SUBBAND BASED MPEG AUDIO MIXING FOR INTERNET STREAMING APPLICATIONS

Patrick De Smet and Thomas Vander Stichele

Ghent University,
Dep. Telecommunications and Information Processing (TW07V),
St-Pietersnieuwstraat 41, B-9000 Gent, Belgium
tel: +32 9 2643416, fax: +32 9 2644295
e-mail: pds@telin.rug.ac.be, thomas@urgent.rug.ac.be

ABSTRACT

In this paper we investigate a special-purpose application of MPEG-1 layer II audio streaming. First, we discuss how two or more already coded MPEG-audio bitstreams can be manipulated and mixed within the coded subband domain by using an appropriate algorithm for combining and recalculating the bit allocation. Next, we will show that very significant speedups can be obtained compared with mixing in the temporal domain. We will also illustrate that, at the same time, the system can act as a bit rate scaling device, using one of the bit allocation-combination algorithms which we implemented. The results obtained with these different algorithms are briefly discussed and compared. Finally, we report on how the developed software was integrated into an internet audio streaming system, which is now able to allow simple, yet efficient, real-time mixing of several MPEG-audio coded signals.

1. INTRODUCTION

The MPEG-1-audio compression standards have been well studied during recent years and several software and hardware implementations have made it a very popular standard for compression, coding, distribution and transmission of high quality audio signals [1][2][3]. However, when manipulating such compressed data streams, e.g. for audio broadcasting or internet streaming applications, several tools or functionalities still are either not available, or have only been partially studied and implemented.

Large internet audio archives are typically compressed and stored on electro-magnetical or optical storage media. Consequently, when volume and/or mixing control operations (e.g. mixing or overlaying, and/or fading in and out) are desired, playing or transmitting the combination of several audio streams is a very important, but far from trivial task. Although several MPEG-audio (de)coding tools are widely available, tools for flexible and efficient mixing are scarce.

Also, for several applications, e.g. internet streaming, it is often desirable to be able to transmit the audio information according to various compression vs. quality trade-offs, i.e. flexible and efficient bit rate scaling functionalities are needed. For example, a signal is typically archived as a high quality (layer II or III) 192 kbit/s file on a harddisk or a CD-subsystem, but needs to be transmitted over a 128 kbit/s TCP/IP connection.

In this paper we focus on how a set of efficient MPEG-1 layer II mixing (and bit rate scaling) algorithms can be implemented. First, we will show that huge mixing speedups can be obtained by using only partial decoding and recoding in the subband domain. A crucial step is the fast recalculation of the bit allocation information for the signal (after mixing in the subband domain). Hence, next, we summarize (results for) several different bit allocation recalculation algorithms that we have studied. We will show that real-time mixing of up to 3 compressed audio signals can easily be obtained on a low-end PC-system while high quality coding of the signals is retained. Finally, we briefly discuss the prototype mixing and streaming system which we developed.

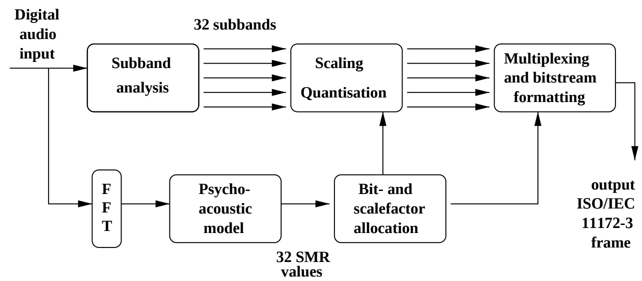

Fig. 1. MPEG-1 encoder structure (layer I and II).

2. MPEG-AUDIO CODING

The general MPEG-audio (layer I and II) coding scheme is illustrated in figure 1. The subband analysis component filters the input signal into 32 equally spaced subbands [1][2][3]. These subbands are then scaled, quantised, formatted and coded into a frame based output bitstream. The quantisation is controlled by the bit allocation and scaling component which determines how many bits will be asigned to each subband. By simulating the psycho-acoustic properties of the human hearing system, the bit allocation routine can decide which subbands should be given the majority of the available bits. The subband analysis filtering, bit allocation and psycho-acoustic modelling are highly computational tasks [4][5].

