Audio Engineering Society

Convention Paper 7081

Presented at the 122nd Convention
2007 May 5–8 Vienna, Austria

Aspects of Scalable Audio Coding

Chris Dunn

Independent Consultant, London, UK

chris.dunn@scalatech.co.uk

ABSTRACT

Banded weight data is transmitted as side information within coded audio bitstreams in order to achieve psychoacoustically-appropriate shaping of quantisation noise. Methods of reducing the information overhead corresponding to weight data are discussed in the context of scalable bitplane coding. Two approaches to coding band weights are compared in terms of coding efficiency and error resilience. In the first, weights are coded as a block of data at the beginning of each frame, using a predictor and Golomb coding of weight prediction residuals to achieve high coding efficiency. This approach is compared to coding weights for bands as they become significant, with weight data distributed across each coded bitstream frame.

1. INTRODUCTION

Scalability has become an important aspect of low bitrate audio coding, particularly for multimedia applications where a range of coding bitrates may be required, or where channel bitrate fluctuates. Fine-grain scalability, where useful increases in coding quality can be achieved with small increments in bitrate, is particularly desirable.

While fine-grain bitrate scalability can be useful, it is important that scalable codecs maintain high coding efficiency with low computational complexity. Error resilience allowing coded bitstreams to be robust against transmission errors is also beneficial, particularly for wireless streaming applications. Bitplane coding is an approach that can meet these goals [1].

This paper addresses issues related to coding banded-weight side information in audio codecs, including techniques to minimise overhead at lower bitrates and realise a degree of error resilience. The discussion is particularly relevant to scalable bitplane codecs where optimal performance is required across a wide range of bitrates.

Two approaches to coding banded weights are evaluated. Weight prediction and block coding of prediction residuals at the beginning of each frame using Golomb-Rice codes is initially considered. This is compared to a technique termed significant weight coding where weights are coded as bands become significant, with weight data distributed across each coded bitstream frame. Significant weight coding is shown to achieve substantial improvements in coding efficiency at lower bitrates. Finally, listening test results are reported for a practical bitplane codec that uses significant weight coding.

2. BANDED WEIGHTS

Fig. 1 shows a generic audio encoding process where a time-domain input signal is transformed to the frequency domain before quantisation and frame packing to a coded bitstream. A psychoacoustic model determines a target noise shaping profile which is used to allocate bits to the transform coefficients such that quantisation errors are least audible to the human ear [2].

In general, perceptual sensitivity to nonlinearity changes as a function of frequency within each coded frame. This is indicated in Fig. 2 which shows the input magnitude spectrum for 1 frame of a harpsichord test signal, and the masked threshold associated with this signal. Non-constant spectral sensitivity to nonlinearity can be exploited in the coding process by spectrally shaping quantisation noise to match masked threshold as closely as possible. Typically this is achieved by approximating the masked threshold function for each frame with a set of banded weights, normalising (dividing) transform coefficients with the weights prior to quantisation, and transmitting weight data as side information in the bitstream to allow re-normalisation (scaling) at the decoder. The banded weights are often referred to as 'scalefactors'.

The width of each weighting band is usually set to increase with frequency in order to approximate the critical bands of the hearing process, and logarithmic quantisation with steps of a few dB allow the weights to be represented as an integer series. The lower trace in Fig. 2 shows the banded weights for the harpsichord frame, where in this example the total number of bands is equal to 32 and weights are quantised in 3 dB steps.

3. BLOCK-BASED WEIGHT CODING

3.1. Predictive Coding

Weight data is typically grouped as a single data block at the beginning of the frame (Fig. 3). Efficient weight coding is required in order to maximise the number of bits available to represent transform coefficient data, and hence maximise overall coding efficiency. Linear coding is a simple method of coding quantised weight data, often with reference to a global scaling factor which can be set to the average weight value for the frame, however linear coding can have significant redundancy which results in poor coding efficiency.

