LOSSLESS AUDIO COMPRESSION IN THE NEW IEEE STANDARD FOR ADVANCED AUDIO CODING

H. Huang, H. Shu, and R. Yu

Institute for Infocomm Research,
Agency for Science, Technology and Research (A*STAR), Singapore

ABSTRACT

The IEEE Standard for Advanced Audio Coding (IEEE 1857.2) is a new standard approved by IEEE in August 2013. The standard comprises both lossy and lossless audio compression tools. This paper presents the lossless audio compression tool, which utilizes a pre-processing procedure for flattening the amplitude envelop of linear prediction residue, and an arithmetic coder that adopts a scaled probability template. The performance of the new IEEE lossless compressor is evaluated and compared with state-of-the-art lossless audio coders. Evaluation results show that the lossless compression performance of the IEEE compressor is about 5% higher than MPEG-4 ALS and 12% higher than FLAC.

Index Terms— Lossless audio compression, IEEE 1857.2, linear prediction, arithmetic coding

1. INTRODUCTION

During the last decade, the Internet has undergone growth at an exponential pace. Coupled with advancement of semiconductor technology and digital signal processing algorithm, online multimedia contents are now pervasive, and there is no sign of slowing down in the speed of new contents being added. A significant portion of those multimedia contents are audio, which are commonly pre-compressed before being put online. Generally, there are two types of audio compression methods: the lossy and the lossless. The former attempts to remove perceptually less important information from the audio data while keeping the sound quality very close, and sometime indistinguishable, to the original audio. Examples of such lossy audio compression algorithms include the MPEG-1 Layer-3 (MP3) and the MPEG-2/4 Advanced Audio Coding (AAC), which can generally achieve compression of audio by more than twenty times, while still delivering good sound quality. The other type of compression algorithms is the lossless, which essentially keeps every bit of information in the original audio data. It is known that state-of-the-art lossless audio compression algorithms can achieve about two times compression.

Lossy audio compression is mainly for general music consumptions, such as playing music from iPods, or listening to networked radio using mobile phones. In contrast, lossless audio compression is mainly used in high fidelity audio reproduction, archival of audio database, and more recently biomedical signal compression, such as lossless ECG compression [1]. In applications like lossless audio archival, the data are generally intended to be preserved for a long period of time, and therefore it is crucial that they can be correctly de-compressed without any loss in the future. Like the lossy counterparts, there are international standards for lossless audio compression and the most recent effort was published as ISO/IEC standards in 2006, namely the MPEG-4 Audio Lossless Coding (ALS) [2], and Scalable Lossless Coding (SLS) [3], respectively. In addition, there were also other open-source lossless audio compression algorithms such as the FLAC [4], which is also popular on the Internet.

In lossless audio compression, a generally adopted approach is to use a combination of linear prediction and entropy coding. A linear predictor first removes redundancy in the input data and generates a prediction residue, which is subsequently encoded by an entropy coder. Linear Predictive Coding (LPC) and Huffman/Golomb-Rice codes are the most commonly used tools for prediction and entropy coding, for example in [1][2][4]. In this paper, a new variation of the compression algorithm is introduced. Essentially, it utilizes an LPC predictor, a novel pre-processor for flattening the prediction residue, and a new entropy coder that is based on arithmetic coding with probability template scaling. The algorithm has been incorporated into the IEEE Standard for Advanced Audio Coding (IEEE 1857.2) [5] as part of the lossless audio compression tool. Evaluation results show that the compression performance of IEEE 1857.2 is better than MPEG-4 ALS, MPEG-4 SLS and FLAC.

2. IEEE 1857.2 LOSSLESS AUDIO COMPRESSION

The block diagram of the IEEE 1857.2 lossless audio compression system is shown in Fig.1, in which the top part illustrates the encoder, and the bottom part the decoder. In encoding, input audio samples are first processed by a predictor, which removes correlations in the input audio samples, and generates a prediction residue. The prediction residue then goes through a pre-processing step where the signal's

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

dynamic range is reduced, or in other words, the amplitude envelop of the signal is flattened. The flattened prediction residue is subsequently coded by an entropy coder into a lossless bitstream. In decoding, a reverse process is performed in which the lossless bitstream is entropy-decoded, post-processed (de-flattened), and losslessly-reconstructed to a decoded signal that is an exact replication of the original input audio.

