Cascaded RLS–LMS Prediction in MPEG-4 Lossless Audio Coding

Haibin Huang, *Member, IEEE*, Pasi Fränti, Dongyan Huang, *Senior Member, IEEE*, and Susanto Rahardja, *Senior Member, IEEE*

***Abstract*—This paper describes the cascaded recursive least square–least mean square (RLS–LMS) prediction, which is part of the recently published MPEG-4 Audio Lossless Coding international standard. The predictor consists of cascaded stages of simple linear predictors, with the prediction error at the output of one stage passed to the next stage as the input signal. A linear combiner adds up the intermediate estimates at the output of each prediction stage to give a final estimate of the RLS–LMS predictor. In the RLS–LMS predictor, the first prediction stage is a simple first-order predictor with a fixed coefficient value 1. The second prediction stage uses the recursive least square algorithm to adaptively update the predictor coefficients. The subsequent prediction stages use the normalized least mean square algorithm to update the predictor coefficients. The coefficients of the linear combiner are then updated using the sign–sign least mean square algorithm. For stereo audio signals, the RLS–LMS predictor uses both intrachannel prediction and interchannel prediction, which results in a 3% improvement in compression ratio over using only the intrachannel prediction. Through extensive tests, the MPEG-4 Audio Lossless coder using the RLS–LMS predictor has demonstrated a compression ratio that is on par with the best lossless audio coders in the field. In this paper, the structure of the RLS–LMS predictor is described in detail, and the optimal predictor configuration is studied through various experiments.**

***Index Terms*—Adaptive prediction, least mean square, lossless audio, MPEG-4 Audio Lossless Coding, recursive least square.**

I. Introduction

LOSSLESS audio coding, as the name suggests, converts a digital audio signal from raw pulse code modulation (PCM) format into a compressed format with a smaller file size. The original audio signal can be perfectly reconstructed from the compressed file. Coupled with continually decreasing storage costs and the increasing growth of processor power, lossless audio compression started to gain popularity with the wide and rapid spread of broadband networks. Applications of lossless audio compression include digital music archival, network music downloading and broadcasting, personal music sharing, and mobile entertainment.

Over the years, various lossless audio compressors were developed by individuals, interested research groups, and commercial entities. For example, APE (by Monkey's Audio) [1] and FLAC (Free Lossless Audio Codec) [2] are popular ones in music file sharing over the internet. OptimFrog [3] and LA (Lossless Audio) [4] provide high compression ratios. Organizations like Apple, Microsoft, and Real Networks also developed their own lossless audio coders. Unfortunately, all these are proprietary coders, which are lacking in large-scale industrial adoption. In response to the industrial demand for a standardized lossless audio coding scheme, the *Motion Pictures Expert Group* (MPEG) issued a Call for Proposal in December 2002 [5]. Various parties responded, and after three years of rigorous competition and productive collaboration, two schemes eventually emerged: the *Audio Lossless Coding* (ALS) and the *Scalable Lossless Coding* (SLS). Both ALS and SLS were adopted by MPEG because of their distinctive strengthes: ALS provides a better compression performance, while SLS can be easily embedded with a lossy audio coder such as the MPEG-4 Advanced Audio Coding (AAC) [6]. The MPEG-4 ALS and SLS were formally published by the International Standard Organization as international standards in March 2006 [7], and June 2006 [8], respectively. Technical details of ALS and SLS are thoroughly explained in [9] and [10]. This paper focuses on the RLS–LMS predictor used in ALS.

In SLS, the input audio samples are first divided into blocks and then converted into transform coefficients by using the integer modified discrete cosine transform (IntMDCT) [11]. The transform coefficients are scaled, quantized, and coded by the AAC encoder to generate a "core" bitstream, which constitutes the minimum quality/rate unit of the final lossless bitstream. For optimal coding efficiency, an error-mapping procedure is employed to remove the information that has already been coded in the core bitstream from the transform coefficients. The residuals are subsequently coded by bit-plane Golomb code [12] to form the final lossless bitstream. SLS provides fine-granular quality/rate scalability, and is well-suited for network music streaming services, where the bitstream can be dynamically truncated according to the available bandwidth.

In ALS, linear prediction is performed on the input audio samples to generate a residual signal, which has a smaller dynamic range than the input signal. The distribution of the residual signal can be closely modeled by a Laplacian (or two-sided geometric) distribution. The residual signal is entropy-coded with the Rice code [13]. For each block of audio samples, either all values can be coded by the same Rice code, or a single block can be further divided into four parts, each encoded with a different Rice code. Alternatively, the residual can also be coded by a more complex and efficient coding scheme called the block Gilbert–Moore code (BGMC) [14].

Manuscript received February 28, 2007; revised September 15, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. George Tzanetakis.

H. Huang, D. Huang, and S. Rahardja are with the Institute for Infocomm Research, Singapore 119613 (e-mail: hhuang@i2r.a-star.edu.sg; huang@i2r.a-star.edu.sg; rsusanto@i2r.a-star.edu.sg).

P. Fränti is with the Department of Computer Science, University of Joensuu, Joensuu, Finland FIN-80101 (e-mail: franti@cs.joensuu.fi).

Digital Object Identifier 10.1109/TASL.2007.911675

In BGMC, the residual distribution is further partitioned into three parts: a central region, flanked by two tail regions. The residuals in the tail regions are simply recentered and coded with Rice codes. Residuals within the central region are further split into those that belong to the least significant bit (LSB) and the most significant bit (MSB) parts. The LSB parts are directly transmitted using fixed-length codes without any processing, while the MSB parts are coded with the more efficient block Gilbert–Moore (arithmetic) code [15].

The MPEG-4 ALS has two different linear predictors: the *linear predictive coding* (LPC) predictor [9], and the *RLS–LMS* predictor [16]. In the LPC predictor, the optimal predictor coefficients are computed using the Levinson–Durbin algorithm [17] for each block of samples. The audio samples pass through a linear predictor whose coefficients are the quantized version of the optimal coefficients. The quantized coefficients are coded together with the residual to form the lossless bitstream.

The RLS–LMS predictor provides an estimate of the current input sample using past input samples. The prediction residual is calculated as the difference between the current input sample and the derived estimate. Unlike the LPC predictor, only the residual is encoded and transmitted. There is no need to code the coefficients of the predictor, as an identical predictor runs in the decoder. The latter is guaranteed to always maintain the same states as that in the encoder. The RLS–LMS predictor consists of a cascade of stages made up of simple predictors. The residual of one predictor stage is passed on as the input of the next stage. The first stage is a first-order predictor with a fixed coefficient value 1. In the second stage, the predictor coefficients are updated using the *recursive least square* (RLS) algorithm [18]. All the subsequent stages in the cascade use the *normalized least mean square* (NLMS) algorithm [18] in the updating of predictor coefficients. The estimate of the current input sample, or the final estimate, is generated by linearly combining the intermediate estimates generated by the cascaded predictor stages. The combiner coefficients are updated using the *sign–sign LMS* algorithm [18]. The residual signal is generated by subtracting the estimate from the current input sample and is subsequently coded by the entropy coder to form the lossless bitstream.

