2003

Scalable and perceptual audio compression

Mohammed Raad
University of Wollongong, mraad@uow.edu.au

SCALABLE AND PERCEPTUAL AUDIO COMPRESSION

By

Mohammed Raad

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
AT
UNIVERSITY OF WOLLONGONG
NORTHFIELDS AVE
WOLLONGONG NSW 2522
AUSTRALIA
FEBRUARY 2003

To all those who value, seek and apply knowledge for the true benefit of mankind.

Table of Contents

List of Tables

List of Figures

Abstract

This thesis deals with scalable perceptual audio compression. Two scalable perceptual solutions as well as a scalable to lossless solution are proposed and investigated. One of the scalable perceptual solutions is built around sinusoidal modelling of the audio signal whilst the other is built on a transform coding paradigm. The scalable coders are shown to scale both in a waveform matching manner as well as a psychoacoustic manner. In order to measure the psychoacoustic scalability of the systems investigated in this thesis, the similarity between the original signal's psychoacoustic parameters and that of the synthesized signal are compared. The psychoacoustic parameters used are loudness, sharpness, tonality and roughness. This analysis technique is a novel method used in this thesis and it allows an insight into the perceptual distortion that has been introduced by any coder analyzed in this manner.

The scalable sinusoidal coder is built around the sorting of perceptually significant sinusoids and the use of the sorted relationship between the sinusoids to implement unique quantization techniques allowing for scalable compression. The results presented, which show the scalability of the coder, also re-enforce the idea that the phase component of the sinusoids has a limited perceptual contribution. The sinusoidal coder is compared to the MPEG-4 AAC coder at various rates with a variety of test material. The obtained results are promising for this scheme. This scheme is not developed further due to the limitations that are imposed on the granularity of the scalability, rather a scheme with much finer granularity is pursued and developed.

The scheme that is pursued is the second scalable scheme proposed in this thesis. It is built around the Set Partitioning In Hierarchical Trees (SPIHT) algorithm and

the Modulated Lapped Transform (MLT). This scheme is shown to produce better compression than the sinusoidal scheme as well as other schemes that combine SPIHT with other transforms. The MLT-SPIHT scheme is combined with masking and a modification to the SPIHT algorithm to increase its compression. The results presented for the MLT-SPIHT scheme with masking show very good quality at rates at or above 56 kbps. The scheme is also shown to be scalable both objectively and perceptually. The scalability of the scheme is the direct result of the use of SPIHT which is a set sorting, bit plane transmission algorithm.

This thesis then extends the scalable concept to approaching and achieving lossless compression in a smooth manner. To facilitate such a coding scheme a number of possibilities are proposed and investigated. An integer transform based scheme is investigated with two integer transforms. The results show limited lossless compression with a loss in perceptual quality at the medium to low rates. This scheme is thus replaced by an MLT-SPIHT based scheme that operates by applying SPIHT to the original signal's MLT coefficients, obtaining a residual signal and applying SPIHT again in the time domain to the residual signal. Lossless compression that is competitive with the current state of the art is achieved whilst maintaining good perceptual quality at the low to medium rates. This scheme is again extended by the development of the Perceptual SPIHT (PSPIHT) algorithm which arranges the bits for transmission in a perceptually significant manner without resorting to quantizing the less perceptually significant bits using a lower resolution than the other coefficients. Whilst there is a cost in terms of lossless performance, the algorithm does allow the focus to be on perceptually significant coefficients at the lower rates. Objective results are presented for this scheme to compare it with the MLT-SPIHT scalable to lossless scheme.

Statement of Originality

This is to certify that the work described in this thesis is entirely my own, except where due reference is made in the text.

This document has not been submitted for qualifications at any other academic institution.

Signed

Mohammed Raad
October 29, 2003

Acknowledgements

Firstly, I must thank my parents, brothers and extended family members for giving me the support I needed during the past few years of study. Without your support, none of this would have been possible.

I wish to thank my supervisors, Dr. Ian Burnett and Dr. Alfred Mertins who have provided me with exceptional guidance, and a three year long brain storming session. I am grateful for the help that both of you have provided.

This work would also have been much more difficult without the financial support of the Motorola Australian Research Centre (MARC). MARC have been helpful industry partners to the Australian Postgraduate Award (Industry)(APAI) through which this work was sponsored.

I would also like to thank all the people whom I've worked with over the past few years at Whisper Laboratories (University of Wollongong) for the many sessions of informal discussion that provided me with the impetus for some of the work that appears in this thesis.

Mohammed Raad
January 20, 2003

List of Acronyms

AAC Advanced Audio Coder

AbyS Analysis by Synthesis

ADPCM Adaptive Differential Pulse Code Modulation

ATRAC Adaptive Transform Acoustic Coder

CELP Code Excited Linear Prediction

DCT Discrete Cosine Transform

DFT Discrete Fourier Transform

DHT Discrete Hilbert Transform

DPAC Differential Perceptual Audio Coder

DPCM Differential Pulse Code Modulation

DSP Digital Signal Processing

DWT Discrete Wavelet Transform

ECG Electrocardiogram

ELT Extended Lapped Transform

EPAC Enhanced Perceptual Audio Coder

ERB Equivalent Rectangular Bandwidth

FFT Fast Fourier Transform

FM Frequency Modulation

GA General Audio

GSM Global System for Mobile communication

HILN Harmonics and Individual Lines plus Noise

HP High Pass

ICT Integer Cosine Transform

ITU International Telecommunications Union

LAR Logarithmic Area Ratios

LP Low Pass

MDCT Modified Discrete Cosine Transform

MLT Modulated Lapped Transform

MOS Mean Opinion Score

MPEG Moving Pictures Experts Group

MP3 MPEG-1 layer III coder

MSE Mean Squared Error

PAC Perceptual Audio Coder

PCM Pulse Code Modulation

PEAQ Perceptual Evaluation of Audio Quality

PNS Perceptual Noise Substitution

PR Perfect Reconstruction

PSPIHT Perceptual Set Partitioning In Hierarchical Trees

QMF Quadrature Mirror Filters

SEEVOC Spectral Envelope Estimation VOCoder

SegSNR Segmental Signal to Noise Ratio

SNR Signal to Noise Ratio

SP Sound Pressure

SPIHT Set Partitioning In Hierarchical Trees

SPL Sound Pressure Level

SQ Scalar Quantization

SQAM Sound Quality Assessment Material

STFT Short Time Fourier Transform

STN Sinusoids plus Transients and Noise

TNS Temporal Noise Shaping

TDAC Time Domain Aliasing Cancellation

VQ Vector Quantization

Chapter 1

Introduction

1.1 Perceptually scalable audio compression

Perceptual audio compression aims to reduce the amount of information that is transmitted or stored whilst facilitating the reproduction of a sound perceptually equivalent to the original. Perceptual audio compression has become the norm in the compression of audio signals as it is now expected that new compression algorithms will have perceptual concepts embedded in them somehow. This of-course has not always been the case; instead, it is the result of the combination of efforts in both signal processing technology and psychoacoustic concepts.

The interest in audio compression in general stems from the simple concept of maximizing return for a given cost, or minimizing the cost for a given return. Whichever way it is looked at, the problem at hand is to be able to provide high quality audio for the lowest transmission cost. Naturally, there are other costs that must be taken into account such as complexity and delay.

This thesis is concerned with addressing the issue of perceptually scalable audio compression, i.e., compression schemes are proposed that scale in perceptual quality

with increasing bit rate. This is a growing area in audio compression as it allows the delivery of different quality audio at varying bit rates. The algorithms that are studied and proposed in this context all aim to provide a single paradigm at different bit rates, a variation from the current standardized practice of employing different paradigms at different bit rates. The advantage of this approach is the ability to smoothly scale in quality with fine granularity up to a perceptual limit that is signal dependent.

Two forms of perceptually scalable algorithms are proposed in this thesis. The first is built around the popular sinusoidal model of audio. The second algorithm is built around a transform paradigm, which is an effective paradigm frequently adopted in audio compression. Chapters 4 and 5 deal with both algorithms respectively.

1.2 Scalable to lossless audio compression

The perceptual scalability of an audio compression scheme is only one aspect that is dealt with in this thesis. The other main issue is scalability to lossless compression. That is, smooth scalability in quality up to the lossless representation of the original signal at an acceptable bit rate. The aim with such a scheme is to allow complete scalability in the sense that given a high enough bit rate (that is below the quantization bit rate of the original signal) an exact copy of the original signal will be obtained. Lossless compression of audio is becoming a field of increasing interest with the increase in bandwidth available for wireless and internet applications.

In this thesis the scalable to lossless issue is dealt with in Chapter 6 where a novel

scheme is proposed; the algorithm maintains smooth scalability from lossy compression to lossless compression whilst offering a lossless compression rate that is competitive with the state of the art in lossless compression.

1.3 Contributions

The contributions of this thesis are the following:

- A unique compression analysis method is proposed to allow more insight into the effects that individual coders have on speech and audio signals. This analysis is based on the use of sensory pleasantness parameters to study the psychoacoustic behavior of the synthesized signal as compared to the original. (Chapter 2) [RRBM02].

- A novel scalable sinusoidal compression scheme is proposed. The scheme is also developed as a variable rate compression scheme. The scalable scheme, although simple in concept is shown to only be slightly outperformed by the MPEG-4 AAC coder. (Chapter 4) [RB01] [RBM01].

- Novel quantization schemes are also proposed that involve spline interpolation and weighted phase quantization. (Chapter 4) [RBM01].

- A perceptually scalable algorithm that is built around the Modulated Lapped Transform (MLT) and a set sorting algorithm known as Set Partitioning in Hierarchical Trees (SPIHT) is proposed. (Chapter 5) [RMB02c] [RMB02b].

- A modified version of SPIHT is proposed that is found to be more suitable to perceptual audio compression. The modified algorithm is developed from the results of a comprehensive investigation into the application of SPIHT for audio compression which also lead to the definition of unique tree sets for audio compression. (Chapter 5) [RMB02b].

• A fine grain scalable to lossless compression algorithm is proposed and investigated. The proposed algorithm is a two stage application of SPIHT to the input audio signal which allows both scalability and acceptable lossless compression. (Chapter 6) [RMB02a].

• An investigation into the application of integer transforms to scalable to lossless audio compression as well as a psychoacoustic analysis of the proposed scalable to lossless scheme. (Chapter 6).

• A new algorithm, based on SPIHT and named Perceptual SPIHT (PSPIHT), is proposed. PSPIHT allows perceptually based set sorting of the frequency domain representation of the audio signal whilst maintaining the ability to scale to lossless compression. (Chapter 6) [RMB03].

1.4 Publications

The following publications have been the result of some of the work presented in this thesis, the chapter in which the work may be found in this thesis is indicated in the parenthesis:

• M. Raad, I.S. Burnett "Audio coding using sorted sinusoidal parameters" Proceedings of ISCAS2001, vol.2, pp 401-404, May 2001. (Chapter 4)

• M. Raad, I.S. Burnett, A. Mertins "Scalable audio coding employing sorted sinusoidal parameters." Proceedings of ISSPA2001 vol.1 pp 174-177, August 2001. (Chapter 4)

• M. Raad, A. Mertins, I. Burnett "Audio compression using the MLT and SPIHT" Proceedings of DSPCS2002 pp 128-132, January 2002. (Chapter 5)

• M. Raad, A. Mertins, I. Burnett "Audio coding based on the Modulated Lapped

Transform (MLT) and Set Partitioning in Hierarchical Trees (SPIHT)" Proceedings of SCI2002, pp 303-306 vol.3, July 2002. (Chapter 5)

- M. Raad, C. Ritz, I. Burnett, A. Mertins "The analysis of speech codecs using psychoacoustic measures" accepted for publication at SCW2002, October 2002. (Chapter 2)

- M. Raad, A. Mertins, I. Burnett, "A scalable to lossless audio compression scheme" accepted for publication at WITSP2002, December 2002. (Chapter 6)

- M. Raad, A. Mertins, I. Burnett, "Scalable to lossless audio compression based on perceptual set partitioning in hierarchical trees (PSPIHT)" accepted for publication at ICASSP03, May 2003. (Chapter 6)

Chapter 2

DSP and psychoacoustic concepts in audio compression

As a starting point in this thesis, this chapter will look at the DSP theory and psychoacoustic concepts that are used in audio compression. The focus is on transform theory, signal modelling and masking. Sensory pleasantness is also considered and introduced as a coder analysis tool. As such, this chapter introduces the first contribution described in this thesis and published in [RRBM02].

2.1 Introduction

Audio compression is one application of signal compression. Digital Signal Processing (DSP) concepts play the central role in the development of algorithms and techniques that will reduce the volume of information that is needed to create a similar (if not exactly the same) copy of the original signal. DSP concepts such as transform theory, signal modelling and quantization form the backbone of current audio compression technology.

Transform theory has developed techniques that allow more compact representations of audio than its time-domain form. The development of lapped transforms and the wavelet transform has aided in representing time domain signals in less error prone ways. Lapped transforms have helped in reducing the blocking effects that are so prominent in block transforms, whilst the wavelet transform has made it possible to represent different time domain signal components with different resolutions. This has meant that the same transform can have high frequency resolution for low frequency components and high time resolution for high frequency components.

The other main set of concepts that aid in the development of effective audio compression algorithms come from the field of Psychoacoustics. The best known contribution of this field to audio compression is the masking model that is an important component of all of the state of the art audio compression schemes. This model defines the signal components the compression algorithm may ignore or modify whilst maintaining a high quality synthesized signal. Psychoacoustic concepts have also helped in the development of more perceptually relevant objective measures that allow judgements to be made with regard to the perceived quality of the audio signal without having to resort to expensive and time-consuming formal listening tests. This ofcourse does not mean that these measures are mature enough to replace formal listening tests, but they are fast and more accurate than waveform distortion measures that are popular in DSP in general. An example of such measures is the standardized objective audio quality measure PEAQ [Tt00].

This chapter begins with a look at the DSP concepts and techniques that are used in audio signal compression. This is followed by a look at perceptual entropy and how it helps reduce the amount of information that must be transmitted. At this stage

sensory pleasantness is discussed and an example of its use to analyze the coding effects of different compression algorithms is presented. Finally, we take a look at audio quality assessment with both subjective as well as objective techniques being described.

2.2 Audio signal processing

This section explores the signal processing aspects of audio compression. The discussion will start with a look at transforms used in audio compression.

2.2.1 Transforms and transform theory

Transform theory is a vast topic which has been the subject of numerous textbooks, which means that this topic is too large to be covered in detail in a thesis such as this. Instead, the focus of this subsection will be on reviewing the main properties of transforms that are utilized in audio compression. More detailed information may be found in [Mer99], [RY90], [Beu84], [YH97], [Mal92], [Yip96], [Hah96], [AR91]. The notation that will be used here is borrowed from [AR91], [Mer99] and [Mal92].

Generally, transforms are used to map a signal from one domain to another. The mapping is denoted by:

$$T(\mathbf{x}) \equiv \mathbf{x} \rightarrow \mathbf{x} \tag{2.2.1}$$

here we are only interested in time-limited transforms and discrete transforms, defined over $[0, M-1]$. Naturally, these transforms normally have continuous time counterparts and discrete time counter parts which are not time-limited, i.e. defined over $(-\infty, \infty)$.

For a known length signal $\mathbf{x} = [x(nM)x(nM+1)\ldots x(nM-M+1)]^T$, the transform $T(\mathbf{x})$ is given by:

$$T(\mathbf{x}) = \mathbf{A}^T\mathbf{x} \tag{2.2.2}$$

where $\mathbf{A}$ is a matrix that is formed from the transform basis functions (the size of this matrix depends on whether a block or lapped transform is being used). The basis functions of the transform may be both real or complex.

For audio compression one is interested in recovering the original signal and hence invertible transforms tend to be used. This means that:

$$(\mathbf{A}^T)^{-1}\mathbf{x} = x \tag{2.2.3}$$

Now, if the matrix $\mathbf{A}$ is invertible then the following relationship holds:

$$\begin{aligned} \mathbf{A}^T &= \mathbf{A}^{-1} \\ \mathbf{AX} &= \mathbf{x} \\ \mathbf{AA}^T &= \mathbf{A}^T\mathbf{A} = \mathbf{I} \end{aligned} \tag{2.2.4}$$

The above is the orthogonality property of a transform. This is clearly an attractive property as it avoids the need for matrix inversion, which leads to simpler implementation and conservation of the energy of the signal such that:

$$\sum_{\ell=0}^{M-1} |x_\ell|^2 = \sum_{\ell=0}^{M-1} |X_\ell|^2 \tag{2.2.5}$$

Further, a transform is said to be linear if it has the following properties:

$$T(\mathbf{x}+\mathbf{y}) = T(\mathbf{x}) + T(\mathbf{y}) \tag{2.2.6}$$

$$T(\alpha\mathbf{x}) = \alpha T(\mathbf{x}) \tag{2.2.7}$$

The transforms that are of interest in this thesis may be classified according to the relationship between the lengths of $\mathbf{x}$ and $\mathbf{X}$. That is, two broad categories may be defined; block transforms and lapped transforms. Block transforms map a vector of length M to another vector of length M, this means that the matrix $\mathbf{A}$ is an $M \times M$ matrix. On the other hand a lapped transform maps a vector of length M into a vector of length L, where $M = kL$ and k is an integer greater than or equal to 2. The difference between the two categories of transforms is that block transforms may operate without any overlap on consecutive blocks whereas lapped transforms can only reconstruct the original signal if successive frames are utilized. In reality, block transforms are normally combined with non-rectangular windows in order to improve the transform's frequency domain selectivity and to reduce edge effects. This effectively means that the block transform operates in a similar fashion to a lapped transform.

Within these two broadly defined categories of transforms, another two subcategories are of interest to this thesis; integer coefficient transforms and integer-to-integer transforms. Integer coefficient transforms employ only integers in the transform matrix $\mathbf{A}$. Most of the popular transforms have integer coefficient counterparts [PD00]. The integer-to-integer transforms that do not have integer coefficients tend to focus on the wavelet style transforms [AK00], [DS97], [CDSY96], and map integers to integers allowing lossless compression of the original signal.

Block transforms

Block transforms have numerous applications including signal filtering, speech scrambling, adaptive filtering, spectral estimation, and of course signal compression. The

block transforms that are of interest here are: the Discrete Fourier Transform (DFT), the Discrete Cosine Transform (DCT), the Hilbert transform and the Discrete Wavelet Transform (DWT). The listed transforms are not limited to having integer coefficients and they are not integer-to-integer. Integer transforms that are analogous to the DFT and the DCT are given in [PD00] and [Cha89].

The DFT is defined as [Mer99]:

$$X(n) = \sum_{\ell=0}^{M-1} x(\ell) W_M^{\ell n} \tag{2.2.8}$$

where $W_M = \exp \frac{-j2\pi}{M}$. The basis functions of the DFT transform matrix are then given by:

$$a_{n\ell} = \frac{1}{M} W_M^{-\ell n} \tag{2.2.9}$$

One of the useful properties of the DFT is the symmetry of the DFT coefficients if the signal is real, i.e., $X_{M-\ell} = X_\ell^*$ where $*$ stands for complex conjugation. This property is helpful in signal compression as one half of the DFT coefficients may be used to deduce the other half.

Spectral estimation has been, and remains, a very important application of the DFT [Mer99], [Mal92]. This application is of particular interest to sinusoidal transform coders as the STFT is simply the DFT with a window, and one of the main techniques of obtaining the sinusoidal model requires the use of the STFT [MQ95]. The DFT is also a popular tool for transferring required calculations from the time domain to the frequency domain, where they may be more efficiently performed. An example of such a simplification is presented in [GS92] where the sinusoidal model analysis-by-synthesis (A-by-S) procedure is conducted in the frequency domain for increased efficiency. Another example is given in [EP00] where the sorting of sinusoidal

parameters is carried out in the frequency domain, again with the aim of increasing computational efficiency.

The DFT is fundamentally a complex sinusoidal transform in that the basis functions of the transform are made up of sines and cosines in the complex plane. Another sinusoidal transform which is of interest in audio compression is the DCT. The DCT is a real transform in that its basis functions are real. It should be noted at an early stage that the DCT is defined in multiple ways, with each definition labelled as a different type; the literature defines exists Type-I, II, III and IV DCTs. The basis functions for all four type are given in the following [Mal92], [Mer99]:

$$a^{I}_{n\ell} = \gamma_\ell \sqrt{\frac{2}{M}} \cos\left[(n+\frac{1}{2})\frac{\ell\pi}{M}\right] \qquad \ell, n = 0, \ldots, M \tag{2.2.10}$$

$$a^{II}_{n\ell} = \gamma_\ell \sqrt{\frac{2}{M}} \cos\left[(n+\frac{1}{2})\frac{\ell\pi}{M}\right] \qquad \ell, n = 0, \ldots, M-1 \tag{2.2.11}$$

$$a^{III}_{n\ell} = \gamma_\ell \sqrt{\frac{2}{M}} \cos\left[(\ell+\frac{1}{2})\frac{n\pi}{M}\right] \qquad \ell = 0, \ldots, M-1 \tag{2.2.12}$$

$$a^{IV}_{n\ell} = \sqrt{\frac{2}{M}} \cos\left[(n+\frac{1}{2})(\ell+\frac{1}{2})\frac{\pi}{M}\right] \qquad \ell = 0, \ldots, M-1 \tag{2.2.13}$$

$$\tag{2.2.14}$$

where

$$\gamma_k = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } k = 0, \text{ or } M \\ 1 & \text{otherwise} \end{cases}$$

The type-II DCT tends to be the one most employed in practice [Mal92]. The main advantage of the DCT in general is that it delivers a higher spectral resolution than the DFT of the same length. That is, a frequency bin in the DFT is approximately double that of a frequency bin in the DCT. This increased frequency resolution comes at a cost as the DCT produces coefficients that cannot be interpreted in terms of phase.

Also, the DCT can miss strong signal components that are $\frac{\pi}{2}$ out-of-phase with the basis functions [Mal92]. However, this problem can be solved by taking a number of consecutive DCTs of the signal.

The DCT finds plenty of applications in spectral estimation [RY90] as well as signal compression [RY90], [Mer99], [Mal92]. The Type-IV DCT has been used in the development of the Modulated Lapped Transform (MLT), which is a popular transform in audio compression [Shl97]. In this thesis the DCT (Type-II) is used in an investigative study that will be presented in Chapter 5 whilst the MLT is the basis for the scalable compression algorithms presented.

The next two transforms to be mentioned here, the Hilbert transform and the DWT are not sinusoidal based transforms. The Hilbert transform is mentioned here for completeness as it finds use in signal envelope calculation for parametric coding [PM00]. The DWT is used as a sub-band transform and has found use in a number of recent audio coders [PS00].

The Discrete Hilbert Transform (DHT) may be calculated through the following [Hah96]:

$$X_\ell = \sum_{n=0}^{M-1} h(\ell - n)x(n) \tag{2.2.15}$$

where

$$h(n) = \begin{cases} \frac{2}{M} \sin \frac{\pi n}{2}^2 \cot \frac{\pi n}{M} & \text{for } M \text{ even} \\ \frac{2}{M} \sum_{q=1}^{\frac{M-1}{2}} \sin \frac{2\pi nq}{M} & \text{for } M \text{ odd} \end{cases}$$

As already mentioned, interest in the Hilbert transform for audio compression originates from the fact that it allows the calculation of the time domain envelope of the transformed signal. To obtain the envelope from the DHT, the following steps are taken [Mer99], [Har98]:

Step one - Form the analytic signal using the equation:

$$\mathbf{x}_a = \mathbf{x} + j\mathbf{X}_{Hilbert} \tag{2.2.16}$$

Step two - Shift the analytic signal into the base band:

$$\mathbf{x}_{bb} = \mathbf{x}_a \exp(-j\omega_0 t) \tag{2.2.17}$$

where ω_0 is the center frequency of the positive bandwidth of signal $\mathbf{x}$

Step three - Determine the envelope by:

$$v = |\mathbf{x}_{bb}| \tag{2.2.18}$$

The final transform that is of interest in this section is the DWT. The DWT is a multi-resolution transform that has good localization in both time and frequency. The multi-resolution nature of the DWT makes it attractive for signal analysis and signal compression, as natural signals are such that there usually exists a small section of the full bandwidth of a signal which is of real significance. The DWT is built on the idea of analyzing an input signal by the use of basis functions that are similar in shape [VK95], [RV91]. These wavelets are dilated and translated versions of one "wavelet" $\psi_1(t)$. This one wavelet is dilated by the use of the following relationship in the continuous time domain [Mer99]:

$$\psi^{(a,b)}(t) = |a|^{\frac{-1}{2}} \psi_1(\frac{t-b}{a}) \tag{2.2.19}$$

This results in the following equation for the evaluation of the continuous wavelet transform:

$$X_{(a,b)} = |a|^{\frac{-1}{2}} \int_{-\infty}^{\infty} x(t)\psi_1(\frac{t-b}{a})dt \tag{2.2.20}$$

The DWT is a sampled version of the continuous time wavelet transform, i.e. letting $a_p = 2^p$ and $b_{pq} = a_p qT = 2^p qT$, then:

$$\psi_{pq}(t) = |a_p|^{\frac{-1}{2}} \psi_1(\frac{t - b_{pq}}{a_p}) \tag{2.2.21}$$

The above sampling and scaling of the wavelet $\psi_1 t$ gives the equation of the DWT as:

$$\mathbf{X}_{(a_{pq}, b_{pq})} = \langle \mathbf{x}, \psi_{pq} \rangle \tag{2.2.22}$$

where $\langle \cdot \rangle$ denotes the inner product. Equation (2.2.22) defines a wavelet series that is sampled dyadically. Using such an expression and defining the space $L_2(\mathbb{R})$ as a sum of subspaces, with each space representing a given frequency band, this can be shown to lead to those subspaces being nested [Mer99]. The relationship between these subspaces is more clearly illustrated using a diagram.

Figure 2.1 illustrates how the DWT operates and what the set of coefficients that are obtained represent. The dyadic nature of the transform means that each band is split into two smaller bands with each application of the transform. The high-pass (HP in the diagram) component, or filter, of the transform produces the detail coefficients of that band. The low pass (LP) filter of the transform produces the "approximation" coefficients of the input signal [VK95]. It must be pointed out here that the DWT is not strictly a time-frequency transform, it is actually a time-scale transform because the base wavelet is scaled and not modulated [Mer99]. However, it is most easily understood for the application at hand in terms of its partitioning of the frequency domain.

The high localization in time for high frequency components and the high localization in frequency for the low frequency components makes the wavelet transform

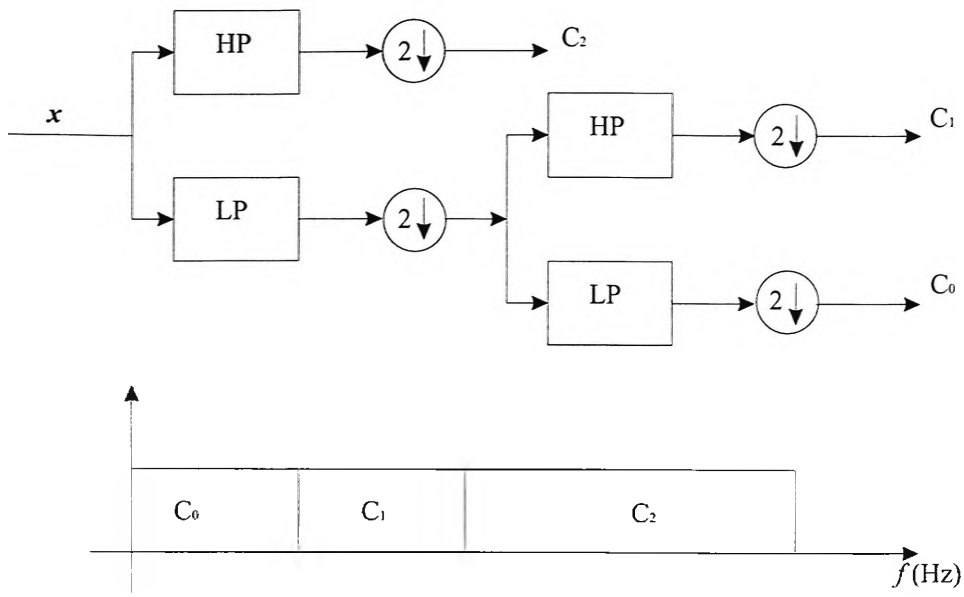

Figure 2.1: The wavelet transform and its sub-band operation

a powerful signal analysis tool [Mer99], [Mal92]. In audio compression, the wavelet transform has been employed in [LP98], [ST93b] and others, with a number of different wavelets as the base wavelet. In this thesis the Daubechies 10 wavelet is used (in Chapter 5) in a comparative study between a number of different transforms for scalable audio compression.

As a final note, although the wavelet transform has been included in this section along with block transforms, it is quite different in structure to the other transforms mentioned. In fact, the wavelet transforms are lapped transforms.

Lapped Transforms

Lapped transforms may be viewed as a generalization of block transforms as lapped transform theory collapses to block transform theory when the basis functions of the transform do not overlap [Mal92]. The overlapping of the basis functions is the distinguishing property of lapped transforms. Originally, lapped transforms were

designed to reduce the blocking effects of block transforms [Mal92]. Blocking effects appear at the edges of blocks that have been coded and reconstructed through the use of a block transform. The reason behind these effects is the discontinuities that may result at the block edges due to quantization noise in the transform domain; this leads to edge samples that are not at their correct level.

In audio compression, if block transform coding was to be applied as it is in image compression (i.e. no-overlap between the blocks or some other mechanism to account for edge effects) then a periodic audible distortion would occur [Mal92]. The reason behind the periodicity of this distortion is the high likelihood of occurrence of edge errors at the end of each coded frame.

As stated earlier, the overlapping basis functions means that given an input block of length M, the resultant number of transform coefficients is L with $M \geq L$ or $M = kL$ with k integer. These simple relationships show that for $k = 1$ the lapped transform is a block transform, thus, for M new samples received by the transform engine, M new transform coefficients are produced [Mal92]. This maintains the overall sampling frequency of the system.

Mathematically, a lapped transform operation is similar to a block transform operation, however, there is the following added restriction to guarantee perfect reconstruction:

$$\mathbf{A}^T\mathbf{D}^m\mathbf{A} = \delta(m)\mathbf{I} \qquad m = 0, 1, \ldots, k-1 \tag{2.2.23}$$

where

$$\mathbf{D} = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$

This is because the basis functions of a lapped transform must be orthogonal to each other (i.e. to the other basis functions in the same block) as well as the

overlapping basis functions.

In terms of implementation, the lapped transform of choice in audio compression is the MLT [Shl97](better known as the MDCT with a sinusoidal window). The MLT has basis functions with an overlap factor of 2, and basis functions given by:

$$a_{\ell k} = \sin\left((\ell + \frac{1}{2})\frac{\pi}{2M}\right)\sqrt{\frac{2}{M}}\cos\left((\ell + \frac{M+1}{2})(k + \frac{1}{2})\frac{\pi}{M}\right) \tag{2.2.24}$$

The sine term in the above equation is the definition of the sinusoidal Perfect Reconstruction window which will be discussed later.

In terms of coding gain, it has been shown experimentally in [Mal92] that well designed lapped transforms have a better performance than block transforms. It has also been shown that lapped transforms can be implemented with their own set of fast algorithms, a useful property for implementation purposes.

Finally, it should be noted that lapped transforms can have any overlap. A lapped transform can be designed for any length frame as long as the relationship between frame length and overlap is maintained. Cases where k is above 2 (i.e. where the overlap is greater than half the frame length) are referred to as the Extended Lapped Transforms (or ELTs) [Mal92]. These transforms have not found much use in audio compression.

Integer transforms

Integer transforms are transforms that have only integer coefficients in their transform matrix. One of the best known integer transforms is the Walsh-Hadamard transform [Beu84], [YH97] which has a transform matrix made up of ± 1 elements only. Integer transforms may be block or lapped [Goy00], [ITN02].

The Walsh-Hadamard transform introduces a new concept to this discussion as it is a "sequency" transform rather than a frequency or scale transform. Sequency refers to the number of zero crossings of a signal and in that sense is a more general descriptor than frequency which only describes periodic signals [Beu84]. The Walsh-Hadamard transform has seen a number of applications in telecommunications and statistics [YH97]. In terms of audio compression, the Walsh-Hadamard transform has not seen significant use. More accurately however we should state that the Walsh-Hadamard transform may see indirect use in audio compression as it can be used as the building block in fast transform algorithms [YH97].

Integer transforms are attractive in applications because they allow a reduction in complexity, in that, when dealing with a quantized signal, integers are being multiplied by other integers; this is a much less complex operation than the multiplication of real numbers. Also, integer transforms allow the potential of lossless representation of the original quantized signal. This is because the end result of the transformation is guaranteed to be integer avoiding irrational numbers which can only be approximated by a finite number of bits. This thesis will present a study into the use of two integer transforms for audio compression in Chapter 6 and a more detailed look at the issues that arise when using such transforms for audio compression will be deferred until then.

Integer-to-integer transforms

Integer to integer transforms are a group of transforms that map a set of integers to another set of integers without the limitation of having to use integer coefficients alone. The advantage of this type of transform (over one that utilizes only integer

coefficients) is the maintenance of good frequency selectivity. Generally, integer transforms have lower performance in terms of frequency selectivity and coding gain when compared to the non-integer transforms [Cha89]. The use of non-integer coefficients allows the transform to maintain frequency selectivity and coding gain closer to those of the normal block or lapped transform from which it has been derived. Note here that the assumption being made is that both the integer transforms and the integer-to-integer transforms are designed from an existing transform. Of course, this is not always the case and there have been a number of techniques presented in the literature which aid in the design of integer and integer-to-integer transforms [ITN02]. However, the most popular approach to the design of the integer-to-integer transforms is the "lifting scheme" [Swe96], [DS97] which allows the decomposition of a transform matrix into a set of matrices that result in an integer-to-integer transform by inserting the quantization at each stage of the new matrix representation. It is also important to mention that integer-to-integer transforms do provide an implementation advantage, as pointed out in [DS97], because of the stage by stage implementation that is made possible by the use of these transforms.

2.2.2 Quantization

Having discussed the transforms that are popular in signal and audio compression, the discussion will now shift to the quantization stages of compression. It may be stated that two stages of quantization exist in signal compression, the first is the expected analogue to digital conversion which is an important step but does not directly influence compression algorithms. The second stage is the quantization of the transform coefficients or the parameters of an adopted signal model.

In a way, the actual analogue to digital conversion may be considered compression as well since the signal will be represented by a finite set of integers instead of being represented by an infinite set of numbers [GG92], [NJ84]. From this point of view one may argue that quantization error is inevitable as one cannot map an infinite set of numbers to a finite set of integers and back without error. The size and properties of the error depends on the size of the integer set and the size of the original number set.

Having obtained a digital representation of the analogue signal that is of acceptable quality, digital compression techniques are applied to reduce the bit rate required. These techniques are typically aimed at a simpler representation of the digital signal which allows a coarser quantization that leads to a lower bit rate while maintaining an acceptable quality signal [GG92], [NJ84].