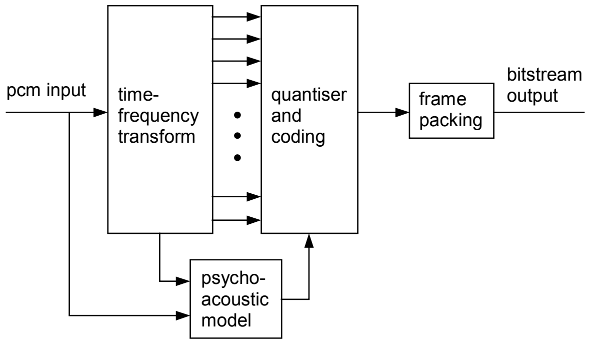


Fig. 1. Perceptual transform encoder for audio compression.

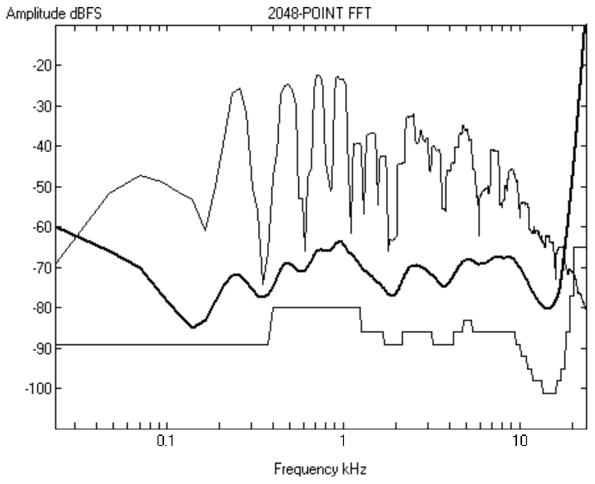


Fig. 2. Input magnitude spectrum (upper trace), corresponding masked threshold (middle bold trace), and banded weights (lower trace, vertically offset for clarity) for 1 frame of harpsichord.

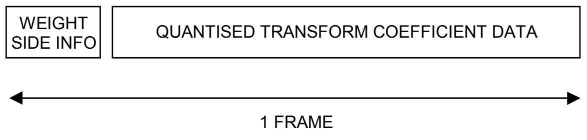


Fig. 3. Bitstream frame packing with block-coded weights.

Because masked thresholds change relatively slowly from band to band, frequency-domain prediction can be used to reduce the information overhead required to code weights. A simple first-order predictor is formed by coding each band weight relative to the previous band weight, in order of ascending frequency. This is equivalent to coding weight residuals $e(j)$ formed from the current and previous band weights $a(j)$,

$$e(j) = a(j) - a(j-1)\,. \qquad (1)$$

The lowest-frequency band weight $a(0)$ is directly coded as a reference weight. First-order prediction is equivalent to differential coding, and is an established approach to coding side information in audio codecs - see for example [3], and [4].

It is possible to further reduce the data rate required to code weights by increasing the prediction order. A second order predictor is obtained by coding residuals derived from the current and previous two band weights,

$$e(j) = a(j) - 2a(j-1) + a(j-2)\,. \qquad (2)$$

As with first-order prediction, the first band weight $a(0)$ is coded directly, while $a(1)$ is differentially coded with respect to $a(0)$. Simulations show that increasing the prediction order to second order results in a 10% reduction in average residual entropy compared to differential coding.

3.2. Golomb-Rice Codes

An efficient entropy coding method should be selected to code prediction residuals. While Huffman coding is used in MPEG-AAC [4, Sec 6.3], an alternative approach that offers reduced computational complexity at no loss to coding efficiency is Golomb coding of transformed residuals. Following second-order prediction, weight residuals have an approximately Laplacian two-sided amplitude distribution, which can be converted to a single-sided distribution using the following transformation:

$$\begin{aligned} e'(j) &= 2e(j) - 1, & e(j) > 0, \\ &= 2\left|e(j)\right|, & e(j) \le 0. \end{aligned} \qquad (3)$$

The resulting distribution is approximately geometric, which allows simple coding with Golomb-Rice codes using an approach similar to that adopted in lossless coding of sampled audio data [5] and image data [6].

For a Golomb code with parameter g, a non-negative integer e is coded as two components – a prefix $\lfloor e / g \rfloor$ coded in unary, followed by suffix $[e \bmod g]$ coded in binary [7]. A particularly simple form of Golomb code, sometimes known as Rice codes, occurs when $g = 2^w$ for some integer wordlength $w \ge 0$ - in this case e can be coded by removing the w least-significant bits from e, coding the remainder as a unary prefix, and appending w binary LSB's. For example, if $e = 9$ and $w = 2$, then the Golomb-Rice code for e is '00101' – here the prefix is '001' = 8, and the remainder is '01' = 1.