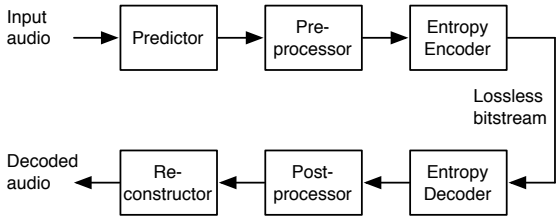


Fig. 1. IEEE 1857.2 lossless audio compression: encoding (top) and decoding (bottom)

2.1. Prediction and Reconstruction

The block diagram of the predictor is shown in Fig.2. Input audio samples are first segmented into frames of fixed length. Linear predictive coding (LPC) is then performed on each frame, with partial-correlation (PARCOR) coefficients computed through the Levinson-Durbin algorithm [6]. The PARCOR coefficients are quantized and sent as ancillary information in the lossless bitstream. The quantized PARCOR coefficients are also locally dequantized and converted to the tap coefficients of a linear predictor, which generates a prediction for each sample in the frame. The difference signal between an input sample and its prediction – the prediction residue – is output to the next processing stage.

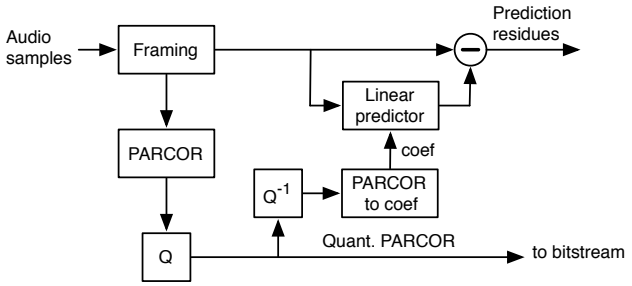


Fig. 2. Linear prediction

In reconstruction, the quantized PARCOR coefficients are extracted from the bitstream, dequantized, and converted to linear predictor coefficients, which are identical to those used in the encoder. The linear predictor generates a prediction, which is added to the decoded prediction residue to reconstruct the original input sample. The block diagram of the reconstructor is shown in Fig.3.

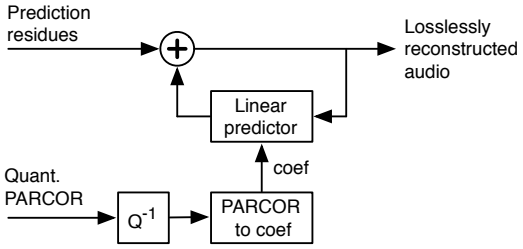


Fig. 3. Lossless reconstruction

2.2. Pre- and Post-processing

In IEEE 1857.2 lossless audio compression, it is designed that each audio frame can be decoded independently without using information from other frames. There are two benefits of this: firstly, compressed audio files can be decoded at a granularity of one frame interval, and secondly, bit errors occurred during transmission do not propagate beyond frame boundaries. As such, intra-frame linear prediction is performed on each audio frame as follows:

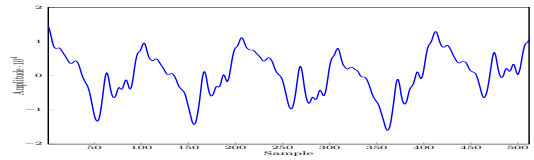
$$y(0) = x(0), \tag{1}$$

$$y(i) = \sum_{j=1}^{i} a_j^{(i)} x(i-j), \quad 1 \le i \le p-1, \tag{2}$$

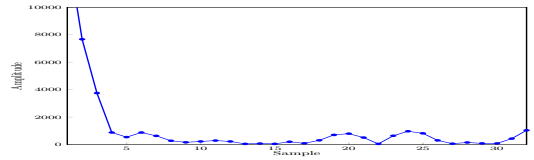
$$y(i) = \sum_{j=1}^{p} a_j^{(p)} x(i-j), \quad i \ge p, \tag{3}$$

where i is the time index to the samples in a frame, $x(i)$ and $y(i)$ denote input samples and their predictions in the frame, the maximal predictor order is p, and the coefficient at each predictor tap is denoted as $a_j, j = 1, \ldots, p$, the superscript in $a_j^{(i)}$ indicates that the coefficients are computed for predictor of order i. Since linear prediction is performed intra-frame, for samples at the beginning of each frame, the orders of the predictor used are very short, which result in larger prediction residues compared to the rest of the frame. An example of such effect is shown in Fig.4, where a segment of audio waveform is shown in the top panel, and the resulted prediction residue signal shown in the centre panel.

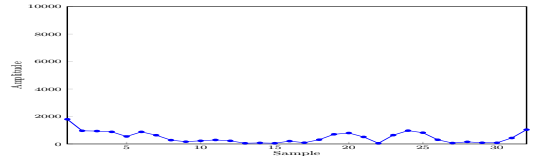
The large residues at the beginning of each audio frame are problematic, as they increase the dynamic range of the prediction residue considerably, which will force the following entropy coder to adopt a large alphabet size in arithmetic-coding, hence significantly increasing the computational complexity of entropy coding. In IEEE 1857.2, this problem is solved by pre-processing the prediction residues as depicted in Fig.5, where the residues are down-shifted by a number of bits to reduce their amplitudes. In this way, the amplitude envelop of the prediction residue is flattened, ensuring a smaller dynamic range of the signal, which can then be coded by the entropy coder using a smaller alphabet size. In the example in Fig.4, the flattened prediction residue is



(a) Input waveform



(b) Prediction residue



(c) Flattened residue after pre-processing

Fig. 4. Example of intra-frame prediction

shown in the bottom panel.

