COMPRESSION OF HIGH-QUALITY AUDIO SIGNALS USING ADAPTIVE FILTERBANKS AND A ZERO-TREE CODER

Y. Karelic and D. Malah

Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa 32000, Israel
E-mail: yairk@tulip.technion.ac.il

ABSTRACT

In this work we present a subband coding system for compressing high-quality audio signals. Two alternative subband decomposition schemes are considered: One consists of time-varying analysis and synthesis filterbanks that adapt their structure to the time-varying properties of the input signal; the other is based on decomposition of each block of samples separately. Both schemes are based on the *wavelet packets* theory and the *best-basis selection* algorithm introduced in [1]. The coder and decoder used in the system are based on the *zero-tree* algorithm [2,3] which was originally developed for coding still images and is adapted here for coding audio signals.

The performance of the proposed system is shown to be superior to the MPEG-Audio layer I standard [4], achieving an improvement of up to 10.6dB in segmental SNR at output bit rate of 64kbit/sec and up to 6.5dB at output bit rate of 128kbit/sec, for a variety of music signals. However, it turns out that most of the improvement is due to the zero-tree coding algorithm.

I. SUBBAND DECOMPOSITION

Introduction

In recent years, the most popular methods for coding high-quality audio signals are based on variants of subband coders. These methods differ in the way in which the frequency scale is partitioned into bands and in the way the information in each band is quantized and coded. One way of decomposing the signal into frequency bands is by using a uniform bandwidth filterbank (as used in the MPEG-Audio layer I standard). More recent, wavelets theory led to the use of equal-Q bands, for which the bandwidth of a band increases linearly with its center frequency. A third method that emerged from that theory is the use of a wavelet-packet decomposition which allows a larger variety of filterbank structures, including the above mentioned decompositions.

Best-Basis Wavelet Packets

When using these filterbanks for the compression of signals, it was observed that some of the structures lead to better quality reconstructed signals than others. Therefore, we attempted to find the filterbank structure that will give the best results. This is done adaptively in the proposed coder by using the best-basis selection algorithm [1].

The algorithm is used to select the best basis (for a given criterion) among a library of orthonormal bases generated by the wavelet packets library. The wavelet packet decomposition of a signal is a multiresolution decomposition and its coefficients can be organized as a full binary tree allowing a fast search algorithm that contains comparisons only between adjacent levels in the tree. The search for the best basis is done with the aim to minimize a cost function. An appropriate selection of a cost function will cause the selected basis to indeed give the best results for a given application. In [1] an additive cost function is used because it enables a faster search for the best basis while in [5] it is suggested to use non-additive cost functions while using the same searching algorithm to give a near best basis selection. The cost function used in [1] for a sequence $\{x_i\}$ is the *entropy,* defined as:

$$M(\{x_i\}) = -\sum_i p_i \ln p_i \quad ; \quad p_i = \frac{|x_i|^2}{\sum_j |x_j|^2}.$$

We examined several cost functions including the entropy and some of those mentioned in [5]. The one that gave the best results was the '$\log_2$' cost function defined as $M(\{x_i\}) = \sum_{i:x_i \neq 0} \lceil \log_2 x_i \rceil$, having the meaning of counting the number of bits needed to represent each coefficient as an integer.

Adaptive Time-Frequency Space Decomposition

The nonstationary nature of audio signals causes the best-basis to change with time and as a result the desired

filterbank structure should also vary in time. That implies that in order to achieve good compression it is better to divide the input signal into short 'stationary' segments and to find for each segment the best basis. Each segment is then coded and the wavelet packet tree (or filterbank structure) is also sent as side information. Using small segments improves the matching between the selected best-basis and the signal but, since the coding of a wavelet packet tree may demand a large number of bits, only a small amount of the available bits will remain for coding the coefficients. Large segments will cause the amount of side information to be comparatively negligible but the best-basis will fail to match the signal. Therefore, the selection of the segment size must take these two considerations into account.