Figure 2 illustrates the general decoding scheme; it basically inverses all the encoder steps. Obviously, compared to the encoder, the computational complexity of the decoder is significantly smaller; most of the decoder computation time is used by the synthesis filter bank.

The popular layer III format (mp3) uses additional advanced techniques such as noise shaping, Huffman coding, bitreservoir

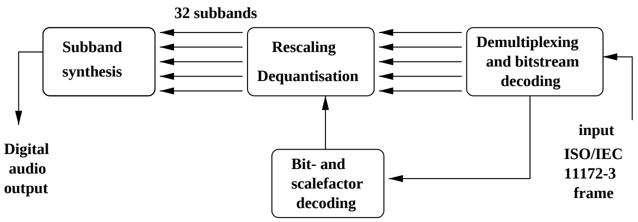

Fig. 2. MPEG-1 decoder structure (layer I and II).

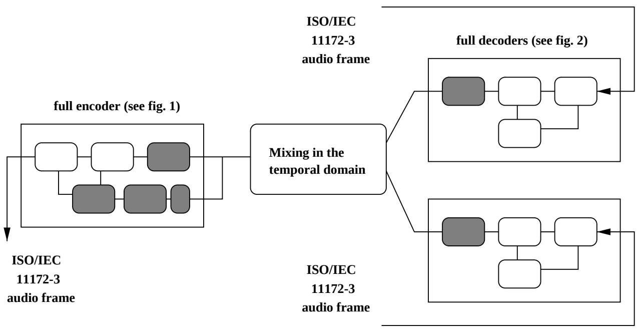

Fig. 3. Mixing and cross-fading of two coded bitstreams using transcoding (the CPU-intensive components are shown filled).

management, window switching, and MDCT coding to improve the layer I and II coding principles; see [1][2].

3. AUDIO MIXING AND CROSS-FADING

3.1. Transcoding and mixing

The most straightforward way to obtain a mixed coded audio stream is illustrated by figure 3. Using this method, each of the coded input signals is fully decoded and scaled according to the defined volume changes (user defined fading in or out, etc.). The signals are then added (in the temporal domain) and fully recoded. Obviously, the computational complexity of this method is very high. Also, recoding can introduce or amplify (quantisation and/or perceptual) errors due to the chained decoder-encoder setup of the system, e.g. due to the use of inaccurate psycho-acoustic models.

3.2. Subband mixing (layer II)

In this section we discuss how high speed mixing can be implemented without fully decoding the layer II coded audio streams. In section 6 we discuss how layer I and layer III (mp3) streams can also be used in the mixing process. Generally speaking, layer II (at the upper part of its bit rate range) has also been accepted as a very efficient complexity vs. quality trade-off. Additionally, since our initial aim was the development of a low cost system for streaming high quality audio on the university LAN and the BELNET research network, we decided that layer II would be an adequate choice.

In this paper we only focus on the situation where two audio streams are being mixed, obviously this can easily be extended to any number of streams (if the necessary amount of computational power is available and the issues discussed in section 4 are handled appropriately).

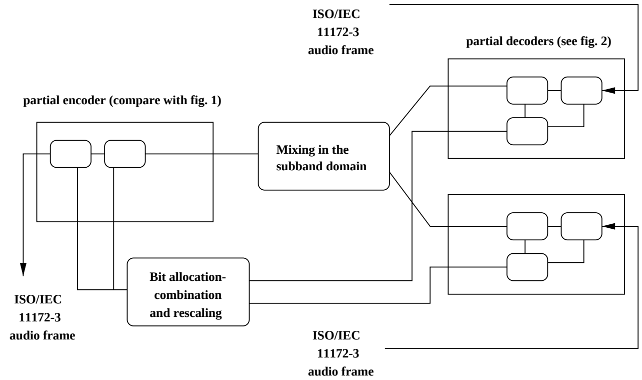

Fig. 4. Mixing and cross-fading two coded bitstreams in the subband domain (compare with figure 3).