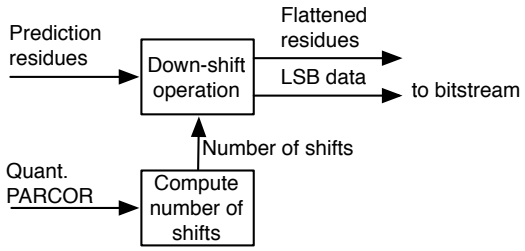


Fig. 5. Pre-processing prediction residue

In pre-processing prediction residues, the number of down-shifts for each residue is computed from the quantized PARCOR coefficients. In the Levison-Durbin recursive procedure [6], there is the following relationship:

$$E_i = (1 - k_i^2) E_{i-1}, \tag{4}$$

where $k_i, i = 1, 2, \ldots, p$ denotes the PARCOR coefficients, E_{i-1} and E_i are the energy of the $(i-1)$-th and i-th prediction residue, respectively. From Equation (4), the following relationship can be obtained:

$$\frac{E_i}{E_p} = \prod_{j=i+1}^{p} \frac{1}{1 - k_j^2}, \quad i = 0, 1, \ldots, p-1, \tag{5}$$

where E_p is the energy of the p-th prediction residue. In practice, only the first p residues in each frame need to be pre-processed, in which the number of down-shifts applied to each residue is derived from Equation (5) as

$$\left\lfloor \sum_{j=i+1}^{p} \log_2 \frac{1}{1 - k_j^2} + 0.5 \right\rfloor, \quad i = 0, 1, \ldots, p-1, \tag{6}$$

where $\lfloor \ \rfloor$ is the floor operator. In Equation (6), The logarithm terms inside the summation can be pre-computed and stored in look-up tables for fast computation.

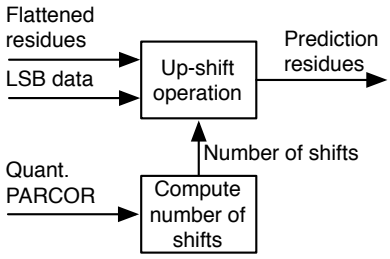


Fig. 6. Post-processing prediction residue

In decoding, a reverse bit-shifting operation is done at the post-processing step, as depicted in Fig.6.

2.3. Entropy Coding

The entropy coder is based on arithmetic coding. The block diagram of the entropy encoder is shown in Fig.7.

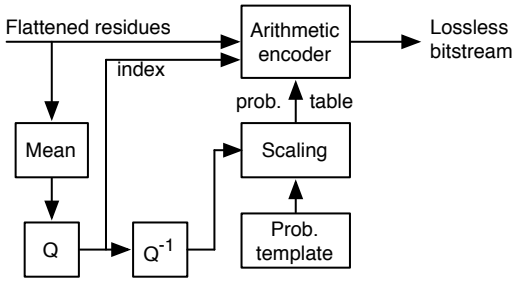


Fig. 7. Entropy encoder

For each frame of flattened prediction residue, the mean value of the frame, μ, is computed as

$$\mu = \frac{\sum_{i=1}^{N} |e(i)|}{N}, \tag{7}$$

where N is the length of the frame, and $e(i), i = 0, 1, \ldots, N-1$, are the flattened residues in the frame. For the purpose of coding, the value μ is logarithmically quantized to an integer, $index$, as

$$index = \lfloor \log_2 \mu + 0.5 \rfloor, \tag{8}$$

which is then locally dequantized to

$$\bar{\mu} = 2^{index}, \tag{9}$$

where $\bar{\mu}$ is the dequanitzed mean of the frame, which is used to scale a probability template to generate a probability table for arithmetic coding as follows:

$$p(s) = f(\lfloor s/\bar{\mu} + 0.5 \rfloor), \tag{10}$$

where s denotes symbols in the probability distributions, $f(s)$ is the probability template, and $p(s)$ is the probability table to be used by the arithmetic encoder.

The probability template is a table containing a set of probability density values, which are trained from a large amount of audio data. Fig.8 shows the probability template (solid line) adopted in IEEE 1857.2, which approximates a Gaussian function (mean -0.1, standard deviation 0.6, dotted line). It is found that using the trained table generally achieves a lossless compression performance 1% better than using the Gaussian function.