By selecting the Golomb-Rice parameter to closely match the transformed weight residual distribution, high coding efficiency can be obtained with low computational complexity. Furthermore, Golomb codes can efficiently code large residual values, avoiding weight clipping that can occur with linear weight coding.

Note that while Golomb codes have previously been shown to be effective for coding transform coefficient bitplane data [1], Golomb codes used to code banded weights will in general be computed independently of any Golomb codes used to code coefficient bitplanes.

3.3. RVLCs

While second-order prediction combined with Golomb-Rice codes achieves compact weight coding, the use of Golomb-Rice codes has a further benefit. If weight residuals are block-coded using Golomb-Rice codes implemented as reversible variable length codes (RVLCs), then coded weight data can be decoded in both forward and reverse directions [8], conferring the advantage of improved resilience to bit errors suffered during bitstream transmission [9]. The error resilience of band weight decoding can be critical for transmission environments with high error rates, since corrupt weight values have the potential to cause high-amplitude noise bursts at the decoder output.

Reversible decoding with RVLCs carries a small overhead relative to uni-directional decoding, since the final band weight must also be directly coded as the reference weight for reverse-direction decoding, and the length of the weight data block is also coded as side information. Typically this overhead is of the order of 20 bits per frame, equivalent to about 1 % of the data rate at 96 kbit/s.

When RVLC decoding detects bit errors in the weight data for a particular frame, the error(s) can often be isolated to a range of bands so that not all band weights have to be flagged as corrupt [9]. Compared to uni-directional block-based weight decoding where any bit errors cause all weights for the frame to be considered corrupt, this can result in a substantial reduction in perceptual impairment for a given bit error rate. Band weights that *are* determined to be corrupt can be interpolated from neighbouring bands and/or previous frames, or transform coefficients in these bands simply muted.

4. SIGNIFICANT WEIGHT CODING

While predictive block coding can reduce absolute band weight overhead, grouping weights together in a single block within each frame suffers the disadvantage of increasing relative overhead as the overall bitrate decreases. This can be a significant drawback for scalable codecs where high coding efficiency is required at lower bitrates.

An approach that can overcome this limitation is the use of an embedded psychoacoustic model [10]. Here transform coefficient weighting is derived from bitplane-coded coefficient data that has already been coded within a scalable bitstream, and does not require band weights to be explicitly coded within the bitstream. While this approach avoids the fixed overhead of block-coded weight data, with potentially improved coding efficiency at lower bitrates, it has two drawbacks. Firstly, the decoder requires a local psychoacoustic model in order to derive a masked threshold and determine the order of coded coefficients. This can significantly increase computational requirements at the decoder, particularly since accurate psychoacoustic models tend to be computationally intensive. A further drawback is that masked threshold is derived from bitplane-quantised coefficient data that is an approximation of the true input spectrum, which can result in inaccurate shaping of quantisation noise.

These disadvantages are overcome, while at the same time retaining the advantage of reduced overhead at lower bitrates, with *significant weight coding.* This technique exploits the fact that at lower bitrates only the most significant bitplanes coded in each frame are decoded, and the majority of transform coefficients remain insignificant - ie are decoded to zero [1]. Similarly, at lower bitrates many bands remain insignificant, containing only insignificant transform coefficients, and there is no requirement to transmit weight data for these bands until later in the frame. Fig. 4 shows how the average number of significant bands per frame increases with bitrate for a codec with 32 bands. In this example, on average only half of all bands are significant at 32 kbit/s, and the remaining band weights are not required to be transmitted within the bitstream until higher bitrates. Even at relatively high bitrates ~ 96 kbit/s it can be seen that on average a few bands remain insignificant.

Significant weight coding is implemented by coding a band weight immediately following the bitplane code that identifies the location of the first significant coefficient coded in each band, so that the decoder has only the information required to reconstruct all significant coefficients to the correct level. Fig. 5 shows the resulting bitstream structure, where band weight data is interleaved with transform coefficient bitplane data as significant bands are progressively identified.