Post Filtering

When dealing with the input signal in a 'prolonged' manner (we denote this as a "prolonged decomposition") we encounter a problem when an adaptive structure filterbank is used. The problem is the result of treating the analysis filterbank in a different way than the synthesis filterbank. This is because that in order to select the best basis we have first to calculate all the coefficients at all levels - that means we always calculate the coefficients using a full binary tree filterbank. In the synthesis part, on the other hand, we use a partial tree according to the tree that was selected by the best basis selection algorithm. When a pair of branches is generated, the filters in the new branches do not have the correct initial state inputs needed to produce valid output values, and therefore they produce errors for a certain period after the switching between different structures. In order to solve this problem we can look at it in a different way: the use of a different filterbank structure for each segment is equivalent to using a full tree filterbank with *time-varying filters*. This is because we can build a full-tree filterbank by completing the missing branches in the tree with filters that have a transfer function of a pure delay (this is done both in the analysis and synthesis filterbanks). As a result we get that the adaptive structure filterbank is equivalent to a fixed structure filterbank with time-varying filters. A problem arises, however, from the switching between different sets of filters from time to time. The switching causes undesired transients in the output signal. The transients occur because, during the transition interval, the analysis/synthesis stages do not have the perfect reconstruction property (when using wavelets) or the near-perfect reconstruction property (when using linear-phase QMF filters), which is a very important characteristic of the system.

To obtain perfect (or near-perfect) reconstruction filterbanks, even during the transition periods, one must

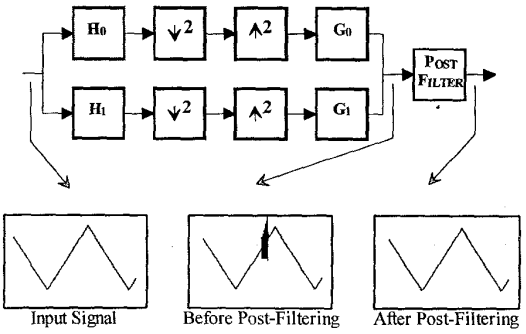

Fig. 1: An example for the operation of a post filter at the output of a time-varying QMF stage

eliminate these undesired transients. Several techniques which achieve this goal were reported [6,7,8]. In the proposed system it is done by using a post-filtering technique suggested in [9] and extending it for the multistage binary-tree filterbanks used in our system. The post-filter is a time-varying filter that affects the output signal during the transient periods only. A simple example for the operation of a post filter is shown in Fig. 1. A triangular wave signal is inserted into an analysis and synthesis stage whose analysis and synthesis filters are switched at a certain moment from pure delays to another perfect-reconstruction set (as happens when a branch is generated). As a result a transient occurs at the output of the stage until it returns to its steady state. The post filter removes the switching effects from the output signal to form a perfect reconstruction time-varying analysis/ synthesis stage. The design of the post-filter is based on the fact that the stage has the perfect reconstruction property so that we can describe the desired output samples during the transition period using the input samples. The post filter time-varying impulse response depends on the two sets of filters and on the switching point (odd or even sample).

The theory of a single stage post-filtering can be extended to multiple-stage post-filtering very easily because of the perfect reconstruction property of each stage. Assume a filterbank whose structure is of a two-level binary tree. If we apply a post filter to the inner level stages (two post-filters are needed) then the inner level behaves as a pure delay of N-1 samples (N is the length of each of the filters), as seen by the outer level. Therefore the delay can be moved to the synthesis filters of the outer level (near the synthesis filters) thus generating an equivalent single stage filterbank with a set of filters having $N+2(N-1)=3N-2$ taps; $2(N-1)$ of them are zeros. From this point on, the design of the outer level post-filter is just the same as for the inner one. As a result,

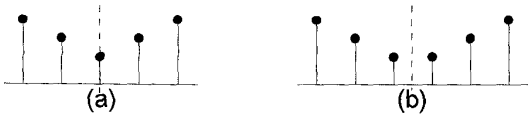

Fig. 2: Symmetric expansion: (a) Full-sample; (b) Half-sample.

by using a post-filter at the output of each level, multiple-stage adaptive filterbanks are designed without having any reconstruction error (in the absence of quantization) or having only minor errors when using near-perfect reconstruction QMF stages.

Segmental Decomposition

As already mentioned, another way to decompose the input signal is to partition it into segments and apply the analysis/synthesis filterbanks on each segment separately. If we find a way to reconstruct each segment perfectly then we can reconstruct the whole signal; this time without having any transients and therefore eliminating the use of post filters. Since each segment is treated independently, a different filterbank structure can be used for coding each segment and the selection of the structure that suits the segment is done by using the best basis selection algorithm as explained earlier.