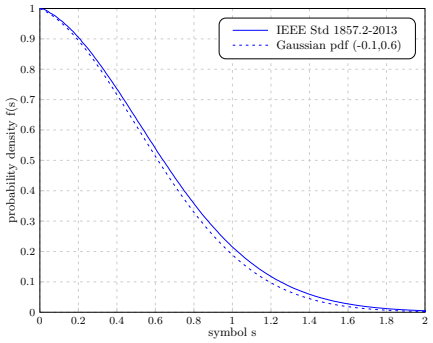


Fig. 8. Probability template for arithmetic coding

In IEEE 1857.2, the arithmetic coder is implemented using the fast algorithm in [7]. During the encoding of each frame, the parameter $index$ is first differentially coded, and then arithmetically coded. Following that, prediction residues in a frame are coded by the arithmetic encoder using the scaled probability table in (10). The block diagram of the corresponding entropy decoder is given in Fig.9.

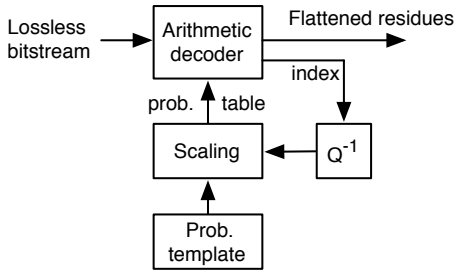


Fig. 9. Entropy decoder

3. PERFORMANCE EVALUATION

The performance of the IEEE lossless audio compressor was benchmarked against other standardized lossless audio coders, including the MPEG-4 ALS, MEPG-4 SLS, and FLAC. In performance benchmarking, the input source used is a 450-second long wav file with 48 kHz sampling rate and 16 bits sampling resolution. The wav file is a concatenation of 15 different, 30-second long audio clips, which are ripped from music CDs of different genres. The performance indicators used are compression rate, and encoding/decoding speed. Compression rate is computed by dividing the file size before compression by that after compression. The more the audio file is compressed, the higher is the compression rate. The encoding/decoding speeds are measured in terms of how fast the encoder/decoder processes relative to playing the audio clip in real-time. For example, if an audio clip is 30-second long, and it takes 2 seconds to compress it, then the compressor is said to run at $30s/2s = 15 \times realtime$. The performance evaluation was tested on a laptop computer with a 1.6 GHz Intel Core i5 processor and 4GB 1333 MHz RAM. The evaluation results are shown in the table below.

Table 1. Performance comparison of lossless audio coders

Lossless Audio Coders	Compress. Rate	Encoding Speed $\times realtime$	Decoding Speed $\times realtime$
IEEE 1857.2	2.18	21.7	38.4
MPEG-4 ALS	2.09	51.1	126.5
MPEG-4 SLS	2.04	8.3	8.4
FLAC	1.94	228.6	392.8

As can be seen from the table, the IEEE coder shows the highest compression rate, which is about 5% higher than MPEG-4 ALS, and 12% higher than FLAC. In terms of encoding/decoding speeds, FLAC is the fastest, followed by MPEG-4 ALS, IEEE 1857.2, and MPEG-4 SLS. This is as expected because the computational complexity of arithmetic coding in IEEE 1857.2 and SLS is higher than that of Huffman/Rice coding used in FLAC and ALS. Nevertheless, on a general-purpose laptop PC used in the benchmarking test, the IEEE compressor provides a decent encoding/decoding speed of about 20/40 $\times realtime$, which is more than sufficient for most applications.

4. CONCLUSION

This paper presents the lossless audio compression tool in the recently approved IEEE Standard for Advanced Audio Coding (IEEE 1857.2). Performance evaluation results show that the lossless compression performance of the proposed method is higher than MPEG-4 ALS, SLS, and FLAC. The encoding/decoding speeds of the IEEE compressor are about 20/40 times realtime on a general purpose laptop PC.

5. REFERENCES

[1] S-L Chen, G-A Luo, and T-L Lin, "Efficient fuzzy-controlled and hybrid entropy coding strategy lossless ecg encoder vlsi design for wireless body sensor networks," *Electronics Letters*, vol. 49, no. 17, pp. 1058–1060, 2013.

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

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

[4] "FLAC - Free Lossless Audio Codec," `http://xiph.org/flac/`, Oct. 2013.

[5] IEEE 1857.2-2013, "IEEE Approved Draft Standard for Advanced Audio Coding," Aug. 2013.

[6] John Makhoul, "Linear prediction: A tutorial review," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

[7] A. Moffat, R. M. Neal, and I. H. Witten, "Arithmetic coding revisited," *ACM Transactions on Information Systems*, vol. 16, no. 3, pp. 256–294, July 1998.