Quantization techniques may be grouped into two broad categories. Scalar Quantization (SQ) and Vector Quantization (VQ). In this discussion, two other categories are also presented (although they also fit under the broad categories SQ and VQ) namely entropy quantizers and perceptual quantizers. These will be discussed separately as they are important components in many audio compression algorithms.

Scalar quantization

Scalar quantization allows the mapping of a single real number to a single real integer. Given a real number x, a scalar quantizer produces the output $\hat{x}$ such that:

$$\hat{x} = \beta x + \epsilon \tag{2.2.25}$$

where β is a gain term and ϵ is an additive noise term [NJ84]. The above equation is a general mathematical representation, the values of β and ϵ depend greatly on

the type of scalar quantizer used as well as on the resolution of the quantizer. The resolution of a quantizer refers to the number of bits per sample that the quantizer utilizes and for this discussion it will be denoted by the letter R. There are three types of scalar quantizers; uniform quantizers, non-uniform quantizers and adaptive quantizers.

A uniform quantizer is one that divides its range equally amongst the integers in its code-book. One of the best known and most popular forms of uniform quantization is Pulse Code Modulation (PCM). PCM tends to be used as an analogue to digital conversion technique [NJ84]. The number of bits per sample used by PCM depends on the bandwidth of the source that must be quantized and on the quality required. PCM has the attractive feature of offering a linear relationship between SNR (in dB) and the number of bits used [NJ84]. In fact for a quantized speech signal with R bits per sample, the SNR can be approximated very well by:

$$SNR_{speech} = 6.02R - 10 \qquad (dB) \tag{2.2.26}$$

For narrow-band speech signals, it is normally adequate to use $R = 16$ bits per sample, giving an overall rate of 128 kbps. A wide-band speech signal (bandlimited to 8 kHz) demands a rate of 256 kbps and CD quality audio requires a rate of 706 kbps per channel if a sampling rate of 44.1 kHz is used, or 768 kbps if 48 kHz is used.

A useful technique for improving the quality of the quantized signal using PCM is to introduce a known high-frequency noise component into the original signal. This is followed by applying PCM to the new noisy signal and then removing the high frequency noise from the quantized signal. This technique is known as dithering [NJ84] and much of the current audio material available is quantized by the use of this technique.

Uniform quantization PCM is only one form of PCM. Other popular forms of PCM use non-uniform quantization. Non-uniform quantization attempts to improve the performance of the scalar quantizer whilst using the same number of bits per sample. This is achieved by the application of increased resolution at regions that cover numbers that have a higher probability of occurrence than other numbers that must be covered by the quantizer. Two standardized techniques of this kind are the A-law PCM and the μ-law PCM. A-law PCM uses the mapping:

$$\hat{x} = \begin{cases} \frac{A|x|}{1+\ln A} sgn(x) & 0 \leq \frac{|x|}{xmax} \leq \frac{1}{A} \\ xmax \frac{1+\ln \frac{A|x|}{xmax}}{1+\ln A} sgn(x) & \frac{1}{A} < \frac{|x|}{xmax} \leq 1 \end{cases} \tag{2.2.27}$$

whilst the μ-law PCM adopts the mapping:

$$\hat{x} = xmax \frac{\ln 1 + \frac{\mu|x|}{\mu max}}{\ln 1 + \mu} sgn(x) \tag{2.2.28}$$

It can be shown that in the case of A-law quantization, the SNR is given by [NJ84]:

$$SNR_A = 6.02R + 4.77 - 20 \log_{10} \frac{A}{1 + \ln A} \tag{2.2.29}$$

thus to obtain an SNR which is equivalent to that of the PCM SNR for speech then A = 87.56. This is the North-American PCM standard. On the other hand for μ-law quantization [NJ84]:

$$SNR_\mu = 6.02R + 4.77 - 20 \log_{10} (1 + \ln (1 + \mu)) \tag{2.2.30}$$

giving a value of $\mu = 255$ for an equivalent SNR as uniform quantization. This is the European PCM standard.

The use of μ-law and A-law PCM is known as log-PCM as the mapping of integers requires a logarithmic operation. Now, log-PCM is a very important technique in the communications community as it allows the quantization of narrow band speech at

R= 8 bits per sample, with a total rate of 64 kbps. This rate obtains a SNR of 38 dB and a Mean Opinion Score (MOS) of 4.5 [NJ84], which is the definition of toll quality speech.

Naturally, variations on PCM exist which allow for high quality signal synthesis with a reduced bit rate. Differential PCM (DPCM) is an important variation on standard PCM. DPCM uses a linear predictor to remove the inter-sample of a signal and transmits the difference between the expected value and the actual value input. When the signal has stationary characteristics , this technique reduces the variance of the signal to be quantized, allowing a better approximation of the signal for a lower rate. The performance of a DPCM quantizer is thus related to the ratio of the variance of input signal to that of the error signal. In fact the SNR is given by [NJ84]:

$$SNR_{DPCM} = 6.02R - 10 + \frac{var(x)}{var(x - \hat{x})} \qquad (dB) \qquad (2.2.31)$$

DPCM normally provides high quality speech (just below toll quality) at rates between 32 kbps and 48 kbps. In these systems, the linear predictor may be adapted in which case the system is referred to as ADPCM.

As previously mentioned, PCM is the format in which most audio material is actually obtained in and is thus the format of the source material used for testing in this thesis. It is also the format which is termed "original" when loss-less compression is applied.

Vector quantization

Vector quantization refers to the mapping of a vector of real numbers to another vector of real numbers chosen from a limited set of vectors [GG92], [NJ84], [Ram99].

Vector quantizers divide the real number plane (that is of the same dimension as the vector to be coded) into 2^N regions (where N is the number of bits used to access each vector in the code-book), known as voronoi regions [GG92], [Ram99]. Each voronoi region contains one vector which is chosen or calculated so that it represents all the vectors that fall in that region adequately.

The code book vector is chosen such that an error criteria is minimized. Normally, the error criteria is the distance criteria between vectors or the weighted distance criteria between vectors [Ram99].

Vector quantizers are designed by dividing the number plane being used into N regions and calculating the code book vectors that minimize the distortion criteria. This is the process that is followed by the well known iterative LBG (Linde-Buzo-Gray) algorithm [GG92]. Variations in the algorithm focus on the distortion measure and thus the division of the voronoi regions, depending on whether a certain distortion measure will lead to a better performance with the source at hand.

Vector quantization can out-perform scalar quantization, for the same bit rate, and the performance tends to increase with the length of the vectors being quantized [GG92]. As the size of the code book increases, the size of the voronoi regions decreases which means that the distortion in turn decreases. However, this improvement in performance comes at the cost of increased complexity. Each unit increase in N means a doubling of the code-book size. If each vector contained k elements, a complete search of the code-book for the best matching vector would require $k2^N$ multiplications and $(k-1)2^N$ additions (assuming that the Euclidean distance measure is used). In contrast, a scalar quantizer using the same number of bits would require $k2^{N/k}$ multiplications and $k2^{N/k}$ additions to find the correct code-word.

The complexity of searching a VQ code-book is considerably greater than the complexity of a scalar quantizer. A number of methods of organizing code-books in a way that leads to a faster location of the required code-book vector have been designed [GG92]. These techniques re-arrange the original structure of the code-book in a way that is informative about the content of the code-book vectors. Tree structured VQ does exactly that, with each node of the tree providing information about the subsequent vectors that are connected to it [GG92]. Tree structured VQ is one technique that reduces the search complexity considerably. Classified VQ is another technique that reduces the search complexity by adopting an identifier code-book and a number of vector sub-code-books. The identifier code-book determines which sub-code-book should be searched for the best matching vector [GG92].

A product code approach may also be adopted to reduce the complexity of VQ search. In this case the vector to be quantized is divided into smaller sub-vectors and each sub-vector is coded using a code-book that contains vectors of the same dimension [Ram99]. Multistage VQ also operates on a similar principle, except here the vector to be encoded is first approximated by a vector from one code-book amongst multiple code-books. The error between the approximation and actual vector is then coded using the second code-book in the set and so on. Multistage VQ is a popular VQ technique because of its memory savings and search complexity reduction [GG92], [Ram99].

Since audio compression has fundamentally been built around transform coding, VQ application to audio has not been as rapid as that for speech where very low rate coders demanded the use of VQ in one way or another. That trend was been changed by the introduction of TwinVQ, an audio coder which achieves very good results

at rates around 16 kbps [PS00], [IMM95]. In this thesis VQ has only been applied in a limited fashion. The choice of scalar quantization over vector quantization has been made because of the coding algorithms applied and developed and not directly because of the disadvantages of complexity and memory consumption.

Entropy coding

Entropy coding refers to the mapping of a number set to another number set in a one-to-one match. That is, entropy coding is lossless [NJ84], [GG92], [GPS94]. One could imagine a real number set being mapped to a binary number set, however the size of each set would be very large and the usual application of entropy codes is the mapping of integers to another set of integers. Compression is obtained through the idea that numbers that are encountered most often are allocated the smallest binary integers, and thus the least number of bits [GG92]. In this way, entropy codes are similar to probability density function (pdf) optimized scalar quantizers that allow for better quantization resolution for sub-ranges that have a high probability of occurrence [NJ84].

The idea behind entropy codes comes from the knowledge that the information carried in a signal is dependent upon the statistics of the signal. The content of information is expressed by the first order entropy of the signal x:

$$H(x) = -\sum p(x) \log_2 p(x) \qquad bits/sample \tag{2.2.32}$$

It can be seen from the above equation that a purely deterministic signal carries no information as $H(x)$ will be zero. That is, no bits need to be spent to communicate the content of that signal. Also, signals that have values with a high probability of occurrence will have a lower entropy than signals that have a pdf that tends towards

a uniform distribution. The first order entropy of the signal sets the lower bound for the entropy code and it is the usual case that entropy codes perform at a rate that is worse than that expected from the calculation of $H(x)$ [GG92].

One of the best known entropy codes is the Huffman code [GG92], [GPS94]. The Huffman code allows the development of a code-book, with variable length codewords, that is decodable and associates the codeword length with the probable frequency of occurrence. The decodability of the Huffman code is due to it obeying the "pre-fix condition", that is if the prefix of each codeword is unique then the variable length code can be decoded [GG92]. Actually, this condition limits the ability of entropy codes to reach the theoretical limit determined by $H(x)$.

The process of designing a Huffman code is iterative. The initialization step involves listing the numbers to be coded in order of decreasing probability (with each number presenting a node in a tree). The two least probable numbers are then combined to produce a single node which has a probability equal to the sum of the two probabilities used to form it. This process is repeated until all the original numbers have been part of producing a node. Finally, each branch of the developed tree is labelled by a 1 or 0 [GPS94], [GG92]. The Huffman code can be extended to the coding of whole vectors [GG92], although this is a rather complicated extension of the algorithm.

An alternative entropy coding technique is arithmetic coding [WNC87]. Arithmetic coding is based on the idea that if the decoder is provided with a unique real number in the range $[0, 1)$ and a model for the coded data, then it can obtain the original message which was used to generate that real number. Again, the process here is iterative in that the encoder repeatedly narrows the range of the real numbers

according to the data encountered. Both encoder and decoder have access to a model which describes the pdf of the number set to be coded and how that pdf is distributed in the range $[0, 1)$. The final real number obtained determines the initial range that is smaller than $[0, 1)$ and thus the first number in the original set being coded. The decoder continues to sub-divide the initial range obtained from the model, with each sub-division corresponding to a number (or character) from the original number set until the final range is reached and the entire original message is extracted [WNC87].

Arithmetic coding has been shown to out-perform Huffman coding in many instances [WNC87] and it does have the advantage of allowing the coding of variable length blocks of data (the data may be numbers or characters) unlike Huffman coding. Arithmetic coding may also result in a zero length code-word in its code-word set [WNC87], [GG92], something that is impossible in Huffman coding. However, arithmetic coding is only uniquely decodable if a unique End of File (EOF) character is used to indicate the end of the block being coded [WNC87].

The final entropy coding technique that will be described here is the Lempel-Ziv code [GPS94], [CT91]. The Lempel-Ziv code is currently a very popular data compression algorithm (it is the basis of the famous zip program) because it is effective and it is fast [CT91]. The algorithm divides a given input string (which is usually binary) into unique substrings. This subdivision of the strings relates each of the substrings such that there is a one bit difference between consecutive substrings [CT91]. The encoder transmits a pointer which tells the decoder which prefix to use as well as the state of the final bit of the substring. The Lempel-Ziv algorithm approaches the Shannon bound (given by $H(x)$) asymptotically [CT91]. Compression is achieved through the fact that the length of the substrings obtained will become longer than

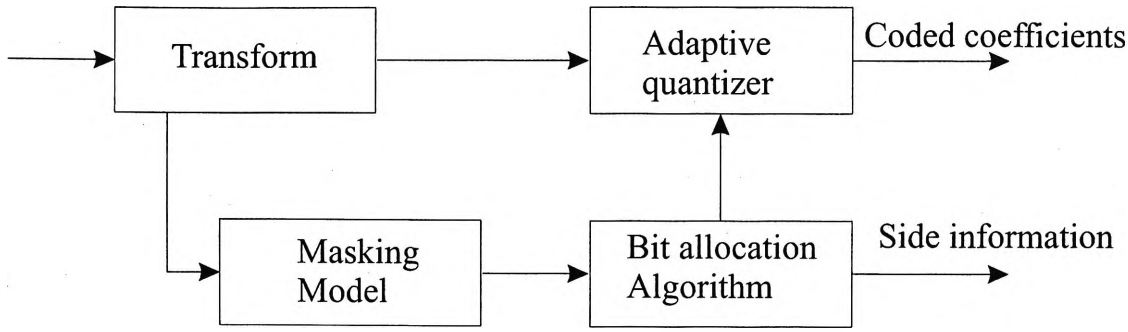

Figure 2.2: Perceptual quantization in transform audio coding

the length of the pointer information transmitted.

Entropy coding techniques play an important role in audio compression as evidenced by their use in the MPEG audio coding techniques [QJ97], as well as the lossless compression algorithms [HS01]. Whilst entropy coding techniques are not usually applied directly to the audio signal, they are very useful in compressing data that is very important to maintaining the quality of the synthesized signal and hence must be coded with no loss, they are also applied directly to the error signal in lossless compression. This thesis deals with entropy coding through the use of the Set Partitioning In Hierarchical Trees (SPIHT) algorithm and its application to lossless audio compression.

Perceptual quantization

Human auditory perception allows the allocation of quantization bits in a manner that maintains a very high quality synthesized audio signal [Nol97]. The use of perceptual quantization in transform coding is illustrated by Figure 2.2.

Perceptual quantization is an implementation of adaptive quantization with the adaptation being controlled by the perceptual model. Human perception is the subject of Section 2.3 so a detailed discussion on the topic will not be given here. The bit

allocation algorithm varies in implementation, however, the aim is the same as other quantization algorithms, to minimize an error criteria with the available number of bits. The error criteria in this case is weighted such that components that are perceptually more significant are allocated more bits than components that are not so significant. Such algorithms are employed in standardized coders such as the MPEG coders [BKS00] as well as proprietary coders such as Dolby's AC-3 [Dav99].

Perceptual bit allocation produces a coded stream that normally requires side information for successful decoding. This is where an entropy code can increase the compression of the overall algorithm, as the side information is usually critical to the sensible decoding of the main bit stream and should be obtained losslessly. This thesis focuses on schemes that utilize perceptual quantization techniques and a number of these techniques are presented and studied in detail in later chapters.

2.2.3 Audio signal models

Recently, audio compression has been approached from a signal model point of view [Goo97], [Lev98], [Ver99]. The reasoning being that a reformulation of the problem will lead to better coding results [HAT96]. The popular approach is based on the harmonic plus stochastic model which divides the signal into a sinusoidal (harmonic) signal and a noise-like (stochastic) signal [Goo97]. Figure 2.3 shows the idea of this decomposition.

This model has been made effective because of advances in both the sinusoidal modelling [GS92] and noise modelling [Goo97]. The model has been further extended in [Ver99] and [Lev98] to include a transient detection component. Figure 2.4 shows the extended model.

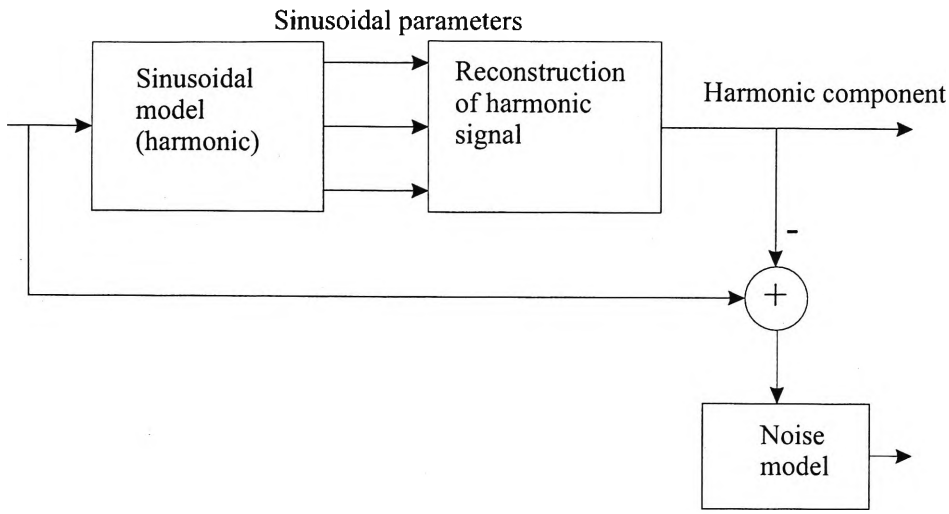

Figure 2.3: The harmonic and stochastic model of audio

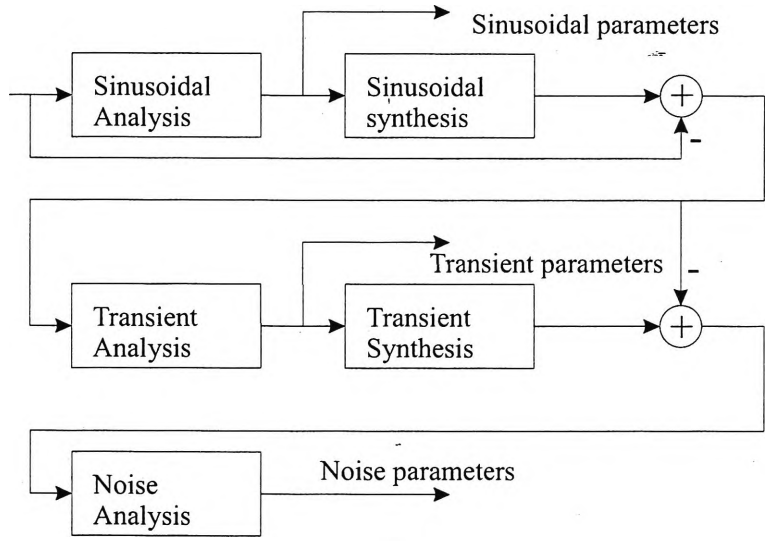

Figure 2.4: The STN audio model

The sinusoidal analysis allows the capture and modelling of the harmonic components, whilst this can be applied with a high enough resolution such that the transients are detected, it is found to be an expensive coding approach to take [Ver99]. The transient detection focuses on the high frequency components of the signal that are inharmonic. Transients, because they have a high frequency, are not essential when the aim is the perceptual similarity between the original signal and the synthesized signal. However, the naturalness of the sound is degraded or even lost if the transients are missing. In other words, a very pleasant sound can be obtained without the transients but a better one is obtained with them. Finally, the noise component also adds to the wholeness of the synthesized audio.

A very attractive property of this approach to coding is that the model can accommodate different types of signals more efficiently. The reasoning is simple; if a frame of a signal is very harmonic the sinusoidal model parameters are then given more prominence as they better model the signal. A frame with a large number of attacks cannot be expected to fit a harmonic model very well and so the transient model would be given prominence. Similarly, a frame that was more noise like should be modelled with a noise model once any harmonic component has been removed [Ver99].

This model has been developed with an eye for scalable audio compression [EP00], [Ver99]. At the lowest rates the harmonic model components of the signal would be focused on, as the rates increase the noise and transient model would be gradually added. The noise model may in some sense be considered more important than the transient model because of the fact that transients may be roughly approximated and one would still maintain a perceptually similar sound, although the transients do add

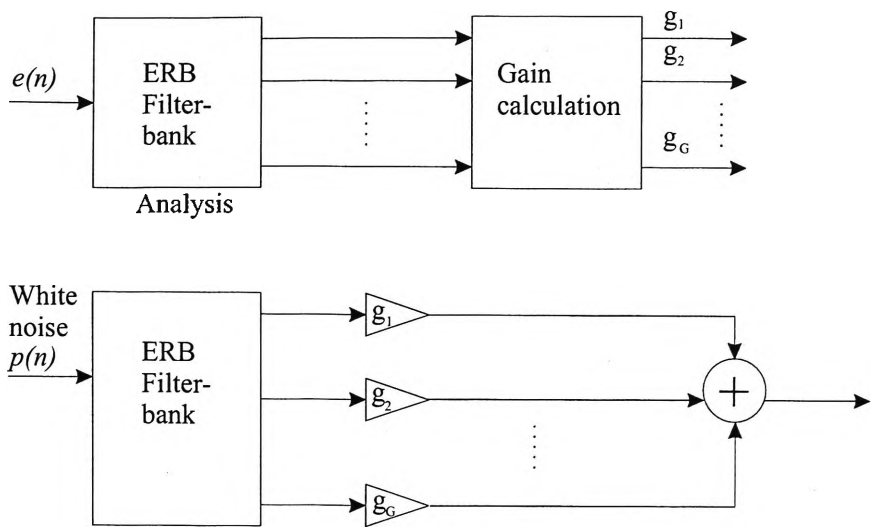

Figure 2.5: Noise modelling through white noise shaping

to the naturalness of the sound. The noise model on the other hand effects the entire spectrum of the signal.

In order to model the noise so that a close perceptual approximation of the original is obtained, Equivalent Rectangular Bandwidth (ERB) filters are used [SA99]. These filters model the Bark scale in the frequency domain (the Bark scale and other perceptual issues are the topic of the next section). The obtained original analysis noise $e(n)$ is itself analyzed using the ERB filters, with the aim of modelling $e(n)$ by the use of white noise and some gain terms [Goo97]. The concept is illustrated in Figure 2.5.

It has been shown in [Goo97] that the gains should be calculated by the use of the following equation:

$$g_\ell = \sqrt{\frac{\sum_{n=0}^{N-1} e_{\ell n}^2}{N\sigma^2 \sum_k h_\ell(k)^2}} \tag{2.2.33}$$

where N here is the number of samples of the ERB band signals, e_ℓ. σ^2 is the variance of the white noise used in the synthesis and $h_\ell(k)$ is the k^{th} filter coefficient of the ℓ^{th} ERB filter in the filter bank. Thus, the model is a coloring process of white noise. The

gains, g_ℓ, are time varying and are updated once per frame. In the reconstruction of the noise signal, the sign of the actual samples is of no concern because of the ear's lack of appreciation for the detail in broad band noise [Goo97]. Also, each gain coefficient is applied to all the white noise components in a given ERB and so the noise is shaped in a "step" manner. That is, there are hard boundaries between the noise components in adjacent ERBs.

The performance of coding systems built on this type of signal model [EP00], [Ver99] is reportedly competitive with transform coding systems at low rates, and may be slightly better. One drawback of this model is its scalable range in that it is difficult to see how it can scale to lossless representation without introducing a different noise model. The presented STN model has been incorporated in scalable compression algorithms that produce perceptually lossless results at high medium rates (around the 80 kbps mark) in [Ver99].

2.3 Perception and perceptual entropy

Having discussed the basic audio compression and signal processing techniques, it is clear that perceptual considerations play an important role in how audio is compressed and modelled. A discussion on human perception has been delayed to this point simply to create an idea of the need for a thorough understanding of this issue as well as to represent signal processing tools that are helpful in implementing methods that try to account for perception of audio by humans.

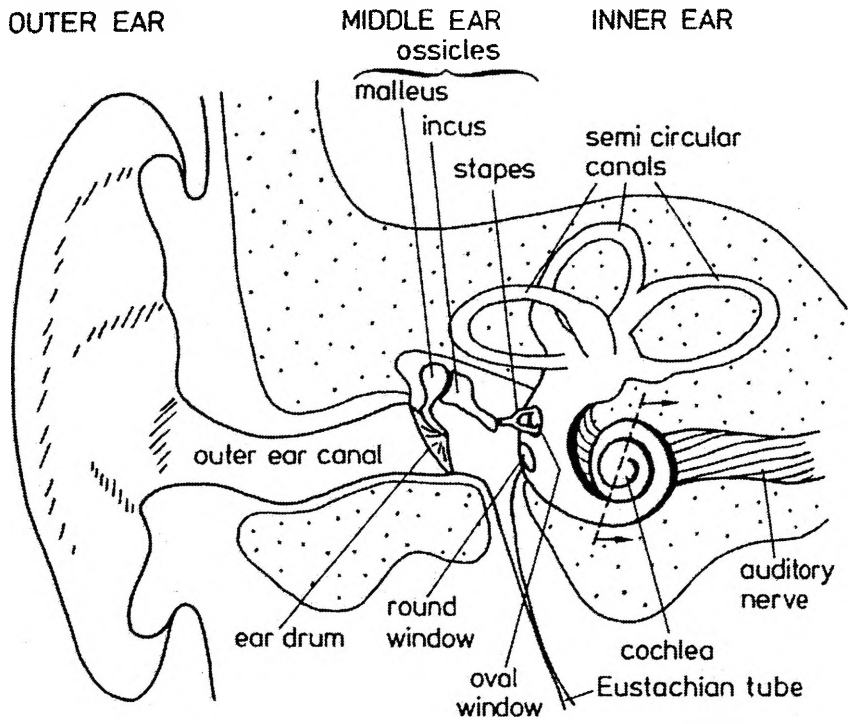

Figure 2.6: A cross section of the human ear [ZF99]

2.3.1 The mechanics of sound perception

To understand how one perceives a sound, much research has been focused on the instrument that is used to receive the sound: the human ear [ZF99], [Moo97], [Moo96], [Hal99]. How we interpret these sounds and what they mean to us is a separate field of research. The purpose behind understanding the operation of the human ear is simple; if one understands the receiver and its limitations then the information that is sent to the receiver can be tailored such that it fits within those limitations. This directly leads to a reduction in the amount of information that must be transmitted [Hal99].

The ear can be divided into three sections; the outer, middle and inner ears [Moo97], [ZF99]. Figure 2.6 shows a cross section of the whole ear [ZF99].

The outer ear is made up of the Pinna (the visible part) and the auditory canal. The pinna significantly modifies incoming sounds and helps in localizing sounds. Sound travels along the auditory canal and causes the ear drum to vibrate. These vibrations are transmitted through the middle ear by the Ossicles to the oval window. The Ossicles is the name of a group of three small bones. The individual names of these bones are the Malleus, Incus and Stapes. In terms of the functionality of the ear, the middle ear has a mechanical function as it transfers a sound from air to the fluids in the Cochlea by the use of impedance matching.

The inner ear, known as the Cochlea, is a spiral shaped structure. It is filled with incompressible fluids and has bony rigid walls. This part of the ear is extremely important to its functionality. The cochlea is divided by two membranes along its length, known as Reissner's membrane and the Basilar membrane. The oval window that can be seen in Figure 2.6 is at the base of the Cochlea which is the first point that encounters a sound.

The apex of the Cochlea is the inner tip at the other end. The apex has a small opening called the Helicotrema through which fluid flows between the two main sections of the Cochlea. These sections are known as the Scala vestibuli and Scala tympani. Fluid flow around the Helicotrema results from inward movement of the oval window. This also causes an outward movement of the round window and membrane. The round window is a second opening in the Cochlea, below the oval window.

It is known that the basilar membrane's vibrational shape depends on the frequency of the stimulant [Moo97]. Pure tones (in psychoacoustic terms a tone is a sound that is capable of triggering a response from the auditory system) produce vibrations of the basilar membrane, which have maxima dependent on the frequency

of the stimulant. That is, the ear produces a time to frequency transformation.

The basilar membrane has hair cells attached to it. The hair cells are part of the organ of Conti. The outer hair cells are those on the outside of the arch closest to the outside of the Cochlea. The outer hair cells are arranged in three rows with each cell having approximately 140 hairs. There are approximately 25000 outer hair cells in total. The inner hair cells are those on the other side of the arch and are arranged in a single row. There are approximately 3500 of these with 40 hairs each.

When the basilar membrane moves, the hair cells move. It is this movement of the hair cells that triggers the neural activity in the nerve cells connected to the hairs. Thus, the sound has been passed from air to the brain where it is processed and so perceived. The human ear is most sensitive to frequencies that match human speech frequencies, i.e. in the range 1 kHz - 5 kHz. It is worth noting that the perception of sounds outside of this range deteriorates with age whereas the perception of sounds that are inside this range is relatively unaffected with age [Moo97].

2.3.2 Modelling the human ear

Modelling the human ear involves the same steps as modelling any system. A known input is used and a model is produced that matches the output of the system being modelled given the same input. The input to the human auditory system must be a sound defined by an accurate measure. To establish a uniform measure of sound, the Sound Pressure Level (SPL) is defined and normally expressed in dB. Sound Pressure (SP) is described in terms of Pascals (Pa), the SPL of a sound is actually a normalized measure given by [ZF99]:

$$SPL = 20\log_{10}(\frac{SP}{SP_0}) \quad dB \tag{2.3.1}$$

where $SP_0 = 20\mu$Pa. The SPL of a sound can be objectively measured. However, this value does not describe how the ear hears the sound as the ear hears sounds with the same SPL with different loudness at different frequencies. To illustrate this, Figure 2.7 [ZF99] shows equal loudness curves. The points on each curve are perceived to have the same loudness, yet there is a clear difference in the intensity of the sounds in dB (the curves are produced by the use of a sinusoid). Such curves may be used to map intensity to loudness (Loudness and its attributes will be discussed in Section 2.4) and they can be used to determine the limits of the auditory system. The bottom curve in Figure 2.7 is the lower threshold of hearing, meaning that any (sinusoidal, or similar) sounds below that level have an extremely high probability of not being heard. This may not be the case for all humans as hearing varies between individuals but it has been found to be a very good description of the lower limit of hearing [ZF99], [Hal99]. The threshold shown is also the threshold in quiet, with an increase in noise that threshold is actually significantly higher. The threshold of hearing in quiet can be approximated by [PS00] (where f is in Hz):

$$Th_q = 3.64(\frac{1000}{f})^{0.8} - 6.5\exp -0.6(\frac{f}{1000} - 3.3)^2 + 10^{-3}(\frac{f}{1000})^4 \qquad (SPL) \tag{2.3.2}$$

In Section 2.3.1 it was mentioned that the ear acts like a mechanical time to frequency transform. As transforms may be looked at as a bank of bandpass filters, then it makes sense to model the ear transform as a bank of bandpass filters. However, the ear transform filter bank has been found to be highly overlapping [Moo97], [ZF99] with asymmetric filter responses. The filter shapes can be derived by the use of the

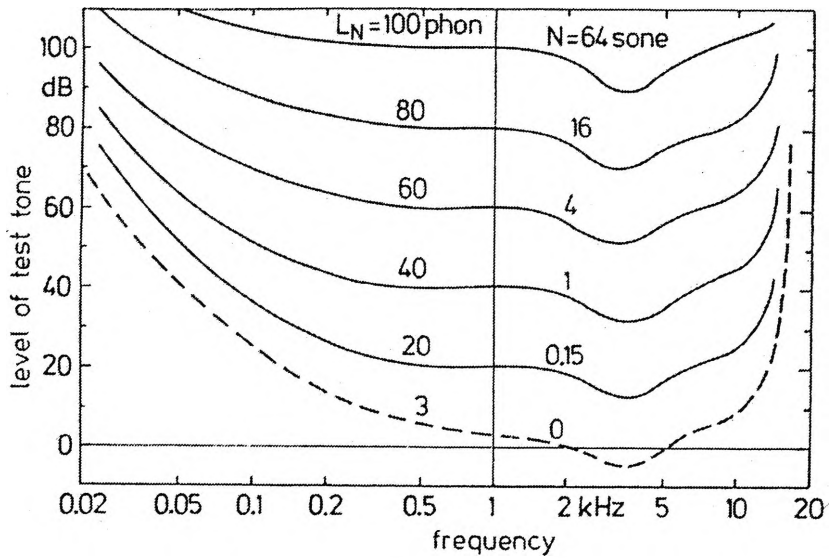

Figure 2.7: Equal loudness curves of the ear [ZF99]

rounded exponential ($roex(p)$) model or the more general $roex(p,r)$ model [Moo96]. The $roex(p)$ model is given by [Moo96]:

$$P(\hat{\omega}) = (1 + p\hat{\omega}) \exp -p\hat{\omega} \tag{2.3.3}$$

In the above, $\hat{\omega} = \frac{f - f_0}{f_0}$ is the normalized deviation from center frequency f_0, $P(\hat{\omega})$ is the magnitude of the filter response (not in dB) and p is a variable that effects the slope of the filter response. The higher and lower frequency slopes of auditory filters are different and so a different value of p is required for the high frequency slope to that used for the low frequency slope. The $roex(p)$ model performs well except in situations where the threshold of hearing is approached [Moo96]. In such situations, it would be better to use the $roex(p,r)$ model which is given by [Moo96]:

$$P(\hat{\omega}) = (1 - r)(1 + p\hat{\omega}) \exp -p\hat{\omega} + r \tag{2.3.4}$$

In this case, r is a constant for both sides of the filter.