The development of the RLS–LMS predictor was motivated by prior work in cascaded prediction for lossless audio compression. In [19], Schuller proposed a cascaded predictor structure with three LMS prediction stages. The final estimate in that study was derived by a linear combination of the intermediate estimates from the three LMS predictors under the so-called predictive minimum description length weighting. In the RLS–LMS predictor, the computation of the linear combiner coefficients is simplified by using the effective, low-complexity sign–sign LMS algorithm. In [20], Yu proposed a cascaded predictor using a low-order RLS predictor followed by a high-order LMS predictor. A final estimate was then given by directly adding the two intermediate estimates from the RLS and the LMS predictors. This predictor can be viewed as a special case of the RLS–LMS predictor that uses only two prediction stages. In Yu's work, the coefficients of the linear combiner take a fixed value 1. For stereo audio input, the RLS–LMS predictor can also perform *joint-stereo prediction*, which exploits the inherent correlation between the left and right audio channels. In this mode, the RLS prediction stage generates the estimate signal for each audio channel by using past samples from both channels. We find that joint-stereo prediction can bring about a 3% improvement in the compression ratio compared to the case where the predictor runs for the left and right channels independently.

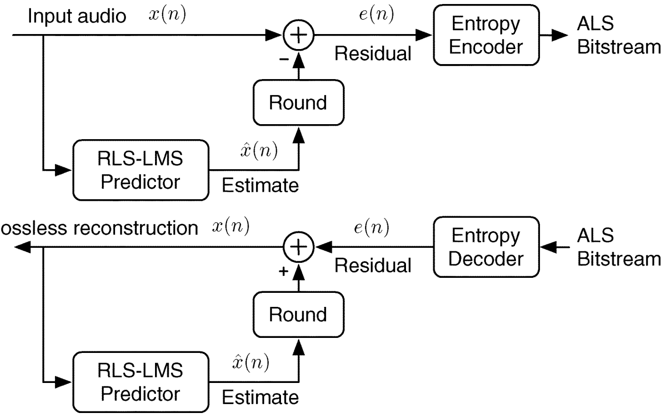


Fig. 1. MPEG-4 ALS encoder (top) and decoder (bottom) with the RLS–LMS predictor.

The rest of the paper is organized as follows: The structure of the RLS–LMS predictor is introduced in Section II. The adaptive algorithms used for updating predictor coefficients are described in Section III, while the joint-stereo prediction is explained in Section IV. The optimal predictor configuration is determined through various experiments with results summarized in Section V. This section also provides comparison results with other lossless coders. A conclusion of the paper is given in the last section.

II. Cascaded RLS–LMS Prediction

The structure of the MPEG-4 ALS is shown in Fig. 1, where the upper part shows the encoder and the lower half showing the decoder. In the encoder, an estimate of the current input sample is generated by the RLS–LMS predictor using past samples. This estimate is rounded to the nearest integer. A prediction residual is generated by subtracting the rounded estimate from the current input sample. The entropy coder subsequently encodes the residual with either the Rice code or the BGMC to form the lossless bitstream.

In the decoder, the above process is reversed. The lossless bitstream is first decoded into the prediction residual by the entropy decoder. The original audio data are then reconstructed by adding the residual to the rounded estimate. The RLS–LMS predictor in the decoder is identical to that in the encoder and maintains the exact same states and coefficients as the latter at all times. Because of the lossless (noiseless) entropy encoding/decoding process, as well as the use of identical predictors in the encoder and the decoder, perfect-reconstruction of the original audio samples is guaranteed by ALS.

The structure of the cascaded RLS–LMS predictor is shown in Fig. 2. The predictor consists of a cascade of simple prediction stages in the sequence of a differential PCM (DPCM) predictor, an RLS predictor, and a series of LMS predictors. The input samples pass through the prediction stages sequentially,

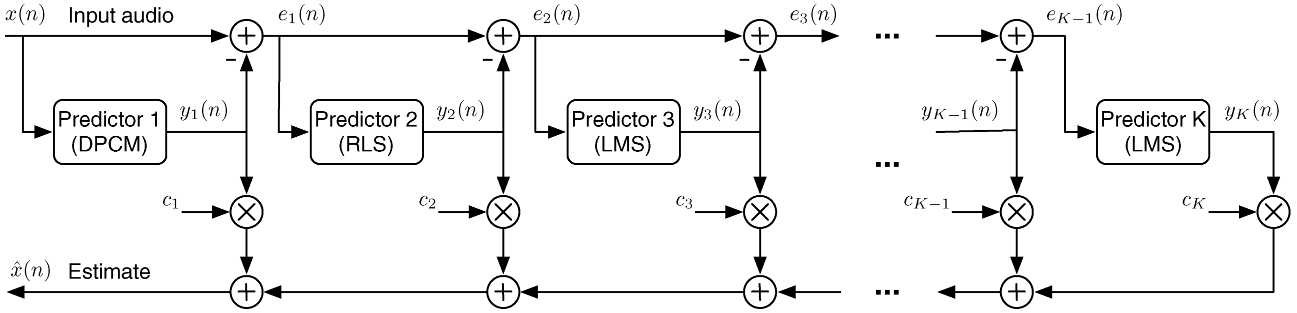


Fig. 2. Structure of the cascaded RLS–LMS predictor.

with the residual of one stage serving as the input to the next stage. The estimates from each prediction stage are summed up by a linear combiner to generate the estimate of the current input sample according to

$$\hat{x}(n) = \sum_{k=1}^{K} c_k(n) y_k(n) \tag{1}$$

where $\hat{x}(n)$ is the estimate of the current input sample $x(n)$, n is the time index of samples, K is the number of prediction stages in the cascade, $c_k(n)$ are the coefficients of the linear combiner, k is the stage index, and $y_k(n)$ are estimates from the prediction stages.

In the kth stage, defining the order of the predictor as M_k, the estimate $y_k(n)$ is given by

$$y_k(n) = \sum_{m=1}^{M_k} a_{k,m}(n) e_{k-1}(n-m) \tag{2}$$

where $a_{k,m}(n)$ are the coefficients of the predictor, m is the tap index, and $e_{k-1}(n)$ is the residual from the previous stage. The residual of the kth stage is given by

$$e_k(n) = e_{k-1}(n) - y_k(n). \tag{3}$$

III. Adaptive Updating of Predictor Coefficients

The RLS–LMS predictor consists of a cascade of prediction stages and a linear combiner. All the prediction stages as well as the linear combiner update the coefficients adaptively, except for the first DPCM prediction stage, which uses a first-order predictor with a fixed coefficient value 1. The RLS algorithm is used in the second stage, and the NLMS algorithm is used in all the remaining stages in the cascade. For the linear combiner, the sign–sign LMS algorithm is used. The following subsections describe the adaptive algorithms used in each part of the RLS–LMS predictor.