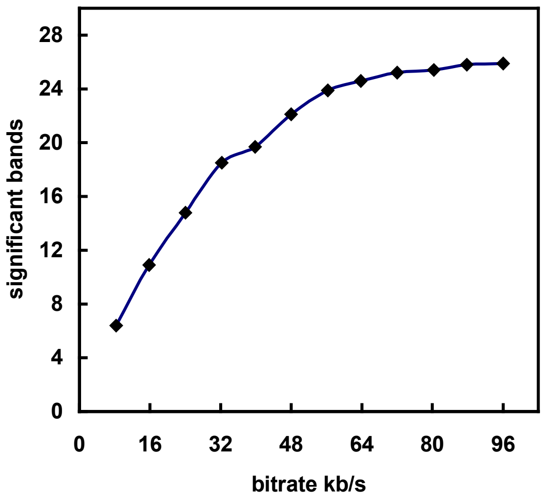


Fig. 4. Average number of significant bands identified in each frame as a function of bitrate for a codec with 32 bands.

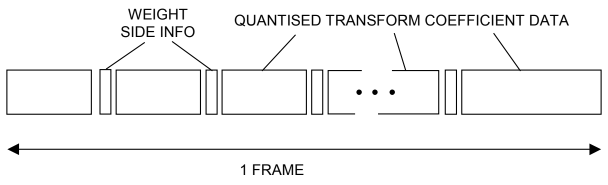


Fig. 5. Data packing for 1 frame of bitstream with significant weight coding.

As with block-coded weights, prediction may be used to reduce residual weight entropy prior to coding. However, because band weights are no longer coded in order of ascending frequency, lower-frequency band neighbours cannot be used to predict the current band weight. Instead weight residuals are formed from a first-order prediction of the previous band weight coded within the bitstream for the current frame. The first weight in each frame is coded directly as a reference weight.

Given a frame of audio transform coefficients $x(k)$ arranged in sign-magnitude format, a two-stage bitplane runlength coding algorithm [1] that incorporates significant weight coding includes the following steps:

- form two coefficient lists: a list of insignificant coefficients (LIC, initially containing all coefficients in the frame), and a list of significant coefficients (LSC, initially empty)
- code the most-significant bitplane level for all LIC members = $\lfloor \log_2 |x|_{\max} \rfloor$
- for each bitplane, beginning with the most significant bitplane:
    - code the significance map:
        - runlength code the positions of newly-significant LIC members, ie those coefficients whose MSB is located within the current bitplane
        - when a LIC member is found to be newly significant:
            - calculate the band index for the coefficient, output the band weight if the band is previously insignificant
            - output the coefficient sign
            - remove this coefficient from the LIC and add to the LSC
    - code refinement bits:
        - for all LSC members added in previous more-significant bitplanes, output the LSB corresponding to the current bitplane
- terminate coding when either the bit allocation for the frame is used or target coding resolution achieved.

4.1. Error Resilience

With significant weight coding, band weights and transform coefficient bitplane data are interleaved and coded in order of significance. In terms of error resilience this arrangement has the advantage that it is not necessary to discard all data within the frame when a bit error is detected in a particular bitplane; data in more-significant bitplanes can be retained and decoded to provide a lower-quality version of the frame. However the interleaved data arrangement makes it more difficult to protect sensitive weight data against bit errors, since bit errors in the coefficient bitplane data can propagate and corrupt band weight data.

A simple method of detecting errors in decoded significant band weights is to embed a CRC word within each bitplane, set to a value determined by the encoded (uncorrupted) band weight data contained within the bitplane. The CRC word is conveniently placed at the end of the bitplane (Fig. 6), so that the CRC generated from decoded weight values can be compared against the encoded CRC. When corrupt band weights are detected, either these weights are interpolated from neighbouring bands and/or previous frames, or the coefficients in these bands are muted, and decoding of the current frame terminates. Practical experiments indicate 16-bit CRC words are required in order to reduce the probability of misdetecting corrupt band weights, requiring significant overhead of about 9% of total bitrate at 64 kbit/s. Although simple to implement, this is a relatively inefficient approach to attaining error resilience, since on average a 16-bit CRC word adds a larger overhead than the band weight data the CRC protects within each bitplane. CRC-based error detection also suffers the disadvantage of increased bitrate granularity, since whole bitplanes must be decoded before the CRC word can validate the data contained within the bitplane.