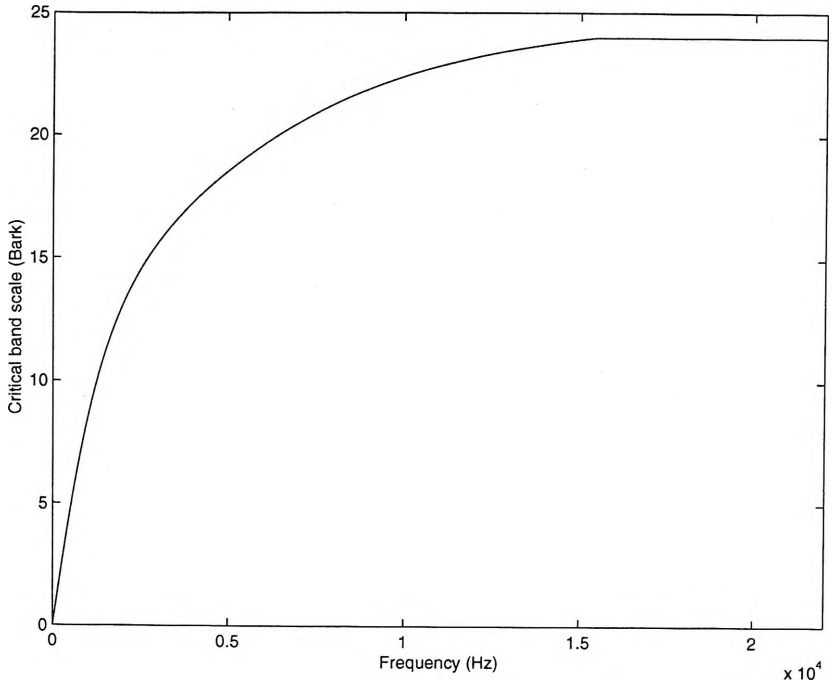

Figure 2.8: Frequency to Bark scale conversion

The use of highly overlapping filters to model the time-frequency transformation in the ear leads to the definition of critical bands, where each filter defines a band. Because of the overlapping bandwidths, the relationship between frequency and the critical band scale is non-linear and is given by [Hal99], [Moo96]:

$$z(f) = 13 \arctan(\frac{76f}{10^5}) + 3.5 \arctan(\frac{f}{7500})^2 \qquad Bark \tag{2.3.5}$$

The critical band value that corresponds to frequency f in Hz is denoted z in the above equation. Figure 2.8 shows the relationship graphically between the frequency scale and the Bark scale. The above equation implies that the relationship between critical band rate and frequency is continuous. Psychoacoustic data has only strictly defined 25 critical bands [ZF99], [Moo97] because of the definition that has been used to describe a critical band. A critical bandwidth is "that bandwidth at which subjective responses rather abruptly change" [Hal99]. Smooth changes within the

Table 2.1: Critical bands with center frequency, lower frequency and upper frequency

$\lfloor z \rfloor$	f_0 (Hz)	f_l (Hz)	f_u (Hz)	$\lfloor z \rfloor$	f_0 (Hz)	f_l (Hz)	f_u (Hz)
0	50	10	100	12	1850	1720	2000
1	150	100	200	13	2150	2000	2320
2	250	200	300	14	2500	2320	2700
3	350	300	400	15	2900	2700	3150
4	450	400	510	16	3400	3150	3700
5	570	510	630	17	4000	3700	4400
6	700	630	770	18	4800	4400	5300
7	840	770	920	19	5800	5300	6400
8	1000	920	1080	20	7000	6400	7700
9	1175	1080	1270	21	8500	7700	9500
10	1370	1270	1480	22	10500	9500	12000
11	1600	1480	1720	23	13500	12000	15500
				24	19500	15500	–

critical band will occur, however, those changes do not contribute a new band. Hence, $\lfloor z(f) \rfloor$ gives the actual band number that the frequency falls in. The relationship between critical band number and bandwidth is given in Table 2.1.

In Table 2.1, the final upper frequency limit is not listed. One may interpolate and add more bands [Hal99] or may simply define the upper frequency limit as the Nyquist frequency for a digital signal (that is sampled above 31 kHz). In either case, there is only limited evidence that any sounds that are relevant to audio and speech applications contain perceptually significant components and such high frequencies [SA99], [Hal99].

Having modelled the structure of the human auditory system, psychoacoustic measurements allow the development of the idea of masking.

2.3.3 Masking

Masking refers to the inaudibility of sounds due to the presence of other sounds [ZF99], [Moo97]. This observable property of the human hearing system is of high significance for audio compression purposes as it allows the identification of sounds that are likely to be missed and so are not necessary to code. The other way of viewing the use of masking in coding is that it defines the effect that broadband noise (the coding noise) will have on the perceptual quality of the signal [Hal99], [Moo96].

The three categories within masking are pre-masking, simultaneous masking and post masking. Pre-masking is probably more clearly described if the term backward masking is used [Hal99] as that implies a sound masking another sound that occurred earlier in time than the masker. It may seem odd that a sound may be masked by a masker that has not yet occurred in time. The explanation for this is related to the nature of sound perception. A sound's loudness is developed by the hearing system, that is the hearing system acts like an integrator and takes time to build the loudness of the sound [ZF99]. The time required is in the range of 5-10 ms and so if another sound is encountered during that period which is significantly larger than the sound being processed, the development of the loudness is interrupted and so masking occurs. Backward masking is primarily a time domain phenomena, like post-masking or forward masking.

Forward masking refers to the masking of sounds by a sound that has already occurred. Loudness in the hearing system does not disappear as soon as the sound ends in the time domain. It has been observed that loudness decreases as a function of time, with a small proportion of the original loudness lasting more than 150 ms [Hal99]. The decrease in loudness resembles an exponential decay. If a sound was

to occur in time after the end of another sound with an intensity that is sufficiently small so that it produces a loudness value below that of the decaying loudness then it will be masked [Hal99], [ZF99], [Moo97].

In order to take advantage of pre and post masking, the algorithm must operate (at least partially) in the time domain. On the other hand, simultaneous masking is primarily a frequency domain effect as it defines how two sounds occurring at exactly the same time may lead to one of them being masked. Simultaneous masking occurs when one of the sounds incident on the hearing system is sufficiently large that it develops a loudness value that is well beyond that of other incident sounds and so the other sounds are masked [ZF99], [Moo97].

There are two types of sounds considered when discussing masking, tones and noise. The interactions that are of interest are noise masking tone, tone masking tone and tone or noise masking noise [Hal99]. For the case of a tone masking a tone, masking is greatest when the frequencies of the tones are very close together. As the difference between the frequencies increases, the masking also decreases. The masking pattern produced by the tone masker also tends to be asymmetric such that a higher level of masking is obtained at higher frequencies than it is at lower frequencies. The study of masking by the use of tone maskers is not very useful for compression purposes, although it has historical significance. The reason is that the coding noise resulting from the compression is not a tone. Also, the use of a tone masker in the actual study of masking leads to non-linear interactions in the auditory system that increase the difficulty of understanding the masking phenomena. This is why more recent studies on masking tend to use noise maskers [Hal99].

In the case of noise masking a tone, the masking results are similar to the case

of tone masking tone in terms of masking pattern shape and asymmetry [Hal99], [ZF99]. However, in this case one can see the operation of the ear's critical bands as the masking pattern is significant for noise components that are in the same critical band as the tone and insignificant otherwise [Hal99], [ZF99]. This effect has been used in determining the bandwidth of critical bands [ZF99], [Hal99].

The final case of noise being masked by either tone or noise is not as thoroughly understood or studied as the other two cases [Hal99]. From an audio compression perspective, masking the coding noise (or being able to) would be attractive. The implementation of such a scheme would mean that not only perceptually relevant signal components are transmitted but also those components that allow the hiding of coding noise would be given preference. The masking patterns of a narrow-band noise being masked are similar in shape, but different in magnitude to those of a tone being masked [Hal99]. A wide-band noise component, that is a noise component that has a bandwidth wider than the bandwidth of the critical band in which the center frequency is located, appears to be very difficult to mask [Hal99]. It is also significant to note that if a tone masker is used, tones that have low frequencies produce a higher masking threshold than higher frequency tones when a noise component is being masked [Hal99].

The observed temporal and simultaneous masking effects have led to the development of psychoacoustic masking models [Moo97], [ZF99]. The simultaneous masking model is a frequency domain model which is dependent on signal energy and critical band analysis of the signal [Moo97], [ZF99], [Joh88a]. On the other hand, the temporal masking model is a time-domain sliding window model [Moo96]. The sliding window model acts like a low pass filter in the frequency domain and so leads to a

smoothing of the time domain signal. The window widths used are related to the center frequency of the signal [Moo96]. The shape of the window is related to the temporal masking effects already discussed, components that are close to the middle of the window are multiplied by a factor equal to or close to unity. Components that are at the edges of the window are multiplied by a factor close to zero with the post-masking side of the window being longer than the pre-masking section of the window [Moo96].

The development of masking models has significantly improved compression rates [Nol97]. One of the earliest steps taken to introduce masking models to audio compression lead to the development of the idea of perceptual entropy [Joh88b].

Perceptual entropy

Perceptual entropy is analogous to information entropy in that it describes the mean number of bits per sample required to faithfully represent the perceptual information in an audio signal. Perceptual entropy is calculated by first calculating the masking threshold Th_i (how to calculate the threshold is discussed in Section 2.3.4) at each component of the spectral representation of the audio signal. This threshold value is equated with the quantization noise of a scalar quantizer, i.e.

$$Th_i = \frac{\Delta^2}{12} \tag{2.3.6}$$

If one assumes the use of a Fourier transform for the calculation of the spectrum of the signal, then the noise must be spread equally across the real and imaginary components. This is followed by determining the ratio of both the real and imaginary components to Th_i in the respective domains and quantization of that ratio:

$$N_{Re,Im}(\omega) = |round(\frac{c_{Re,Im}}{Th_{i_{Re,Im}}})| \tag{2.3.7}$$

If N_{Re} or N_{Im} is above zero then the number of bits required to transmit either N_{Re} or N_{Im} (or both) is calculated by using the $\log_2(\cdot)$ operation:

$$R_N = \log_2 (2N_{Re \text{ or } Im} + 1) \tag{2.3.8}$$

The perceptual entropy is then given by [Joh88b]:

$$H_p = \frac{\sum R_N}{length(FFT)} \tag{2.3.9}$$

In [Joh88b] it was found that most of the tested material had a perceptual entropy in the range $[0.1, 2.1]$ bits per sample. The most probable entropies were observed to be between 1 and 2 bits per sample. At a sampling rate of $f_s = 44.1$ kHz this means that the compressed audio files should range between 44.1 kbps and 88.2 kbps. As will be seen in Chapter 3, most current audio coders perform in that range. It should be remembered at this stage that H_p refers to the bits required to re-synthesize all the perceived information in the signal. The audio signal may be coded at lower rates and the noise shaped to create less annoying artifacts. The noise shaping approach may be viewed as parametric modelling the audio signal.

The development of H_p as a measure of bit rate required has shown, objectively, that for an audio coder to be of high quality and still have a low to medium bit rate, it must take perception into account [Vel92]. The reasoning is obtained by studying the calculation of H_p. In the worst case scenario, all of the spectral components are above the masking threshold by the same amount as they are above the threshold in quiet. However, this is simply the original signal representation. Hence, if the

masking threshold is taken into account only a reduction in the number of bits per sample required is possible. Coding algorithms that do not take H_p into account can thus only perform at an average rate that is worse than similar algorithms that do take stock of H_p.

2.3.4 Calculating the masking threshold

In this section the details of how the simultaneous masking curve can be calculated are given. The early method of calculation is due primarily to Johnston [Joh88a]. This technique was adopted, with modifications, by MPEG in its perceptual transform coder [BKS00]. The MPEG model produces a masking threshold value at each bin of the spectral representation of the audio signal. Johnston's early approach, in contrast, produces a single threshold value for each critical band in frequency. Our own informal experiments have shown that both models produce similar results, however, the MPEG model was observed to be more "aggressive" at the higher frequencies of the signal which leads to a slightly more muffled sound when the spectral components that are labelled perceptually insignificant are removed completely. These issues will be discussed further in the coming chapters. It should also be pointed out that the MPEG model being used here is the MPEG model 1.

Johnston's algorithm

Johnston's algorithm is diagrammatically given in Figure 2.9. The equations required for each stage are given below.

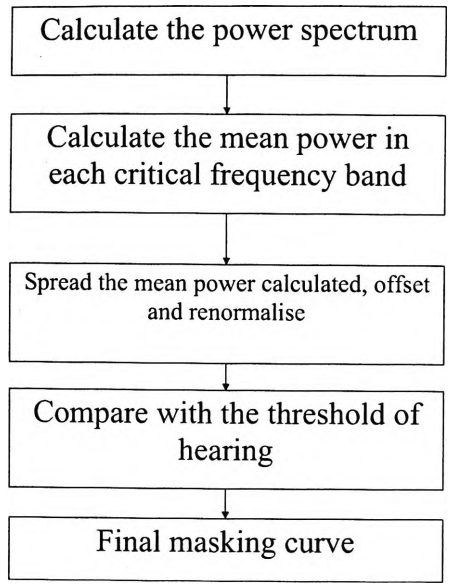

Figure 2.9: The generation of the Johnston masking curve

Step 1- Calculation of the power in each critical band.

$$P_j = \sum_{\omega_l j}^{\omega_u j} (Re(c_\omega)^2 + Im(c_\omega)^2) \tag{2.3.10}$$

where c_ω is the Fourier transform coefficient at frequency ω

Step 2- Applying the spreading function across the whole spectrum.

$$Q_j = S_{ji} Pj \tag{2.3.11}$$

$$10 \log_{10} S(z) = 15.81 + 7.5(z + 0.474) - 17.5\sqrt{(1 + (z + 0.474)^2)} \quad dB \tag{2.3.12}$$

In the above, z is the difference in critical band scale between the frequency components j and i.

Step 3- Offsetting the obtained threshold.

$$O_j = (\alpha(14.5 + j) + (1 - \alpha)5.5) \tag{2.3.13}$$

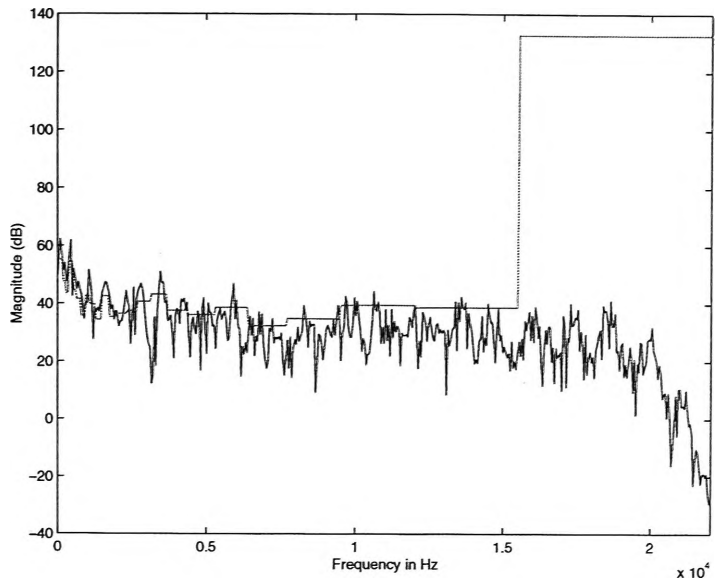

Figure 2.10: An example of the masking curve generated via Johnston's technique

$$\alpha = min(\frac{SFM_{dB}}{-60}, 1) \tag{2.3.14}$$

$$SFM_dB = 10log_{10}\frac{E_{geometric}(P_j)}{E_{Arithmetic}(P_j)} \tag{2.3.15}$$

$$Th'_j = 10^{log_{10}Q_j - \frac{O_j}{10}} \tag{2.3.16}$$

Step 4- Re-normalizing the obtained threshold.

$$Th''_j = Th'_j(\frac{P_j}{Q_j}) \tag{2.3.17}$$

Step 5- Final threshold

$$Th_j = max(Th''_j, Th_q) \tag{2.3.18}$$

where Th_q is the threshold in quiet.

Figure 2.10 shows the final masking curve for an example audio frame obtained by the use of the listed algorithm.

MPEG model 1 algorithm

The MPEG model 1 algorithm is listed here to illustrate the difference between the models adopted by MPEG and the Johnston algorithm. Model 1 is recommended for use with MPEG-2 layers I and II. Model 2 is the recommended model for layer III, it has higher computational complexity for an improvement in overall performance [Dav95]. The whole algorithm is completed in 5 steps [PS00]. The assumed sampling rate is 44.1 kHz.

Step 1- Normalize the audio signal to SPL scale and obtain the power spectrum using 512 samples per frame (11.6 ms) with an overlap of 32 samples. The Hann window is used in the spectral analysis. In the following equation, N is the length of the DFT and q is the number of bits used in quantization.

$$x_{SPL} = \frac{x(n)}{N(2^{q-1})} \tag{2.3.19}$$

$$P(i) = 90.302+10\log_{10}|\sum_{k=0}^{N-1}\{\frac{1}{2}[1-\cos(\frac{2\pi k}{N})]\}x_{SPL}(n)\exp -j(\frac{2\pi ik}{N})|^2 \qquad 0 \leq i \leq \frac{N}{2} \tag{2.3.20}$$

Step 2- Identify the tonal and noise maskers using the following definitions:

$$\zeta_{tonal} = P(i) \qquad \text{iff } P(i) > P(i \pm 1) \text{ AND } P(i) > P(i \pm \Delta_i) + 7 \text{ dB} \tag{2.3.21}$$

$$\Delta_i \in \begin{cases} 2 & 2 < i < 63 \\ [2,3) & 63 \leq i < 127 \\ [2,6) & 127 \leq i \leq 256 \end{cases} \tag{2.3.22}$$

Noise maskers are found from components that are not in the Δ_i range of tonal

maskers, and the power of the tonal and noise maskers are computed using the following:

$$P'_{tonal}(i) = 10\log_{10}\sum_{k=-1}^{1} 10^{P(i+k)/10} \qquad dB \tag{2.3.23}$$

$$P'_{noise}(i') = 10\log_{10}\sum_{k} 10^{P(k)/10} \qquad dB \qquad \forall P(k) \ni P'_{tonal}(i, i\pm 1, i\pm \Delta_i) \tag{2.3.24}$$

here

$$i' = (\prod_{k=l}^{u} k)^{\frac{1}{l-u+1}}$$

Step 3- Reduce the number of maskers by discarding the maskers that are below the threshold in quiet Thq. Replace any two maskers (from the remaining set) that are within 0.5 Bark of each other by the stronger of the two. Finally, the remaining maskers are re-organized according to the following:

$$\begin{aligned} P'_{tonal,noise}(k) &= P'_{tonal,noise}(i) \\ P'_{tonal,noise}(i) &= 0 \\ k &= \begin{cases} i & 1 \le i \le 48 \\ i + (i mod 2) 49 \le i \le 96 \\ i + 3 - ((i-1) mod 4) 97 \le i \le 232 \end{cases} \end{aligned} \tag{2.3.25}$$

Step 4- Calculate the individual masking thresholds for both noise and tonal maskers.

$$Th_{tonal}(k,j) = P'_{tonal}(j) - 0.275z(j) + S(k,j) - 6.025 \qquad dB \tag{2.3.26}$$

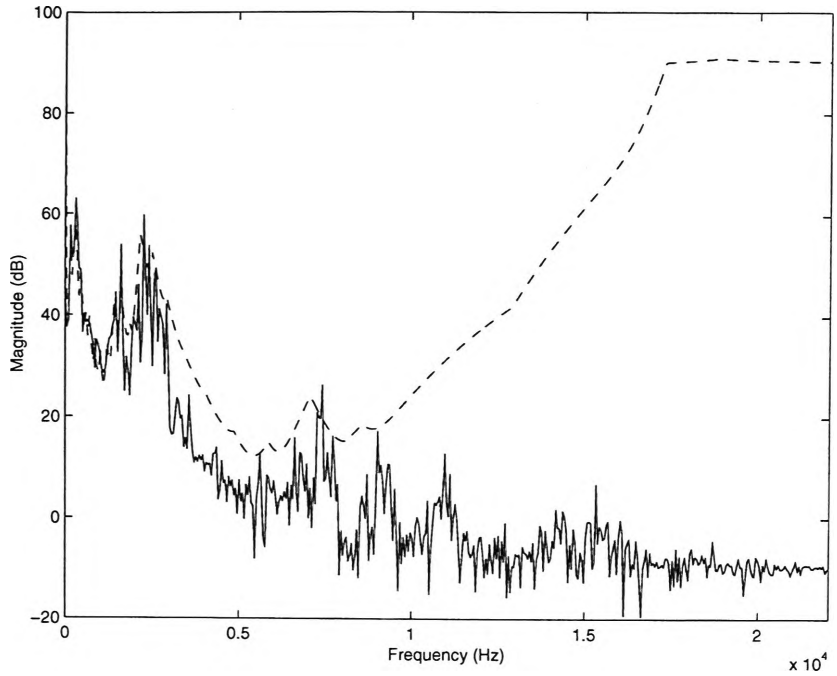

Figure 2.11: An example of a masking curve generated by MPEG's perceptual model1

In the above, z is the bark scale variable and S is the spreading function given by:

$$S(k,j) = \begin{cases} 17\Delta_z - 0.4P'(j) + 11 & -3 \leq \Delta_z < -1 \\ (0.4P'(j) + 6)\Delta_z & -1 \leq \Delta_z < 0 \\ -17\Delta_z & 0 \leq \Delta_z < 1 \\ (0.15P'(j) - 17)\Delta_z - 0.15P'(j) & 1 \leq \Delta_z < 8 \end{cases}$$

$$Th_{noise}(k,j) = P'_{noise}(j) - 0.175z(j) + S(k,j) - 2.025 \qquad dB \qquad (2.3.27)$$

Step 5- Calculate the global masking threshold by combining the I tonal maskers with the J noise maskers in the following manner:

$$Th(k) = 10\log_{10}(10^{\frac{Th_q(k)}{10}} + \sum_{i=1}^{I} 10^{\frac{Th_{tonal}(k,i)}{10}} + \sum_{j=1}^{J} 10^{\frac{Th_{noise}(k,j)}{10}}) \qquad dB \qquad (2.3.28)$$

Figure 2.11 shows the masking curve obtained for a sample frame of audio using the above algorithm. Compared with the masking curve obtained by the use of the Johnston masking model, the major differences include the increase in resolution and

the increased aggressiveness at high frequencies. The increased resolution is an advantage as it allows the quantization scheme more flexibility in bit allocation. The increased flexibility is the result of not having to define the quantization scheme according to the critical band because each spectral component now has an autonomous masking threshold. On the other hand, the increased aggressiveness in masking at the higher frequencies may be a disadvantage for sounds that are more noise-like in nature because the higher frequencies add to the naturalness of the sound. A carefully designed coding scheme would take this into account.

2.4 Sensory pleasantness and its contributing factors

Sensory pleasantness describes the acceptability of a given sound to the human ear. The contributing factors to sensory pleasantness are sharpness, roughness, loudness and tonality. This section looks at sensory pleasantness factors from a coding perspective. The factors of loudness, roughness and sharpness are used to analyze the sensory effect that the coding noise has on the synthesized signal. To this end, an example analysis is presented where two well known speech codecs are analyzed by the use of these measures. The sensory pleasantness factors are utilized through out this thesis to aid in the development of the scalable perceptual algorithms that are presented.

Sharpness may be viewed as a measure of the density of loudness across the spectrum in different critical bands. Sharpness is most heavily influenced by the center frequency of the sound as well as the spectral content [ZF99]. The unit of

sharpness is the "acum". Sharpness increases for sounds with greater loudness spread across more critical bands, i.e. if the spectral envelope (which ultimately determines the loudness) is spread in significance across a large number of critical bands then the sound will be sharp. In order to model this effect, Zwicker and Fastel in [ZF99] proposed the following simple equation:

$$S = 0.11 \frac{\int_0^{24} N' g(z) z dz}{\int_0^{24} N' dz} \tag{2.4.1}$$

where S is the sharpness, N' is the loudness in the given critical band (called the "Specific Loudness") z and $g(z)$ is a weighting function. The weighting function is only plotted in [ZF99] but a close approximation is given by:

$$g(z) = \begin{cases} 1 & 0 \leq z < 16 \\ 8 \times 10^{-5} z^{3.42} & 16 \leq z \leq 24 \end{cases}$$

Loudness has been central to the discussion about perception and masking in the previous sections. It is measured in units of "phon" and is a relative measure indicating Sound Pressure Level (SPL) of a 1 kHz signal that would sound as loud as the given sound. Loudness is a sensation that is developed by the hearing system, that is, a sound incident to the hearing system will not be heard instantaneously. An interesting fact about loudness is that a sound consisting of two tones separated in frequency will be perceived to be louder than a sound of the same energy but consisting of a single tone. As the frequency separation is increased, the loudness increases. In [ZF99], the loudness is modelled by the following equation:

$$N = \int_0^{24} N' dz \tag{2.4.2}$$

The Specific loudness is not simply the excitation level at a given frequency in a

band, although it is related to it. The equation relating the excitation level and the specific loudness is:

$$N' = 0.08\left(\frac{E_{TQ}}{E_0}\right)\left[\left(0.5 + 0.5\frac{E}{E_{TQ}}\right)^{0.23} - 1\right] \text{ Sone/Bark} \tag{2.4.3}$$

where E_{TQ} is the excitation at the threshold in quiet, E_0 is the excitation as related to a reference intensity of $I_0 = 10^{-12}\ W/m^2$ and E is the excitation of the sound of interest.

Having described Loudness and Sharpness, the two remaining sensations that contribute to sensory pleasantness are Roughness and Tonality. Roughness describes the inability of the ear to distinguish tonal components. That is, a sound that is noise-like sounds rough. As the ear begins to distinguish the tonal components of a sound, the roughness of the sound decreases. In this way Roughness may be considered as an opposite effect to Sharpness. The model proposed in [ZF99] for Roughness is based on the assumption that the hearing system is only capable of detecting changes in excitation as is by:

$$R = 0.3\frac{f_{mod}}{kHz}\int_0^{24}\frac{\Delta L_E(z)dz}{dB/Bark} \tag{2.4.4}$$

In Equation (2.4.4) ΔL_E is the change in the sensation level in dB, this is different to the change in excitation level but may be calculated from it (see [ZF99]). The term f_{mod} is the modulating frequency of the sound, where it has been assumed that an amplitude modulation model is sufficient to represent the sound.

The final sensation to look at when studying sensory pleasantness is Tonality. In [ZF99] it is suggested that tonality must be judged subjectively as no appropriate model exists. It is noted, however, that tonality decreases with increasing critical

band rate spread, that is as the noise becomes more noise like it becomes less tonal. It should be noted here that the tonality of the sound can be approximated by using the Spectral Flatness Measure (SFM) [Joh88a]. In [Joh88a] it is suggested that as the SFM increases, the tonality decreases which matches what is reported in [ZF99].

Example analysis of speech coders

As an example of how the sensory pleasantness factors can be used in the analysis of speech and audio coders, the following experiment was carried out using the FS1016 coder [JPC89] and the G729 coder [St98], both well known speech coders [RRBM02]. The aim is to determine how the coding techniques effect the psychoacoustic properties of the coded signals. In this experiment, ten files (five female and five male) of narrow band speech were used to compare the behavior of the four factor models. These files were extracted from the ANDOSL database [and], re-sampled to 8 kHz and bandlimited between 300 Hz and 3.4 kHz. The models presented in the previous section were then used to calculate the mean Loudness, Sharpness, Roughness and Tonality of each speech file (the original, FS1016 and G729 coded speech). The results obtained for the compressed versions of each file have been normalized to the results of the original file. As such, a relative measure is obtained showing how the compression techniques tested compare to each other and the original.

Figures 2.12 and 2.13 present the normalized mean loudness and sharpness of the compressed files while Figures 2.14 and 2.15 present the roughness and tonality values. It can be seen from Figure 2.12 that the G729 coder results in a louder synthesized sound than the FS1016 coder while Figure 2.14 indicates that the FS1016 coder produces the rougher sound, Figure 2.13 shows that the FS1016 coder also produces

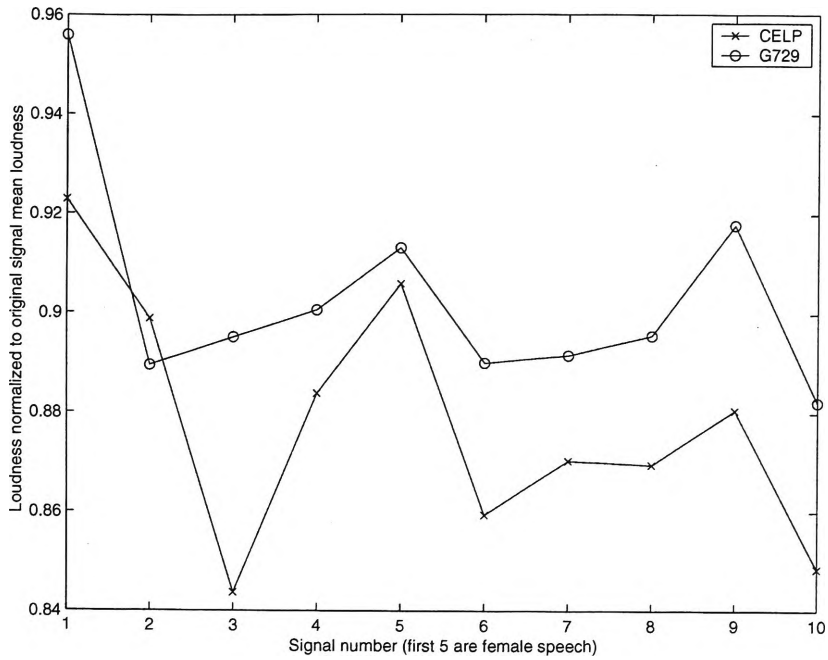

Figure 2.12: Normalized loudness of the coded narrow band speech

a sharper sound than the G729 coder. Finally, Figure 2.15 shows that the tonality of the synthesized sounds varies but tends to be higher than the tonality of the original sound.

The results show that the FS1016 coder is rougher and sharper than the G729 coder. According to [ZF99], as the roughness of a sound increases, the pleasantness decreases and similarly as the sharpness increases pleasantness again decreases. The results are in line with subjective results that suggest that the G729 coder produces synthesized speech of higher perceptual quality than the FS1016 coder [Cox95]. On the other hand, the loudness of the G729 coder is higher than that of the FS1016 coder which suggests that the G729 coder is less pleasant than the FS1016 coder. It should be noted that the loudness results presented in [ZF99] show a considerable amount of scatter and the model presented is less accurate than the models presented for sharpness and roughness.

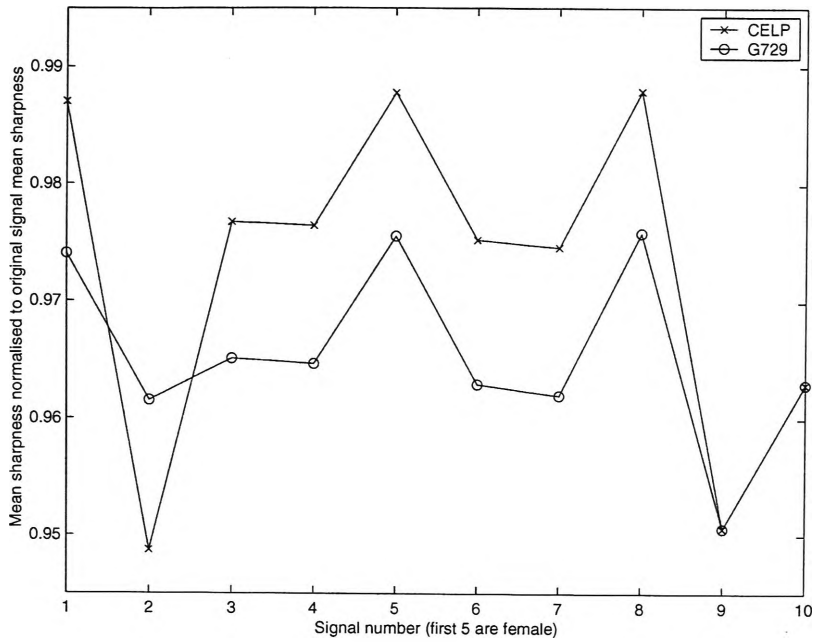

Figure 2.13: Normalized sharpness of the coded narrow band speech

The inconclusiveness of the tonality result can be simply explained as a direct result of the fact that both coders utilize linear prediction and post filtering as the basis of speech coding. Linear prediction, by the use of an all-pole model generally increases the tonality of a signal and this is deliberately enhanced further by post filtering, hence the tonality of the synthesized speech appears to be higher than the original speech. Significantly, in a consistent manner between the speech files similar curve shapes result.

The previous discussion shows how the pleasantness contributing factors can be used to adjust a coding scheme so that it is tuned towards producing more pleasant coding noise. The aim is not to hide the noise or to mask it, it is to shape it so that the final result is a more acceptable sound.

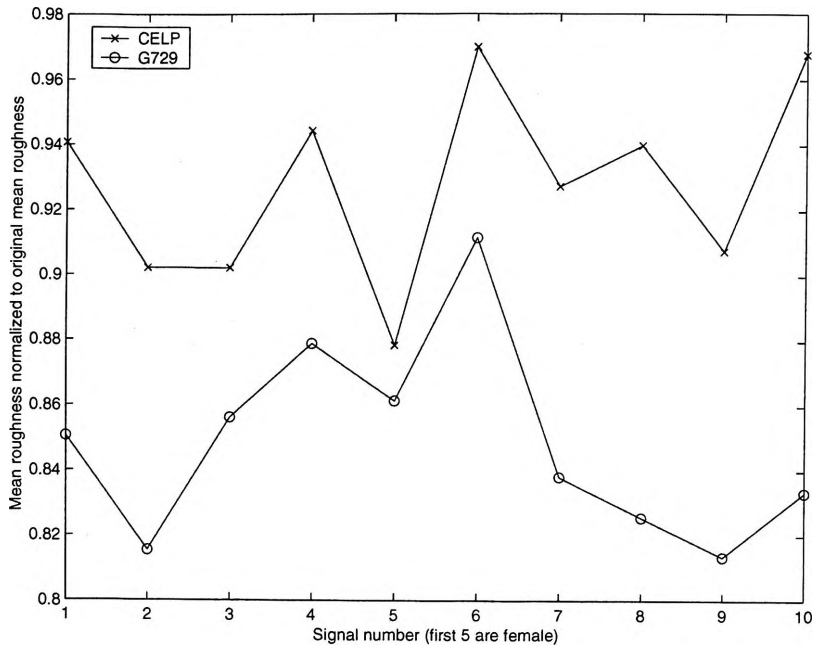

Figure 2.14: Normalized roughness of the coded narrow band speech

2.5 Audio quality assessment

The previous discussions about masking and pleasantness as well as the description of the use of masking in perceptual coding of audio raises the question of how should one assess the quality of the synthesized audio. If a waveform matching algorithm is being used, one could reliably use the SNR as an indicator of the quality. However, when one considers the possible number of signal components that are actually masked, and hence coarsely coded, the SNR becomes inaccurate as an indicator of perceptual signal quality. For example, it is well known that signals that have an SNR of 13 dB can still sound very similar to the original signal [BBT96]. When one compares this value with that of 38 dB used to indicate toll quality speech [NJ84], then the inaccuracy of the SNR becomes clearer.