A. DPCM Predictor

As the first predictor in the cascade, the DPCM predictor is a simple first-order predictor with the coefficient set to 1, i.e., the previous input sample is used as the estimate of the current input sample. The DPCM predictor is given by

$$y_1(n) = x(n-1) \tag{4}$$
$$e_1(n) = x(n) - y_1(n) \tag{5}$$

where $y_1(n)$ is the estimate of the first prediction stage, $e_1(n)$ is the prediction residual, and $x(n-1)$ is the previous input sample.

B. RLS Predictor

The RLS predictor is the second predictor in the cascade. The RLS algorithm is used to adapt the predictor coefficients. The algorithm is initialized by setting the inverse autocorrelation matrix $\mathbf{P}$ as follows:

$$\mathbf{P}(0) = \delta\mathbf{I}$$

where δ is a small positive number, $\mathbf{I}$ is an $M_2 \times M_2$ identity matrix, and M_2 is the order of the RLS predictor. The predictor coefficient vector $\mathbf{a}_2(n)$, defined as

$$\mathbf{a}_2(n) = [a_{2,1}(n), a_{2,2}(n), \ldots, a_{2,M_2}(n)]^T$$

is initialized by

$$\mathbf{a}_2(0) = \mathbf{0}.$$

Here, the symbol T denotes the operation of vector transpose.

Define

$$\mathbf{e}_1(n) = [e_1(n-1), e_1(n-2), \ldots, e_1(n-M_2)]^T$$

as the RLS predictor input vector, for each instance of time, $n = 1, 2, \ldots$, the following calculations are made:

$$\mathbf{v}(n) = \mathbf{P}(n-1)\mathbf{e}_1(n) \tag{6}$$
$$m = \begin{cases} \frac{1}{\mathbf{e}_1^T(n)\mathbf{v}(n)}, & \text{if } \mathbf{e}_1^T(n)\mathbf{v}(n) \neq 0 \\ 1, & \text{else} \end{cases} \tag{7}$$
$$\mathbf{k}(n) = m\mathbf{v}(n) \tag{8}$$
$$y_2(n) = \mathbf{a}_2^T(n-1)\mathbf{e}_1(n) \tag{9}$$
$$e_2(n) = e_1(n) - y_2(n) \tag{10}$$
$$\mathbf{a}_2(n) = \mathbf{a}_2(n-1) + \mathbf{k}(n)e_2(n) \tag{11}$$
$$\mathbf{P}(n) = \mathrm{Tri}\{\lambda^{-1}(\mathbf{P}(n-1) - \mathbf{k}(n)\mathbf{v}^T(n))\}. \tag{12}$$

In (12), λ is the forgetting factor that is a positive value slightly smaller than 1. Tri$\{.\}$ denotes the operation of first computing the lower triangular part of $\mathbf{P}(n)$, and then copying the values in the lower triangular to the upper triangular according to

$$p_{i,j} = p_{j,i}$$

where $p_{i,j}$ is the element of matrix $\mathbf{P}(n)$ at the ith row and the jth column.

There is a slight difference between the above RLS algorithm and the standard version in [18]. The denominator in (7) is given by $\{\mathbf{e}_1^T(n)\mathbf{v}(n)\}$ instead of $\{\lambda + \mathbf{e}_1^T(n)\mathbf{v}(n)\}$ as defined in the standard RLS algorithm. The reason of why λ is neglected from the summation is that λ is several orders of magnitude smaller than $\{\mathbf{e}_1^T(n)\mathbf{v}(n)\}$ when the input signals are 16-bit PCM samples.

C. LMS Predictor

The RLS–LMS predictor has a series of LMS prediction stages. The NLMS algorithm [19] is used to adapt the coefficients of the LMS predictors. For the LMS predictor in the kth stage, the coefficient vector

$$\mathbf{a}_k(n) = [a_{k,1}(n), a_{k,2}(n), \ldots, a_{k,M_k}(n)]^T$$

is initialized by

$$\mathbf{a}_k(0) = \mathbf{0}$$

where M_k is the order of the predictor.

Define

$$\mathbf{e}_{k-1}(n) = [e_{k-1}(n-1), e_{k-1}(n-2), \ldots, e_{k-1}(n-M_k)]^T$$

as the input vector to the kth prediction stage, for each instance of time, $n = 1, 2, \ldots$, the following calculations are made:

$$y_k(n) = \mathbf{a}_k^T(n-1)\mathbf{e}_{k-1}(n) \quad (13)$$

$$e_k(n) = e_{k-1}(n) - y_k(n) \quad (14)$$

$$\mathbf{a}_k(n) = \mathbf{a}_k(n-1) + \frac{e_k(n)\mathbf{e}_{k-1}(n)}{1 + \mu_k \mathbf{e}_{k-1}^T(n)\mathbf{e}_{k-1}(n)} \quad (15)$$

where $\mu_k \geq 1$ is the stepsize of the NLMS algorithm.

D. Linear Combiner

The linear combiner multiplies a set of coefficients to the estimates from the DPCM, RLS, and LMS prediction stages. The results are summed up together to provide the estimate of the current input sample. The sign–sign LMS algorithm is used to adapt the coefficients of the linear combiner.

The coefficient vector is defined as

$$\mathbf{c}(n) = [c_1(n), c_2(n), \ldots, c_K(n)]^T$$

where K is the number of prediction stages in the cascade. The input vector is given by

$$\mathbf{y}(n) = [y_1(n), y_2(n), \ldots, y_K(n)]^T.$$

The estimate of the RLS–LMS predictor is calculated as

$$\hat{x}(n) = \mathbf{c}^T(n)\mathbf{y}(n). \quad (16)$$

The linear combiner coefficients are updated according to

$$\mathbf{c}(n+1) = \mathbf{c}(n) + \alpha\, \mathrm{sgn}[\mathbf{y}(n)]\, \mathrm{sgn}[x(n) - \hat{x}(n)] \quad (17)$$