In order to split a finite length signal into high and low frequency bands (as in a QMF stage) and then to reconstruct it perfectly, there is a need for information about the signal outside the relevant interval. Since we treat each segment independently there is a need to expand the signal, either periodically (cyclic expansion) or by mirroring (symmetric expansion). The advantage of using the symmetric expansion is that the generated signal to be decomposed is continuous thus eliminating undesired edge effects in the low frequency band. There are several types of symmetric expansions. The one we chose is the 'half-sample' symmetric expansion (see Fig. 2) since it assures that the signals at the high and low frequency bands are symmetric. This allows the use of segmental decomposition when dealing with binary-tree filterbanks. The segmental decomposition causes some inefficiency in coding the signal because edge effects are present in the high frequency band and therefore many bits may be needed for coding the coefficients at the segment boundaries. This implies that, for coding independent segments, longer segments will have less energy leakage; thus leading to a better reconstructed signal under the same bit rate restriction. That can be seen in Fig. 3 - the selected best-basis for a low-frequency (10Hz) sine wave in two cases: (a) A short segment (1024 samples); (b) A long segment (4096 samples). Note that for the long segment the best-basis gives the expected tree while for the short segment some energy leaks to higher frequency bands .

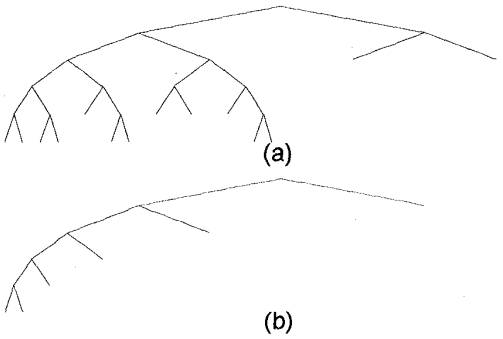

Fig. 3: The selected best-basis for a low frequency sine wave: (a) For a short segment (1024 samples); (b) For a long segment (4096 samples).

II. THE ZERO-TREE CODER

As mentioned earlier, the zero-tree algorithm was found useful in subband coding of still images [2,3]. The basic assumption is that most of the signal's energy is concentrated in the lower frequency bands. Under the above assumption there is a high probability that if the energy in some frequency band is lower than a certain threshold, the energies of the higher bands will remain below that threshold as well. In that case a 'zero-tree' code is sent for the entire set of bands thus saving bits.

The performance of the algorithm depends strongly on the existence of the above assumed signal characteristics. These characteristics hold also for audio signals since indeed most of the energy is usually concentrated in the lower frequencies. Another reason why we can adapt this assumption for audio signals is the fact that the human hearing system in much more sensitive at low frequencies than at high frequencies.

In the zero-tree algorithm the coefficients values are not transmitted in a pre-defined order but both the encoder and decoder share the same index ordering algorithm based on the magnitude of the coefficients. Assuming the coefficients are ordered according to their magnitude (or that their ordering is known to both encoder and decoder) then the transmission is done according to the bit-significance order, that means that the bit that will lower the reconstruction error the most will be sent first while the least significant bits will be sent last (that means that the signal is reconstructed progressively).

In the zero-tree algorithm the ordering of the transmission of coefficient values is not done by sending the indices of the coefficients, but by sending coefficient significance information. Below we explain the notion of significance:

A coefficient c_n is called a *significant coefficient* with

respect to a given k if it satisfies $|c_n| \geq 2^k$; otherwise it is called an *insignificant coefficient.*

A subset S_m is defined to be a *significant subset* with respect to a given k if it contains at least one significant coefficient with respect to the given k; otherwise it is defined as an *insignificant subset.*

For the ordering process we have to split the set of coefficients into subsets and check for the significance of each subset. If a subset is found to be insignificant then all of its members are insignificant coefficients; if a subset is significant the decoder needs more information in order to find the significant coefficients in S_m. For that purpose the encoder and decoder share splitting rules of a significant subset into smaller subsets. After splitting, each new subset must be examined for significance. The process is repeated until a magnitude check is applied to all significant subsets that include only one member. Then k is decremented and the comparison process is repeated until k=0. Since the splitting rules are known to the decoder there is no need to send the indices of the transmitted coefficients.