As is shown in figure 4, the two partial decoders only reconstruct the subband domain samples which are then added and recoded into the new bitstream. The CPU-intensive components, see figure 3, which can be omitted include the subband synthesis filterbanks, the analysis filterbank, the FFT and psycho-acoustic model, and the normal, full search bit allocation.

Obviously, an important component of the proposed system is the bit allocation-combination and scaling module; see figure 4. This is discussed in the next section.

4. BIT RATE SCALING

A remaining problem when applying subband mixing is that the required bit rate of the mixed stream is (almost) always higher than the bit rate of the partially decoded input streams. This is due to the fact that we neither can nor wish to determine the psycho-acoustic properties of the (original) audio signals. Hence, mixing the partially decoded input bitstreams will require an independent, bit rate constrained re-evaluation and combinatation of the bit allocation sets.

Remark that in fact this is a bit rate scaling problem; the bit rate of the coded signal —the input MPEG-audio signals after subband mixing— has to be reduced to a lower level that can be handled by the transmission channel — the single output bitstream. Here, bit rate scaling (in the coded domain; see also [6]) corresponds to lowering the bit allocation, and thus the available number of quantisation steps for each of the 32 MPEG-subbands, while trying to spread the resulting added quantisation noise over the subbands in a computationally efficient and intelligent way.

In [7] we implemented five different algorithms to do this; the algorithms developed by Nakajima et al. [6] partially served as a basis. However, we slightly adapted these algorithms, and also developed some new variations in order to retain very high processing speeds. Also, we only discuss results for mild bit rate scaling functionalities, i.e. the (higher) bit rate needed by the mixed signal was reduced, and we only investigated down scaling from 192 kbit/s to 128 kbit/s (see section 3.2; design goals). For fully functional bit rate scaling additional algorithms are being considered.

The first two algorithms have the following initialisation step: the maximum bit allocation for each subband over all input streams is determined and for each non-zero allocated subband the allocation is increased with one allocation step.

After this initialisation, algorithm 1 then proceeds iteratively; it decreases the bit allocation of the highest subband until the subband is removed or the bit rate constraint is satisfied.

The second algorithm first makes single step decreases over all subbands before repeating the same procedure.

The third algorithm is similar to the second one, but the initial bit allocation in each subband is set to the value used by the input stream with the highest sum of subband sample values (in each subband being considered).

Algorithm 4 is similar to algorithm 3, but considers the sum of absolute values of the subband samples.

Finally, algorithm 5 does not consider bit allocation steps but initially calculates which bit allocation should be chosen in order not to introduce quantisation errors bigger than the accumulated error of the combined input streams; this method considers the number of possible quantisation levels rather than the bit allocation values. The down scaling step of algorithm 5 is identical to the one used by algorithm 2.

A more detailed explanation of all the algorithms mentioned above is available in [13].

5. SIMULATION RESULTS

The initial MPEG-codec source code [14] was adapted, extended and compiled on a standard MMX-200 Mhz pentium PC running Linux (kernel version 2.0.36).

Using the proposed subband domain mixing procedure, the total mixing time for two audio sequences could be reduced by a factor 8 to 12; enabling the mixing of up to three layer II files in real time.

Due to a lack of space we can not report all bit allocation-combination comparisons and simulation results that we have obtained; more results are reported on the WWW: see [13]. These results include comparisons of the obtained bit allocations for dual input 192 kbit/s to mixed 192 kbit/s, dual 128 kbit/s to mixed 128 kbit/s, and dual 192 to mixed (and additionally bit rate scaled) 128 kbit/s. Additionally, in [13] we also report on bit allocation results for a transcoded mix compared with an initial time domain based and afterwards coded mix, and differences obtained when using two different psycho-acoustic models.

However, figures 5 and 6 illustrate results for dual 192 kbit/s input to 192 kbit/s output mixing. Figure 5 illustrates the average bit allocations of the mixing algorithms and the result obtained when using the transcoding method (see figure 5: "trans" results).

Figure 6 illustrates the average quadratic difference between the bit allocation of the transcoded signal and each of the implemented mixing algorithms.

