \leftarrow \alpha_{ij} P_j(a) \prod_{l \in \hat{L}_{ij}} R_{lj}^a$$
   where
   $$\hat{L}_{ij} = \{l : H_{lj} \neq 0, l \neq i\},$$
   and α_{ij} is a scaling factor determined to ensure that $\sum_a Q_{ij}^a = 1$. This computation can be performed using a forward-backward algorithm.
4. Pseudo-posterior probability computation: $\forall 1 \leq j \leq N$, and $\forall a \in GF(q)$ compute
   $$Q_j^a \leftarrow \alpha_j P_j(a) \prod_{l \in L_j} R_{lj}^a$$
   where $L_j = \{l : H_{lj} \neq 0\}$ and α_j is a scaling factor determined to ensure that $\sum_a Q_j^a = 1$.
5. Tentative Decoding: $\forall 1 \leq j \leq N$ assign
   $$\hat{x}_j = \arg\max_a (Q_j^a)$$
6. Convergence Check: If $\mathbf{H}\hat{\mathbf{x}} = \mathbf{0}$ decoding is complete: assign the last K symbols of $\hat{\mathbf{x}}$ to $\hat{\mathbf{m}}$ and terminate; else increment iteration count $\Omega \leftarrow \Omega+1$, if $\Omega > \Omega_{max}$ declare decoder failure and terminate, else go to step 2.

Fig. 16. Outer q-ary LDPC decoding algorithm.

$$\hat{P}_D = \frac{\sum_i \sum_\eta F_i(\eta) P_D B_{i+1}(\eta - 1)}{\underline{D}}$$

$$\hat{P}_T = \frac{\sum_i \sum_\eta F_i(\eta) P_T [P_f(1 - \delta(\hat{t}_{i+\eta}, \mathbf{w}_i)) + (1 - P_f)\delta(\hat{t}_{i+\eta}, \mathbf{w}_i)] B_{i+1}(\eta)}{\underline{D}}$$

$$\hat{P}_f = \frac{\sum_i \sum_\eta F_i(\eta) P_T P_f (1 - \delta(\hat{t}_{i+\eta}, \mathbf{w}_i)) B_{i+1}(\eta)}{\hat{P}_T \underline{D}}$$

$$\hat{P}_I = \frac{\sum_i \sum_\eta F_i(\eta) (\frac{P_I}{2}) B_i(\eta + 1)}{\underline{D}}$$

ACKNOWLEDGMENT

The authors would like to express their gratitude to M. Celik for help with the pitch-based watermark embedding and to M. C. Davey for assistance with the insertion–deletion codes. The authors would also like to thank the anonymous reviewers for their comments which have helped to significantly improve the manuscript.

REFERENCES

[1] P. Loo and N. G. Kingsbury, "Motion estimation based registration of geometrically distorted images for watermark recovery," in *Proc. SPIE: Security Watermarking of Multimedia Contents III*, Jan. 2001, vol. 4314, pp. 601–617.

[2] G. Caner, A. M. Tekalp, G. Sharma, and W. Heinzelman, "Local image registration by adaptive filtering," *IEEE Trans. Image Process.*, vol. 15, no. 10, pp. 3053–3065, Oct. 2006.

[3] V. Licks and R. Jordan, "Geometric attacks on image watermarking systems," *IEEE Multimedia*, vol. 12, no. 3, pp. 68–78, Jul.–Sep. 2005.

[4] G. Sharma and D. J. Coumou, "Watermark synchronization: Perspectives and a new paradigm," in *Proc. 40th Annu. Conf. Info. Sciences and Syst.*, Princeton, NJ, Mar. 22–24, 2006, pp. 1182–1187.

[5] J. K. O. Ruanaidh and T. Pun, "Rotation, scale and translation invariant spread spectrum digital image watermarking," *Signal Process.*, vol. 66, no. 5, pp. 303–317, May 1998.