For efficient compression the significant-subset splitting rules should satisfy the following: a subset that is expected to be insignificant should have as many coefficients as possible while a subset that is expected to be significant should have the lowest possible number of coefficients (preferably only one - the significant coefficient). That is the reason for the use of subband coding with the zero-tree algorithm since the lower bands are expected to have significant coefficients while the higher bands are expected to be insignificant.

Another property of the signals in the subbands is "self-similarity", which implies that a phenomenon occurring at a certain time will affect the signals in many subbands (that means that every coefficient at a given level can be related to a set of coefficients at the previous finer level at that time). Therefore the coefficients, that may come from each of the levels, should correspond to the same time interval. Fig. 4 shows two examples of possible *ordering trees*. Each rectangle represents one coefficient and the arrows indicate the way that the coefficients are arranged in the ordering tree. In the case of a uniform filterbank (upper figure) all the bands represent the same resolution level, therefore the number of coefficients in each band is the same. The resulting ordering tree is that each coefficient in a certain band points to a coefficient in the next upper frequency band. In the wavelet-like filterbank (lower figure), on the other hand, the time resolution of bands gets finer as the center frequency of the band goes higher, therefore the number of coefficients is doubled as we go from one band to a higher band. As a result, the ordering tree in the case of

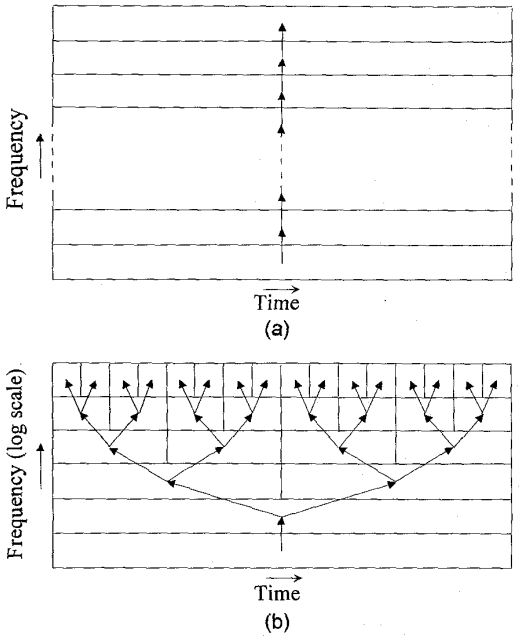

Fig. 4: Two examples for possible *ordering trees*: (a) Equal bandwidth filterbank; (b) Wavelet-like filterbank.

wavelet-like filterbanks is a full binary tree. In both cases the coefficients in the lowest frequency band are the roots of the tree while the coefficients in the highest band are the leaves of the tree, according to the assumptions that most of the energy is concentrated in the low bands.

When describing the zero-tree algorithm it was mentioned that instead of sending the indices of the coefficients, the results of coefficient significance tests are sent. In our system we used the same method as in [3] and defined four states describing the significance of a node and its sub-tree. In order to further save bits, only state changes are transmitted. Since the probabilities of these state changes are not equal we further compress the signal by using an arithmetic coder for the side information [10].

III. RESULTS

As discussed above, the proposed system consists of an adaptive analysis/synthesis filterbank and a zero-tree coder (including an arithmetic coder). Two types of analysis/synthesis filterbanks were considered: The one that handles the input signal in a prolonged manner, and the second that handles the signal as a collection of independent segments. First we examined the performance of the proposed system, using a fixed tree-structured (QMF-based) 32-band *uniform* filterbank, as compared to the standard MPEG Audio layer I coder which uses a different 32-band uniform filterbank. The

tests were performed on four audio signals, each of them sampled at 44.1kHz and lasts 20-25sec. Two output bit rates were selected: 64kbit/sec and 128kbit/sec. The signals named 'mozart' and 'vivaldi' contain harmonic music; 'vega' contains a female vocal song while 'wir' is a piece of rock music. The obtained results showed better objective (improvement by up to 9.4dB in segmental SNR) and subjective (by informal listening) performance of the zero-tree based coder. Since the standard coder uses a psychoacoustic model it is not designed to be optimal in the sense of minimum mean squared-error. Therefore, another test of the standard coder was performed, this time without the psychoacoustic model block. The resulting SNR values did not go any higher for 128kbit/sec while for 64kbit/sec the results were 0.32-2.07dB higher than before but still much lower than the system with the zero-tree coder. The subjective performance (by informal listening) of the uniform filterbank system with the zero-tree coder was in all cases at least the same as that of the standard coder. The results of the objective tests are listed in Table 1 (rows 'MPEG' and 'uniform') for both output rates.