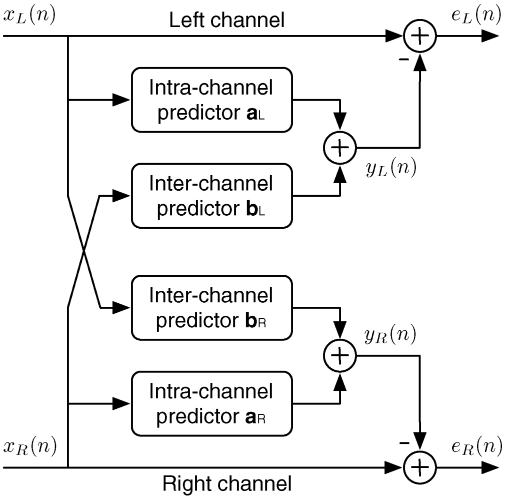

$x_L(n)$
Left channel
$e_L(n)$
Intra-channel predictor $\mathbf{a}_L$
Inter-channel predictor $\mathbf{b}_L$
$y_L(n)$
Inter-channel predictor $\mathbf{b}_R$
$y_R(n)$
Intra-channel predictor $\mathbf{a}_R$
$x_R(n)$
$e_R(n)$
Right channel
-
-

Fig. 3. Joint-stereo prediction.

where the sgn function is defined as

$$\mathrm{sgn}[r] = \begin{cases} 1, & r > 0 \\ 0, & r = 0 \\ -1, & r < 0 \end{cases}. \quad (18)$$

If the input to the sgn function is a vector, the output is also a vector that contains signs of each individual elements in that vector. The stepsize α takes a small positive value.

IV. Joint-Stereo Prediction

For mono audio signals, the correlation that linear prediction tries to reduce is among audio samples within the same channel. This type of correlation is called *intrachannel* correlation. On the other hand, for stereo audio signals correlation also exists between samples in different channels. This type of correlation is referred to as *interchannel* correlation. Both intrachannel and interchannel correlations are exploited by the RLS–LMS predictor through a joint-stereo prediction, where past samples from both L and R audio channels are used in estimating the current sample of each channel. This joint-stereo prediction is implemented in the second prediction stage as illustrated in Fig. 3.

In Fig. 3, the intrachannel predictor $\mathbf{a}_L$ generates an estimate for the L channel by using L channel samples. At the same time, the interchannel predictor $\mathbf{b}_L$ generates another estimate by using samples in the R channel. The two estimates are added together to give the output estimate of the RLS prediction stage for the L channel. Let M_a and M_b be the orders of the intrachannel predictor $\mathbf{a}_L$ and interchannel predictor $\mathbf{b}_L$, respectively, and the L channel estimate is given by

$$y_L(n) = \sum_{m=1}^{M_a} a_{L,m} x_L(n-m) + \sum_{m=1}^{M_b} b_{L,m} x_R(n-m) \quad (19)$$

where $a_{L,m}$ and $b_{L,m}$ are coefficients of predictor $\mathbf{a}_L$ and predictor $\mathbf{b}_L$, respectively. $x_L(n)$, and $x_R(n)$ are input samples in the L and R channels.

Similarly, the R channel estimate is given by

$$y_R(n) = \sum_{m=1}^{M_a} a_{R,m} x_R(n-m) + \sum_{m=0}^{M_b-1} b_{R,m} x_L(n-m) \quad (20)$$

where $a_{R,m}$ and $b_{R,m}$ are coefficients of the intrachannel predictor $\mathbf{a}_R$ and the interchannel predictor $\mathbf{b}_R$, respectively. Note that the second summation term in (20) starts from 0 instead of 1 as in (19). The reason is that, in the ALS decoder, audio samples are reconstructed in the order of

$$[\ldots, x_L(n-1), x_R(n-1), \\ x_L(n), x_R(n), x_L(n+1), x_R(n+1), \ldots].$$

This order ensures that the L channel sample $x_L(n)$ is consistently decoded before the R channel sample $x_R(n)$, which motivated the use of $x_L(n)$ for the decoding of $x_R(n)$.

In joint-stereo prediction, the coefficients of the intra- and interchannel predictors are updated using the RLS algorithm given in Section III-B. For the L channel, the input vector and coefficient vector are given by

$$\begin{aligned}\mathbf{x}_L(n) &= [x_L(n-1), \ldots, x_L(n-M_a), \\ &\quad x_R(n-1), \ldots, x_R(n-M_b)]^T \quad (21) \\ \mathbf{w}_L(n) &= [a_{L,1}(n), \ldots, a_{L,M_a}(n), b_{L,1}(n), \ldots, b_{L,M_b}(n)]^T \quad (22)\end{aligned}$$

respectively. For the R channel, the input vector and coefficient vector are given by

$$\begin{aligned}\mathbf{x}_R(n) &= [x_R(n-1), \ldots, x_R(n-M_a), \\ &\quad x_L(n), \ldots, x_L(n-M_b+1)]^T \quad (23) \\ \mathbf{w}_R(n) &= [a_{R,1}(n), \ldots, a_{R,M_a}(n), b_{R,1}(n), \ldots, b_{R,M_b}(n)]^T \quad (24)\end{aligned}$$

respectively. In joint-stereo prediction, the RLS routine is first called to update the L channel intra- and interchannel predictors, and then called again to update the R channel predictors. There are also two $\mathbf{P}$ matrices, one for each channel.

V. Predictor Optimization and Experimental Results

During the standardization process, extensive tests were conducted to optimize the parameters of the RLS–LMS predictor. To benchmark the performance of various lossless compressors, MPEG used a common test set [21] that consists of sampled waveforms of 15 different types of music. The sampling frequency/resolution used are: 48 kHz/16 bit, 48 kHz/24 bit, 96 kHz/24 bit, and 192 kHz/24 bit. Each waveform lasts 30 s, and the total playtime of the whole test set is 25 min. This test set was also used to run various experiments in this paper.

A. Predictor Signals and Residual Distribution

Fig. 4 illustrates a segment of typical input and output signals of the RLS–LMS predictor. The typical probability density distributions of the residual signals are shown in Fig. 5 for the following three predictor configurations:

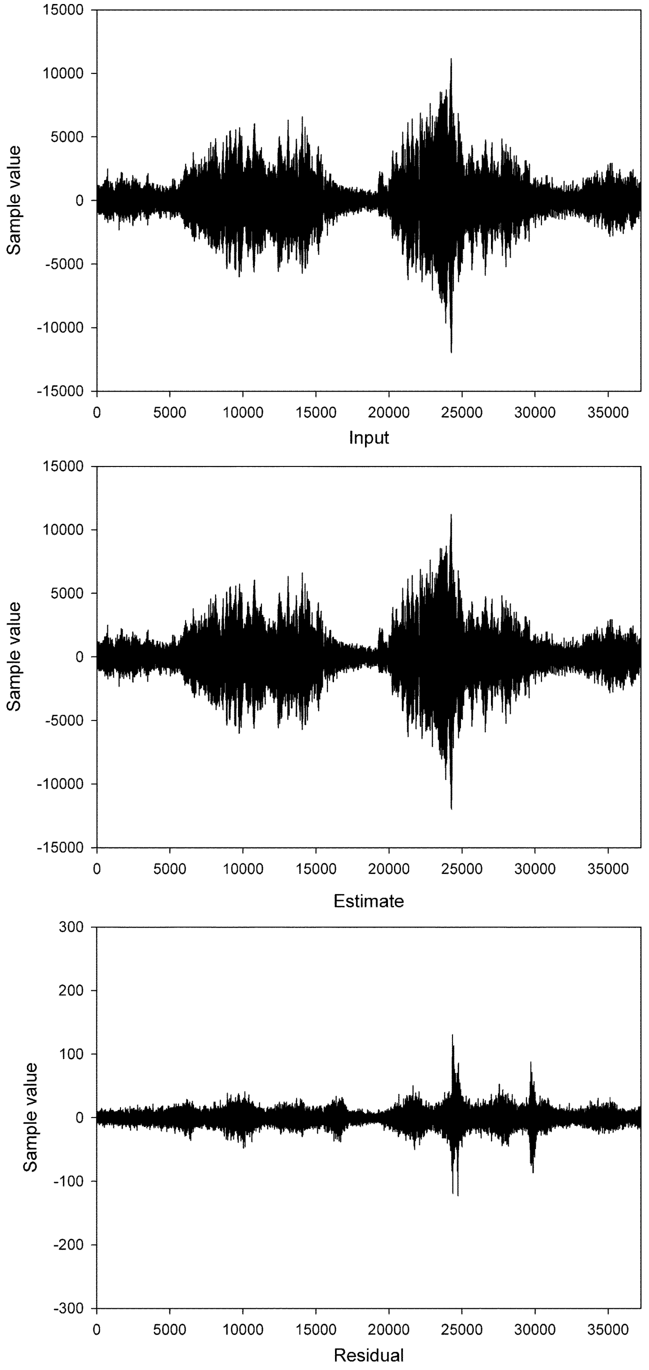


Fig. 4. Input and output signals of the RLS–LMS predictor.

1) DPCM (using only the first prediction stage);
2) DPCM + RLS (using the first and second prediction stages);
3) DPCM + RLS + LMS (using all the prediction stages).

Fig. 5 shows that when the number of prediction stages increases, the residual distribution tends to concentrate towards the center. The entropies of the distributions in the figure are: 1) 9.63, 2) 6.54, and 3) 6.08, respectively. The results show that the entropy is not reduced enough by the DPCM predictor alone, and significant improvement can be obtained by the subsequent

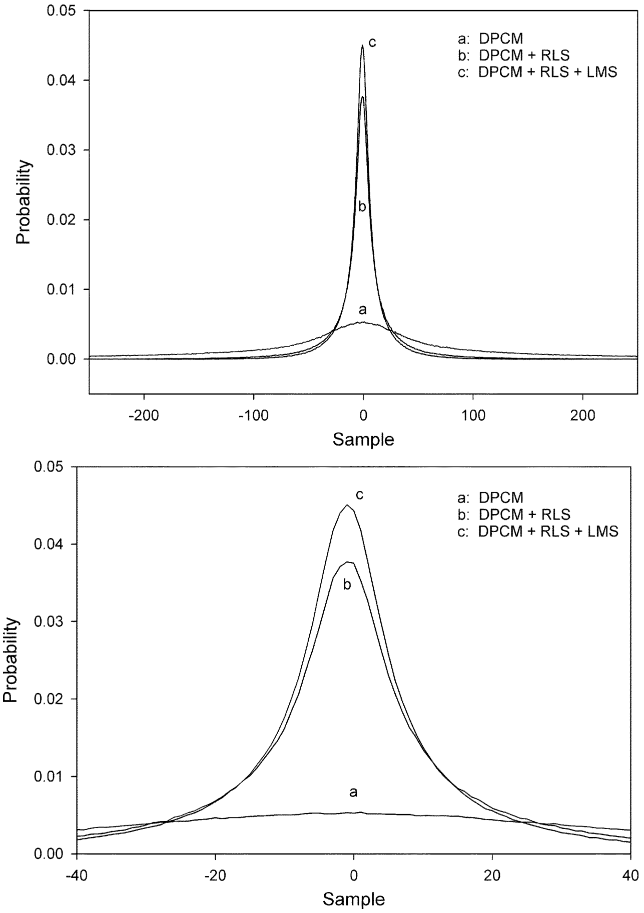


Fig. 5. Residual distributions for three predictor configurations (above) and a close-up of the central region of the distributions (below).

TABLE I
COMPRESSION RATIOS FOR VARIOUS RLS–LMS PREDICTOR CONFIGURATIONS

No.	Predictor Configuration	Compression Ratio
1	DPCM + RLS + LMS	2.172
2	RLS + LMS	1.935
3	DPCM + LMS	2.126
4	DPCM + RLS	2.089
5	DPCM + mono RLS + LMS	2.109
6	DPCM + LMS + RLS	2.165

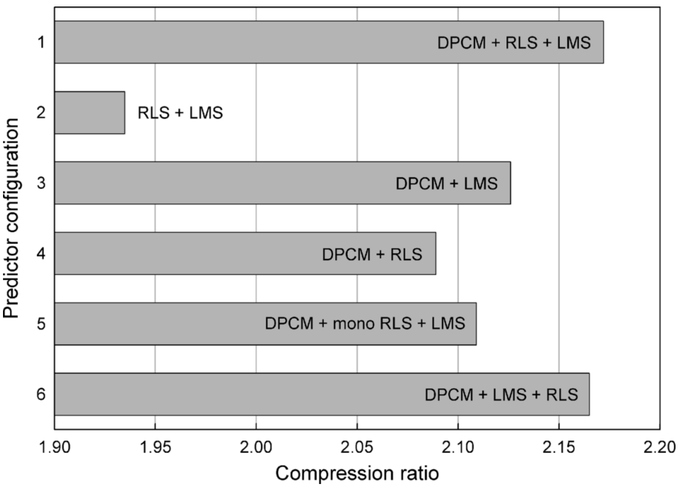


Fig. 6. Compression ratios for various RLS–LMS predictor configurations.

TABLE II
COMPRESSION RATIOS FOR VARIOUS CONFIGURATIONS OF LMS PREDICTION STAGES

No.	LMS Configuration	Compression Ratio
1	300 + 72 + 12	2.162
2	192 + 128 + 64	2.157
3	128 + 128 + 128	2.152
4	64 + 128 + 192	2.153
5	12 + 72 + 300	2.155
6	64 + 256 + 64	2.156
7	160 + 64 + 160	2.155

RLS predictor. The LMS predictor reduces the entropy further by about 10%.