[6] R. Caldelli, M. Barni, F. Bartolini, and A. Piva, "Geometric-invariant robust watermarking through constellation matching in the frequency domain," presented at the IEEE Int. Conf. Image Proces., Sep. 2000.

[7] M. Alghoniemy and A. Tewfik, "Image watermarking by moment invariants," presented at the IEEE Int. Conf. Image Process., Sep. 2000.

[8] M. Celik, G. Sharma, and A. M. Tekalp, "Pitch and duration modification for speech watermarking," in *Proc. IEEE Int. Conf. Acoustics Speech Sig. Process.*, Mar. 2005, pp. 17–20.

[9] P. Bas, J.-M. Chassery, and B. Macq, "Geometrically invariant watermarking using feature points," *IEEE Trans. Image Process.*, vol. 11, no. 9, pp. 1014–1028, Sep. 2002.

[10] C. W. Honsinger, P. W. Jones, M. Rabbani, and J. C. Stoffel, "Lossless recovery of an original image containing embedded data," U.S. Patent 6 278 791, Aug. 21, 2001.

[11] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proc. IEEE*, vol. 87, no. 7, pp. 1079–1107, Jul. 1999.

[12] G. Csurka, F. Deguillaume, J. J. K. O'Ruanaidh, and T. Pun, "A Bayesian approach to affine transformation resistant image and video watermarking," in *Proc. 3rd Int. Information Hiding Workshop*, 1999, pp. 315–330.

[13] M. Kutter, S. K. Bhattacharjee, and T. Ebrahimi, "Towards second generation watermarking schemes," in *Proc. IEEE ICIP*, Oct. 1999, vol. 1, pp. 320–323.

[14] M. U. Celik, E. Saber, G. Sharma, and A. M. Tekalp, "Analysis of feature-based geometry invariant watermarking," *Proc. SPIE: Security and Watermarking of Multimedia Contents III*, vol. 4314, pp. 261–268, Jan. 2001.

[15] M. C. Davey and D. J. C. Mackay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 687–698, Feb. 2001.

[16] L. R. Bahl and F. Jelinek, "Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition," *IEEE Trans. Inf. Theory*, vol. IT-21, no. 4, pp. 404–411, Jul. 1975.

[17] Q. Cheng and J. Sorensen, "Spread spectrum signaling for speech watermarking," in *Proc. IEEE Int. Conf. Acoustics Speech and Sig. Process.*, May 2001, vol. 3, pp. 1337–1340.

[18] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. Englewood Cliffs, NJ: Prentice-Hall, 1978.

[19] Uninett AS, Jan. 14, 2004. [Online]. Available: http://www.uninett.no/voip/codec.html.

[20] K. Hellwig, Full Rate Speech Transcoding. [Online]. Available: http://www.3gpp.org/ftp/Specs/archive/06_series/06.10/ 3GPP TS 06.10.

[21] S. Bruhn, AMR Speech Codec General Description. [Online]. Available: http://www.3gpp.org/ftp/Specs/archive/26_series/26.071 3GPP TS 26.071.

[22] M. Mouly and M.-B. Pautet, *The GSM System for Mobile Communications*. Palaiseau, France: Telecom Publishing, 1992.

[23] C. P. Wu and C.-C. J. Kuo, "Comparison of two speech content authentication approaches," *Proc. SPIE: Security and Watermarking of Multimedia Contents IV*, vol. 4675, pp. 158–169, 2002.

[24] B. Chen and G. W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1423–1443, May 2001.

[25] E. Molines and F. Charpentier, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diaphones," *Speech Commun.*, pp. 453–467, 1990.

[26] B. Sklar, *Digital Communications: Fundamentals and Applications*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2001.

[27] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165–167, Jun. 1998.

[28] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(q)," in *Proc. IEEE Inf. Theory Workshop*, Jun. 1998, pp. 70–71.

[29] M. C. Davey, "Error correction using low density parity-check codes," Ph.D. dissertation, Inference Group, Cavendish Lab., Univ. Cambridge, Cambridge, U.K., Dec. 1999.