As can be seen from these figures, four of the algorithms perform quite well. Algorithm 1 does not work well, but this corresponds to the bandwidth cut-off nature of the algorithm. Algorithm 5 tends to favour the higher (less perceivable) subbands too much. Further fine tuning should improve this.

Finally, simple subjective listening tests have indicated that (when using algorithm 4 and) depending on the signals being recoded, often only after training noticable differences can be perceived between the transcoded and the subband mixed signals. For practical applications as discussed below (see section 8), high quality audio can always be retained.

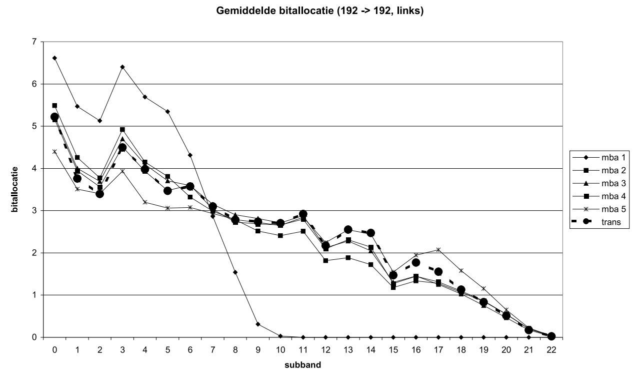

Fig. 5. Average bit allocation results when mixing two pop-songs with the different bit allocation-combination algorithms and the transcoding method (1000 frames).

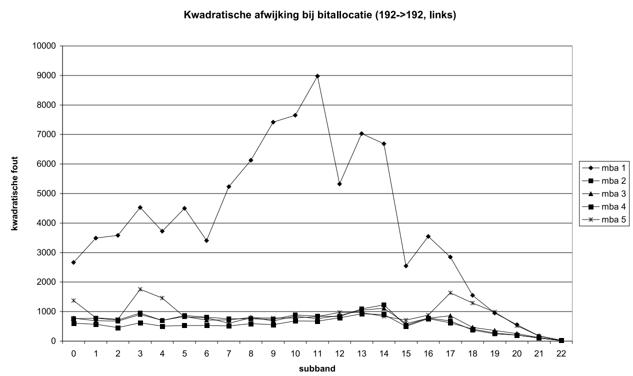

Fig. 6. The average squared bit allocation error of the implemented algorithms with respect to the transcoding bit allocation results.

6. MIXING WITH LAYER I AND III

The system that was described above mainly focusses on efficient layer II (re)coding. However, we have also implemented functionalities for translating layer I and layer III frames into layer II frames. Some of the issues discussed below will require further research.

6.1. Layer I

Although layer I has found very little use in practical applications, we have developed some routines which also enable the fast recoding and mixing of layer I audio frames [8]. However, we have only implemented simple procedures to translate the layer I bit allocation values (layer I uses a direct linear quantiser with symmetric zero) into the bit allocation tables used in layer II (see [1]; tables B2.a–B2.d restrict the number of subbands and the quantisation possibilities, but add "grouping"). These translation algorithms are being improved further.

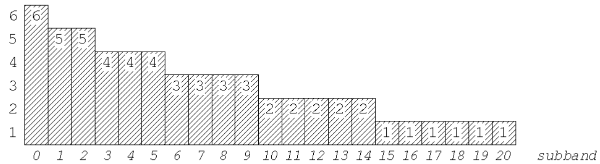

Fig. 7. Fixed bit allocation used for layer III to layer II translation (192 kbit/s).

6.2. Layer III

In order to enable mixing with layer III files the following procedure was implemented [7]. First, the mp3-bitstream is decoded until subband domain samples are obtained (this requires fast Huffman decoding and IMDCT calculation etc.). Then, the subband domain samples are mixed, and a fixed bit allocation, as is e.g. illustrated in figure 7, is used for all the layer III frames.

7. INTERNET STREAMING

In order to stream the coded audio onto the internet we decided to use an adapted version of the Icecast HTTP based streaming tool [9].

This tool is a quite popular, free and OpenSource counterpart of the commercial Shoutcast system [10]. Icecast streaming can be used with several MPEG-audio players as e.g. FreeAmp, XMMS, mpg123, Xaudio, WinAMP, Sonique, MacAMP and SoundJAM.