B. Various Predictor Configurations

The RLS–LMS predictor can be configured in a number of ways. A few intuitive configurations are listed in Table I. These configurations are compared in terms of compression ratio, which is defined as

$$\text{compression ratio} = \frac{\text{original filesize}}{\text{compressed filesize}}. \tag{25}$$

In Table I, six predictor configurations are compared. Among the six predictor configurations, Configuration 1 is the selected configuration with cascaded prediction stages in the sequence of DPCM, followed by RLS, and finally LMS. Configurations 2–4 contain only two stages in the cascade, with one stage turned off. Configuration 5 uses only intrachannel prediction (as inferred from the term "mono") with no interchannel prediction. In Configuration 6, the position of the RLS and LMS prediction stages are swapped. To facilitate our analysis, the results listed in Table I are plotted in Fig. 6. The results show that Configuration 1 produces the highest compression ratio. This configuration is actually the one adopted by the standard. A 3% improvement in compression ratio is also found by comparing Configuration 1 (using joint-stereo prediction) with Configuration 5 (using only intrachannel prediction).

C. Configuration of LMS Prediction Stages

The RLS–LMS predictor contains a cascade of LMS prediction stages. Each stage is an LMS predictor of a certain order. Various configurations of the LMS predictor cascade are compared in Table II, where the second column lists the different combinations of LMS predictor orders. Each of the configurations in the table is comprised of three LMS stages, and has a total predictor order of 384.

In the first two configurations listed in Table II, the LMS cascade has predictor orders of descending values. Configuration 3 uses predictors of equal order, while in Configurations 4 and 5, the predictor orders are increasing in sequence. The final two configurations show two alternative combination patterns of

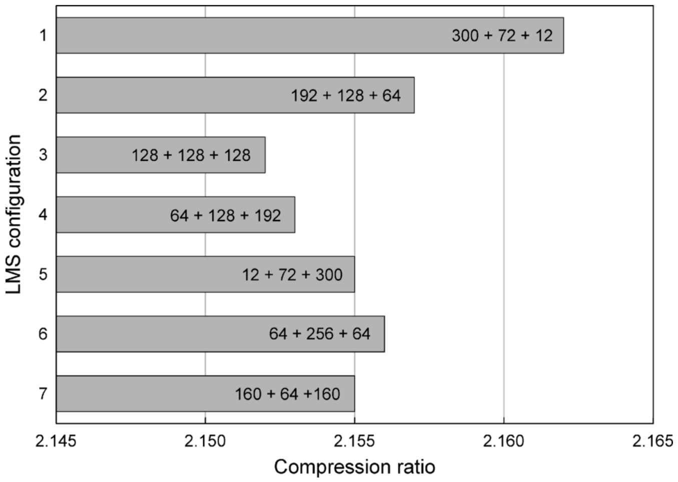


Fig. 7. Compression ratios for various configurations of LMS predictor stages.

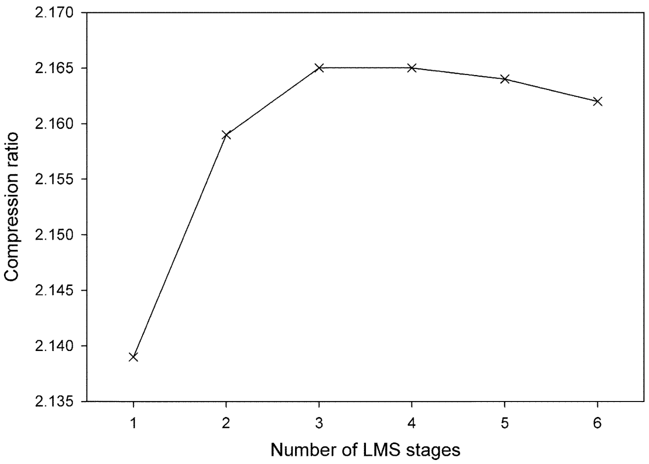


Fig. 8. Compression ratios for configurations with one to six LMS prediction stages.

TABLE III
COMPRESSION RATIOS FOR CONFIGURATIONS WITH ONE TO SIX LMS PREDICTION STAGES

No.	Predictor Configuration	Compression Ratio
1	512	2.139
2	384 + 128	2.159
3	384 + 112 + 16	2.165
4	384 + 90 +30 + 8	2.165
5	360 + 100 + 36 + 12 + 4	2.164
6	300 + 120 + 60 + 22 + 8 + 2	2.162

TABLE IV
COMPRESSION RATIOS FOR LINEAR COMBINER CONFIGURATIONS WITH ZERO TO FIVE NONUPDATE COEFFICIENTS

No.	Configuration of Combiner Coefficients					Compression
	DPCM	RLS	LMS1	LMS2	LMS3	Ratio
0	*	*	*	*	*	1.985
1	1	*	*	*	*	2.158
2	1	1	*	*	*	2.172
3	1	1	1	*	*	2.171
4	1	1	1	1	*	2.169
5	1	1	1	1	1	2.167

"short-long-short" and "long-short-long," respectively. For ease of comparison, the results of Table II are plotted in Fig. 7. The results show that the highest compression ratio is provided by Configuration 1, which is also the selected configuration of the RLS–LMS predictor. Configuration 1 confirms the observation in [19] that predictor orders should optimally be in a sequence of descending values. In addition, we find that a better compression ratio can be obtained when the LMS predictor orders are separated by wide margins, as demonstrated by the performances of Configurations 1 and 2.

D. Number of LMS Prediction Stages

The RLS–LMS predictor can be configured to use different numbers of LMS prediction stages. Table III lists the configurations where one to six LMS stages are used. The total order of LMS predictors in each configuration is fixed at 512.

In Table III, the LMS prediction stages are configured in a descending pattern of predictor orders, as suggested in Section V-C. The results are also shown in Fig. 8. The compression ratio is found to peak at three LMS prediction stages, which suggests that the RLS–LMS predictor needs no more than three LMS stages.

E. Configuration of Linear Combiner

In the RLS–LMS predictor, the final estimate is generated by multiplying the intermediate estimates obtained from the prediction stages by the linear combiner coefficients and summing up the results. The coefficients of the linear combiner are updated by the sign–sign LMS algorithm. We find that the best compression ratio can be obtained by updating only part of the combiner coefficients, while setting the others to 1. The configurations of the linear combiner coefficients are illustrated in Table IV, where the five central columns list the values of the combiner coefficients. Each of these columns corresponds to the prediction stage that is indicated by the column header. Value "1" in the table indicates that the coefficient is fixed to 1, while a "*" means that the coefficient is adaptively updated. Six configurations of the combiner coefficients were tested, with the number of fixed, nonupdate coefficients increasing from zero to five. The corresponding compression ratios are plotted in Fig. 9. As evident from the graph, the highest compression is achieved by fixing the first two combiner coefficients to 1, while adaptively updating the other three coefficients.