Nevertheless, accurate audio quality assessment is very important. The reason

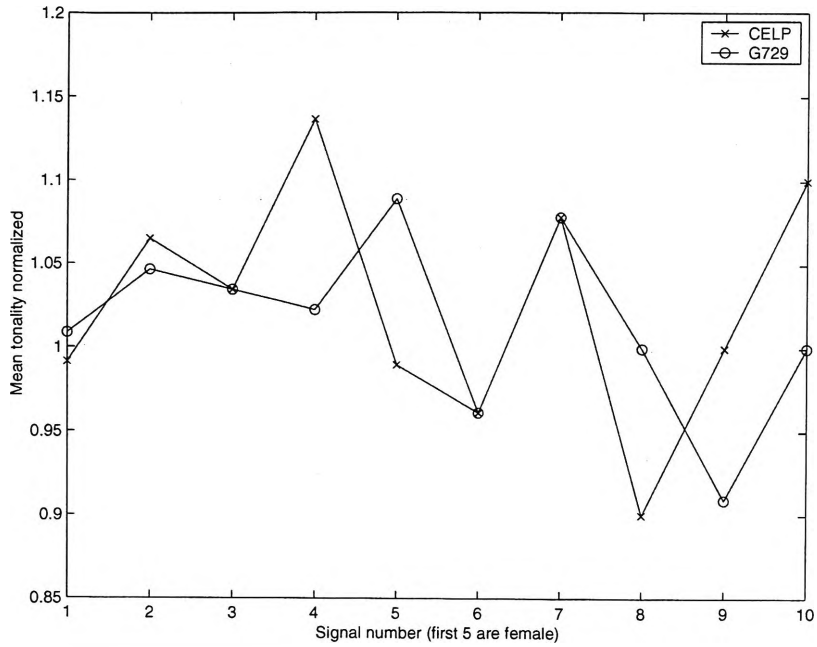

Figure 2.15: Normalized tonality of narrow band speech

is that audio compression algorithms are part of large systems (such as broadcast systems or telephone systems) that have considerable commercial value. That commercial value is enhanced when high quality material is delivered at low cost. Thus, it is necessary to use assessment techniques that result in an accurate indication of the quality of the transmitted or stored processed audio. To date, the most accurate assessment techniques have involved subjective testing [Ryd96], [PS00]. The subjective tests aim to obtain an average value of the perceived coding noise for a particular algorithm. However, subjective tests are costly, time consuming and the results tend to be relevant only in the context of the conducted test [Ryd96]. These cons have given impetus to the development of objective perceptual measures that quantify the perceptual response of people to audio coded by the use of a particular algorithm [Tt00]. In the following, subjective test techniques will be described. This will be

followed by an explanation of recent advances in objective quality assessment.

2.5.1 Subjective quality assessment

Subjective quality assessment is a multi-step process that beings with the design of the test to the statistical analysis of the results obtained [Ryd96]. Subjective audio assessment has been primarily based on the ITU-R Recommendation BS.116 [Ryd96]. This recommendation specifies that the test should be a double-blind test, conducted with a trained set of listeners and involve a continuous scale between 1 and 5 [Ryd96], [PS00]. The term double-blind means that neither the person conducting the test nor the listener know the order of the material being played. The listener is given three sources: A, B and C. The reference signal (the one being compared to) is always at source A. The coded signal and the reference signal are then randomly assigned to sources B and C. The listener is asked to indicate on a scale form 1 to 5 the impairment that is heard when comparing sources B and C with A. The scale is actually continuous and is divided as shown in Table 2.2. The listener is trained before the testing. The purpose of the training is to help reduce the variance of the results amongst different listeners rather than to instruct the listener on how they should vote [Ryd96]. The training of listeners has been found to be beneficial as different people allocate different importance to the impairments heard [PS00]. The tests should also be performed with one subject at a time [Ryd96].

The analysis of the obtained results must be approached carefully. Straight statistical analysis techniques of ignoring outlying results should be avoided [Ryd96], rather, if any results are going to be ignored at all, all the results associated with

Table 2.2: ITU-R Rec. BS.1116 small impairment scale

Perceived Impairment	Difference grade	Absolute grade
Imperceptible	0.0	5.0
Perceptible but not annoying	-0.1 → -1.0	4.9 → 4.0
Slightly annoying	-1.1 → -2.0	3.9 → 3.0
Annoying	-2.1 → -3.0	2.9 → 2.0
Very annoying	-3.1 → -4.0	1.9 → 1.0

an individual subject should be ignored [Ryd96]. This practice is used because different people have different bias or preference to audio material and this preference will influence all of their judgements of audio quality. The number of subjects used affects the conclusions that may be drawn from the test. The larger the sample space, the more general the conclusions. A sample space of 20 subjects is usually deemed adequate [Ryd96]. However, one cannot simply compare coders across different tests with different subjects. If two coders are to be compared then ideally they should be included in the same test so that the same subjects rate the coders. Otherwise it has to be made clear that different groups of subjects were used to judge the coders [Ryd96], [PS00]. The reason for this is consistency, although if the tests are sufficiently large in terms of number of subjects then one would expect that the results across different tests should be approximately the same. Finally, the score given to the coder using the BS.116 approach is an average difference grade (which is a negative number) . For a coder to be deemed transparent, none of the audio material tested should have a score below -1.00 difference grade. Very good quality coders score an average difference grade greater than -1.00.

The test described previously applies to audio coders that are of high quality and (usually) medium to high bit rate. This test is thus not very useful when testing

Table 2.3: ITU-R Rec. P800/P.830 scale

Quality	Score
A much better than B	+3
A better than B	+2
A slightly better than B	+1
A same as B	0
A slightly worse than B	-1
A worse than B	-2
A much worse than B	-3

low rate coders which are guaranteed to introduce audible impairments. An alternative test is the ITU-R Recommendation P.800/P.830 which allows a seven grade comparison between two audio signals. The subjective results range from source A much better than source B to the opposite. The scale is not continuous, rather the subject has a choice between seven grades [PS00]. This test is used for making a choice between available coders that are known to introduce artifacts into the audio signal. Table 2.3 shows the grading criteria of this test.

A similar subjective test that has been very popular in speech coder evaluation is the Mean Opinion Score (MOS) test [DPH93]. This test utilizes a scale similar to the 5-Grade impairment scale as suggested (shown in Table 2.4) in ITU-R Recommendation BS.562-3 to judge the quality of the coded speech. This test results in a an average score, with very high quality coders scoring close to 4.5 which is the MOS for toll quality speech [NJ84].

Considering the effort that is required to produce acceptable test results, the use of subjective tests in the development phase of audio coders is an expensive and time-consuming approach. It is also an imperfect approach, as much research is still to be done on how the results of such tests should be analyzed to reflect the true nature of

Table 2.4: ITU-R Rec. BS.562-3 scale (MOS)

Quality	Score
Very annoying (close to noise)	1
Annoying	2
Slightly annoying	3
Perceptible (but not annoying)	4
Imperceptible	5

the perceived quality of the coder [Ryd96]. Subjective tests are highly influenced by factors that are not related to the audio coder itself. Besides the already mentioned individual tastes and bias, there is also the equipment bias and the environment bias [Ryd96]. The equipment on which such tests are conducted rarely matches consumer equipment and so such tests are not truly reflective of how consumers will assess such coders [Ryd96]. The environment in which such tests are conducted is also not reflective of every-day environments that often contain unpredictable noise and masking factors [Ryd96]. Having said that, subjective tests remain the only respected way of choosing audio coders [Ryd96], [PS00].

2.5.2 Objective quality assessment

The time and financial cost of subjective tests make them impractical in a number of areas that require audio quality assessment [Tt00], [BBT96]. For example, the on-line assessment of audio quality cannot be carried out subjectively. Distributors of audio over telecommunication networks (including television and telephone) need some way of determining the audio quality in a near instantaneous time frame. Complete subjective tests are also impractical in the development phase of audio compression

algorithms. The prohibitive nature of subjective test costs lead to "informal" listening tests that quite often result in an inaccurate assessment of how test subjects will respond. This can lead to costly development errors. The existence of these requirements and others is the reason behind the interest in objective quality measures that correspond to the subjective quality measures already mentioned.

The first measure that will be mentioned here is the SNR and its derivatives. The SNR is a useful measure for waveform matching algorithms. It has already been pointed out that a low SNR does not necessarily mean bad signal quality, however it should be remembered that a high SNR does mean good signal quality, for obvious reasons. The SNR is rarely applied as a global measure when assessing sound signals as it can be heavily biased by impulsive sections of speech and audio, rather the Segmental SNR (SegSNR) is normally employed [DPH93], [GG92], [NJ84]. The SegSNR is simply the average of the frame SNR value calculated for frames ranging in length between 15 and 25 ms (to maintain stationarity). The frames on which the SegSNR is calculated are not necessarily the same frames chosen in the coding process, this can help produce a better indication of the global quality of the signal. Other objective measures used include the Itakura distance measure, frequency weighted SegSNR and the weighted spectral slope measure [DPH93]. All of these measures compare the spectral representation of the original signal with that of the processed signal. Various processing is applied to the original signal to obtain a representation of the underlying perceptual signal, although perceptual issues are not taken into account directly [DPH93].

The objective measures mentioned thus far have only had limited success with predicting the subjective evaluation of speech and audio signals. An attempt to

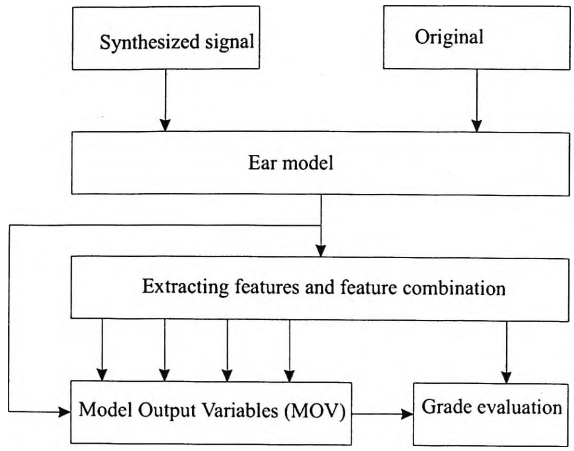

Figure 2.16: An overview of the PEAQ scheme

capture the subjective evaluation by humans of audio quality is presented in [BBT96]. A very brief description of how this measure works would be to say that it compares the evaluated response to both the original signal and the processed signal and grades the result. Results presented in [BBT96] show that there is good correlation between the predicted results and actual subjective results. Note, however, that the system compares the original signal with the processed signal. This is reflective of popular audio assessment techniques but is not reflective of MOS testing (which does not always include the original). An objective model for speech quality assessment has been presented in [HK00] with good correlation obtained between MOS scores and the presented objective measure. The objective measure presented in [HK00] is a correlation factor between the original signal's auditory response parameters and that of the processed signal, the higher correlation the expected MOS score.

The perceptual objective measures discussed thus far are only a sample of the work that has been conducted in this area [Tt00], as an exhaustive discussion of all the techniques proposed would be too lengthy to include here. However, one measure that demands some attention is PEAQ which is the ITU's relatively new standard for

objective audio assessment [Tt00]. Figure 2.16 shows the operation of this measure from an overview perspective. PEAQ includes a model for the human ear (which may be implemented by the use of a filter-bank or the FFT) which performs the time to frequency transformation of the original and processed signal. The transformed signals are used to calculate the excitation patterns, loudness and modulation patterns in a similar fashion to that mentioned in Section 2.4 and [ZF99]. These patterns are fed into a subjective model and a neural network which attempt to mimic human response to the given signals. The outputs of the whole system are a distortion index and an objective difference grade (the equivalent of the subjective distortion grade). Results presented in [Tt00] and [TS00] show very good correlation between this measure and the subjective results. PEAQ does have two modes of operation, a basic mode and an advanced mode, the basic mode is the FFT based approach which increases the computational efficiency of PEAQ. A simple analysis of the PEAQ system would imply a high computational complexity for the filter model based approach, this issue is not addressed in [Tt00], however the filter based model works considerably better than the FFT based model indicating that a heavy computational load would have to be tolerated to obtain an accurate prediction of the subjective quality.

2.6 Summary and conclusion

This chapter has outlined the main concepts that form the basis of audio compression. Transform theory has been reviewed from the perspective of transforms that have been used in audio compression algorithms. The discussion has shown the evolution in the transforms used in audio coding; from block transforms to the lapped and wavelet

transforms. Quantization was also looked at in a similar manner. Adaptive and non-adaptive techniques were discussed where the adaptation is carried out based on signal statistics or perceptual measures. Recent signal models have also been discussed.

The perceptual concepts presented in this chapter have focused on explaining masking, its causes and applications. Sensory pleasantness was also presented, however, this is of more interest as an analysis tool than a coding tool. The example analyzed two speech codecs and allowed a discussion of the perceptual effects that the coding algorithms employed have on the synthesized speech. This led to the discussion about audio quality assessment, both objective and subjective.

It can be concluded from the presented discussions that the application of transforms in audio compression should focus on lapped or wavelet style transforms, this is the case with current audio coders as will be seen in the next chapter. Signal models can be an effective way of dealing with audio compression, again the next chapter will show the current popularity of these models. Finally, it is noteworthy that the development of audio compression algorithms is streamlined by the availability of effective objective measures as reliable subjective tests are costly and time consuming. The pleasantness analysis presented has been shown to be of potential use in audio compression design.

Chapter 3

A Review of Perceptual Audio Compression

This chapter presents a review of the audio compression literature that is relevant to this thesis. The chapter builds an overall picture of the most popular approaches to audio compression by focusing mostly on standardized as well as commercially successful techniques. This chapter will show the shift in perceptual audio compression from algorithms aimed at optimizing the perceptual results for a set bit rate to algorithms that can easily scale from low rates to high rates. The growing interest in lossless audio compression is also discussed.

3.1 Introduction

Audio compression may be seen as beginning in earnest during the mid 1980's [PS00], previously much of the focus was on the compression of speech signals which have a significantly narrower bandwidth and are one specific component of the audio field. Since then, one of the most popular and successful approaches to audio compression has been transform coding [PS00], [EEM+97]. This approach involves the conversion

of the audio signal from the time domain into the frequency domain and the distribution of the quantization bits across the frequency bins in some desirable manner. Another approach has focused on modelling the audio signal as either the sum of simpler signals or the combination of parametrically synthesized signals.

In this chapter a thorough review of audio compression techniques will be given. The focus will be on perceptual transform techniques as well as perceptual parametric techniques that have shown promising performance. To facilitate a more complete discussion of these algorithms, the starting point will be a general look at the principles behind these schemes.

It will be seen that audio coders tend to be medium rate coders or higher (that is, around the 40 kbps mark and above). Low rate coders (starting from as low as 6 kbps) will also be looked at. The scalable coders will be given more attention than others as the focus of this theses is on scalable perceptual audio compression. It will also be seen that the driving force behind the continuing research in audio compression is the best achievable quality at a given bandwidth.

The following section outlines the basic concepts of audio compression. This is followed by a description of specific audio compression standards, that is the MPEG audio standards, as well as the major commercial algorithms available including PAC, ATRAC and AC-2/3. Other coders will also be reviewed according to category, that is, whether they are transform based coders, parametric coders and hybrid coders.

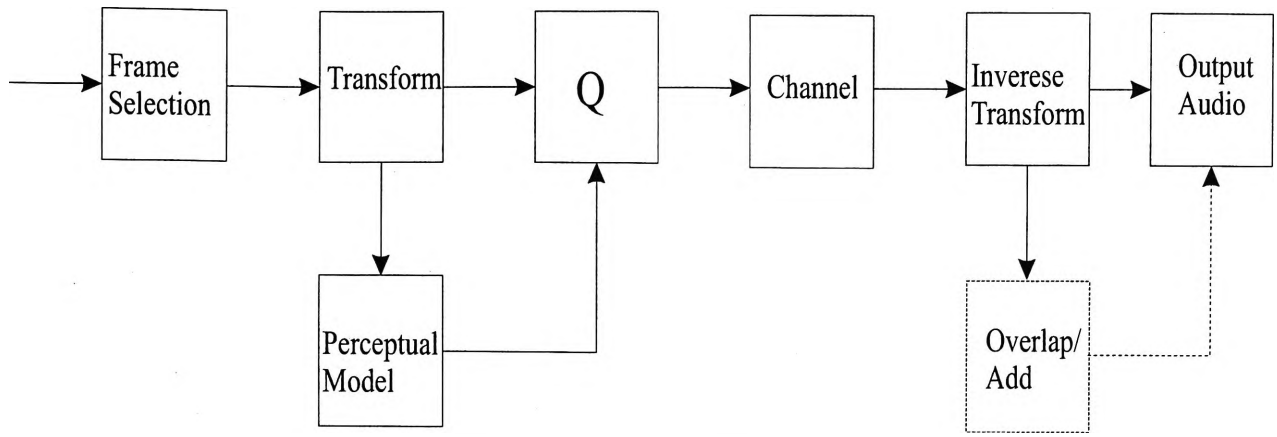

Figure 3.1: Transform coding of audio

3.2 Transform and sub-band audio compression basics

Transform audio compression is diagrammatically described by Figure 3.1. Although transform coding is not the same as sub-band coding, the structure of the coding schemes are usually very similar. The most obvious difference between transform coding and sub-band coding is the division of the spectrum that each coding scheme applies. However, it will be seen that this distinction is quite often blurred in audio coders. The following subsections will discuss the major components of these schemes.

3.2.1 Frame selection

Transform coding is typically built to operate on a frame by frame basis, the frame length chosen depends on the algorithm used and it may be variable . At 44.1 kHz, frame lengths of 512 and 1024 samples are popular although other frame lengths may be used, for example a frame length of 882 samples is employed in MPEG-4 [BKS00]. In terms of time, frame lengths between 15 ms and 25 ms are normally

chosen to maintain the pseudo-stationarity of the audio segment and to ensure low delay with minimal perceptual effects. The issue of pseudo-stationarity is of more concern to parametric coding schemes because of the assumption that the probability distribution of the signal is independent of time, however, the other considerations of delay and perceptual effects are quite relevant to transform coding as well. The meaning of "perceptual effects" will become clearer in later sections of this chapter; here we shall only mention that the human perceptual system is time dependent. This means that some coding artifacts may be hidden by correctly spreading them in time. Dividing the audio signal into separate frames has the added advantage of allowing the editability of the audio signal [EP00], whereby the signal may be changed by either changing its parametric or transform representations.

There have been a number of algorithms proposed that actually adapt the frame length [BKS00][Ver99][Lev98]. This variation in frame size is aimed at improving the perceptual performance of the coding algorithms. However, the uncertainty principle sets a limit on the performance of these algorithms. The uncertainty principle states that there is a trade-off between the time domain resolution and the frequency domain resolution. Mathematically, this trade-off is given by [Mer99]:

$$\delta_t \delta_\omega \geq \frac{1}{2} \tag{3.2.1}$$

where δ_t and δ_ω are the resolution in time and in frequency respectively. The variation in the frame length is usually used as a tool for the reduction of pre-echo distortion [BKS00],[PS00] whereby the impulsive sections of the audio signal are coded by the use of shorter frame lengths than steadier sections. Pre-echo effects are an issue in transform coding as the quantization error in each bin of the frequency domain is

distributed across the entire frame in the time domain. When an impulsive component of the signal occurs at the end of a frame, the frequency domain quantization errors cause audible errors in the time domain which occur before the synthesized impulsive section; i.e. they are heard before the impulsive section and hence the name 'pre-echo' [PS00].

As a direct result of the uncertainty principle, the change of frame length in the time domain resolution leads to an opposing change in the frequency domain resolution. Thus, increasing the time domain resolution will lead to a decrease in the frequency domain resolution. This may result in undesirable frequency domain effects for the transform being employed, however, the perceptual gains are generally viewed as more important than this potential drawback; hence, the deployment of frame length varying algorithms [Ver99][Lev98][HJ97].

The frame selection algorithm in an audio compression scheme must also account for the overlap between the frames. Overlap between audio frames becomes necessary when one employs a non-rectangular window to improve the frequency domain performance of the transform being used. The window of traditional choice is the Hamming window [GS92], however, the recent popularity of the Modulated Discrete Cosine Transform (MDCT) has lead to the deployment of PR windows which have better frequency domain properties whilst allowing the PR of the original audio signal [Mal92].

Windowing also necessitates overlap in order to avoid time and frequency domain aliasing of the original signal [RS78]. As the multiplication by a window in the time domain has a filtering effect in the frequency domain, the time-domain sampling of the frequency representation of the signal changes accordingly. In order to recover the

original signal it is important to ensure that the frequency domain representation is evaluated in the time domain at twice the "time-domain bandwidth" of the window, just as it is equally important that the signal's time domain representation is sampled at twice the frequency domain bandwidth of the signal [RS78]. In the case of a Hamming window, the shape of the window is defined by the equation:

$$w(n) = \begin{cases} 0.54 - 0.46\cos\frac{2\pi n}{M-1} & 0 \leq n < M \\ 0 & \text{otherwise} \end{cases} \tag{3.2.2}$$

and the bandwidth of the window is given by:

$$W = \frac{2f_s}{M} \tag{3.2.3}$$

where W is in Hz, fs is the sampling frequency and M is the length of the window. The sampling rate of the frequency domain representation is $2W$. Thus, for a 44.1 kHz sampled signal, if a window length of 1024 samples is used then $W = 86$ Hz and $2W = 172$ Hz. This gives the necessary condition of evaluating the frequency domain representation every 256 samples. Thus, the overlap for a Hamming window of length 1024 samples would be 768 samples.

Other windows have been developed to specifically address the issue of time-domain aliasing [Mer99] and so require a different overlap to the Hamming window to obtain a perfect reconstruction of the original signal. One such window is the sine window [Mal92] given by:

$$w(n) = \frac{1}{\sqrt{M}} \sin\left((n + \frac{1}{2})\frac{\pi}{M}\right) \qquad 0 \leq n \leq \frac{M}{2} \tag{3.2.4}$$

In the case of the sine window, the overlap required is half the frame length. Under-sampling of the frequency domain representation of a signal and maintaining PR is

possible under special circumstances [RS78]. Perfect reconstruction is not generally the aim of an audio compression scheme, the main aim is usually perceptual perfect reconstruction where the synthesized signal and the original signal sound exactly the same without necessarily having zero error between them. In this thesis, however, one of the aims has been the development of a lossless compression scheme, in which case PR is a specified requirement.

3.2.2 The transform

The previous discussion on frame selection is related directly to the choice of transform. At this point it is worthwhile mentioning the reasoning behind transforming a signal from the time-domain into another domain. Transforms are designed to act like decorrelating devices [Beu84], i.e., the correlation between the transform coefficients is meant to be much less than that of the original coefficients or samples. In this way, most of the energy of the signal will be concentrated in a few coefficients leading to a smaller error between the original signal and the reconstructed signal for a given bit rate [Mal92]. If one was to employ a scalar quantizer in both the time domain and the frequency domain, then the gain of transform coding is defined as the increase in Signal-to-Noise-Ratio (SNR) for transform coding over Pulse Code Modulation (PCM). This can be calculated from the following equation [Mal92]:

$$\Gamma = \frac{\frac{1}{M}\sum_{\ell=0}^{M-1}\sigma_{ell}^2}{\left(\prod_{\ell=0}^{M-1}\sigma_{ell}^2\right)^{\frac{1}{M}}} \tag{3.2.5}$$

In (3.2.5) the σ_ℓ's are the transform coefficient variances and Γ is the coding gain (M is the frame length).

It is necessary to point here to the different types of transforms that may be used

in transform coders. Using the equation for coding gain as a measure of optimality, the optimal transform that may be used in combination with a scalar quantizer is the Karhunen-Loéve transform [Mal92][Mer99] as it minimizes the geometric mean of the transform coefficient variances (which happens to be the denominator of Equation (3.2.5)). However, currently, lapped transforms continue to be the most popular transforms in the proprietary and standardized coders [BKS00][Dav99][Nol97][PS00]. The wavelet transform is also quite popular as it allows the audio signal to be divided into non-uniform frequency bands that are hierarchical in nature [PS00][SJ96][ST93b]. Transforms that combine both time and frequency parameters have been used for the analysis of music signals [PWS96], although their use in audio compression is rare.

3.2.3 The perceptual model and quantization

Having transformed the audio frame, the obtained coefficients must be quantized for digital transmission or storage. The coefficients may not be quantized explicitly (implicit quantization is used in integer-to-integer transforms for example) however, it is the usual case that they are [GG92]. Transform coders tend to employ scalar quantizers with good results [Goy00]. These scalar quantizers are usually adaptive as they take into account perceptual effects [BKS00], [PS00], [EEM+97]. Taking account of perceptual effects dictates that the more perceptually significant coefficients should be quantized using more bits than the other coefficients. Stated in another way, quantization noise is shaped such that it corrupts perceptually insignificant coefficients rather than perceptually significant coefficients.

The perceptual model may operate in both the time and frequency domains. As an aside, this is a useful application of time-frequency transforms in audio compression

as such transforms may allow the use of the total perceptual model to reduce the required bit rates. The perceptual model helps to identify the perceptually significant components of the audio signal, hence reducing the overall bit rate required for the perceptual reconstruction (and not necessarily the waveform reconstruction) of the original signal [Joh88a].

Transform audio compression is the core of the compression techniques presented in this thesis. The advantages that this technique offers are related to scalability and perception. The use of a transformation in compression allows signal reconstruction both partially and completely. This clearly helps when one is determined to allow scalability in the coding scheme. The disadvantage is that at low and very low rates, the quality of a transform coder tends to decrease at an increasing rate [KP95]. Also, at the higher rates, the quality increases at a decreasing rate (i.e. it saturates). The sharp decrease can be addressed by employing hybrid coding techniques which combine both parametric and transform coding. On the other hand, the saturation may be addressed by designing the coder to scale to lossless compression. The saturation is not a concerning phenomena in low rate compression, simply because it is of general interest to have good quality at low rates and in doing so saturation is guaranteed, especially in terms of perception.

The parametric coders which are generally employed for low rate compression are the next focus of this chapter.

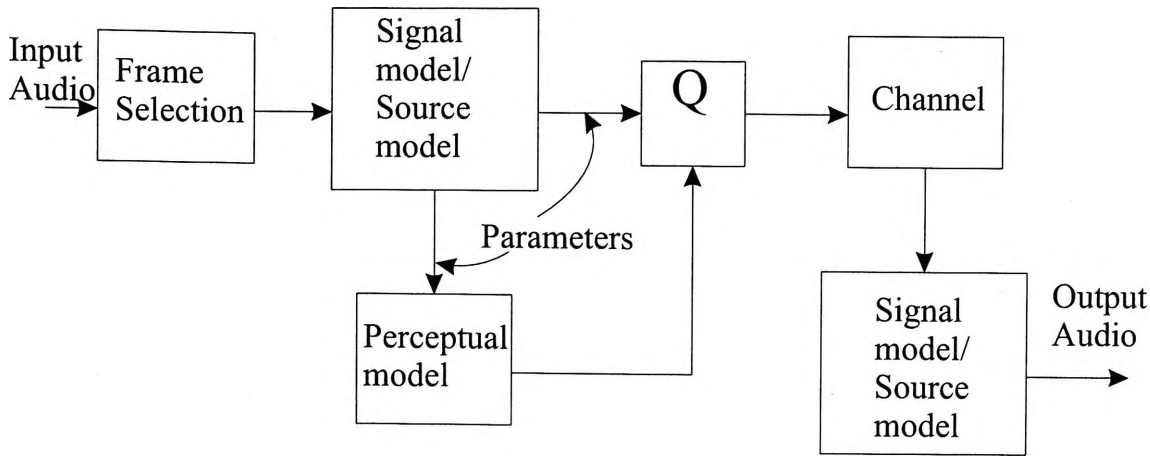

Figure 3.2: Parametric audio coding basics

3.3 Parametric audio compression basics

Parametric signal compression is concerned with the description of the given signal by the use of some model. Figure 3.2 shows the basic components of current popular parametric audio coders. The fundamental difference between a transform coder and a parametric coder is the signal or source model component. A popular signal model for audio is the sinusoidal model [GS92][EP00][PS00] which defines the signal as a sum of sinusoids. The parameters of this model are the amplitudes, phases and possibly frequencies (depending on the implementation actually used). In order to generate the signal model a transform could be used as a tool. In the case of sinusoidal modelling, the Short Time Fourier Transform (STFT) was used to generate the original sinusoidal model [MQ95].

Another popular audio modelling techniques utilizes linear prediction. This is primarily used in speech compression as the speech production process is modelled by an all-pole filter that is excited by colored noise [RS78]. The linear predictor model in audio compression has been applied extensively in lossless audio compression [HS01]

but it has not enjoyed the same popularity in medium and low rate audio compression [PS00].

Frequency Modulation (FM) has also been proposed as a possible model of audio [PS00][Cho73]. Limited success has been reported, however the idea is a novel one and it is based on the observation that an FM signal spectrum in many ways resembles an audio signal spectrum [Cho73].

In terms of signal models for scalable coders, the Sinusoids plus Transients and Noise (STN) model has been shown to provide a tool for allowing smooth scalable compression [Ver99][Lev98]. The STN model, as the name suggests, decomposes the audio signal into sets of sinusoids, transients and noise. The sinusoids are used in the same manner as in the sinusoidal model while the transients are added to the synthesized signal at higher bit rates to increase the similarity between the original signal and the synthesized signal. The noise is inserted at low bit rates, that is before the transients, and is used to color the spectrum of the synthesized signal. The reason behind using colored noise is that the sinusoidal model on its own produces a highly tone like spectrum at low bit rates [Ver99]. The sinusoidal model is also inappropriate for a number of audio signals that have a noise like composition [Ver99]. The term "noise like" refers to the similarity between the spectrum of the audio signal and that of white noise, i.e., a flat spectral shape. For example, the Harpsichord tends to produce a noise like signal. Figure 3.3 shows the Power Spectral Density (PSD) of a tune played on a Harpsichord. The flatness of the spectrum is an indication that such a signal cannot be modelled by a sum of only a few sinusoids. Results presented in Chapter 4 will show the difficulty that a pure sinusoidal model has with representing such a signal.

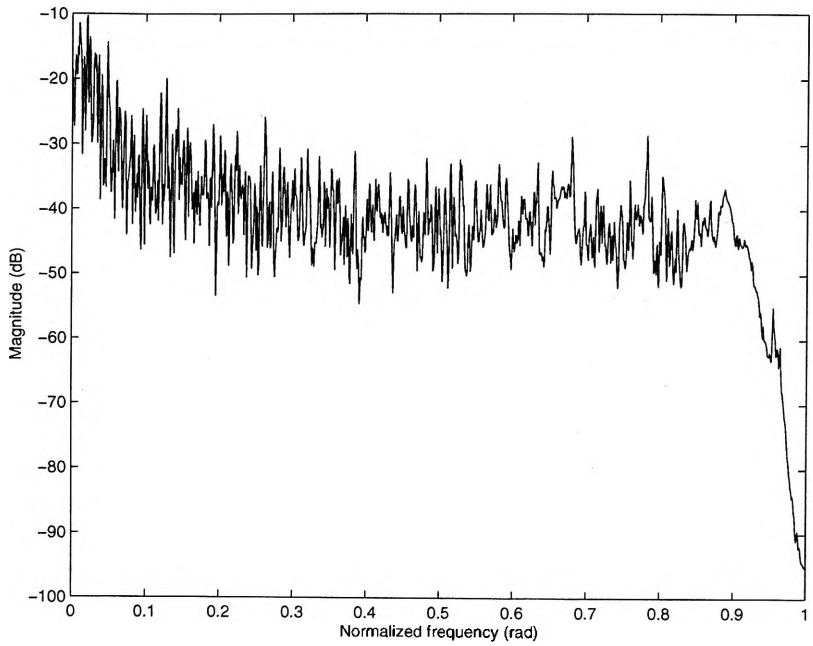

Figure 3.3: PSD of the Harpsichord signal

The term "noise model" is somewhat misleading, as most such models try to shape white noise by coloring it spectrally [PS00][Ver99][SA99]. The shaping of white noise may be carried out in a number of ways, and noise modelling is still a rich area of research [Ver99], [SA99]. Fundamentally, the techniques apply time-varying filters to a known white-noise sequence in order to obtain the color required.

Returning to Figure 3.2, one may notice that the signal model may be used in frame selection. The model determines the length and/or overlap of the frames. This is particularly true in the case of sinusoidal modelling[MQ95][GS92] where the frame length tends to vary depending on the pitch of the input signal. The variation in frame length aids some parametric coders in better representing the input audio signal.

Finally, a comment about the quantization schemes employed in parametric coders. Unlike transform coders that must quantize coefficients that are uncorrelated, parametric coders must efficiently represent groups of parameters. It is noticeable that

Vector Quantization (VQ) is used in parametric coders more often than in transform coders [PS00]. The reason is that parametric coders seek a high subjective similarity between the original and synthesized signal at low rates without being concerned with waveform matching. It is well known that in audio and speech compression there is only a small link between SNR and subjective quality [Tt00]. This means that the approach taken by subjective parametric coders will produce very good quality at low rates, however, this same property is also a hinderance in that scalability is limited. Throwing an increasing number of bits at a parametric coder improves the subjective quality up to the ceiling set by the parametric model used and is unlikely to improve the objective quality [KP95].

3.4 Scalable audio compression- why and how ?

To this point the discussion has been focused on the basics of a number of audio coding techniques; transform and parametric coding. The main aim of this thesis is the presentation of new scalable coding techniques that may have a wide variety of applications. Scalable compression should allow the extraction of different quality audio from the same bit stream. Alternatively, one may view a scalable compression technique as one that does not have to be altered to obtain higher quality for an increased bit rate. Possible applications of such schemes include preferential sale of audio, the advertising of good quality audio at the lower rates and so on.

The current direction of research in scalable audio compression tends to focus on the parametric compression approach [Lev98][Ver99][EP00][BKS00]. The primary reasoning behind adopting such an approach is that it is possible to decompose the

signal into a number of representations that may be combined together in an embedded bit stream. A completely embedded bit stream should ideally add value for every bit transmitted, in the current available literature, the bit streams that are referred to as embedded actually add value for a defined number of bits together. Scalable transform coders tend to be rarer than their parametric counter-parts. A scalable transform coding scheme was presented in [JMN+99] where the MDCT was combined with TwinVQ quantization in a hierarchical manner to obtain scalability. Again, this scheme obtained tends to deliver large step scalability rather than fine grain scalability. A scheme presented in [LP98], which will become of increased relevance in later chapters, utilizes a quantization technique that allows very fine grain scalability, although the actual coding scheme described in [LP98] was not presented as a scalable coder.

3.5 Lossless audio compression - why and how ?

Lossless audio compression enables the reproduction of the exact digital audio signal using a reduced bandwidth or memory. The current state of the art in lossless audio compression applies some kind of decorrelation tool to the time domain samples and codes an error signal by the use of an entropy code [HS01]. The decorrelation tool employed is usually a linear predictor, or less often, a transform [HS01],[LPN97]. Figure 3.4 illustrates the basic technique underlying lossless compression.

The reason behind lossless compression is very simple; if very high quality lossy compression requires a large bandwidth then that bandwidth would be more appropriately utilized if the resultant synthesized signal was an exact copy of the original.