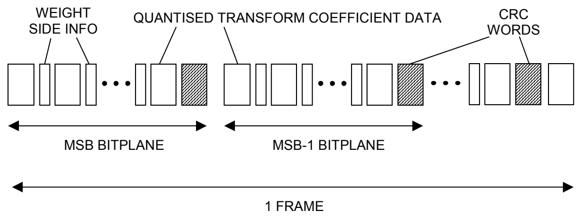


Fig. 6. Data packing for 1 frame of error resilient bitstream with significant weight coding and bitplane CRC words.

In Section 3.3 we discussed how RVLCs can be used to achieve a degree of error resilience with block-based weight coding. As described, significant weight coding where weight codes are interleaved with transform coefficient bitplane data cannot use RVLCs to facilitate bi-directional weight decoding. A compromise solution is to group weights for bands that will become newly significant in a bitplane at the start of that bitplane, and code using RVLCs. This arrangement requires additional side information for each bitplane, consisting of the directly coded final band weight for the bitplane, and the length of the RVLC-coded data block for the bitplane. In total this amounts to approximately 12 bits for each bitplane, equivalent to 6% overhead at 64 kbit/s and more compact than the CRC approach. Using RVLCs in this manner also has the advantage of not increasing average bitrate granularity compared to unprotected significant weight coding.

5. CODING EFFICIENCY

An experimental codec was constructed to compare the coding efficiencies of block-based and significant weight coding across a range of bitrates. The encoder design follows the generic structure outlined in Fig. 1, with a block-switched MDCT time-frequency transform similar to that used in MPEG-AAC [4]. With a frame length of 1024 samples, long-block frames use a single 2048-sample transform window, while short-block frames use 8 overlapping 256-sample windows. A custom psychoacoustic model is used to spectrally shape quantisation errors so as to be minimally audible, and accounts for most of the computational requirement in the encoder. Because psychoacoustic model calculations are confined to the encoder, decoder complexity is low. The quantisation stage features bitplane runlength coding [1]. The following results were obtained without implementing error-resilient source-coding techniques (ie bitplane CRC words and RVLCs were not used).

Table 1 compares the coding efficiencies of the weight coding techniques with long-block frames, by measuring average weight overheads at 3 bitrates. Each result was obtained by averaging data for 3 test signals with 44.1 kHz sample rates. At 64 kbit/s both approaches require about 5% of overall bitrate, however at 16 kbit/s significant weight coding requires about 12% less of the overall bitrate to code weight data.

For short block frames the potential improvements in coding efficiency can be substantial, as weight updates may be required several times within each frame. Table 2 records worst-case short-block overheads for 44.1 kHz sampled material with significant transient content. It is seen that the overhead reduction for significant weight coding is notable even at 64 kbit/s. At 16 kbit/s block-based weight coding can require most of the available bitrate, leaving few bits to code transform coefficients and resulting in audible 'dropouts' during transient sections. This does not occur with significant weight coding, where even at 16 kbit/s the majority of the overall bitrate is available to code coefficient data.

BITRATE kbit/s	AVERAGE LONG-BLOCK WEIGHT OVERHEAD %	
	BLOCK-CODED WEIGHTS	SIGNIFICANT WEIGHT CODING
64	5.2	4.9
32	10.5	7.3
16	20.9	8.6

Table 1. Average band weight overheads for long-block frames.

BITRATE kbit/s	WORST-CASE SHORT-BLOCK WEIGHT OVERHEAD %	
	BLOCK-CODED WEIGHTS	SIGNIFICANT WEIGHT CODING
64	24.3	15.9
32	49.2	20.0
16	75.0	22.6

Table 2. Worst-case band weight overheads for short-block frames.

Perceptual entropy coding efficiency was developed in [1] as a coding efficiency metric that is effective at revealing performance differences between quantisation algorithms. Average short-block perceptual entropy results are shown in Fig. 7 for the experimental codec with the two weight coding approaches considered. The results indicate a substantial performance advantage for significant weight coding at low bitrates.