[30] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[31] D. J. C. MacKay, "Good error correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[32] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewoods Cliffs, NJ: Prentice-Hall, 1983.

[33] P. Boersma and D. Weenik, Praat: Doing phonetics by computer. [Online]. Available: http://www.fon.hum.uva.nl/praat.

[34] Ohio State Univ., Speech Corpus. [Online]. Available: http://buckeyecorpus.osu.edu.

[35] Open Speech Repository. [Online]. Available: http://www.voiptroubleshooter.com/open_speech.

[36] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[37] D. J. C. MacKay, Optimizing sparse graph codes over GF(q) [Online]. Available: http://www.cs.toronto.edu/~mackay/gfqoptimize.pdf.

[38] R. G. Gallager, *Low Density Parity Check Codes*. Cambridge, MA: MIT Press, 1963.

[39] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Hoboken, NJ: Wiley, Jun. 2005.
[40] D. Coumou and G. Sharma, "Watermark synchronization for feature-based embedding: Application to speech," in *Proc. IEEE Int. Conf. Multimedia Expo.*, Toronto, ON, Canada, Jul. 9–12, 2006, pp. 849–852.
[41] P. Bas and A. L. Guerro, "Several considerations on the security of a feature-based synchronization scheme for digital image watermarking," presented at the First Wavila Challenge, Barcelona, Spain, May 2005.

David J. Coumou (M'92) received the B.Sc. and M.Sc. degrees in electrical engineering from the Rochester Institute of Technology, Rochester, NY, in 1992 and 2001, respectively, and is currently pursuing the Ph.D. degree at the University of Rochester, Rochester, NY.

He is a Technical Manager with the ENI Products Division of MKS Instruments, Inc., Rochester, where he is responsible for the development of RF metrology and control. His research interests include multirate, adaptive, and statistical signal-processing, source and channel coding, digital communications, and watermarking. He holds six issued U.S. Patents and has six additional patent applications that are under review by the U.S. Patent office.

Mr. Coumou has been a Chapter Officer for the Rochester chapter of the IEEE Signal Processing Society since 2003 and is currently Treasurer. From 2004 to 2007, he was Co-Chair of the annual Western New York Image Processing Workshop in Rochester. He is listed in *Who's Who* and is a member of SPIE.

Gaurav Sharma (SM'00) received the B.E. degree in electronics and communication engineering from the Indian Institute of Technology Roorkee (formerly the University of Roorkee), Roorkee, India, in 1990; the M.E. degree in electrical communication engineering from the Indian Institute of Science, Bangalore, India, in 1992; and the M.S. degree in applied mathematics and Ph.D. degree in electrical and computer engineering from North Carolina State University (NCSU), Raleigh, in 1995 and 1996, respectively.

From 1992 through 1996, he was a Research Assistant with the Center for Advanced Computing and Communications in the Electrical and Computer Engineering Department at NCSU. From 1996 through 2003, he was with Xerox Research and Technology, Webster, NY, initially as a member of the research staff and subsequently becoming Principal Scientist. Since 2003, he has been an Associate Professor in the Department of Electrical and Computer Engineering and in the Department of Biostatistics and Computational Biology at the University of Rochester, Rochester, NY. His research interests include multimedia security and watermarking, color science and imaging, genomic signal processing, and image processing for visual sensor networks. He is the editor of the *Color Imaging Handbook* (CRC, 2003).

Dr. Sharma is a member of Sigma Xi, Phi Kappa Phi, Pi Mu Epsilon, IS&T, and the IEEE signal processing and communications societies. He was the 2007 Chair for the Rochester section of the IEEE and served as the 2003 Chair for the Rochester chapter of the IEEE Signal Processing Society. He is Vice-Chair for the IEEE Signal Processing Society's Image and multidimensional signal processing (IMDSP) technical committee and is a member of the IEEE Standing Committee on Industry DSP. He is an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and the *Journal of Electronic Imaging*.