F. Comparison of Lossless Compressors

A number of state-of-the-art lossless compressors were used to benchmark the performance of the RLS–LMS predictor. The experiments were run on a Pentium IV 2.4-GHz PC, and the results are summarized in Table V. The lossless compressors used are divided into two categories: standard coders from MPEG and nonstandard proprietary ones. The first category includes the MPEG-4 SLS RM8 [22] and the MPEG-4 ALS RM18 [23] running in two predictor modes: the RLS–LMS mode and the

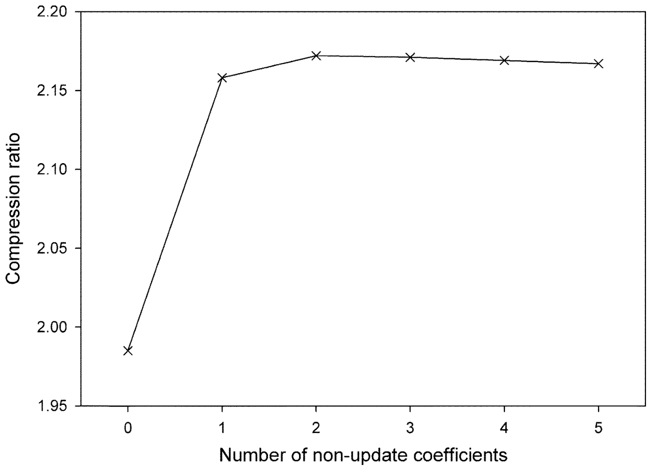


Fig. 9. Compression ratios for linear combiner configurations with zero to five nonupdate coefficients.

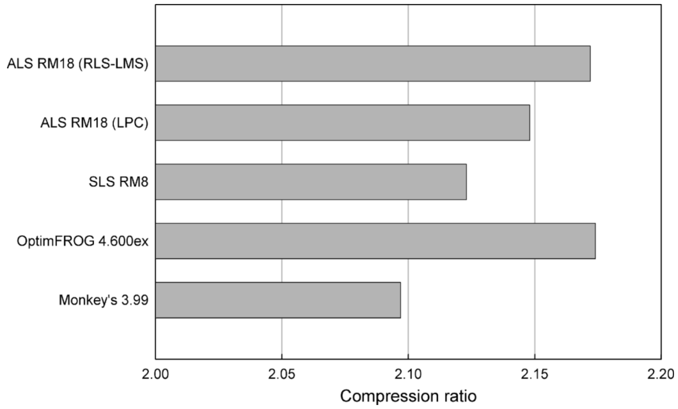


Fig. 10. Comparison of lossless compressors.

TABLE V
COMPARISON OF LOSSLESS COMPRESSORS

Lossless		Compression	Speed (× realtime)	
Compressors		Ratio	Encode	Decode
Standard	ALS RM18 (RLS-LMS)	2.172	0.4	0.4
	ALS RM18 (LPC)	2.148	20	40
	SLS RM8	2.123	22	25
Non-standard	OptimFROG 4.600ex	2.174	1.2	2.1
	Monkey's Audio 3.99	2.097	6	6

LPC mode. Both these coders were implemented in 32-bit fixed-pointed C code. Coders for the second category are selected according to the following considerations: OptimFROG [3] is reported [24]–[26] to be one of the top lossless audio coders in terms of the compression ratio, and is therefore used as a major benchmark for our comparison. Because of the widespread popularity over internet, the Monkey's coder [1] is also included in the comparison. The compression ratio results are plotted in Fig. 10.

Clearly, the highest compression ratio is provided by OptimFROG 4.600ex, closely followed by the ALS RM18 coder running in the RLS–LMS predictor mode. This suggests that the latter becomes one of the top performers in the field in terms of the lossless compression ratio. Among the five tested coders, the ALS RM18 coder running in the RLS–LMS predictor mode gives the slowest encoding/decoding speed. This slow speed of the coder results from the high complexity of the RLS–LMS predictor, which consists of a computationally intensive RLS filter, as well as large-order LMS filters.

The ALS reference software RM18 was implemented in 32-bit fixed-point arithmetics. In the RLS–LMS predictor, to guarantee convergence of the adaptive coefficients update recursions, computations inside the RLS recursions must be performed with a high numerical precision [18]. This requirement is generally not a problem for implementations done in double-precision floating-point, but remains a challenging issue for fixed-point implementations because of the much smaller dynamic range to represent values in fixed-point. In the RLS and NLMS recursions, each multiplication/division operation is coupled with necessary prescaling, postscalings, and range comparison instructions to keep the numerical precision high. As a result, a multiplication done in one floating-point step can only be achieved by several fixed-point steps. This overhead with fixed-point implementation also contributes to the high complexity of the RLS–LMS predictor.

The MPEG-4 lossless audio compression standard provides a range of coders to handle different application scenarios. For example, the SLS coder provides lossless bitstream that can be arbitrarily truncated to cater for different transmission bandwidth requirements. The ALS coder running in the LPC predictor mode has a very high encoding/decoding speed. Among all the coders in the standard, the ALS coder in the RLS–LMS predictor mode provides the highest level of audio compression.

VI. CONCLUSION

This paper provides a detailed description of the cascaded RLS–LMS predictor in the MPEG-4 ALS standard. The predictor consists of a cascade of linear prediction stages in the sequence of a DPCM predictor, followed by an RLS predictor, and finally a series of LMS predictors. A linear combiner then sums up the intermediate estimate signals from these prediction stages to generate the final estimate signal of the RLS–LMS predictor. Various configurations were experimented with the predictor to optimize the predictor settings. The results from these experiments provide valuable insights and guidelines to the field of cascaded adaptive filter design. The ALS coder running the RLS–LMS predictor demonstrates a compression ratio that is on par with the best lossless audio coders in the field. An important work in the future is to reduce the high computational complexity of the predictor. Some potential strategies are optimizing parameters of the adaptive algorithms so that the orders of the RLS and LMS predictors can be kept low and choosing adaptive algorithms that are less computationally intensive than the RLS and NLMS algorithms. The core portion of the RLS–LMS predictor is made up of a highly efficient, fast-tracking cascaded adaptive filter, which also has a wide range of applications including: system identification, blind source classification and separation, channel equalization, beam-forming, and noise cancelation.

References

[1] *Monkey's Audio*, [Online]. Available: http://www.monkeysaudio.com/.

[2] *FLAC*, [Online]. Available: http://flac.sourceforge.net/.

[3] *OptimFROG*, [Online]. Available: http://www.losslessaudio.org/.

[4] *Lossless Audio*, [Online]. Available: http://www.lossless-audio.com/.

[5] "Final call for proposals on MPEG-4 lossless audio coding," ISO/IEC JTC1/SC29/WG11 Moving Picture Experts Group, Shanghai, China, 2002, N5208.

[6] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, K. Akagiri, H. Fuchs, M. Dietz, J. Herre, G. Davidson, and Y. Oikawa, "ISO/IEC MPEG-2 advanced audio coding," *J. Audio Eng. Soc.*, vol. 45, no. 10, pp. 789–814, 1997.

[7] "Audio lossless coding (ALS), new audio profiles and BSAC extensions," 2006, ISO/IEC 14496-3:2005/Amd 2:2006.