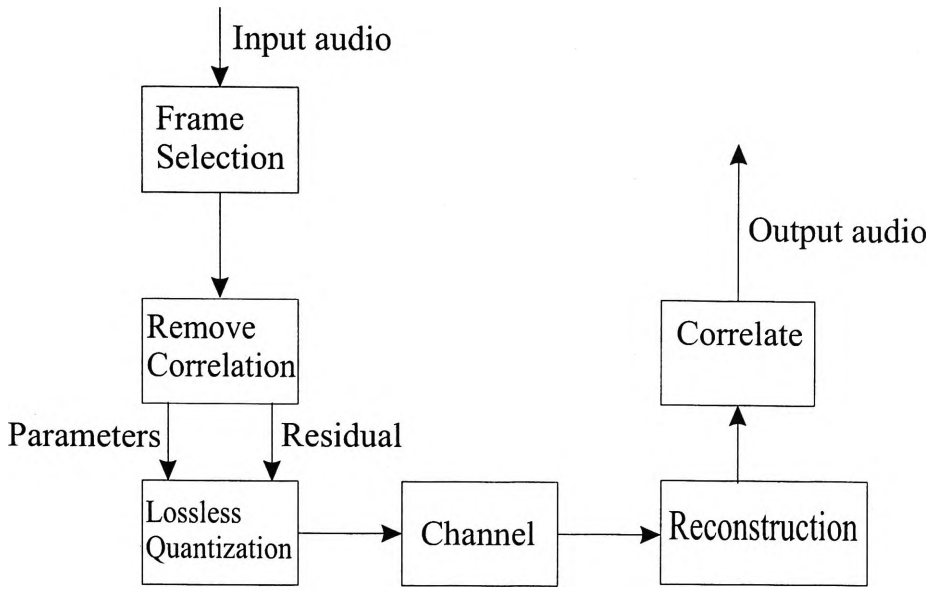

Figure 3.4: The concept of lossless audio compression

This is a logical argument, since early compression techniques (which were lossy) operated above the 100 kbps mark [PS00], i.e. at a compression ratio of 7:1. The current state of the art averages a 3:1 compression ratio [HS01]. Thus, if the lossless compression ratio were to be improved, one would find a high level of demand for such compression schemes. Chapter 6 will present a scalable to lossless scheme that actually maintains fine grain scalability. Hence, a more elaborate discussion on lossless audio compression will be deferred until then as here only the basic concepts behind different forms of audio compression are being introduced.

Having given a rather quick look at how digital audio compression is achieved, it is now appropriate to take the time to examine the state of the art in audio compression. A look at all the audio compression algorithms that have been developed over the complete history of audio compression would be impractical in a thesis such as this. Instead, the focus of the review of the state of the art will be on standardized audio coders as well as popular commercial products. The other coding algorithms will be

mentioned, but only briefly.

3.6 Audio compression standards

In the early 1990's the International Organization of Standardization (ISO) started an audio compression standardization process through the setting up of the MPEG (Moving Pictures Experts Group) audio activity [Nol97]. The aim of the exercise was to standardize an audio coder that produced transparent audio at high compression ratios. The first standard produced, MPEG-1, was the first standard aimed at high quality audio compression. This has been followed by the development of MPEG-2 and 4. MPEG-7 and 21 have also been developed, but these last two standards deal with content description and content delivery respectively.

3.6.1 MPEG-1

The first effort of MPEG-1 resulted in a coder that offered three layers of compression; layers I, II and III. Layer III being the famous MP3 coder [Nol97]. The three layers are organized in the order of increasing compression, complexity and quality. Table 3.1 lists the compression bandwidths required to achieve transparent compression for stereo signals using the MPEG-1 coders. Mono signals require rates slightly greater than half of the rates listed as the compression of a stereo signal has the advantage of reducing inter-channel redundancy.

Figure 3.5 shows the block diagram representation of MPEG-1 layers 1 and 2. Figure 3.6 illustrates the MPEG-1 layer 3 coder. As can be seen from both figures, MPEG-1 relies on the sub-band analysis of the original audio signal by the use of

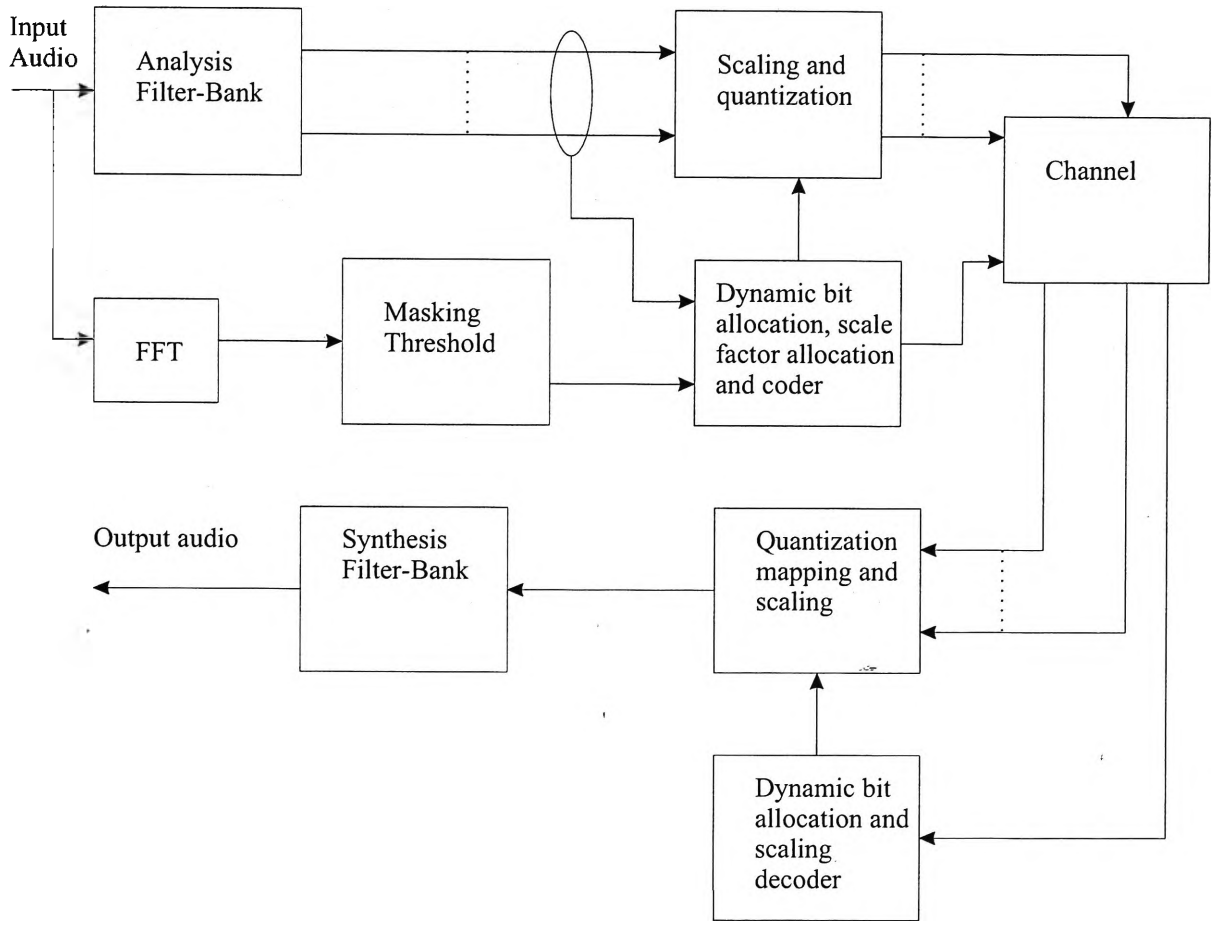

Figure 3.5: Block diagram of the MPEG-1 layer 1 and 2 codec

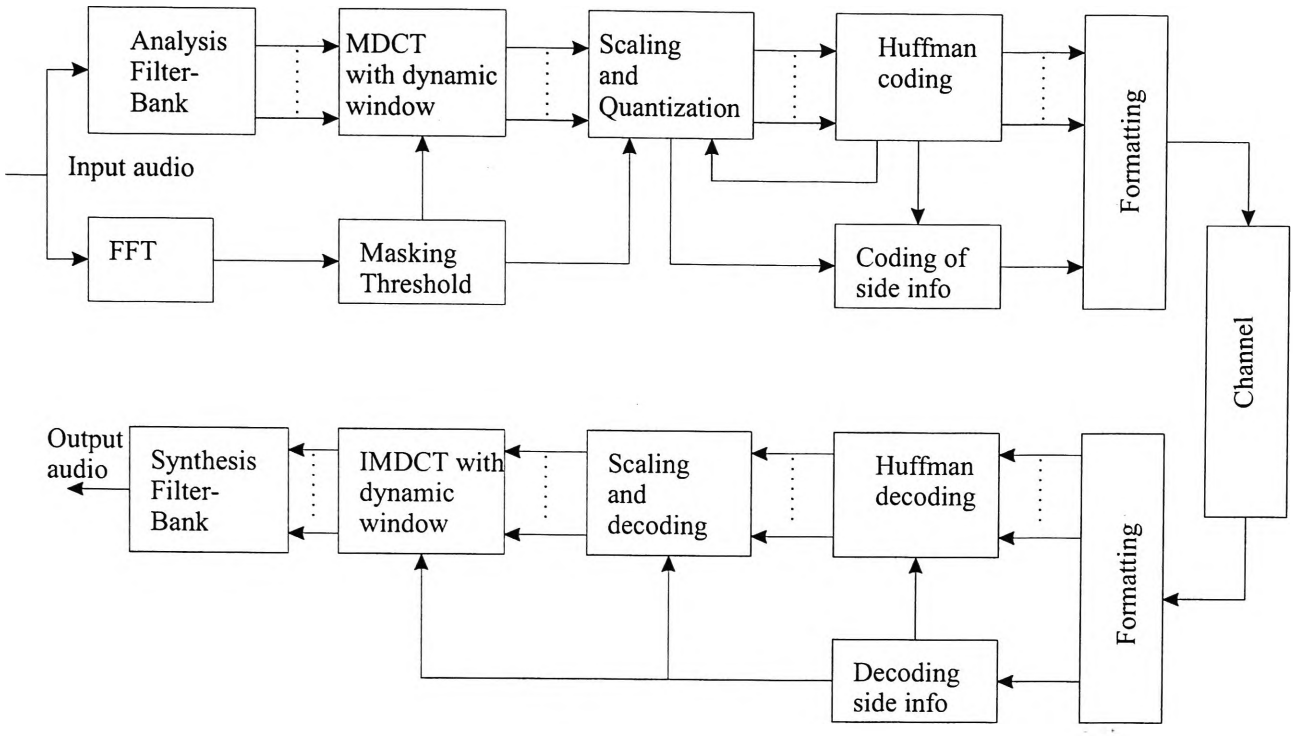

Figure 3.6: Block diagram of the MPEG-1 layer 3 codec

either a filter bank (layers 1 and 2) or the MDCT (layer 3). Two psychoacoustic models were defined for MPEG-1 (with different accuracy and complexity). MPEG model 1 is utilized for layers 1 and 2 whilst MPEG model 2 is utilized by layer 3. All three layers use the defined psychoacoustic model to determine how the frequency domain representation of the original signal should be quantized. This means that side information is produced to enable the decoder to correctly synthesize the coded signal. This method of using the psychoacoustic model is quite popular, although not very efficient, as noted in [LJ02].

For increased compression MPEG-1 layer 3 relies on an entropy coding scheme of the coder bitstream. This approach, whilst an effective compression method, does result in a variable rate scheme which could be unattractive when only a constant bandwidth is available. To counter this problem, MPEG defined a bit reservoir scheme

Table 3.1: Mean rates for transparent audio compression in MPEG-1 stereo

Layer	Mean bit rate (kbps)
I	384
II	192
III	128

that maps the variable rate bitstream into a constant rate stream limiting the bit rate to the reservoir's maximum level [Nol97].

3.6.2 MPEG-2

MPEG-1 has been a widely applied standard [BKS00], [Nol97]. The major step forward in terms of lower rate audio compression in MPEG-2 is embodied in the Advanced Audio Coder (AAC) which is not backward compatible with MPEG-1. Figure 3.7 shows the layout of the AAC coder. The filter bank used in AAC is a 1024 line MDCT (thus a frame length of 2048 is used). The perceptual model is MPEG model 2. The other significant tools in AAC for all audio signals (that is mono, stereo and multichannel) include the Temporal Noise Shaping (TNS) tool and the backward prediction tool. The TNS allows the shaping of time domain noise in order to reduce pre-echo distortion as well as improve the overall perceptual quality.

The AAC coder has three modes of operation; the main profile full AAC coder, the low complexity profile and the sampling rate scalable version. The low complexity profile does not use the TNS or backward-prediction tools whilst the sampling rate scalable version allows different sampling rates through the use of a hybrid filter bank. MPEG-2 offers high audio quality for five channels at 320 to 384 kbps [Nol97]. MPEG-2 can handle up to 46 channels and it has a simulcast mode, allowing the transmission of both MPEG-1 and 2 bit streams for backward compatibility.

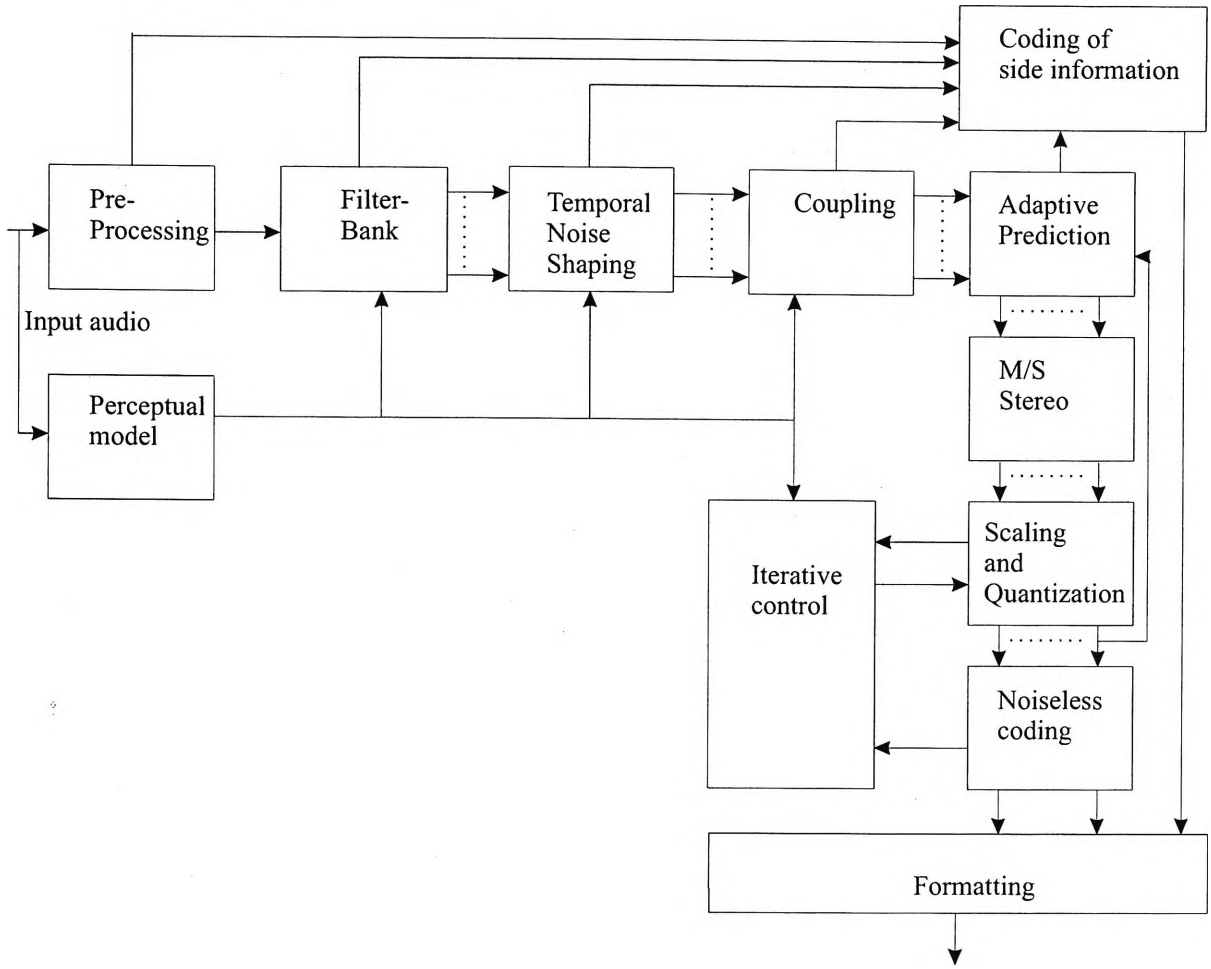

Figure 3.7: The MPEG-2 AAC coding scheme

3.6.3 MPEG-4

The most recently completed compression standard in the MPEG series is MPEG-4 (Note that MPEG-7, which is a more recent standard, deals with content description and MPEG-21, which is being developed deals with content delivery). The main theme of MPEG-4 was the development of audio compression objects [BKS00]. This allowed MPEG-4 to include techniques for scalable audio compression. The MPEG-4 activity did not find that any single audio compression algorithm submitted for consideration could scale from low to high bit rates with satisfactory results [BKS00]. This meant that MPEG-4 adopted different audio compression algorithms for different bit rates. At the lowest rates, text to speech tools were adopted. At the next level of bit rates it is recommended that the HVXC low rate clean speech coder is used. The CELP telephone speech coder comes next and the General Audio (GA) coding tool (shown in Figure 3.8) is defined to operate from the CELP region (12 kbps to 24 kbps) to higher rates. Scalability is achieved by the transmission of a base layer followed by refinement layers if the bit rate permitted is sufficient. The base layer could be generated by any one of the coders adopted. The same is true for the refinement layers. However, the refinement layers should only contain information that will improve the quality provided by the base layer and not retransmit previously transmitted information [BKS00].

The MPEG-4 GA tool introduces the use of Perceptual Noise Substitution (PNS) and TwinVQ to the MPEG standards. PNS is based on the idea that signal components that are noise-like may be replaced by other, similarly noise-like, signals [Nol97]. The definition of this tool is consistent with the object oriented approach to audio compression that MPEG-4 adopted.

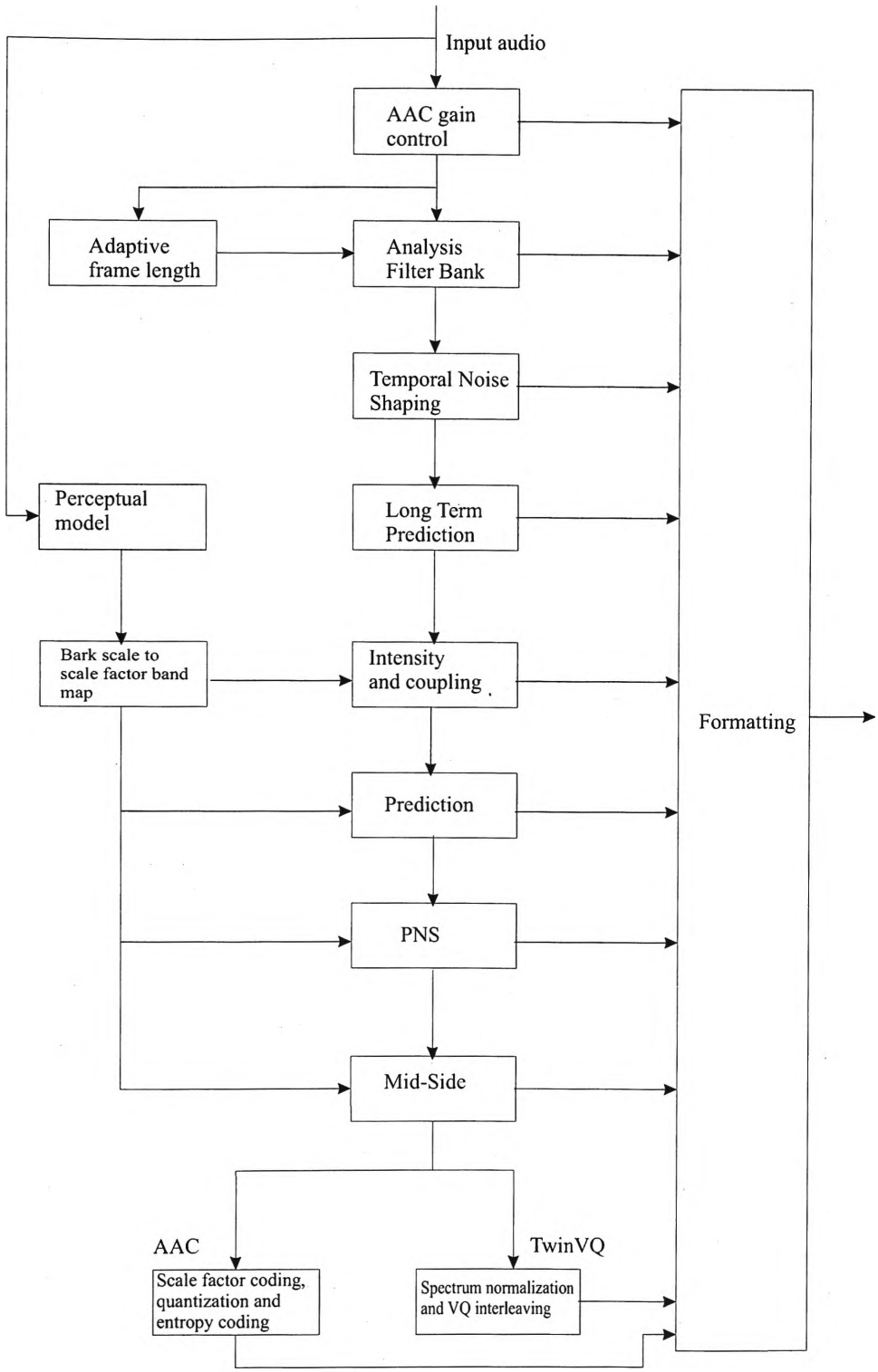

Figure 3.8: The MPEG-4 General Audio coder

As for the TwinVQ, it is aimed at increasing the quantization efficiency of the GA tool at lower rates. The TwinVQ [IMM95] coding scheme utilizes the MDCT to analyze the incoming audio signal as well as linear prediction to approximate the linear redundancy in the signal. The linear prediction spectral envelope is used to divide the MDCT coefficients and hence obtain a frequency domain residual signal. This residual is further whitened by the use of a fine structure predictor whose spectral approximation is again used to divide the calculated residual of the MDCT coefficients. The fine structure envelope is determined by the use of three previous frames' fine structure envelopes. The normalized coefficients obtained following the fine structure division are split into equal length vectors whilst being interleaved as described in [MH88], where the coefficients are decimated by a factor equal to the number of sub-vectors to be produced and allocated to the sub-vectors such that the sub-vectors each have an approximately equal number of low and high frequency coefficients. A perceptual weighting is applied to the sub-vectors (before being quantized) by weighting each sub-vector by a transformed version of its linear predictor envelope. It has been reported that TwinVQ outperforms MPEG layer II for 48 kHz sampled signals coded at 64 kbps, and MPEG layer I for 32 kHz sampled signals at 32 kbps [PS00].

The foregoing discussion about the MPEG standards highlights the need for an embedded bitstream scalable coder. Whilst the defined method of scalability attempts to achieve harmony between a number of compression algorithms, it lacks control over every bit transmitted. Instead, sections of bits are grouped together to produce a base layer and enhancement layers. This thesis proposes compression algorithms in Chapters 5 and 6 that enable the control of every bit used in the compression algorithm.

The MPEG coding standards have come about as a result of the co-operation of a number of commercial enterprizes [Nol97]. The contribution of large commercial enterprizes such as AT&T Bell and Dolby were primarily based on coding algorithms that each had implemented as a product. The following review of popular commercial products should clarify the similarities between those products and the coding algorithms adopted by MPEG.

3.7 The Dolby AC series of coders

The AC-2 and AC-3 series of coders were designed to be low complexity transform audio coders [FBD+96] to address commercial entertainment services such as cinema audio and digital television audio. The AC coders TDAC transforms (such as the MDCT). AC-2 and AC-3 adopt a similar overall approach to the coding of audio with differences in the detail. The commonalities between AC-2 and AC-3 are shown in Figure 3.9 which shows the structure of both encoders with the dashed lines indicating paths that are only applicable to AC-3.

As indicated by Figure 3.9, quantizing the frequency domain representations in both AC-2 and 3 involves the approximation of the spectral envelope. The AC-2 spectral envelope encoding picks the peaks in the critical bands and quantizes them. These values are used as gain factors in the quantization of the remaining spectral components. The approximated spectral envelope is transmitted to the decoder and it is used at both the encoder and decoder to approximate the masking threshold of the signal, avoiding the transmission of perceptual side information [FBD+96] [Dav99]. AC-2 embodies four algorithms with varying delay and bit rate (as well as quality). The frame length used for the analysis of the audio signal also varies

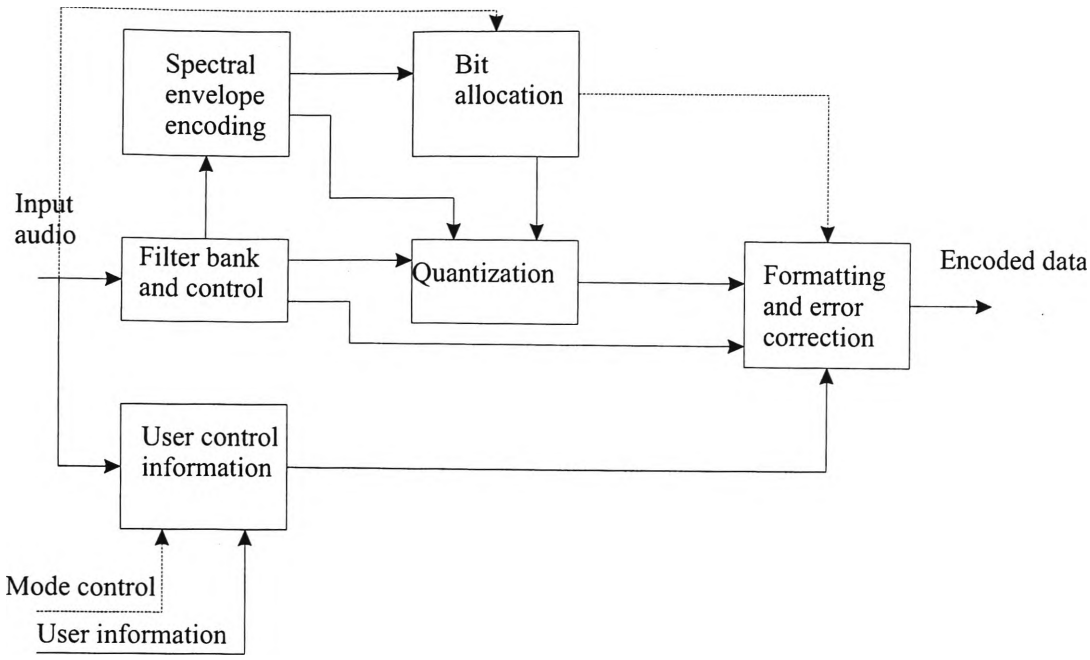

Figure 3.9: The AC-2 and AC-3 encoding schemes

according to the algorithm used. The lowest delay algorithm uses a 128 sample frame length at 48 kHz sampling, to produce a coded bit stream at 192 kbps with 7 ms total delay. The coding delay may be increased to 12 ms by the use of the second AC-2 algorithm which also produces a 192 kbps bit stream. The increased delay is the result of consecutive 128 sample frames sharing information and hence using the allocated bandwidth more efficiently leading to increased quality. The third AC-2 algorithm requires 512 sample frames and operates at 128 kbps with the cost of the delay increasing to 45 ms. The fourth AC-2 algorithm, named AC-2A, allows the adaptation of frame length between 512 samples and 128 samples depending on the signal characteristics. The adaptation of frame length is used to localize transients more accurately and so improve the perceptual performance of the coder.

In contrast, AC-3 spectral envelope encoding utilizes a finer resolution than the

AC-2 envelope encoding. In AC-3, rather than grouping together spectral coefficients according to the critical band to which they belong, the coefficients are lumped together in groups of ones, twos or fours. These new "bands" are grouped with corresponding bands from up to five other frames in time before transmission. This introduces a delay in the AC-3 coding algorithm [FBD+96]. AC-3 also uses a more complete masking model to determine the masking threshold, whilst maintaining frame length variability according to signal characteristics. AC-3 produces high quality audio compression at a stereo rate of 192 kbps and a surround sound (5.1 channels) rate of 384 kbps [Dav99].

3.8 AT&T Perceptual Audio Coding (PAC)

The origins of the PAC coder can be found in [Joh88a], [Joh88b] and [Joh89]. PAC is a popular commercial product that is built with a hierarchical paradigm in that a core coder is used for all modes of PAC and various enhancements are added to this core coder depending on the required application [SJDQ99]. Here our concern is with the mono PAC, and so it is illustrated in Figure 3.10. PAC uses 2048 sample frames which are transformed by the use of the MDCT. This was found to be the optimal frame length in terms of coding gain in a study conducted during the design of PAC [SJDQ99]. The perceptual model provides the masking threshold to the bit allocation algorithm which tries to hide the quantization noise.

Similarly to the other products and standards discussed thus far, window switching is used in PAC to address the differences between transient and stationary regions. In a more recent redesign of PAC (named EPAC or Enhanced PAC) window switching is replaced with transform switching. The MDCT is switched to the wavelet transform

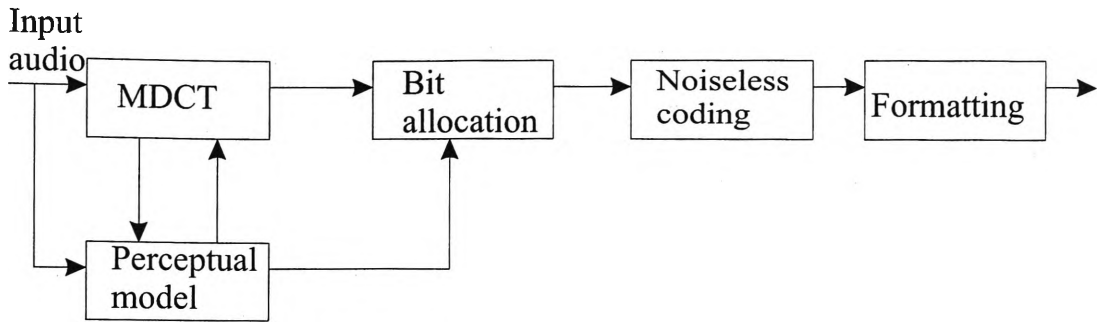

Figure 3.10: The mono PAC encoder

which allows a higher time resolution for transient signal segments. This switching between the MDCT and the wavelet transform is possible because of the link between lapped transforms and the wavelet transform as discussed in [Mal92].

Once the frequency domain representation has been perceptually quantized, the quantized representation is entropy coded to allow a lower bit rate, albeit a slightly variable one. The bit rates of PAC vary between 16 kbps of mono signals up to 1024 kbps for surround sound signals, whilst stereo signals are transparently coded at rates close to 128 kbps [SJDQ99][JSDQ96].

The similarities between the PAC coding paradigm and that adopted by MPEG are noteworthy. Two similarities that seem to standout are the use of a core coder with enhancement layers as well as entropy coding. As discussed previously, whilst these techniques are effective audio compression techniques, the scalability and bitstream control that these schemes offer is limited, i.e., the scalability tends to be coarse and the variable rate bitstream means that it has to modified for transmission over a constant bandwidth channel.

3.9 Sony's ATRAC

The Adaptive Transform Acoustic Coder (ATRAC) was designed with a very specific performance aim, namely to deliver a compression ratio of approximately 4.8 : 1 whilst maintaining a simple enough implementation to allow its installation in small devices [AKY+99], [TSS+96]. The structure of ATRAC is shown in Figure 3.11. The time to frequency transformation is performed on 512 sample frames by the use of QMF filter banks compounded with the MDCT for increased frequency domain resolution in each of the sub-bands. The time to frequency transformation actually utilizes three sub-bands and each sub-band's signals are applied to an MDCT. This sub-band and transform compounding (which is similar to that in the MPEG standards) results in 52 bins in the frequency domain. The spectral component quantization is carried out according to some of the masking behavior of the incoming signal which like all the schemes reviewed thus far allocates bits in a noise hiding exercise. Also, in a similar manner to the compression techniques reviewed thus far, window switching is used to handle transient regions of the time domain audio signal. In the case of ATRAC, the frames used may be as small as 1.45 ms [SJDQ99].

As mentioned previously, ATRAC compresses mono audio signals at a rate of 146 kbps. This has been improved upon by the design of ATRAC2 which allows compression of such signals at 64 kbps. This has been achieved by a more parametric approach to signal compression, where the tonal component of the signal is separated from the signal spectrum and coded separately [AKY+99]. This approach is quite unique to ATRAC as a commercial product, and is in some ways similar to the approach taken in MPEG-4 to low rate audio compression [EP00]. Having separated the perceptually significant tonal components from the less significant noise components, they are

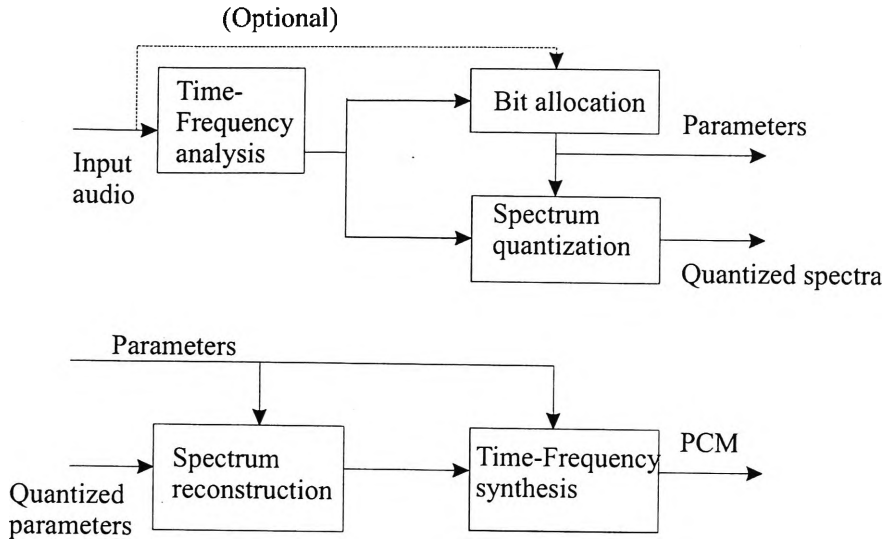

Figure 3.11: Sony's ATRAC

quantized separately as shown in Figure 3.12. Both the tonal components and the noise components are Huffman encoded after quantization to reduce the overall bit rate. Finally, the pre-echo distortion control of ATRAC2 is worth mentioning as it is also slightly different to the other schemes mentioned thus far. In ATRAC2, pre-echo distortion is reduced by increasing the magnitude of signal components just before a transient section to better distribute the quantization bits.