An obvious disadvantage of our current prototype system is that it still requires LAN, xDSL or cable modem network access, in order to receive the high bit rate and high quality audio signal. To enable reliable playback of the audio stream it is often also a good idea to increase the buffersize of the player tool, so that peeks in network load and jitter can be compensated for. Reduction of the required bandwidth could be obtained by further down scaling of the bit rate (see section 4). But, obviously, this would also have an impact on the audio quality. Indeed, preliminary experimental results have confirmed that the use of simple yet efficient algorithms for bit rate scaling of mixed signals, without any recalculation of the psychoacoustics of the combined signals, causes additional negative coding effects *if* the signals have considerably different psycho-acoustic properties *and* the target bit rate of the mixed signal is significantly lower; i.e. the bit rate vs. quality trade-of curve degenerates faster w.r.p. to the normal (trans)coding approach.

8. THE PROTOTYPE MIXER AND STREAMER

The software was developed for an audio broadcasting and internet streaming system. It is being put to use by the student radio channel at Ghent University; see [11]. Currently, RealAudio G2 (TM) streaming technologies are used to provide listeners with simple, i.e. non-mixed, concatenations of medium quality audio signals. In the near future high quality MPEG-1 audio mixing and streaming functionalities will also be provided.

The current prototype system is a standard PC with a number of harddisks containing the already (layer II and III) encoded songs, jingles, etc. The mixing functionalities overlap and add the jingles at the end and the beginning of songs, and can also shape the volume of the songs as they fade in or out.

Currently, the prototype system can be listened to by accessing the WWW: see [11] or [12]. For future reference, we recommend visiting our WWW page [13] for more up-to-date information.

9. CONCLUSION

In this paper we have discussed the subband domain based mixing and bit rate scaling of MPEG-audio signals. We have shown that using various different algorithms for recalculation and combination of the input bit allocation values, the total mixing time for two already coded audio sequences can be significantly reduced, enabling the mixing of up to three layer II files in real time on a 200 Mhz PC. We briefly outlined how the proposed subband mixing implementation was integrated into an internet audio streaming server, and indicated some possibilities for future research.

10. REFERENCES

[1] ISO/IEC 11172-3, "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5Mbit/s", 1993.

[2] P. Noll, "MPEG Digital Audio Coding", IEEE Signal Processing Magazine, pp. 59-81, September 1997.

[3] "Audio and video digital radio broadcasting systems and techniques", Proc. of the 6th Tirennia International Workshop on Digital Communications (1993), Elsevier Science B.V. 1994.

[4] P. De Smet, "Performantie-evaluatie en DSP-implementatie van een real-time MPEG-audio encoder" (Performance evaluation and DSP-implementation of a real-time MPEG-audio encoder), student thesis, Ghent University, 1995.

[5] Hyen-O Oh, Sung-Youn Kim, Dae-Hee Youn, Il-Whan Cha, "New Implementation Techniques Of A Real-Time MPEG-2 Audio Encoding System", proc. of ICASSP'99, 1999.

[6] Y. Nakajima, H. Yanagihara, A. Yoneyama, and M. Sugano, "Mpeg audio bit rate scaling on coded domain data," in *Proc. of ICASSP'98*, vol. VI, pp. 3669–3672, May 1998.

[7] T. Vander Stichele, "MPEG-audiocompressie voor multimedia-netwerktoepassingen" (MPEG-audio compression for multimedia network applications), student thesis, Ghent University, 1999.

[8] W. Demeulenaere, "MPEG-audiotranscodering voor multimedia-netwerktoepassingen" (MPEG-audiotranscoding for multimedia network applications), student thesis, Ghent University, 2000.

[9] `http://www.icecast.org/`

[10] `http://www.shoutcast.com/`

[11] `http://urgent.rug.ac.be/`

[12] `http://cleo.rug.ac.be:8000/mix/`

[13] `http://telin.rug.ac.be/~pds/` in `papers/icassp2001/`

[14] C source code tarball `mpegaudio.tar.gz` available at `http://www.funet.fi/pub/sci/audio/mpeg/`