[8] "Scalable Lossless Coding (SLS)," 2006, ISO/IEC 14496-3:2005/Amd 3:2006.

[9] T. Liebchen, T. Moriya, N. Harada, Y. Kamamoto, and Y. Reznik, "The MPEG-4 audio lossless coding (ALS) standard – Technology and applications," in *Proc. 119th AES Conv.*, New York, Oct. 2005, preprint 6589.

[10] R. Yu, R. Geiger, S. Rahardja, J. Herre, X. Lin, and H. Huang, "MPEG-4 scalable to lossless audio coding," in *Proc. 117th AES Conv.*, San Francisco, CA, Oct. 2004, preprint 6183.

[11] Y. Yokotani, R. Geiger, G. D. T. Schuller, S. Oraintara, and K. R. Rao, "Lossless audio coding using the IntMDCT and rounding error shaping," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 6, pp. 2201–2211, Nov. 2006.

[12] R. Yu, S. Rahardja, X. Lin, and C. C. Ko, "A fine granular scalable to lossless audio coder," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 4, pp. 1352–1363, Jul. 2006.

[13] R. F. Rice, "Some practical universal loiseless coding techniques," JPL, 1979, Tech. Reps 79-22.

[14] Y. A. Reznik, "Coding of prediction residual in MPEG-4 standard for lossless audio coding (MPEG-4 ALS)," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP'04)*, Montreal, QC, Canada, May 2004, vol. 3, pp. 1024–1027.

[15] E. N. Gilbert and E. F. Moore, "Variable-length binary encodings," *Bell Syst. Tech. J. 38*, pp. 932–967, Jul. 1959.

[16] H. Huang, S. Rahardja, X. Lin, R. Yu, and P. Franti, "Cascaded RLS–LMS prediction in MPEG-4 lossless audio coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP'06)*, Toulouse, France, May 2006, vol. 5, pp. 181–184.

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

[18] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[19] G. D. T. Schuller, B. Yu, D. Huang, and B. Edler, "Perceptual audio coding using adaptive pre- and post-filters and lossless compression," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 6, pp. 379–390, Sep. 2002.

[20] R. Yu and C. C. Ko, "Lossless compression of digital audio using cascaded RLS–LMS prediction," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 532–537, Nov. 2003.

[21] "Audio Research Labs," [Online]. Available: http://www.audioresearchlabs.com/

[22] MPEG-4 SLS Reference Software. [Online]. Available: ftp://vpsp.i2r.a-star.edu.sg.

[23] MPEG-4 ALS Reference Software. [Online]. Available: http://www.nue.tu-berlin.de/forschung/projekte/lossless/mp4als.html.

[24] "Hydrogen Audio," [Online]. Available: http://wiki.hydrogenaudio.org/.

[25] "Performance comparison of lossless audio compressors," [Online]. Available: http://members.home.nl/w.speek/comparison.htm/.

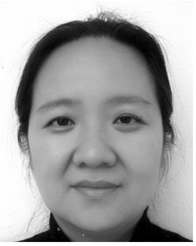
[26] "Compression and speed of lossless audio formats," [Online]. Available: http://web.inter.nl.net/users/hvdh/lossless/lossless.htm.

Haibin Huang (M'07) was born in Henan, China, in 1972. He received the B.S. degree from Xi'an Jiaotong University, Xi'an, China, in 1993, the M.S. degree from National University of Singapore in 1996, and the Ph.D. degree from the University of Joensuu, Joensuu, Finland, in 2007.

He has been a Research Scientist with the Institute for Infocomm Research, Singapore, since 2003. His research interests include audio signal processing, adaptive signal processing, and interactive media.

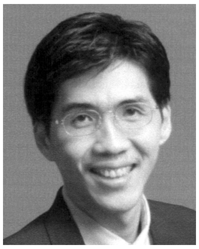
Pasi Fränti received the M.Sc. and Ph.D. degrees in computer science from the University of Turku, Turku, Finland, in 1991 and 1994, respectively.

From 1996 to 1999 he was a Postdoctoral Researcher at the Academy of Finland. Since 2000, he has been a Professor at the University of Joensuu, Joensuu, Finland. His primary research interests are in image compression, clustering, and speech technology.



Dongyan Huang (SM'04) was born in Xi'an, Shaanxi, f China, in 1963. She received the B.Sc. degree in control and information engineering and the M.Sc degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 1985 and 1988, respectively, and the Ph.D. degree in signal processing from the Conservatoire National des Arts et Métiers Paris (CNAM), Paris, France, in 1996.

In December 1996, she began her postdoctoral research work on low-delay high-quality audio and speech codec design at UFR de Mathématiques et Informatique, Université René Descartes, Paris V. From December 1997 to December 2002, she was a Senior Research Engineer with the Institute of Microelectronics, Singapore. She is currently a Senior Research Fellow with Institute for Infocomm Research, Singapore. Her research interests include lossy/lossless audio and image compression, recognition and synthesis of expression in speech and audio, adaptive filtering, and robust wireless multimedia communications.



Susanto Rahardja (M'97–SM'03) received the B.Eng. degree from the National University of Singapore and the M.Eng. and Ph.D. degrees from the Nanyang Technological University (NTU), Singapore, all in electrical and electronic engineering.

He has been the Director of the Media Division at the Institute for Infocomm Research, Singapore, from 2002 to 2007. His research interests are in audio/video signal processing, spread spectrum and multiuser detection techniques for CDMA applications, digital signal processing algorithms, and implementations and logic synthesis, of which he has more than 180 publications in internationally refereed journals and conferences. He is also an Associate Professor at the School of Electrical and Electronic Engineering, NTU.

Dr. Rahardja was the recipient of the Institution of Electronic Engineers (IEE) Hartree Premium Award for the best journal paper published in IEE Proceedings in 2002. In 2003, he received the prestigious Tan Kah Kee Young Inventorsaŕ Gold award in the Open Category for his contributions on scalable to lossless audio compression technology. Since 2002, he has been actively participating in the international ISO/IEC JTC1/SC29/WG11 (Moving Picture Expert Group, or MPEG) where he contributed to MPEG-4 scalable to lossless system (SLS) and the technology is incorporated in ISO/IEC 14496-3:2005/Amd.3:2006. He also contributed technology to the MPEG-4 Audio Lossless System (ALS) where it is now incorporated in ISO/IEC 14496-3:2005/Amd.2:2006. In recognition for his contributions to the national standardization program, he was awarded the Standards Council Merit Award by SPRING Singapore in 2006. He has served on several boards, advisory, and technical committees in various IEEE- and SPIE-related professional activities in the areas of multimedia. He is currently serving as an Associate Editor for the IEEE Transactions on Audio, Speech, and Language Processing and the *Journal of Visual Communication and Image Representation*.