Before discussing other transform based coders that have been presented in the literature, it is worthwhile to consider the similarities of the commercial coders just presented. In an overview comparison, the coders are quite similar. Their difference lie more in the detail. However, all of the coders presented are fundamentally transform based and require the transmission of quantization information as well as the quantized, or coded, information. Also, the coders presented determine both sets of information separately before combining them into a single bitstream. This means that the bitstream produced can not be referred to as embedded. The advantages of an embedded bitstream are scalability and increased error resilience, as only part of

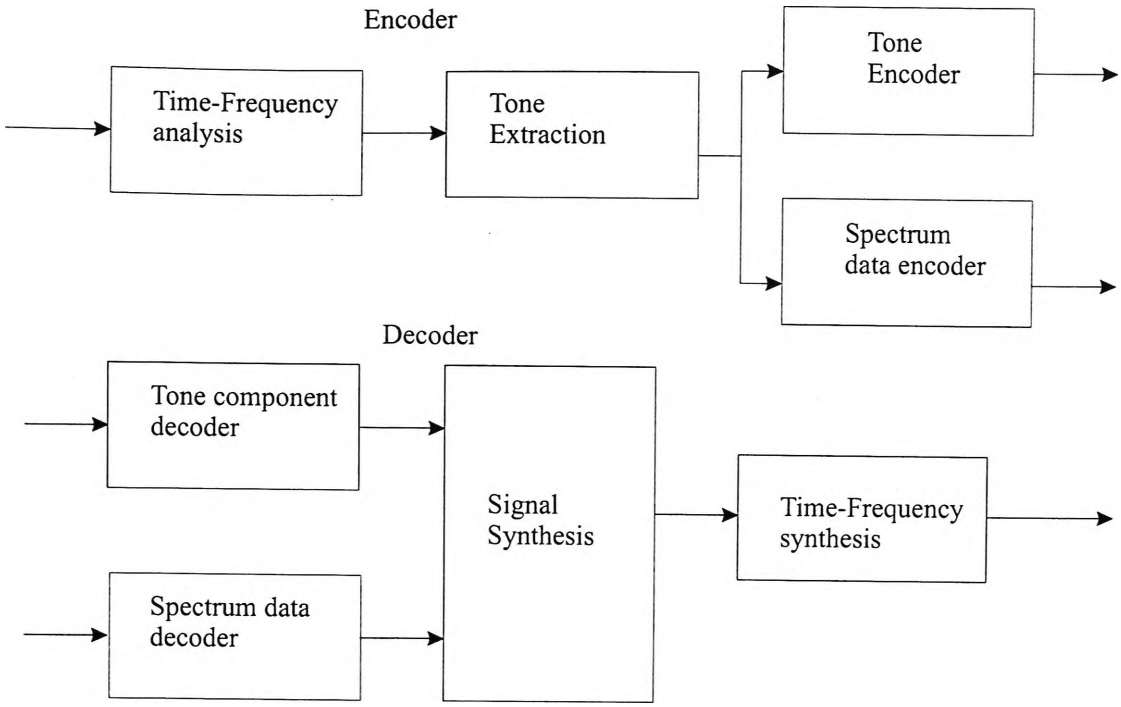

Figure 3.12: Second generation ATRAC encoding

the bitstream may be used to reconstruct some meaningful (although perhaps noisy) representation of the original signal.

To this point, the coders reviewed have either been standardized or are popular commercially. Naturally, there are many other audio coders that do not fit into those categories. In the following a look will be taken at such coders that have been found in the literature. The first group of coders looked at will be the transform and sub-band coders. These have been grouped together because, as the forgoing discussions about MPEG, AC-2, AC-3, PAC and ATRAC have shown, these coders tend to be very close in concept and structure.

3.10 Other transform and sub-band coders

As mentioned previously, the development of PAC started with the development of the perceptual transform coder [Joh88a] which used a 2048 point FFT combined with 128 sub-band quantizers to achieve good quality audio at around the 90 kbps mark. This has been improved upon by the hybrid coder proposed in [BJ90] which utilized a filter bank that more closely resembles the auditory filters. The improved time and frequency resolution is implemented by the use of a QMF filter bank to obtain four octave sub-bands, each sub-band is further divided by the use of a uniform transform. This allows higher frequency resolution for the lower frequencies and increased time resolution for the higher frequencies. The main disadvantage of this scheme was the increased complexity, however, this coder formed the structure of MPEG's layer III (mp3) coder.

A DFT based system that utilized redundancy across signal frames was proposed in [MC89]. This approach produced high quality audio at 96 kbps for signals bandlimited to 15 kHz. However, this coder had difficulty handling signals with substantial harmonic content [PS00]. Linear prediction is utilized in the coding scheme which quantizes spectral magnitudes and differential phase to achieve the high quality reported. This coder is very similar to the AAC coder introduced in MPEG-2. The use of the MDCT in the AAC has resulted in high quality audio being achieved at significantly lower rates (around 64 kbps). A coder that appears to combine the properties of the previous three coders mentioned was proposed in [BHJ+91]. This coding scheme also utilized the MDCT and achieved good quality audio at 64 kbps. The coder in [BHJ+91] also used dynamic window switching to address the issue of pre-echoes.

In an attempted improvement to PAC, a differential coding scheme (DPAC) was introduced in [PM95]. DPAC operates on the same principle as key frame extraction, where frames that are judged to be significant according to a perceptual criteria are coded with a greater number of bits to frames that are judged to be insignificant. This is achieved by coding the significant frames and only coding the difference between the coded significant frames and the insignificant frame. The MDCT is the transform chosen in this coder, and transform coefficients that are deemed to be of high perceptual significance in an insignificant frame are coded in the same way as the coefficients of a significant frame. This coder is reported to have outperformed the coder of [Joh88a] for some types of audio below the 88 kbps mark. As such, this may be viewed as a low rate perceptually lossy alternative to the coder of [Joh88a]. Another technique for improving the perceptual performance of perceptual transform coders was outlined in [Sch96], where noise substitution was proposed for noise-like sections of the audio spectrum. The noise-like sections of the spectrum are replaced by noise signals that have the same average power, temporal envelope and frequency range. The DFT is used in this process, as it allows spectral magnitude shaping that is separate from the phase (which is deemed irrelevant in this process). The idea of noise substitution has found its way into the MPEG-4 standard (as mentioned earlier) however it is a technique of significant complexity.

Whilst scalar quantization remains a popular choice in transform and sub-band audio coders, vector quantization alternatives have been proposed in [CG90] and [IMM95]. In [CG90], VQ was combined with the DCT to compress 15 kHz bandlimited signals. Frame lengths of 512 samples were used with a simplified masking model and the DCT coefficients were combined into 29 groups before being applied to

a multi-stage VQ process. The scheme proposed resulted in a variable rate coder that delivers good audio quality at rates between 55 and 106 kbps. The scheme in [CG90] applied VQ directly to the DCT coefficients, in contrast the TwinVQ scheme proposed in [IMM95] applies weighted interleave VQ [MH88] to residual transform coefficients. This scheme has already been briefly described in the discussion of MPEG-4. The TwinVQ scheme has been found to outperform the AAC only at rates below 16 kbps [PS00]; nevertheless, it has been included in the MPEG-4 standard because of its error resilience and performance at low bit rates.

A "Masking Pattern Adapted Sub-band Coding" scheme was proposed in [TSL87]. In this scheme a 24 non-uniform filter bank was used to mimic the operation of the critical bands in the human auditory system. The filters had 64 taps and the sub-band samples were processed in 2 ms frames. The bit-allocation is carried out using a simplified masking model for simultaneous masking as well as temporal post-masking. For 15 kHz band limited signals, the coding scheme delivered high quality audio for bit rates ranging between 80 and 100 kbps. The MUSICAM scheme [DLU91], which was adopted as the basis for MPEG-1 layers I and II, is another sub-band coding scheme. MUSICAM analyzes the audio signal with a 32 band filter bank and processes the sub-band samples in 8 ms frames. A 1024 FFT is used in parallel with the sub-band analysis to obtain a high resolution spectral analysis for the application of a psychoacoustic model that is very similar to the MPEG-1 model 1. MUSICAM achieved near transparent compression at 96 kbps for mono signals, but this performance suffered when sharp attacks were encountered. This coder has also been shown to have high error resilience.

Transform and sub-band coding have become more interleaved with the injection

of the wavelet transform into audio compression algorithms. A globally optimized and applied wavelet coding scheme was proposed in [ST93a] where wavelet packet signal analysis was combined with VQ and perceptual masking to achieve near transparent CD-quality audio compression at rates between 48 and 64 kbps. A total of 29 sub-bands were used to mimic the time to frequency transformation of the auditory system. During the analysis of each frame, the wavelet basis functions are chosen to minimize the bit rate for a given distortion. This scheme is referred to as "global" because the same wavelet chosen in the optimization is applied to all the sub-bands [PS00]. The wavelets belong to the Daubechies family of wavelets.

A scalable DWT coder has also been proposed in [SJ98] where the wavelet tree structure is varied to reduce the perceptual entropy of the signal in each of the sub-bands. Zero-tree coding is applied to achieve very good quality for most of the tested signals at 45 kbps. Yet another wavelet based compression scheme was proposed in [LP98]; this scheme will be the subject of further discussion in Chapter 5.

The wavelet based schemes have shown considerable promise because of the temporal and frequency resolution that can be achieved for high and low frequency components respectively. To further improve the analysis of audio signals, sinusoidal and wavelet hybrid schemes have been proposed in [HAT96], [BD98] and [PSP96]. In [HAT96], the signal decomposition was based on the idea that an audio signal is a combination of tonal, transient and noise components. As such, the tonal components have been represented by the use of sinusoids whilst the transient components are analyzed by the wavelet transform. Sinusoidal amplitudes are quantized to satisfy masking threshold requirements whilst the phase components are uniformly quantized. The sinusoidal components are used to represent the low frequencies in the

audio signal. Wavelet transform components are quantized in two ways; if they fall below 11 kHz they are quantized individually otherwise a parametric representation is used. The noise components are represented in terms of mean, variance and decay constants rather than being quantized directly. This scheme achieved very good quality at rates in the range of 44 kbps. The scheme proposed in [BD98] operated on similar principles but utilized least squares linear prediction for the sinusoidal extraction. The scheme in [BD98] is reported to have achieved quality to MPEG-1 layer III at rates ranging between 60 and 70 kbps. Finally, the coder of [PSP96] adapted the analysis frame to achieve better perceptual quality. A unique masking threshold shifting was used in [PSP96] to determine the perceptual bit allocation at varying rates.

Good quality audio compression has also been reported by the use of time varying filter banks in [SJ96], [PJ95], [HJ97] and [PN96]. In [PJ95] better quality audio compression than the MPEG-1 layer III coder at 48 and 64 kbps was reported. The adaptation is achieved by applying a second stage filter bank depending on the required time-frequency representation. Scalar quantization is applied to the sub-band coefficients in a perceptual manner to achieve the performance mentioned. In comparison, the coder proposed in [PN96] adapts the wavelet decomposition in both depth (number of stages) and breadth (bandwidth) to achieve a minimum bit rate criteria. This coder has achieved high quality compression at 55 kbps.

Another form of hybrid coding involves the use of sub-band coding with linear prediction. Recently, one such scheme has been proposed in [YK02] where a warped LP filter is used for the pre and post processing of the audio signal. It is argued that this approach will distribute the noise in the sub-band domain in such a manner that

the synthesized sound should be better perceptually. It is shown that the perceptual entropy in the sub-bands is less than the perceptual entropy of the original signal and hence the excitation of the WLP filter is quantized in the sub-band domain. Essentially the idea is to perceptually decorrelate the audio signal by the use of an LP filter. The warped spectrum is used to better match the ear's own decorrelation process. The results showed improved perceptual performance at 56 kbps over the MPEG-1 layer II coder at the same rate. A similar approach has been taken in [ND01] except for the use of the wavelet transform for the sub-band filtering phase, the other significant difference with the scheme in [YK02] is the way that the quantization is carried out.

It is also worth mentioning in the completion of this subsection a number of papers that have appeared recently that address quantization issues in transform coders. In [RT00] a backward adaptation scheme is presented that is similar in a sense to adaptive delta modulation and similar in concept to the backward adaptation adopted in MPEG-4. The scheme adapts the perceptual model on a sample by sample basis, thus changing the quantization resolution of the sub-band filters. Entropy coding is applied to the output of the adapted quantizers to reduce the overall bit rate. In [NK00], a study of quantization bit allocation schemes is presented. One of the schemes is Signal to Masker Ratio (SMR) based and the other is energy based (that is the energy above the masking threhold). It is found that the SMR scheme performs better perceptually and should be the one employed for low rate coding. It is also concluded that a combination of the two schemes would probably be more advantageous than either scheme on its own. A variable size vector entropy codebook design algorithm is presented in [Sho01] to address the issues of complexity

in design. The vector entropy scheme is similar in some ways to the Lempel-Ziv algorithm [GPS94] and it does result in a compression gain of approximately 37% on the Huffman entropy coding scheme of audio and speech material. Finally, in [Ron00] both vector and scalar quantization are combined in a hybrid quantization scheme for sub-band audio compression by obtaining the MNR (Mask to Noise Ratio) for both the scalar and multistage vector quantizer. The quantizer that provides the best MNR is chosen for use. This approach tries to deliver a balance between the complexity in design and training of VQ and the transform coding performance of scalar quantization. The coding scheme mentioned reportedly outperformed MPEG-1 layer II at 56 kbps.

3.11 Parametric audio coders

The focus of this section thus far has been on transform and hybrid coding algorithms. This is simply an indication of the popularity of transform coding in audio compression. Yet there have been a number of compression algorithms presented that attempt to model the audio signal. These algorithms are different to the speech coding algorithms which attempt to model the production process of the sound, rather, these algorithms attempt to divide the audio signal into a number of simpler signals that can be more efficiently coded and transmitted. The basics of the algorithms presented here have already been discussed in Section 3.3 and so only brief summaries of each approach will be presented in this section.

Sinusoidal modelling of audio signals was thoroughly studied in [GS92] where an analysis by synthesis method of sinusoidal extraction was proposed. This technique has been adopted by coders presented in [EP98] and [EP00], where the sinusoidal

components are extracted according to their perceptual significance. This means that a scalable coder is made possible as more significant components are extracted first whilst less significant components are extracted at a later stage. The coder described in [EP00] and [PM00] has been adopted in the MPEG-4 audio coding standard for low rate scalable audio compression (ranging from about 6 kbps to 24 kbps). The results presented for this coder did not show a significant improvement over the MPEG-4 AAC coder at similar rates, however its scalable structure makes it a useful coder.

The sinusoidal model remains popular for audio signals, and a significant number of papers continue to be published about improvements to the model. In [PS01], a scheme of sinusoidal component selection is reported whereby the sinusoids are chosen according to their contribution to the excitation of the human auditory system rather than their SMR. The reported results show a perceptual improvement. This approach highlights the difference between SMR and the auditory excitation pattern. The calculation of the excitation pattern is presented in detail in [ZF99] and has been used in the calculation of the pleasantness parameters in Section 2.4. In a way, the approach of [PS01] is a mix between sinusoidal modelling and noise shaping as it tries to get a more complete sinusoidal representation. Noise shaping has been discussed in 2.2.3, and recently the work reported in [VDk01] has shown that an efficient solution to the shaping of re-quantization noise that is perceptually dependant is possible. Whilst re-quantization noise is not the same as coding noise, the fact that one can shape a noise component perceptually is encouraging for audio compression purposes.

Modifications to the original sinusoidal model (described in [MQ95]) have been proposed recently in [VHK01], [VHvdPK01], [NHD98] and [JK00b]. The work in [VHK01] and [VHvdPK01] proposes to modify the original signal by shifting the

transients such that they are located at the start of an analysis frame. The main reason behind this is the well documented problems that the sinusoidal model has with efficiently modelling transients. This problem was addressed in [NHD98] by the application of exponentially modulated amplitudes to the sinusoidal model. However, this modification still suffered from modelling transients that were far from the start of the analysis frame. It is reported in [VHvdPK01] that the shifting of the transients so that they are at fixed positions with regards to the framing method used does not alter the perceptual content of the audio signal, but it does significantly improve the modelling efficiency of the exponentially modulated sinusoidal model. With the same aim of increasing modelling efficiency, the work reported in [JK00b] proposes the use of a small number of frequency harmonics and a time-varying fundamental frequency. This method provided some encouraging results.

Other parametric coding approaches have involved Frequency Modulation (FM) coding [Cho73] and linear prediction [Sin90], [BD98], [CW96], [HLK96]. In FM modelling, the spectrum of the audio signal is approximated by that of an FM signal. The search for a good match is carried out iteratively until the residual signal is below the masking threshold. FM modelling seems to work very well for sounds with a single fundamental frequency (that is for single instrument sounds) but has difficult modelling multi-instrument material [PS00]. Improvements to this model are required before it can become a mainstream audio model.

The linear prediction coder of [Sin90] utilizes a multi-pulse excitation approach. The linear predictor used was a 24th order predictor. The pulse positions and magnitudes are derived by the use of a perceptual optimization weighting criteria. This coder achieved good quality audio at rates around the 128 kbps mark. In contrast,

the LP coder of [CW96] used sinusoidal modelling of the excitation to obtain good quality audio at around the 96 kbps. The excitation of the linear predictor is modelled by the use of seven sinusoids. This approach seems slightly counter-intuitive as one would expect a residual signal to be closer to noise than the original, and it is well known that a noisy signal is not efficiently modelled by the use of a sinusoidal model. However, in LP based coders, it is the position and magnitude of the pulses in the excitation which influence the quality of the synthesized signal, and one can expect a sinusoidal model to be able to model such pulses. In a similar manner, the coder reported in [BD98] transform coded the LP residual using a three level wavelet transform. Finally, warped linear prediction was used in [HLK96] for audio compression. Frequency warping allows the linear predictor to analyze the audio on a scale that is closer to the bark scale and so provides a more perceptually accurate representation.

3.12 Scalable audio coders

In the previous discussions about transform and parametric coders, some scalable audio coders were mentioned. This is simply because scalable audio coders can fall into either category, or a hybrid category. However, it should have become clear that the majority of audio compression techniques proposed operate at an optimal bit rate for which they were defined. The case for scalable audio coding is quite simple; different quality audio should be available at different rates whilst maintaining the relationship that as the bit rate increases so does the quality of the synthesized audio. However, bit rate scalability is only one form of scalability; another is complexity scalability. A good example of this is the MPEG-1 standard, (described previously) which achieves higher quality and higher compression with increasing complexity. Yet

another form of scalability is bandwidth scalability, which means the ability to code wider band signals as the bit rate increases.

Most of the scalable coders found in the literature adopt the MPEG-4 method of bit rate scalability, where a base layer is coded by the use of an efficient scheme and the residual produced between the original signal and the base layer synthesized signal is coded by the use of a similar or completely different scheme. This is certainly the case for the scalable coder presented in [KS96] where scalable compression is achieved by combining the wavelet transform with entropy coding in a multi-layer system. The bottom layer is treated as the low quality layer and the second layer is the enhanced quality layer. As the bit rates of the bottom layer increases, the necessary bit rate of the second layer decreases because the residual signal decreases in dynamic range. The coder presented in [CL01] applies the same principle of multi-layer scalability with the enhancement layer being coded by the use of zero tree coding [Sha93]. The nature of zero tree coding implies scalability because more significant coefficients are transmitted first. A similar wavelet/zerotree approach was also taken in [ACRG99] where the wavelet transform employed was adapted according to a perceptual criteria. The output of the wavelet transform was quantized perceptually and mapped to a zero tree, resulting in variable rate compression of audio signals at 1.5 to 2.5 bits per sample. Zero tree coding was also used in [DWJ00] which presented a complexity scalable algorithm that achieved better audio quality with increasing complexity.

The multi-layer approach to scalable compression has also been used with an LP based coder in [JK00a]. The algorithm presented was based on low-delay CELP. The coder presented operates in a number of modes, with increasing delay. The scalability of the coder was set in integer multiples of the base layer, which varied between 2.66

kbps and 10.67 kbps depending on the mode used. In a similar fashion, a bandwidth scalable coder was presented in [KCG00]. The base layer was encoded by the use of the standard G.729 coder whilst the enhancement layer was coded by the use of CELP. The coder reportedly achieved better quality than the 16 kbps MPEG-4 CELP coder at the same rate. Here bandwidth scalability means that the base layer produces a signal that is narrower in bandwidth than the base plus enhancement layers synthesized signal.

In an alternative approach to the schemes already discussed, the coder presented in [VA01] utilizes a two-dimensional transform that first transforms the time domain signal to a time-frequency representation and then transforms the time frequency representation into a frequency-modulation frequency representation. TDAC transforms are used for the first stage and then a similar method to the TDAC transform is applied to the magnitude time-frequency spectrum to obtain a very small number of effective coefficients. Huffman entropy coding plus scalar quantization is applied to the phase and magnitude of the final representation. Only the phase of the first 5 kHz of the complete bandwidth is transmitted and noise substitution is used for insignificant magnitudes. The coder shows very good performance at 32 kbps and allows scalability because of a priority of transmission of components based on the perceptual significance of each component. Although very effective, this coding scheme seems to be more of an off-line coding scheme because of the delay required by the time-frequency transformation in the first stage.

An object oriented approach to scalable compression has been taken in [JMIM01] and [JMN+99] where a hierarchical scheme is developed based on a TwinVQ coding block. Each layer or 'block' adds to the quality of the coded audio as it codes the

residual that remains from the previous layer. This scheme is unique in that the same coder is applied at each stage creating the impression of an object oriented coding scheme. However, because of this object hierarchy, the scalability achieved in large step scalability with each enhancement layer adding 8 kbps to the overall bit rate. This coder was shown to outperform the MPEG-4 AAC at 8, 16 and 24 kbps for 24 kHz sampled audio and so it has been adopted by MPEG as part of the MPEG-4 standard.

Yet another approach to scalable compression has been the signal model approach of [VM00], [VM00] and [EP00]. This approach has been discussed in some detail in a number of preceding sections (see Sections 2.2.3 and 3.11) and so they will not be elaborated on any further.

3.13 Lossless audio coding

The final group of coders that will be considered in this chapter are the lossless audio coders. The discussion here will only be a brief; relevant aspects of lossless audio compression that need to be elaborated upon will be addressed in Chapter 6.

Lossless audio compression aims to recreate an exact copy of the original signal whilst using a lower rate to the original. Currently, lossless audio coding is being approached from a signal model perspective [HS01][CL97][BOvdV96][Qui01]. The signal is typically modelled using a linear predictor, which may either be FIR or, as in the case of [CL97], IIR. The aim of using a linear predictor is the decorrelation of audio samples in the time domain and the reduction of the signal energy that must be coded [HS01]. The coefficients of the linear predictor are coded as well as the excitation of the predictor, which is typically coded using an entropy code. The

combination of the linear predictor with the variable length entropy code leads to a perfect reconstruction of the audio signal. The typical compression ratio of such coders typically depends on the nature of the audio signal being coded and may range between 1.4 and 5 [HS01].

Another approach to lossless compression of audio signals involves the use of transform coding as presented in [LPN97]. This approach is actually very similar in nature to the linear prediction approach as it utilizes a transform coder to produce a lossy compressed version of the original signal and an entropy code to compress the generated error signal between the lossy compressed signal and the original signal. The use of the transform coder decorrelates the audio samples and hence the transform coder operates on the same basic principles of decorrelation and entropy coding as the linear prediction based lossless coders [HS01]. The compression ratios reported in [LPN97] again varied with the nature of the input audio signal and ranged between 2.2 and 3.2.

An approach of interest to this thesis was taken in [MIJM00] where a lossy coder was combined with an entropy coding scheme to achieve lossless compression. This means that the one bit stream may be decoded to either obtain a lossy version of the original signal or a lossless version of the original. However, the gap between the lossy and lossless performance is not smoothly transitioned. At the time of writing this thesis scalable to lossless coders were reported in [GHKB02] and [LJ02]. The work in [LJ02] proposes a scalable to lossless scheme that would be expected to be more smoothly scalable because of the compression algorithm used. Granularity is claimed to eight bits. the issue of scaling from a lossy to a lossless scheme is a growing field of interest and is the subject of Chapter 6.

3.14 Discussion and conclusions

This chapter has focused on the state of the art in audio compression. The majority of the audio compression techniques reviewed have been found to be transform coding techniques, however, parametric coding techniques are becoming more popular with the increasing interest in very low rate audio and scalable audio compression. It has been shown that a number of hybrid schemes have achieved very good quality at reasonable bit rates, these schemes try to combine the advantages of both parametric and transform coding. These hybrid techniques can also be seen in the recent audio coding standards that use techniques such as noise substitution (which is primarily a parametric idea) to achieve better perceptual quality at lower rates.

Improving quality at current rates will remain of interest in the near future however the direction of audio compression seems to be towards high quality scalable schemes that allow different audio quality at different rates with a smooth increase and decrease in quality. The Internet has a vital role to play in the development of such coders because of the highly variable channel capacity. If the channel capacity on such networks is to be stabilized then these types of coders will become of less interest, however, this is unlikely in the near future. On the other hand, recent increases in channel capacity on cellular and wireless networks has meant that it is now possible to transmit a very high quality audio signal to mobile customers. This will increase the interest in lossless audio compression. The interest in lossless audio compression has also been driven by popular consumer products such as DVD entertainment systems. At this stage there is no existing smoothly scalable coder that can scale to lossless quality with single bit granularity.

It must also be remembered that whilst the focus of this chapter has been on

compression schemes that are applied mainly to mono signals, audio is very much a scenic experience and so interest in compression schemes for multi-channel audio signals will remain an area of research.

As a result of the presented review of the current literature, this thesis addresses the development of scalable compression schemes that are both perceptually and objectively scalable. The development of a smoothly scalable scheme from lossy compression to lossless compression with fine grain scalability is also addressed.

Chapter 4

Perceptually Scalable Sinusoidal Compression of Audio

This chapter introduces a perceptually scalable sinusoidal coder for full band audio. The sinusoidal representation of the audio signal is obtained by the use of the Short Time Fourier Transform (STFT) and re-arranged in such a manner that the most perceptually significant parameters are transmitted first, using the limited available bandwidth in a perceptually efficient manner. Other schemes of obtaining the sinusoidal coefficients are also discussed and compared to the scheme employed. Finally, the output of the coder is analyzed using sensory pleasantness measures in order to quantify the noise introduced due to the compression in an objective perceptual manner. Thus, this chapter deals with the scalable sinusoidal coding contribution of this thesis which is also described in [RB01] and [RBM01].

4.1 Introduction

Sinusoidal compression of audio and speech signals involves two steps; first the derivation of sinusoids that, when summed together, reconstruct the original signal and

secondly the quantization of the parameters that define the sinusoids [MQ95]. In this way, Sinusoidal compression may be viewed as an application of the Fourier Transform to audio signals and the quantization of the transform coefficients in an efficient manner. However, the representation of a signal as a sum of sinusoids does not necessarily have to be based on the Fourier Transform. The work in [GS92] shows how sinusoids may be extracted from the original signal in an iterative manner. Further, it has been shown that sinusoids may be removed in such a way that perceptually significant sinusoids are extracted before the perceptually insignificant sinusoids [EP00, PM00].

Sinusoidal coders are normally considered parametric coders, in that the sinusoidal parameters are the coded components for transmission [MQ95]. Compression is obtained in such coders by realizing that in a given sound there will only exist a small number of sinusoids that contain significant information [MQ95]. Here, the word 'small' is used in a relative context to the number of sinusoids that are required to reconstruct the audio or speech signal in order to acquire a high Signal to Noise Ratio (SNR). This chapter describes sinusoidal coding according to the work of [MQ95] and [GS92] in some detail in the next section. The contribution of [GS92] is further discussed as it applies to the work on the HILN coder in [EP00].

The work of [GS92],[MQ95] and [EP00] is used as background material for the development of a sinusoidal compression scheme that allows scalability. Two versions of the compression scheme are presented, the first is a variable rate coder that has been developed for compression at medium rates and the second version is the scalable version which may be used from low to high rates. Both versions of the coder are based on sorting perceptually significant coefficients to allow the transmission of those

coefficients first. Objective results are presented, in terms of the Segmental Signal to Noise Ratio (SegSNR) and sensory pleasantness parameters. Informal listening tests have shown that both versions of the coder perform acceptably well when compared to the MPEG-4 AAC coder.

4.2 Sinusoidal Compression of Speech and Audio

4.2.1 The Short Time Fourier Transform (STFT) Approach

The work of [MQ95] describes the original sinusoidal compression scheme developed for speech compression. The original speech signal s of length N is approximated by the model

$$\hat{s}[n] = \sum_{i=1}^{L} A_i \cos(\omega_i n + \phi_i) \qquad n = 0, \ldots, N-1 \tag{4.2.1}$$

L is the total number of sinusoids used in the synthesis and is time varying. It has been reported in [MQ95] that to obtain a good quality narrow-band synthesized speech signal, as many as 80 sinusoids may be required. To develop a low rate sinusoidal coder, it was recognized from an early stage that the sinusoidal model must be reduced to a model that is describable by a few parameters [MQ95]. As a result the sinusoidal model is generally reduced to a harmonic model given by:

$$s_H[n] = \sum_{i=0}^{L(w_0)} A_i \cos(i\omega_0 n + \phi_i) \tag{4.2.2}$$

From Equation 4.2.2 it can be seen that the STFT can be used to derive the harmonic sinusoidal model, as all the frequencies involved in the model are multiples of one frequency. The frequency chosen as ω_0 is the pitch frequency of the original signal, which may be obtained by the use of a number of well known algorithms [Ram95][RS78]. In [MQ95] a pitch estimation algorithm is proposed that fills the requirements of sinusoidal coding well.

The overall sinusoidal coding of speech system is illustrated in Figure 4.1. As has been described thus far, the analysis involves the determination of the sinusoidal parameters by the use of the STFT. The Figure shows that only the peaks of the amplitudes are actually selected for use in the model, this is to allow further compression by generating an amplitude envelope and representing that efficiently instead of dealing with each individual amplitude. The amplitude peaks are selected by the use of the SEEVOC algorithm (which is described in detail in [MQ95]). SEEVOC basically selects the peaks of the amplitudes in search intervals defined by multiples of the pitch frequency. The amplitude envelope is then generated by using linear interpolation between the selected amplitudes, although other interpolation methods may be used (for example cubic spline interpolation).

As the STFT is being used for the analysis of the speech signal on a frame by frame basis, side-lobe leakage must be addressed. In the original system developed by [MQ95], the Hamming window is used to reduce the side-lobe leakage. This naturally widens the main-lobe, which is a problem in this case because the peaks of the obtained spectrum (that is the STFT generated spectrum) are being used to approximate the sinusoidal amplitudes. To justify such an approximation it has been reported that the frame length used for analysis should be set at 2.5 times the pitch

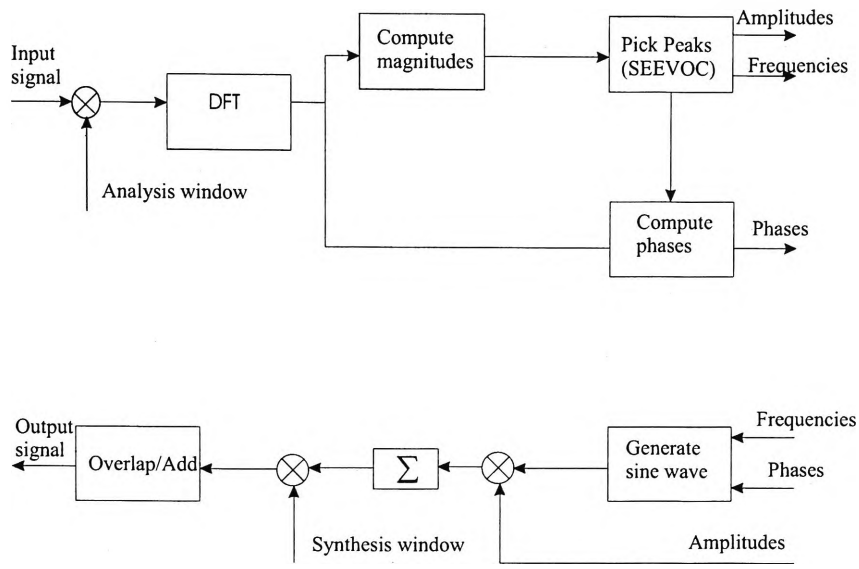

Figure 4.1: The sinusoidal model based on the STFT

length [MQ95]. In unvoiced speech, the frame length is kept equal to the last voiced speech frame or extended to 20 ms if it is smaller than that.

To synthesize the speech signal after analysis, the overlap-add method is used. This allows the elimination of discontinuities between frames, whilst avoiding the use of sinusoidal tracks. Sinusoidal tracks are sometimes used in sinusoidal coding to avoid discontinuities at frame boundaries. The designers of the original sinusoidal coder argued for the overlap-add method ahead of sinusoidal tracks primarily because of simplicity of implementation.

Having obtained the pitch of the signal, and the amplitude envelope the final step towards achieving a low-rate implementable system is to adequately model the phase. The phase may be modelled by using a technique described in [MQ95] which relies on the cepstral coefficients of the original signal. The cepstral coefficients are obtained by taking the log magnitude of the STFT coefficients followed by the inverse STFT. To achieve good quality at low rates, an all-pole model is used to code the magnitude

envelope, the signal is assumed minimum phase and only harmonic frequencies are used. Good quality has been reported at 4.8 kbps [MQ95] and reasonable quality at 2.4 kbps. Formal listening tests conducted by [Pri95] show that the sinusoidal coder at 2.4 kbps performs as well as the 4.8 kbps STU-III CELP coder in quiet and slightly better in the office environment.

4.2.2 The Analysis By synthesis Approach (A-by-S)

A considerable number of researchers have used the analysis by synthesis approach to sinusoidal modelling of an audio or speech signal [GS92][EP00][PM00]. This approach is summarized simply by stating that it decomposes a given signal into a sum of sinusoids by extracting individual sinusoids one at a time. The Mean Squared Error (MSE) between the original signal and the synthesized signal is minimized in each iteration and so the sinusoids are extracted in order of energy significance.

Modifications to this system have been proposed in [PM00, EP00] where the "Harmonics and Individual Lines plus Noise) (HILN) is described. The change introduced focuses on the extraction of the sinusoids in a perceptually significant manner. The HILN scheme is illustrated in Figure 4.2.

An important feature of the A-by-S scheme is the incorporation of $v[n]$, the windowed time domain envelope of the original signal. This is important because it shapes the "sinusoids" in such a way that significantly fewer sinusoids may be needed to closely approximate the original signal. The envelope may be obtained by low pass filtering the input signal [GS92] or by using the Hilbert transform [EP00], as the envelope of the input signal may be defined as the magnitude of the sum of the

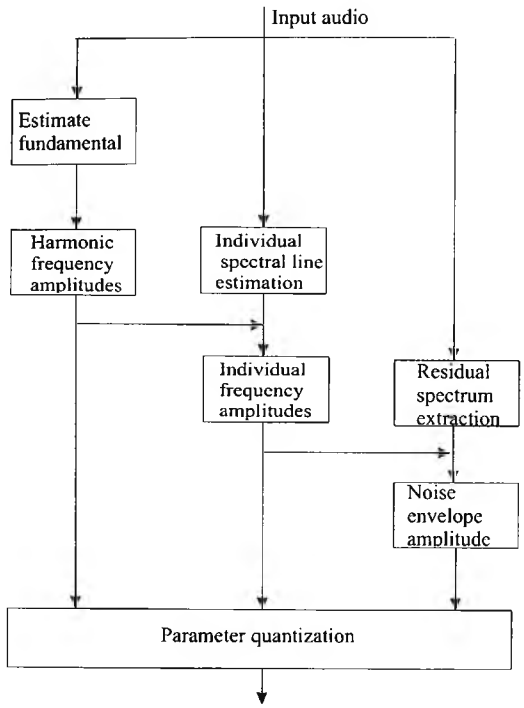

Figure 4.2: The HILN scheme

original signal and the imaginary component of the Hilbert transform. That is,

$$v[n] = \sqrt{s^2[n] + \{H(s[n])\}^2} \tag{4.2.3}$$

In Equation (4.2.3) {H(s[n])} is the Hilbert transform of $s[n]$.