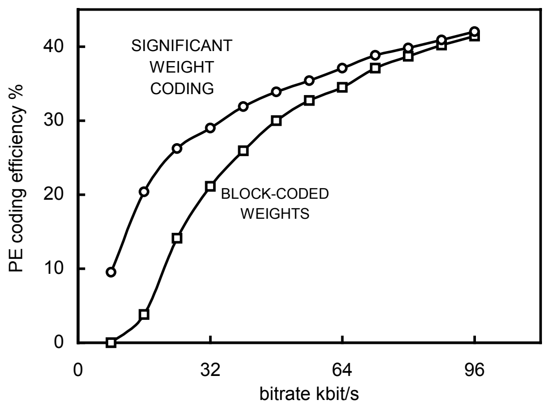


Fig. 7. Perceptual entropy coding efficiency for short-block frames with block-coded weights and significant weight coding.

6. SUBJECTIVE RESULTS

In this Section we report the results of subjective tests that compare the coding performance of the 2-channel experimental codec incorporating bitplane runlength (BPRL) coding and significant weight coding against publicly available reference codecs. A Win32 version of the experimental BPRL decoder and scalable bitstreams are available for demonstration purposes at http://www.scalatech.co.uk/download.htm

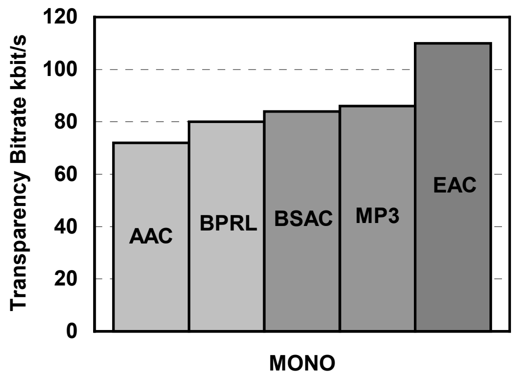
Informal but carefully controlled listening tests compared perceptual transparency bitrates achieved with the BPRL codec against two fixed-rate and two scalable reference codecs:

- MPEG-1 Layer 3 (MP3) - FhG mp3 codec - fixed bitrate
- MPEG-4 AAC Low Complexity Profile (AAC) [4] - Apple Quicktime Pro 6.5 - fixed bitrate
- Microsoft Embedded Audio Coder (EAC) [10] - scalable
- MPEG-4 Bit Sliced Arithmetic Coding (BSAC) [11], [12] - scalable.

A measure of transparency was obtained for each codec by averaging transparency bitrates across four demanding 44.1kHz-sampled test pieces. The tests were conducted using both single-channel and stereo material. Results for the experimental BPRL codec indicate average transparency bitrates of 80 kbit/s for mono signals, and 132 kbit/s for stereo material (Fig. 8). This represents an improvement in coding efficiency compared to fixed-bitrate MP3, and also the two scalable reference codecs. While the BPRL mono result is close to optimised AAC performance, the shortfall in stereo performance compared to AAC may relate to the absence of intensity stereo coding with the tested BPRL implementation.

An error resilient version of the BPRL codec with error detecting CRC words embedded in each bitplane was also tested (see Section 4 for a detailed discussion). With 16-bit CRC words, transparency bitrates were about 10% higher than the unprotected codec (86 kbit/s for mono material, 154 kbit/s for stereo signals).

Of particular interest was the low-bitrate subjective performance of the error resilient version of the BPRL codec incorporating significant weight coding, in comparison to the other scalable codecs tested. Since the BSAC and EAC codecs investigated did not include error-resilient design features, the comparisons were conducted under error-free conditions. The BPRL and BSAC codecs were judged to have similar performance at 64 kbit/s, but at 32 kbit/s the BPRL codec was generally preferred. While the EAC codec performed well at 32 kbit/s for some signals, other signals suffered from clearly audible coding artifacts. At the higher rate of 64 kbit/s, the BPRL codec was preferred to EAC. Sound file samples from these tests are available for demonstration purposes at http://www.scalatech.co.uk/



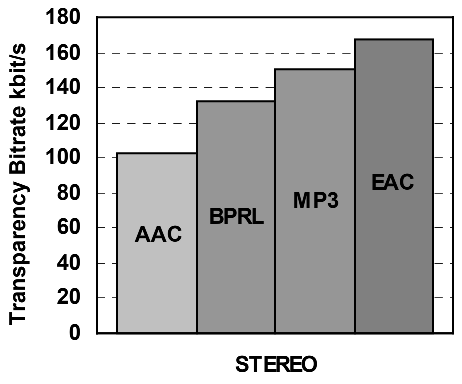


Fig. 8. Transparency bitrates for comparison codecs.