Table 1 - Different examined coders vs. the standard coder (segmental SNR in dB for rates of 128kbit/sec and 64kbit/sec)

coding scheme	bit rate	mozart	vivaldi	vega	wir
MPEG	128	35.70	35.10	25.79	35.51
'uniform'	128	40.19	40.48	30.33	39.88
'prolonged'	128	41.35	41.65	31.33	41.05
'segmental'	128	41.16	41.59	31.33	41.03
MPEG	64	19.67	19.61	16.03	23.53
'uniform'	64	28.58	29.05	21.55	30.40
'prolonged'	64	29.67	30.21	22.22	31.14
'segmental'	64	29.67	30.18	22.20	31.19

The next step was to test the objective performance of the system with the adaptive filterbanks (both schemes) using several cost functions. The results obtained when using the zero-tree coding algorithm, including the arithmetic coding block, with the 'prolonged' and the 'segmental' types of decompositions, and using the '$\log_2$' cost function, are also listed in Table 1. The system using the 'prolonged' decomposition approach provided slightly better results, on average, but it requires many more computations and a larger memory storage in the decoder because of the post filtering. The results obtained using the 'entropy' cost function were about 0.3dB lower than those with '$\log_2$' on average, while the other cost functions, taken from [5], gave much worse results - about 5dB lower.

Using the zero-tree coding block in the standard coder gave better results (up to 1.2dB) than when using it in our 'uniform' system, showing that the zero-tree coder is responsible to most of the improvement. This result also shows that our filterbank needs more careful design to reach the maximum potential of the system.

IV. CONCLUSION

Two schemes of adaptive filterbanks subband coding systems were introduced. Both were shown to be superior to the MPEG-Audio layer I standard. The coding of a signal using a 'prolonged' decomposition was shown to give somewhat better results than coding independent segments of the signal separately, but with much higher complexity and storage requirements.

Using the zero-tree coder as the encoding block in the MPEG-Audio layer I improved the performance of the standard coder.

It appears that in order to further improve the performance of the proposed system, there is a need for a better design of the filters in the analysis/synthesis filterbanks.

REFERENCES

[1] R. R. Coifman, M. V. Wickerhauser, "Entropy-Based Algorithms for best Basis Selection", IEEE Transactions on Information Theory March 1992, vol. 38, no. 2, pp. 713-718.

[2] J. Shapiro, "An Embedded Wavelet Hierarchical Image Coder", ICASSP 1992, vol. IV, pp. 657-660.

[3] A. Said and W. A. Pearlman, "Image Compression Using the Spatial-Orientation Tree", IEEE Int. Symp. on Circuits and Systems May 1993, vol. I, pp. 279-282.

[4] ISO/IEC 11172-3:1993 Information technology - "Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s - Part 3: Audio".

[5] C. Taswell, "Near-Best Basis Selection Algorithms with Non-Additive Information Cost Functions". In Moeness G. Amin, editor, Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis, pp.13-16, Philadelphia, PA, October 1994.

[6] C. Herley, J. Kovacevic, K. Ramchandran, M. Vetterli, "Tiling of the Time-Frequency Plane: Construction of Arbitrary Orthogonal Bases and Fast Tiling Algorithms", IEEE Trans. on Sig. Proc. December 1993, vol. 41, no.12, pp. 3341-3359.

[7] R. A. Gophinath "Factorization Approach to Time-Varying Filter Banks and Wavelets", ICASSP 1994, vol. III pp. 109-112.

[8] T. Chen, P. P. Vaidyanathan, "Time-Reversed Inversion for Time-Varying Filter Banks", Asilomar 1993, pp. 55-59.

[9] I. Sodagar, K. Nayebi, T. P. Barnwell III, M. J. T. Smith, "A Novel Structure for Time-Varying FIR Filter Banks", ICASSP 1994, vol. III, pp. 157-160.

[10] I. H. Witten, R. M. Neal and J. G. Cleary, "Arithmetic Coding for Data Compression", Communications of the ACM, vol. 30, pp. 520-540, June 1987.