There are a number of clear advantages to this to this approach. Firstly, scalability is possible because individual sinusoids are being extracted one may extract up to a pre-defined limit (in terms of MSE, or SNR) and the quality of the synthesized signal increases proportionally. This advantage has helped HILN be installed as part of the MPEG-4 standard for scalable audio compression.

Editability (that is, ease of editing the synthesized signal) is also an advantage of this scheme [PM00]; since individual sinusoids may be varied, the resultant signal is editable from its very basic components. This immediately implies a good synthesis technique for general sounds and music especially. Also, the use of the envelope in

the Analysis-by-Synthesis loop leads to the system being adequate for both harmonic and inharmonic systems [GS92].

The actual extraction algorithm is iterative with the initial synthesized signal being zero. For a given frequency ω (which is chosen from a uniformly sampled set of frequencies $\omega_i = \frac{2i\pi}{N} \qquad i = 0, \ldots, N/2$) sinusoidal parameters are extracted such that the MSE is minimized. To achieve this, the following set of equations must be used [GS92]:

$$A_i = \sqrt{a_i^2 + b_i^2} \tag{4.2.4}$$

$$\phi_i = -\arctan \frac{b_i}{a_i} \tag{4.2.5}$$

$$a_i = \frac{\gamma_{22}\psi_1 - \gamma_{12}\psi_2}{\Delta} \qquad b_i = \frac{\gamma_{11}\psi_2 - \gamma_{12}\psi_1}{\Delta} \tag{4.2.6}$$

$$\Delta = \gamma_{11}\gamma_{22} - \gamma_{12}^2 \tag{4.2.7}$$

$$E_i = E_{i-1} - a_i\psi_1 - b_i\psi_2 \tag{4.2.8}$$

$$\psi_1 = \sum_{n=-N}^{N} e_{i-1}[n]v'[n] \sin \omega_i n \tag{4.2.9}$$

$$\psi_2 = \sum_{n=-N}^{N} e_{i-1}[n]v'[n] \cos \omega_i n \tag{4.2.10}$$

$$\gamma_{11} = \sum_{n=-N}^{N} v'^2[n] \cos \omega_i n^2 \tag{4.2.11}$$

$$\gamma_{12} = \sum_{n=-N}^{N} v'^2[n] \cos \omega_i n \sin \omega_i n \tag{4.2.12}$$

$$\gamma_{22} = \sum_{n=-N}^{N} v'^2[n] \sin \omega_i n^2 \tag{4.2.13}$$

$$v'[n] = \sqrt{w_a[n]}v[n] \tag{4.2.14}$$

In Equations (4.2.4) to (4.2.14), E_i is the sum of the squared error for the i^{th} frequency component, $e_i[n]$ is the actual error between the windowed input and the synthesized signal and $v[n]$ is the time domain envelope of the signal being modelled. In [GS92] the window used W_a is the Hamming window, whereas the window used in [EP00] was a perfect reconstruction cosine window. The frequency ω_i is the one chosen for which E_i is minimum. In the case of HILN, the above set of equations are weighted perceptually in order to extract the significant perceptual as well as energy sinusoids.

The most noticeable disadvantage of the Analysis-by-Synthesis technique is that the computational load of extracting individual sinusoids is quite heavy. Both [GS92] and [EP00] propose techniques which reduce the computational burden of individual sinusoidal extraction, and both utilize the frequency domain to do so. The frequency domain A-by-S technique proposed in [GS92] moves the calculations from the time domain to the frequency domain through the use of the Fourier transform. The equations all have their equivalent frequency domain equations and the procedure gains speed of computation by the calculation of sinusoidal components in groups.

The fast HILN algorithm relies heavily on the Matching pursuit algorithm in the frequency domain [EP00]. The matching pursuit algorithm allows the decomposition of a signal into a sum of its underlying components. The components are chosen from a "highly redundant" code book or dictionary. In the case of HILN, the components are sinusoids. Once the sinusoidal components have been extracted they are sorted with the aid of a perceptual model and hence the perceptually significant sinusoids are extracted from the system first.

Finally, a note about the coding techniques used in both [GS92] and [EP00, PM00].

The work presented in [GS92] focused on the decomposition into signal components through the use of A-by-S and so no coding (i.e. quantization for transmission) issues were addressed. On the other hand, the work in [EP00] focused on developing a coder and so the quantization techniques have been described, they are only summarized here.

The sinusoidal parameters are quantized by first grouping them into "new" sinusoids and "continuing" sinusoids, meaning those sinusoids that are tracked from a previous frame. The new sinusoidal amplitudes are coded by the use of a relative quantization technique; that is, relative to a maximum amplitude previously transmitted. The new frequencies are transmitted by the use of a "sub-division" code, that is one that continuously sub-divides the region where the frequency may be located. For continuing parameters, the changes in the parameters are transmitted. The envelope is modelled by a linear predictor and the LPC parameters are coded using Logarithmic Area Ratios (LAR). As mentioned previously, HILN is scalable and tends to use a bandwidth that varies according to the bit rate. More details of this coding technique will be described in latter sections of this chapter.

4.3 A new scalable sinusoidal coder

4.3.1 Motivation

Representing an audio signal as a sum of sinusoids has the attraction of building complex signals from a set of simple signals. The original work on compressing speech signals by the use of such a representation was aimed at low rate applications, as explained in the previous section. This aim limits the scalability of the coder as it

forces it into the parametric coding paradigm.

On the other hand, the coder presented as part of MPEG-4 (HILN) does allow scalability, even perceptually, although the extraction of the sinusoids one at a time is computationally cumbersome. This technique is made less computationally demanding by the use of the matching pursuit algorithm leaving the performance of such a scheme dependant on the design of the code-book. The scheme, like the parametric sinusoidal coder, also works with a variable bit rate. This variation in bit rate results primarily from using a varying frame length, that is dependant on the pitch estimate of the input signal. Having a variable rate coder may not be desirable in some circumstances, especially if the rate is governed primarily by the content of the signal. If the frame length was to be fixed, the variability in the rate may be reduced, depending on what is implemented at the quantization stage. Similarly, the development of a scheme that utilizes a fixed frame length allows a simpler implementation.

The adoption of frequency domain sinusoid extraction has helped reduce the complexity of the HILN scheme significantly [PEF98]. This indicates that the combination of the extraction method with a fast transform would be useful in terms of reduction of complexity.

Finally, a perceptual sinusoidal model is of more interest than a straight waveform matching model, simply because it allows greater compression and the identification of signal components (i.e. sinusoids) that are significant to the human ear.

The possible combination of computationally efficient components into a scheme that was scalable perceptually and objectively motivated the development of the scalable sinusoidal coder presented in the following sub-sections. The coder may be implemented as either a variable rate scheme or as a scheme that is scalable to a

maximum allowable rate. The next sub-section gives an overview of the scheme.

4.3.2 System Overview

The block diagram representation of the designed sinusoidal coder is shown in Figures 4.3 and 4.4. The proposed architecture is built around the sorting of the amplitudes according to both energy content and perceptual significance. To obtain the sinusoidal model in this case, the short-time Fourier transform is utilized, however, a short fixed frame length is used meaning that the frame length is independent of the pitch of the signal. The frame size was chosen to be equal to the GSM speech coder size of 20 ms [EV99]. Generally, the frame size used in speech and audio coders must take a number of considerations into account. Amongst these considerations are delay, perceptual effects, statistical stationarity of the signal and time frequency localization.

The delay introduced in a speech or audio coder that is aimed at operating "on-line" (that is, much like today's cellular phone speech compression algorithms [EV99]) must be small enough to not cause problems with regards to transmission and echo [EV99]. The proposed scheme has no algorithmic delay besides the initial frame size, and 20 ms is sufficiently small to avoid transmission problems [EV99].

Perceptual effects that must be taken into account include the pre-echoes that may result from transform representations [PS00], although this does not appear to be a problem in parametric coders. The length of the frame does effect the existence of pre-echoes, as discussed in [PS00] . As for statistical stationarity, it is well known in speech compression that frame sizes of 15 to 25 ms show pseudo-stationarity, longer frame sizes lose this quality [RS78]. Stationarity allows the modelling of signals by the use of linear prediction and other techniques.

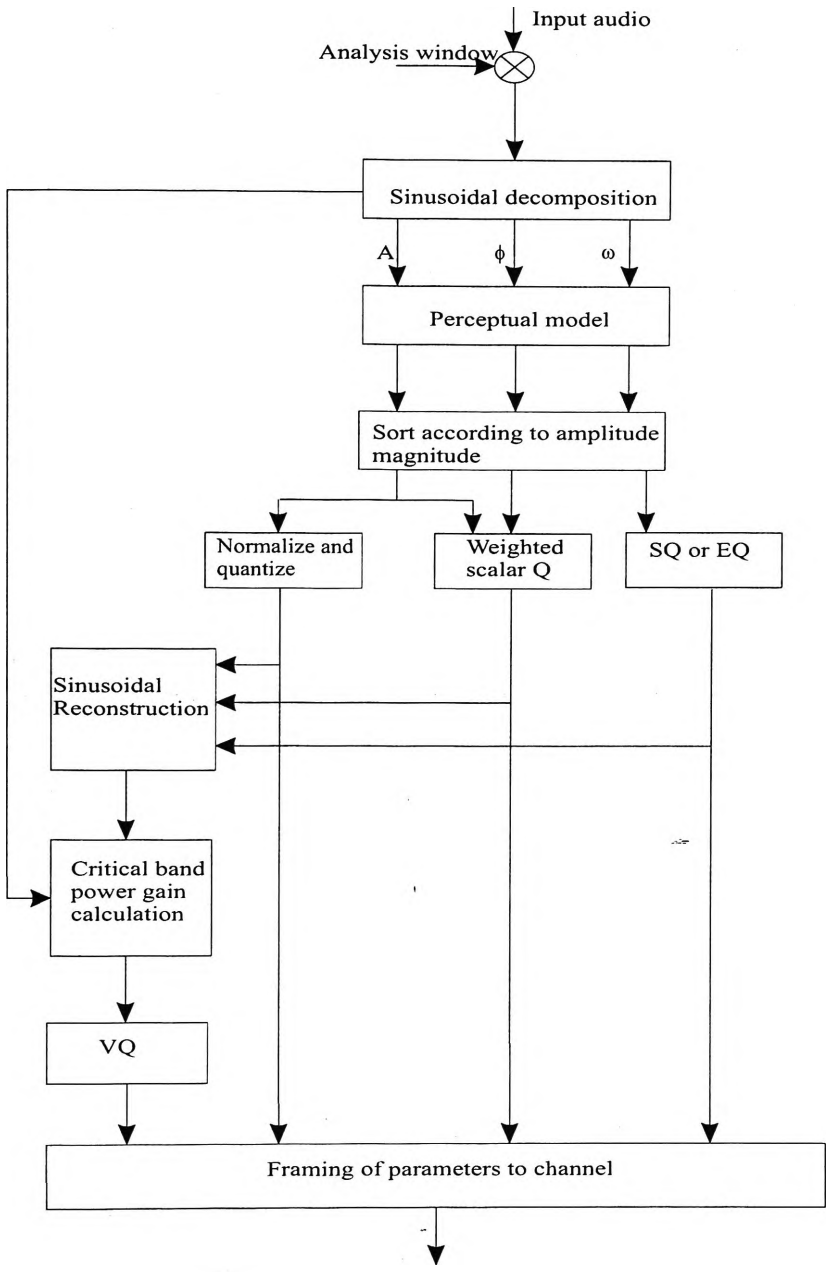

Figure 4.3: The sinusoidal encoder proposed

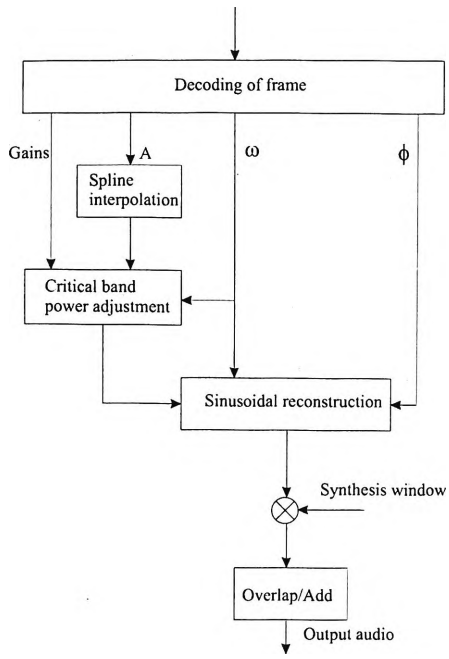

Figure 4.4: The scalable sinusoidal decoder

Overall, a frame size of 20ms provides a compromise for all of the mentioned issues. Compared with other standardized coders [BKS00] a constant frame size of 20 ms is slightly smaller than that normally used. At 44.1 kHz, 20 ms corresponds to 882 samples. The frame selection is carried out by the use of overlapping windows. As mentioned previously, sinusoidal based coders tend to use the Hamming window [MQ95]; however, HILN uses the cosine window [EP00, PM00]. In this work a perfect reconstruction sinusoidal window is used [Mal92]. Figure 4.5 shows a comparison between the Hamming window and the sinusoidal window in the time domain, Figure 4.6 shows the comparison in the frequency domain. Also included in both figures is the proprietary AC-3 window. The aim of using a window is to reduce the frequency leakage when using a transform , it can be seen from the presented figures that all three windows perform similarly in the frequency domain. The AC-3 window and the sinusoidal window both perform better than the Hamming window and both

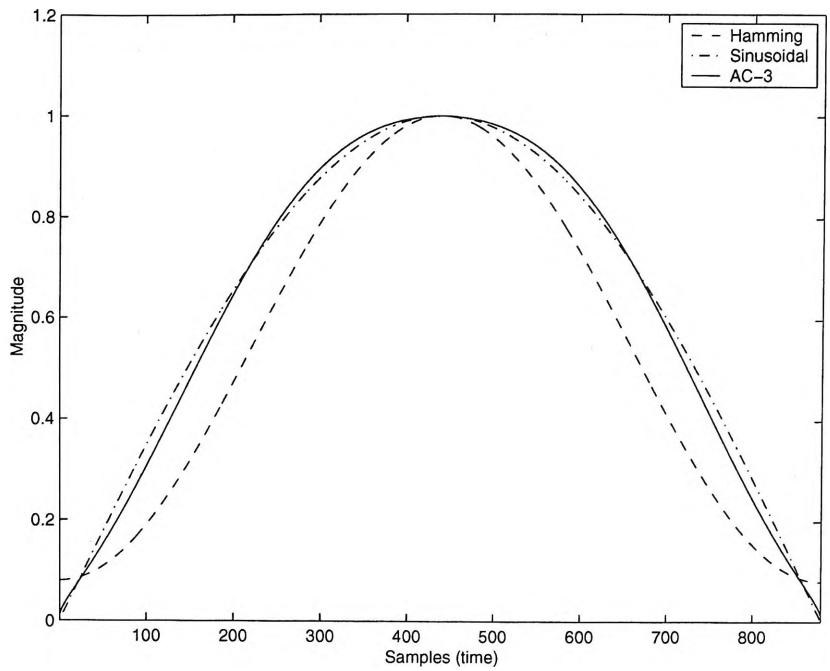

Figure 4.5: The Sinusoidal window, AC-3 window and Hamming window

allow perfect reconstruction with only 50% overlap. The overlap is of little concern since in the original sinusoidal coding scheme overlap-add was used to remove the need for sinusoidal tracking, that is, the use of the sinusoidal window fits well with the general sinusoidal coding scheme. The fact that the sinusoidal window is perfect reconstruction also means that the original signal may be losslessly synthesized should an appropriate coding scheme be utilized. Lossless audio compression is the subject of Chapter 6 of this thesis.

Once a frame has been selected, the sinusoidal model is generated by the use of 4.2.1. The parameters are input to a perceptual model, which acts in a manner best described as a perceptual entropy filter in that perceptually insignificant coefficients parameters are removed from the full set of parameters. The new set of parameters are then sorted according to energy content. The sorted set of parameters are quantized using various techniques, depending on the implementation required, and a gain factor

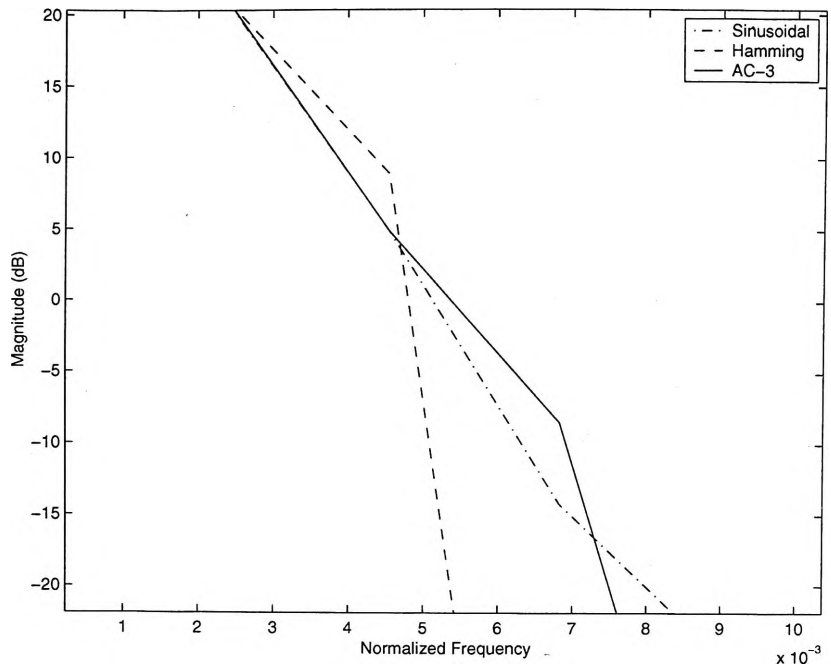

Figure 4.6: The Sinusoidal window, AC-3 window and Hamming window in the frequency domain

is included to improve the quality of the synthesized signal. The aforementioned is a summary of the main body of the encoder, the following sub-sections will elaborate on each component.

4.3.3 The Perceptual Model

Perception and perceptual models have been discussed in detail in Chapter 3. As mentioned in that chapter, simultaneous masking is an effect that can be modelled in the frequency domain by the calculation of the masking curve. The masking curve was originally presented by Johnston [Joh88b] as a way of calculating the perceptual entropy of a signal. The term entropy was used because of the strong similarity with the information theoretic definition of entropy in that the greater the entropy (both perceptual and informatic) the more the resources required to faithfully represent it.

In the coding scheme presented here, the masking model is used to calculate the perceptual entropy contribution of the sinusoidal components in the sinusoidal representation across the critical bands. Thus, by adopting the Johnston technique an average value for the frequency domain masking in each critical band is obtained. The sinusoidal components that fall below this value are considered perceptually redundant. In other words, these components do not contribute to the perceptual entropy of the signal and these components can thus be removed by giving them a weighting of zero.

This application of the masking model differs from the technique used in, for example, MPEG-4 and AC-3 where the method determines a limit for the quantization noise. In those techniques the frequency components are given a weighting that determines the distribution of the quantization bits across the bandwidth of the signal. In our case, the perceptually redundant components are removed to allow the synthesis of a perceptually equivalent signal in a step-by-step fashion. To facilitate the development of this signal in a step-by-step fashion, the remaining sinusoidal components are then sorted according to energy content.

4.3.4 Sorting the Parameters

From Equation (4.2.1), it is clear that the energy content of each of the samples in a frame is determined by the sum of the square of the magnitudes of the amplitudes. As this is the case, the amplitudes are sorted in order of decreasing magnitude. The sorting of these amplitudes allows the coder to concentrate on the parameters that contribute most to the reconstruction of the perceptually equivalent signal. The sorting also provides the added advantage of producing a monotonic relationship

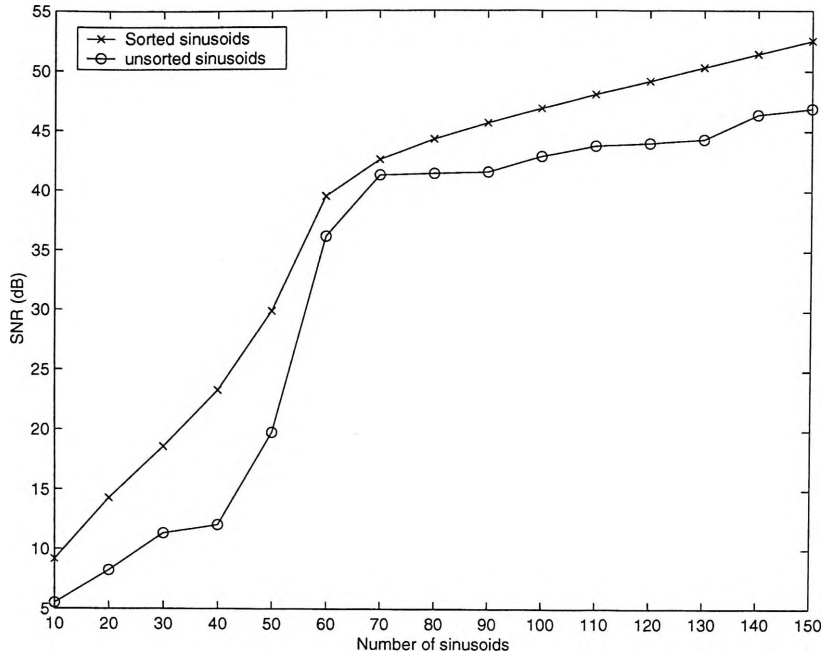

Figure 4.7: Comparing the SNR between the sorted set of sinusoids and the unsorted set used for synthesis

between the ordered amplitude magnitudes. This relationship is exploited through the modelling of the sorted amplitudes by the use of either a monotonically decreasing function or by the use of interpolation. This is unlike other sinusoidal schemes and provides a mechanism to obtain a smoothly scalable audio coding scheme.

When the sinusoidal components are sorted, and the signal is reconstructed from the sorted sinusoidal components it was found, through informal listening tests, that 50 components may be used to produce good quality audio; significantly less than the reported 100 in [MQ95]. Thus, the sorting leads to a quite significant reduction in the number of sinusoids that should be used to obtain a good quality synthesized signal.

Figures 4.7, 4.8, 4.9 4.10 have been included to further illustrate this point. Figure 4.7 gives the Signal to Noise Ratio (SNR) between the original signal and the

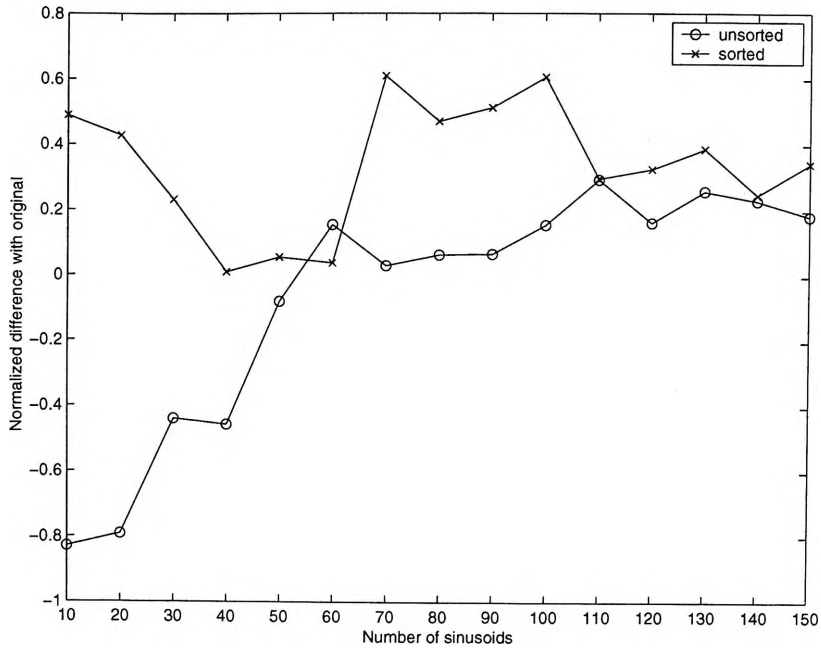

Figure 4.8: Comparing the Roughness between the sorted set of sinusoids and the unsorted set used for synthesis

reconstructed signal as more sinusoids are added. The advantage gained by sorting the sinusoids can be clearly seen. Figure 4.8 shows the relative roughness between the original and reconstructed signal for both sorted and unsorted sinusoids. The roughness, as explained in Chapter 2, is a pleasantness contributing factor which is used in this work along with other indicators to determine in an objective way the subjective effects of the coding scheme on the original sound. The closer the relative roughness values are to 1 the more like the original signal is the synthesized sound. Figure 4.8 shows an interesting result in that for a small number of sorted sinusoids the roughness appears to be closer to the original than for a larger set (eventually the error in the roughness decreases again). This result requires some explanation. The roughness is effected by the modulation frequency of the signal as well as the excitation. The modulation frequency is derived from the envelope which is usually effected primarily by the low frequencies. The selection of the amplitudes according

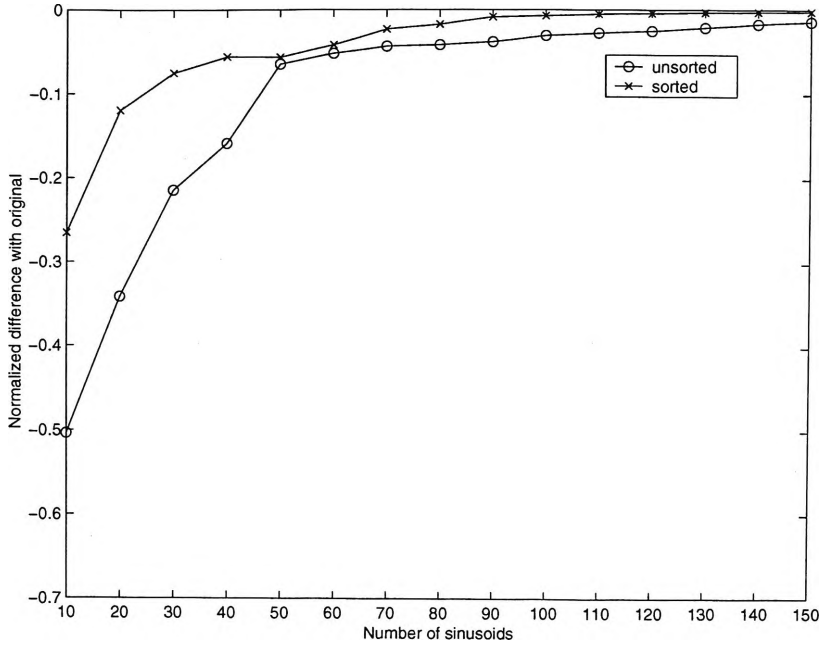

Figure 4.9: Comparing the Loudness between the sorted set of sinusoids and the unsorted set used for synthesis

to amplitude magnitude may (as in this case) lead to some higher frequencies being selected at the expense of lower frequencies; thus altering the envelope information more than the excitation information leading eventually to a divergence of roughness values. To show that the excitation is not the contributing factor to this effect Figure 4.9 has been included, the benefit of the sorting in this case is clear. The same conclusion may be drawn from Figure 4.10 where the sorting of the amplitudes aides in the reproduction of a similar sharpness value.

However, the presented results do not take into account the masking encountered. Figures 4.11 and 4.12 show the same set of results for the roughness and sharpness with the masked components removed. The objective here is to observe whether the masking model used has a detrimental effect on the psychoacoustics of the synthesized signal when combined with the sorting. It can be seen from both figures that in some instances there is even a gain in using the masking model with the sorting rather than

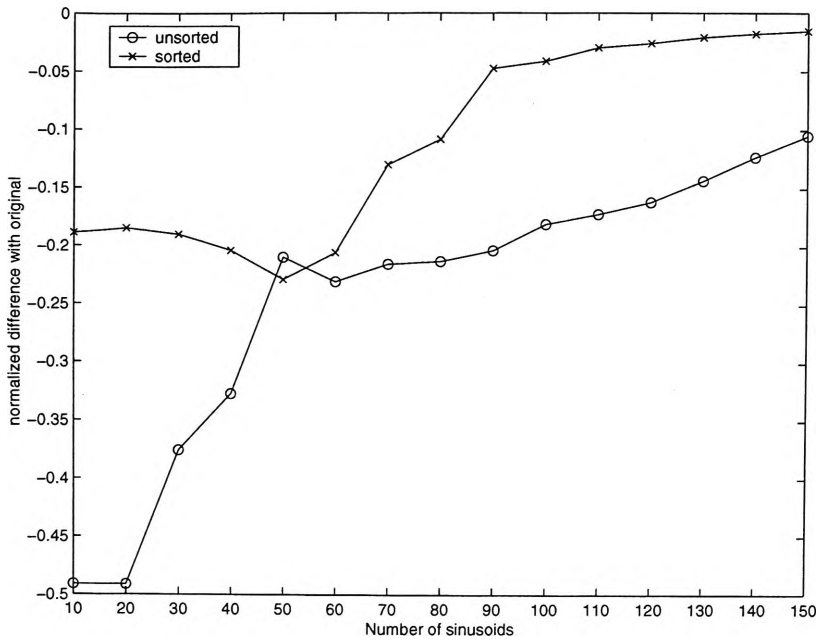

Figure 4.10: Comparing the Sharpness between the sorted set of sinusoids and the unsorted set used for synthesis

a loss in reproducing the psychoacoustic effects of the original signal.

Comparing Synthesis by Sorted Components with A-by-S

Both the A-by-S technique and the sorted sinusoidal components approach aim to reduce the number of sinusoidal components required for a faithful reconstruction of the original signal. As such, it is appropriate, having presented both techniques, to compare (objectively) the relative performance of the two; The important measure is the number of sinusoids required to faithfully reconstruct the signal. Sixteen different files were used in this experiment (the files listed in Table 4.2) and a sampling of the results are shown in Figures 4.13 to 4.15. The results chosen are for files x1, x2 and x5, i.e., a signal that has both harmonic and noise properties, an extremely harmonic signal and a signal with a highly noise like structure. The results are given in terms of Mean Squared Error (MSE) versus the number of sinusoids used in the reconstruction.

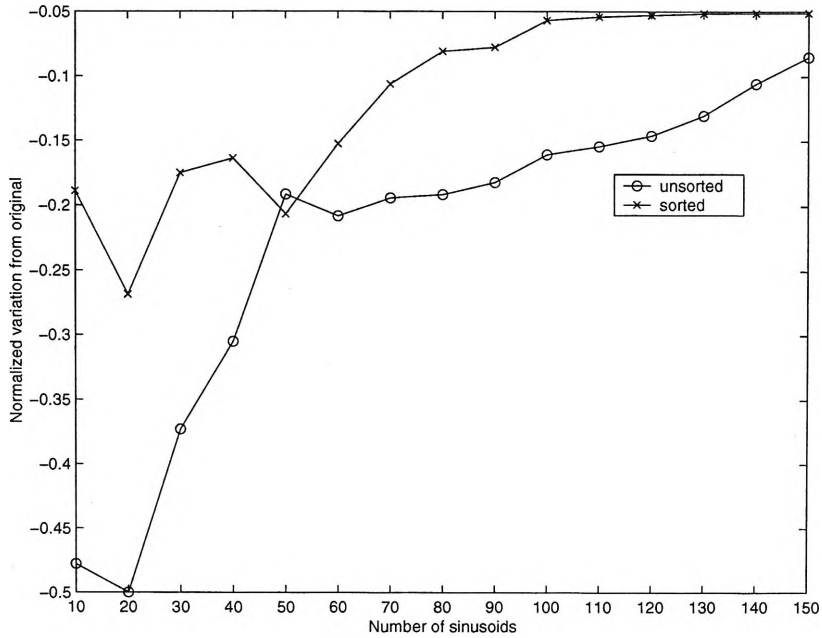

Figure 4.11: Comparing the Sharpness between the sorted set of sinusoids and the unsorted set used for synthesis with the use of a masking model

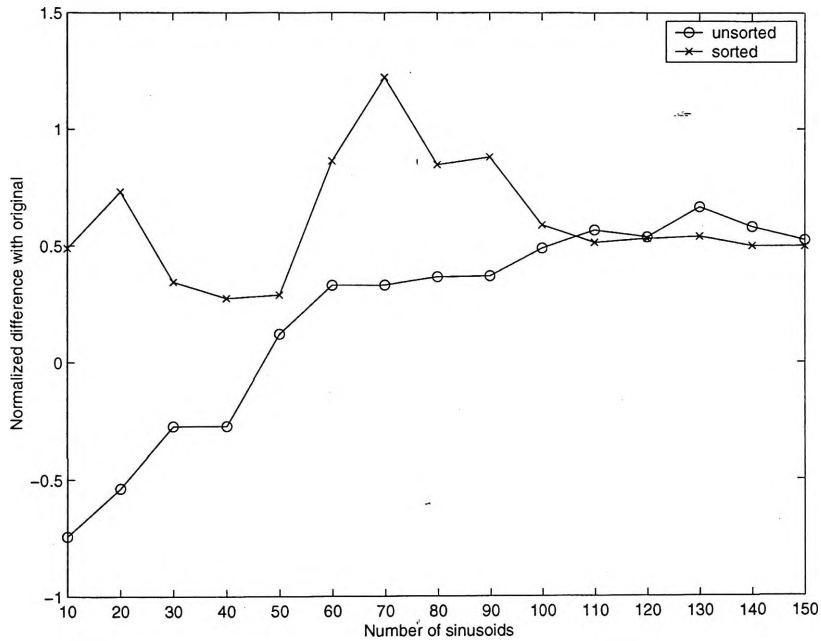

Figure 4.12: Comparing the Roughness between the sorted set of sinusoids and the unsorted set used for synthesis with the use of a masking model

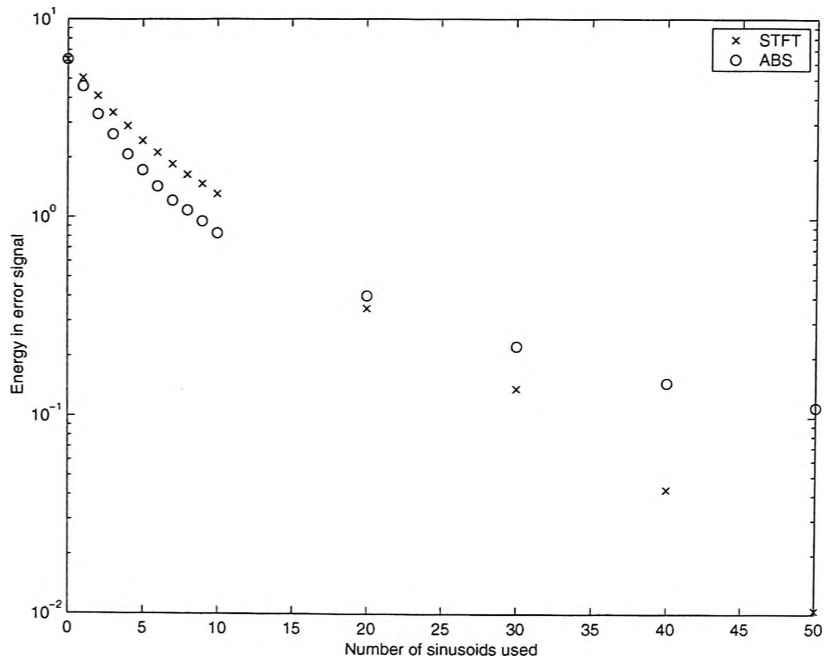

Figure 4.13: Comparing the ABS approach to the STFT approach for x1

The presented results show that the difference between the ABS approach and the sorted STFT sinusoids approach is small, with the ABS only holding a clear advantage for a limited number of sinusoids in all cases. The ABS approach does possess a higher advantage for the most noise like signal than the less noise like signals, however one cannot say that there is a significant difference between the two approaches in terms of MSE (compared with the original signal).