7. CONCLUSIONS

This paper has compared two approaches to coding banded-weight side information in scalable audio codecs. Block-based coding where weights are grouped together at the start of each frame was initially considered, where second-order prediction followed by Golomb-Rice coding of prediction residuals achieves compact coding. Error resilience can be achieved by using RVLCs to facilitate bi-directional decoding with only a small additional bitrate overhead.

Significant weight coding, where weights are distributed throughout the frame, is an alternative to block-based coding. For applications where error resilience is not critical, significant weight coding can achieve substantial coding efficiency advantages compared to block-based coding, particularly for short-block frames. Subjective tests with an experimental codec employing bitplane runlength coding and significant weight coding suggest transparency bitrates that are competitive with commercially-available fixed- and scalable-rate codecs.

A simple approach to achieving a degree of error resilience with significant weight coding is to embed a CRC word in each bitplane, however this carries a significant bitrate overhead and also increases bitrate granularity. A promising alternative strategy with a smaller overhead involves grouping weights required for each bitplane at the start of the bitplane and coding with RVLCs. Further work is required to establish the best overall approach to error-resilient weight coding for scalable codecs in realistic transmission environments.

8. ACKNOWLEDGEMENTS

This work was initially supported by the UK Department of Trade and Industry. Further support was received from the National Endowment for Science, Technology and the Arts.

9. REFERENCES

[1] C. Dunn, "Scalable Bitplane Runlength Coding," presented at the 120th Convention of the Audio Engineering Society, Paris, 20-23 May 2006, *J. Audio Eng. Soc. (Abstracts)*, vol. 54, p. 700 (2006 July), preprint 6749.

[2] J. D. Johnston, "Transform Coding of Audio Signals Using Perceptual Noise Criteria," *IEEE J. Select Areas in Communications*, vol. 6, pp. 314 - 323 (1988 Feb.).

[3] Y. Mahieux and J. P. Petit, "Transform Coding of Audio Signals at 64 kbit/s," Proc. Globecom 90, pp. 518 – 522 (1990 Nov.).

[4] M. Bosi et al, "ISO/IEC MPEG-2 Advanced Audio Coding," *J. Audio Eng. Soc.*, vol. 45, no. 10, pp. 789 – 812 (1997 Oct.).

[5] T. Robinson, "SHORTEN: Simple Lossless and Near-Lossless Waveform Compression," Cambridge University Engineering Dept. Technical Report CUED/F-INFENG/TR.156 (1994 Dec).

[6] M. J. Weinberger, G. Seroussi, and G. Sapiro, "LOCO-I: A Low Complexity, Context-Based, Lossless Image Compression Algorithm," Data Compression Conference (DCC) 1996.

[7] S. W. Golomb, "Run-length Encodings," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 399 - 401 (1966 July).

[8] J. Wen and J. D. Villasenor, "Reversible Variable-Length Codes for Efficient and Robust Image and Video Coding," Proc. 1998 IEEE Data Compression Conference, Snowbird, Utah, pp. 471 - 480 (1998 Mar.).

[9] R. Sperschneider et al., "Error Resilient Source Coding with Differential Variable Length Codes and its Application to MPEG AAC," 112th AES Convention (2002 May, preprint 5555).

[10] J. Li, "Embedded Audio Coding (EAC) with Implicit Auditory Masking," Proc. ACM Multimedia 2002, Nice France (2002 Dec.).

[11] S. H. Park et al., "Multi-Layer Bit-Sliced Bit Rate Scalable Audio Coding," presented at the 103rd Convention of the Audio Engineering Society, New York, Sep. 1997 (preprint 4520).

[12] ISO/IEC JTC1/SC29/WG11 N2803, "MPEG-4 Audio Version 2 (Final Committee Draft 14496-3 AMD1)", Vancouver, Canada (1999 July).