The results presented clarify a trade-off that may be made between the performance and complexity of the two schemes. The sorted sinusoids approach underperforms the A-by-S approach only when very few sinusoids are being used for reconstruction, yet it can be inferred (from the presented algorithms for both approaches) that it is more efficient in comparison to the A-by-S. In this work, the trade-off mentioned has been made in favor of the computational complexity as the MSE gain is quite small.

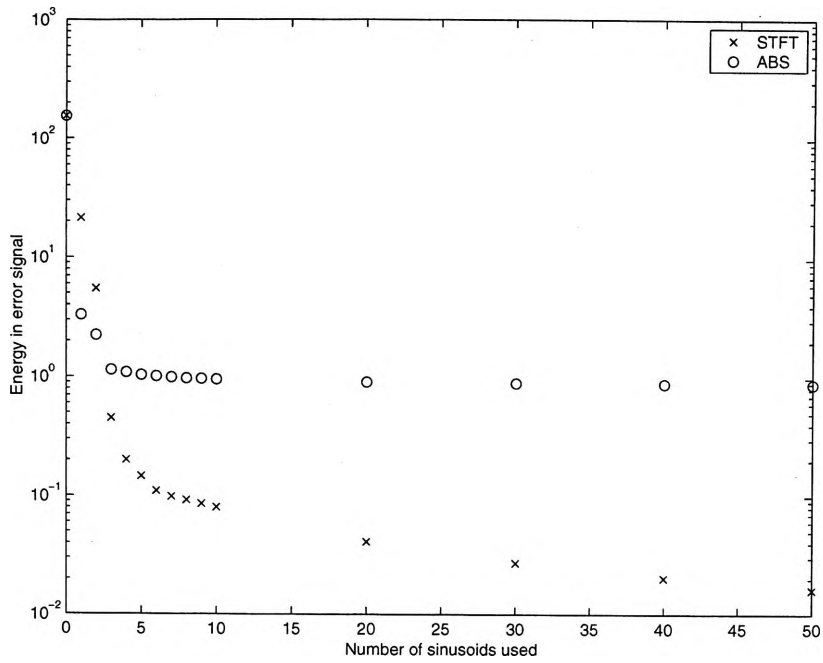

Figure 4.14: Comparing the ABS approach to the STFT approach for x2

4.3.5 Frequency Domain Gains

As only a relatively small number of sinusoids is being utilized in the synthesis of the audio signal, the energy of the synthesized signal in each critical band may be considerably less than the original. This change in energy is translated into a change in the masking curve level, as the masking curve is primarily a function of the energy in each critical band [Joh88a]. In order to counter significant energy loss, twenty five frequency domain gains are used, each corresponding to a critical band. These gains adjust the average energy of the synthesized signal in each critical band to ensure the energy in each band is approximately the same for the synthesized and original signal.

It was found experimentally that more weight should be applied to the low frequency gains than the high frequency gains. The experiments simply involved listening to the resultant synthesized signal. Without this weighting high frequency

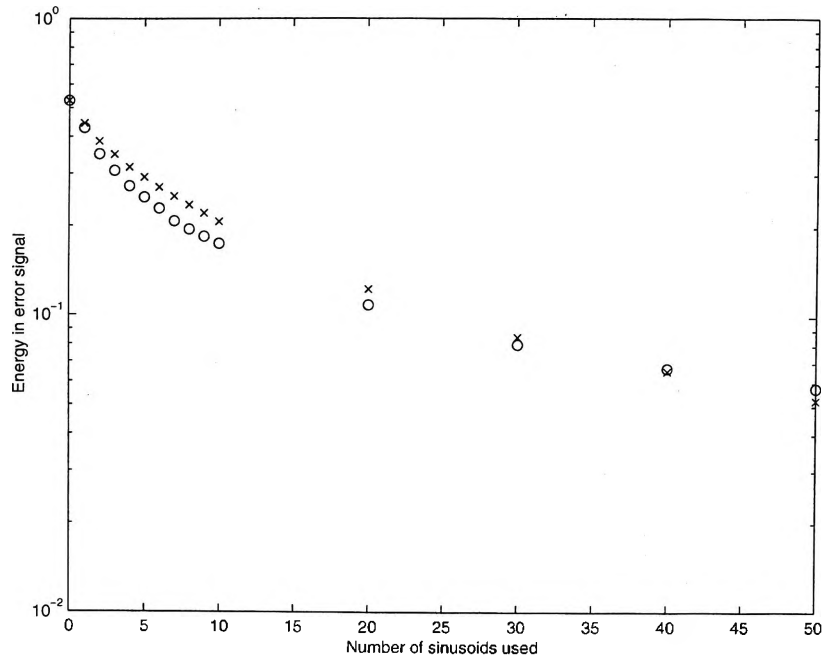

Figure 4.15: Comparing the ABS approach to the STFT approach for x5

"scratches" (i.e. sounds that sound like a scratch) may occur in the synthesized audio when the gains are quantized. The weighting is actually applied in the log domain, this makes sense as the calculation of the original masking curve is carried out in that domain. This is equivalent to the use of a power law on the actual calculated gains. The un-quantized gains are given by:

$$\gamma_\ell = \frac{P_{o\ell}}{P_{s\ell}} \qquad 0 \leq \ell \leq 24 \tag{4.3.1}$$

where γ_ℓ is the gain in a given band, $P_{o\ell}$ is the power of the original signal in the critical band ℓ and $P_{s\ell}$ is the synthesized signal power in that band. The gains are weighted by the use of the following equation:

$$\gamma_{f\ell} = (\gamma_{i\ell})^{x_\ell} \qquad 0 \leq \ell \leq 24 \tag{4.3.2}$$

$\gamma_{f\ell}$ and $\gamma i\ell$ are the final and initial gains respectively and x is a weight varying between 0 and 1. An example of a weighting vector is shown in Figure 4.16. The

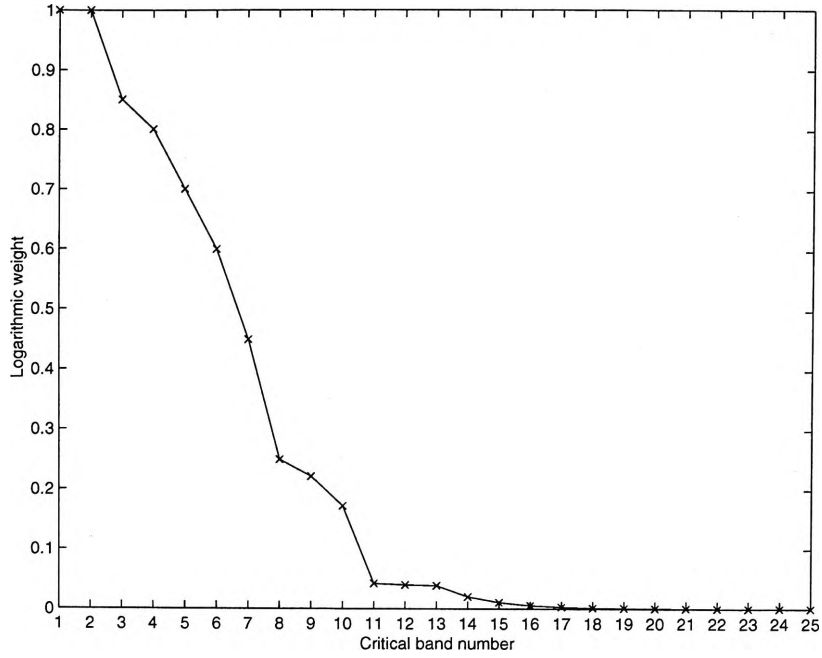

Figure 4.16: An example weight vector used in the described coder

power law technique allows the gains to be adjusted between their original value and 1. This permits the adjustment of sinusoids belonging to the low frequency band whilst leaving others unchanged. The critical band gains have been found to improve the perceived quality of the synthesized signal in informal listening tests.

4.3.6 Quantization

The sorting of the sinusoids according to energy content allows preferential transmission of the parameters as well as the exploitation of the relationship between consecutive amplitudes. Scalar quantization is used for the quantization of most of the parameters in this work. This may seem wasteful, but, as will be explained later, scalar quantization is necessary for some parameters and sufficiently efficient for the quantization of other parameters.

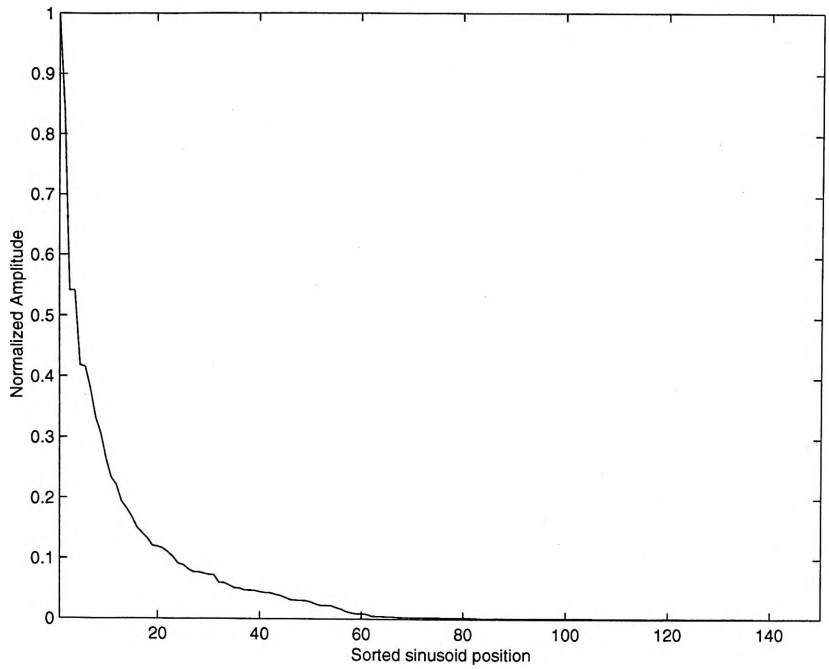

Figure 4.17: An example of a set of sorted amplitudes

Quantizing the Amplitudes

The sorted amplitudes have a monotonic relationship. An example set of sorted amplitudes is shown in Figure 4.17. The gradient of the amplitude magnitudes is related to the signal being coded, with tonal signals having a much higher rate of decrease than non-tonal signals.

Two approaches for quantizing the amplitudes have been considered. The first models the amplitudes by the use of a smooth function, such as an exponential, and the second uses spline interpolation. In the case of the exponential modelling, the amplitude curve is modelled by attempting to fit an exponential to it that has the form:

$$\hat{A} = \beta \exp(\alpha i) \tag{4.3.3}$$

where $\hat{A}$ is the set of approximated amplitudes, β and α are coefficients that are

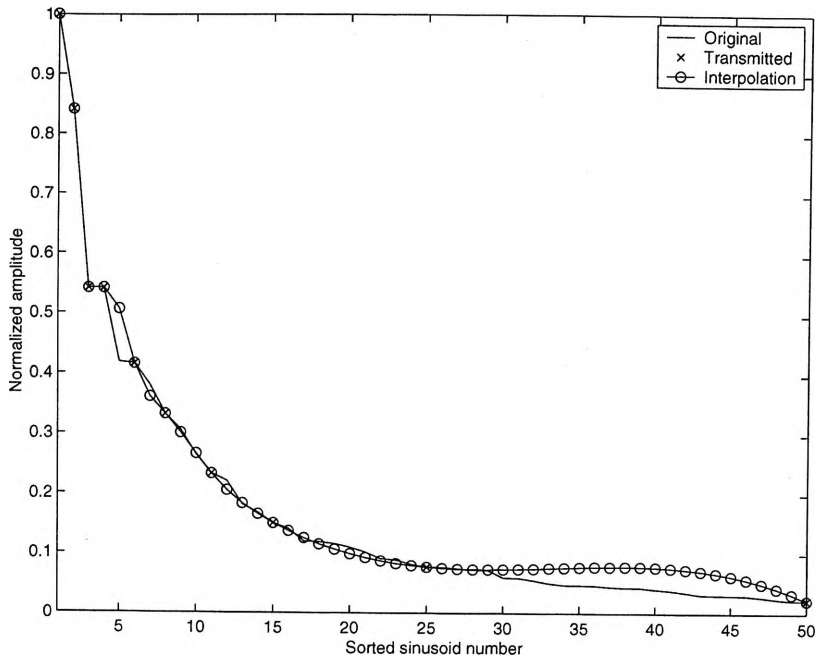

Figure 4.18: An example of spline interpolating between selected sorted amplitudes

determined adaptively to minimize the MSE of the model and i is the sorted index of the amplitude. On the other hand, the interpolation technique involves the selection of a number of amplitudes, quantizing those amplitudes and using a spline interpolator for points between those amplitudes. Figure 4.18 shows an example of a sorted set of amplitudes from which ten amplitudes have been transmitted corresponding to positions {1,2,3,4,6,8,11,15,25,50}. The greater the number of amplitudes transmitted, the less distortion that is encountered. This feature allows the coder to scale, something that the exponential model cannot do. However, this technique does rely heavily on the accurate quantization of the transmitted amplitudes.

Finally, Figure 4.19 shows a comparison between the performance of the exponential model of the amplitudes and the spline interpolated result. The test has been conducted using 50 sorted sinusoids of an audio signal. The exponential result is shown as being constant as it has been obtained for the modelling of 50 amplitudes.

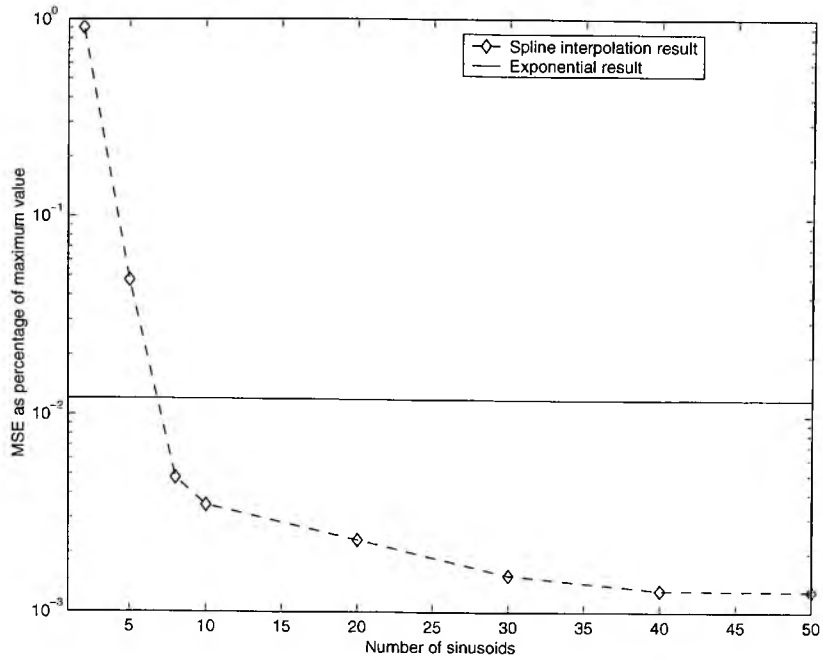

Figure 4.19: A comparison of exponential interpolation and spline interpolation in terms of MSE

The spline interpolator's performance improves at a decreasing rate with an increase in the number of amplitudes transmitted. Note that the knee point of the interpolator is around the 10 amplitude mark. Ten amplitudes are normally transmitted in the non-scalable version of this coder. In the scalable version, the number of amplitudes varies according to the available bit rate.

Quantizing the Phase

As mentioned previously, to obtain a low rate coder one finds it necessary to model the phase as in [MQ95]. This is not an ideal solution, yet it produces good results at low rates. The coding scheme being presented here has not been specifically aimed at low rates, it is a medium rate coder when implemented in its non-scalable form, and a medium to high rate coder when implemented in its scalable form. As such, the phase

information has been maintained. Maintaining the phase has been acknowledged in [MQ95] as necessary to achieve a high quality synthesized sound. In the work involving A-by-S [GS92],[EP00] modelling, the envelope of the original signal implicitly captures the phase information of the sinusoidal model and so is quite important to the overall quality of the synthesized audio [EP00].

In the non-scalable coder being described here, the phase is quantized using weighted scalar quantization; where the amplitudes are used as the relative weights. That is, phases associated with large amplitudes are quantized more accurately than phases associated with smaller amplitudes. The scalable version of the proposed coder used in this work utilized a uniform scalar quantizer for the phase to allow simpler coder implementation.

Quantizing the Indices

The indices indicate the spectral position of the sinusoids before sorting. As the sinusoids have been re-arranged, the indices are required by the decoder for correct synthesis. It has been found experimentally, that even a small level of distortion in the indices leads to annoying artifacts in the synthesized signal. Thus, it was decided that to avoid the potential of having such artifacts the indices would be coded losslessly.

For a frame length of 880 samples, 441 sinusoids are possible using the STFT method. Thus, nine bits per index would be required for fixed length lossless coding. In the scalable implementation of this coder a fixed length scalar quantizer is used. However, it was found that the first order entropy of the indices collected over many experiments was 4.24 bits per index.

Thus, the theoretical limit for coding the indices losslessly is 4.24 bits per index.

Table 4.1: The ten most probable frequencies and their Huffman codes

Index value	Frequency (HZ)	code word	Prob. of occurrence
7	350	11101	0.257
8	400	11111	0.253
9	450	11110	0.252
10	500	11011	0.251
6	300	11010	0.249
11	550	11001	0.248
5	250	11000	0.248
12	600	10001	0.242
4	200	10011	0.238
13	650	10010	0.236

Using this result, a Huffman code was designed for the indices based on the probability of appearance in the most energetic 50 sinusoids. A Huffman code allocates more bits for indices that have a low probability of appearance. The code is a prefix code [GG92] in that the prefix of each code word is unique and thus the code is decodable. The code is loss-less and variable in length. The most probable 10 indices and the equivalent code are given in Table 4.1. Other loss-less variable length codes may be used, such as arithmetic coding [WNC87], or a Lempel-Ziv style code [GPS94]. The Huffman code designed resulted in a variable length code set for the indices with a mean length of 6.089 bits per index, a reduction of 32 % on the fixed length code.

Quantizing the Gains

To quantize the frequency domain gains, vector quantization was employed. Vector quantization (VQ) allows whole vectors to be coded with a single codeword [GG92]. Each code word represents an entry in a pre-defined code book. In this work a 10 bit code book (1024 entries) was trained to quantize the gains with each gain vector having 25 elements. The use of this code book, in combination with the weighting of

the gains, produced insignificant distortion in the synthesized audio.

4.4 Results

In this section two sets of results will be presented; both objective and subjective (the focus of the results will be on the scalable scheme rather than on the variable rate scheme. The reason for this is that the variable rate scheme is not as versatile as the scalable scheme). Comparing the two schemes, one can say that the variable rate scheme performs better than the scalable scheme at around the 40 kbps, which is the rate for which the variable rate scheme was designed for. However, by just listening to the synthesized audio it was found that the difference is not significant enough to warrant a complete subjective comparison of the two schemes. Instead, only the objective results of the variable rate scheme will be given. As for the subjective results, the scalable scheme is compared to the MPEG-4 AAC transform coder at four rates; 16, 32, 42 and 64 kbps. Also, pleasantness factors (Loudness, Sharpness and Roughness) are given for some of the test material at those rates to illustrate the objective perceptual scalability of the sinusoidal coder.

The test material used is listed in Table 4.2 and was obtained from [mpe]. This same test material is used throughout this thesis.

The first set of results included here are segmental signal-to-noise-ratio (SegSNR) results. As these results are not related to a perceptual measure they require normalization. To allow the comparison of the SegSNR results Table 4.3 lists the SegSNR of the original signals with only the perceptually irrelevant material removed, i.e., signal components that were deemed to have zero perceptual entropy were not coded. The resultant sound is transparent, yet the SegSNR is quite low; in fact, in none of the

listed test files does the SegSNR reach 30 dB, which is 8 dB below the toll quality definition of speech [NJ84]. This is an expected result [Ryd96], and by comparing the rest of the SegSNR results presented here to those in Table 4.3 one can approximate how close the synthesized signal is to the perceptually significant signal.

4.4.1 Variable rate coder objective results

Table 4.4 lists the results obtained for the variable rate coder. One can see a clear difference between the SegSNR values listed in Tables 4.3 and 4.4 indicating a loss in perceptual quality. The variable rate coder operated at an average rate of 38.9 kbps and by simply listening to the synthesized audio one can tell that the coder is not transparent at that rate. However, the coder sounded much like the MPEG-4 AAC transform coder (Verification Model or VM) operating at 42 kbps (which is also not transparent at that rate). The quality of the coder can be increased with an increase in rate by using, for example, finer quantization of the amplitudes and less interpolation. It would be expected that the SegSNR would then approach that listed in Table 4.3. It must also be pointed out that Table 4.3 lists SegSNR results without any quantization, i.e., none of the perceptually significant parameters are quantized, and that must be taken into consideration when comparing the results listed in the two tables.

4.4.2 Scalable rate coder objective results

The scalable set of results are shown in Figures 4.20 to 4.23. Each figure shows the SegSNR of the synthesized files at the given rates. All of the files surpass the variable rate version by the 64 kbps mark, and most of the files show clear scalability. The

Table 4.2: The Signal Content

Signal Name	Signal Content	Signal Name	Signal Content
x1	Bass	x9	English Female Speech
x2	Electronic Tune	x10	French Female Speech
x3	Glockenspiel	x11	German Female Speech
x4	Glockenspiel	x12	English Male Speech
x5	Harpsicord	x13	French Male Speech
x6	Horn	x14	German Male Speech
x7	Quartet	x15	Trumpet
x8	Soprano	x16	Violoncello

Table 4.3: SegSNR of the perceptually significant signals

Signal	SegSNR (dB)	Signal	SegSNR (dB)
x1	22.3	x9	20.4
x2	15.5	x10	20.8
x3	22.7	x11	17.3
x4	29.1	x12	18.0
x5	10.53	x13	16.9
x6	26.7	x14	17.9
x7	26.5	x15	28.9
x8	25.9	x16	17.1

Table 4.4: SegSNR results for variable rate coder

Signal	SegSNR (dB)	Signal	SegSNR(dB)
x1	13.4	x9	10.9
x2	10.7	x10	10.6
x3	15.3	x11	9.6
x4	16.9	x12	10.7
x5	7.3	x13	9.1
x6	17.3	x14	9.6
x7	14.1	x15	16.4
x8	16.3	x16	12.5

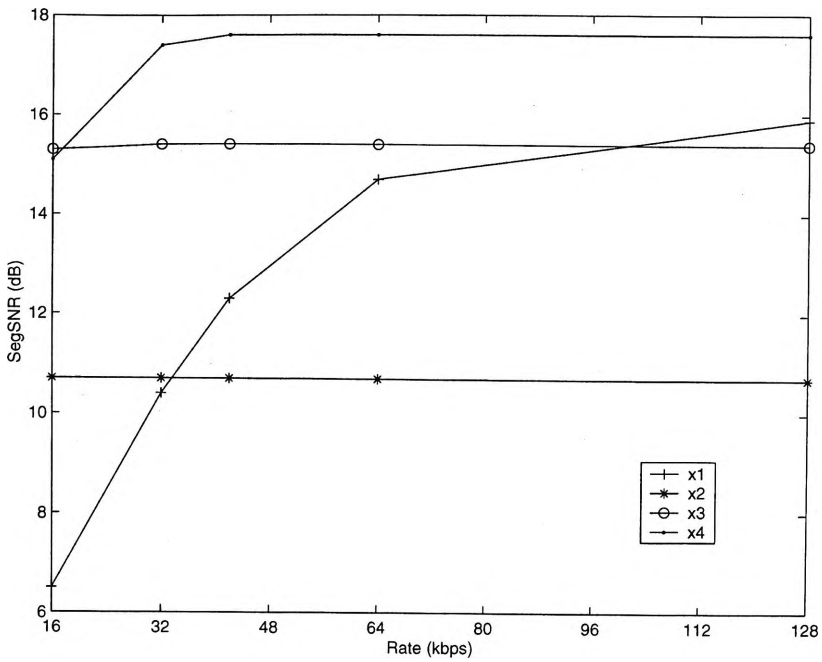

Figure 4.20: SegSNR results for the scalable sinusoidal coder for signals x1 to x4

files that do not show scalable behavior are highly tonal files which are modelled quite well at 16 kbps, with the given limitations of quantization noise and limited phase representation discussed earlier. The worst performing files in terms of SegSNR are the files that have strong resemblance to noise, in terms of Power Spectral Density (PSD) representation. The Harpsichord signal (x5)is one such signal. The deficiency of the sinusoidal model for noise like signals is well documented (see [Goo97] for example) and these results clearly display this effect.

Another objective method of analyzing the results of the scalable coder has been described in Chapter 2, where the sensory pleasantness parameters were shown to allow a psychoacoustic analysis of the synthesized audio signal. In this case (for a scalable coder) one would require that a smooth transition in pleasantness be achievable. It has already been noted that a decrease in roughness, sharpness and loudness

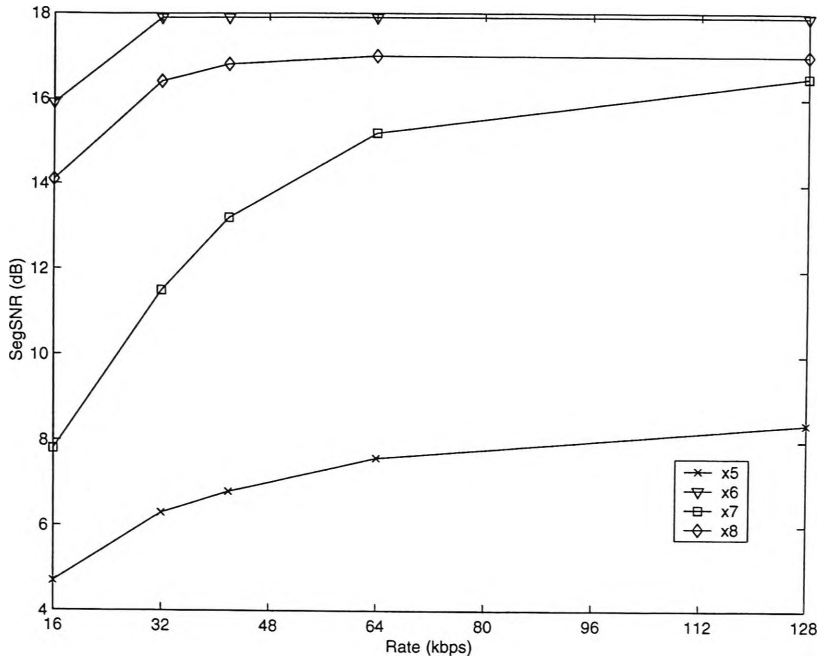

Figure 4.21: SegSNR results for the scalable sinusoidal coder for signals x5 to x8

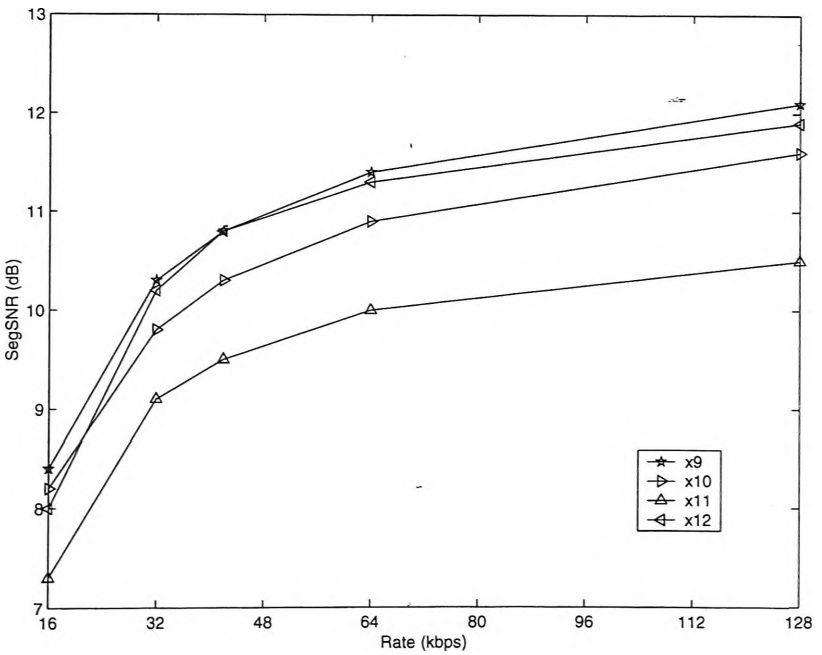

Figure 4.22: SegSNR results for the scalable sinusoidal coder for signals x9 to x12

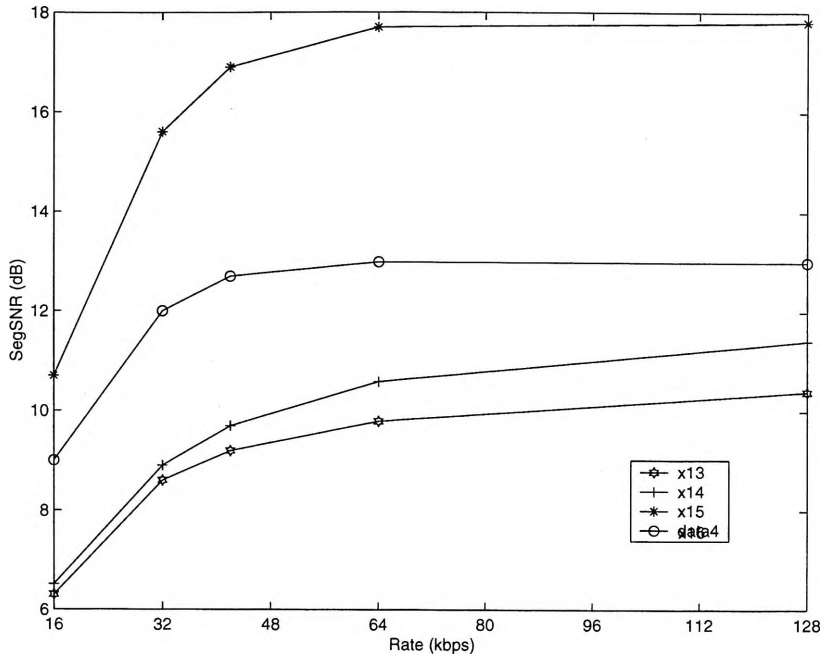

Figure 4.23: SegSNR results for the scalable sinusoidal coder for signals x13 to x16

would mean an increase in sensory pleasantness. Figures 4.24 to 4.27 show the mentioned pleasantness factors for the scalable sinusoidal coder for four of the sixteen test files. The plotted results are given in terms of mean variation from the original value as a percentage. This means that if the plotted values are heading towards zero then the synthesized signal should be heading towards having the same psychoacoustic properties as the original signal. When this is combined with a positive signal to noise ratio then one can state that the perceptual properties of the synthesized sound and the original sound should be the same.

It can be seen that the sinusoidal coder does scale relatively smoothly when analyzed using the loudness (N), sharpness (S) and roughness (R). The percentage variation decreases significantly for most of the files shown as the bit rate increases. It can also be seen that each file varies in terms of which psychoacoustic factor is most accurately produced; this is related to the content of the file. In this case, four

quite different files have been presented (in terms of content as well as spectral representation), hence one would expect different psychoacoustic factors to be distorted differently by the coding process.

More insight can be gained by considering the obtained results individually. For x1, a low frequency dominated original signal, this is well matched by the sinusoidal model and so the scalable coder displays a smooth reduction of psychoacoustic variation between the synthesized signal and the original to within 10% of the original signal's value. In the case of x5, a much noisier type of signal, one observes a similar reduction in the variation of psychoacoustic parameters except that the starting point is significantly higher than that for signal x1. As for x6, it is a very harmonic type of signal and so it is modelled well at low rates (resulting in low mean variations to begin with) with slightly decreasing variation as the rate increases. The roughness seems to show a somewhat odd behavior in that it stays significant beyond the 32 kbps mark. This is an indication that the envelope of the synthesized signal is not being improved upon because the contributing factors to a sounds roughness are its excitation level and its modulation frequency. The excitation level is the primary indicator of the loudness and as that is clearly improving then the gap in roughness must originate from the modulation frequency. However, the modulation frequency is approximated by the frequency of the envelope of the signal. Thus, the envelope of the synthesized signal is not being improved for higher rates.

In the interest of comparing the scalable sinusoidal coder with an existing popular coder, the same files were coded using the MPEG-4 AAC transform coder at the rates 16, 32, 42, 54 and 64 kbps. The aim was to observe the effect that this coder has on the psychoacoustic measures and compare it with the scalable sinusoidal coder.

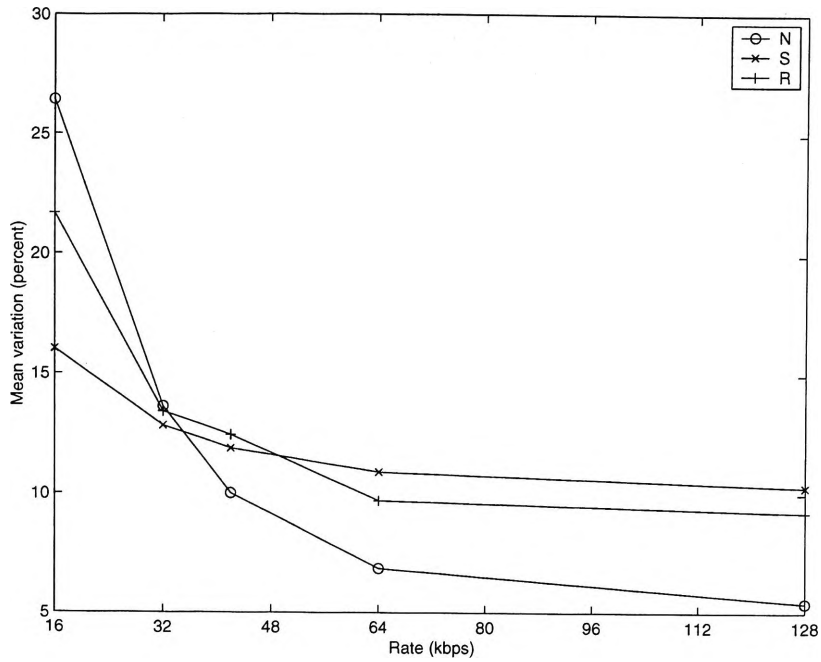

Figure 4.24: Pleasantness factors results for the scalable sinusoidal coder synthesized file x1

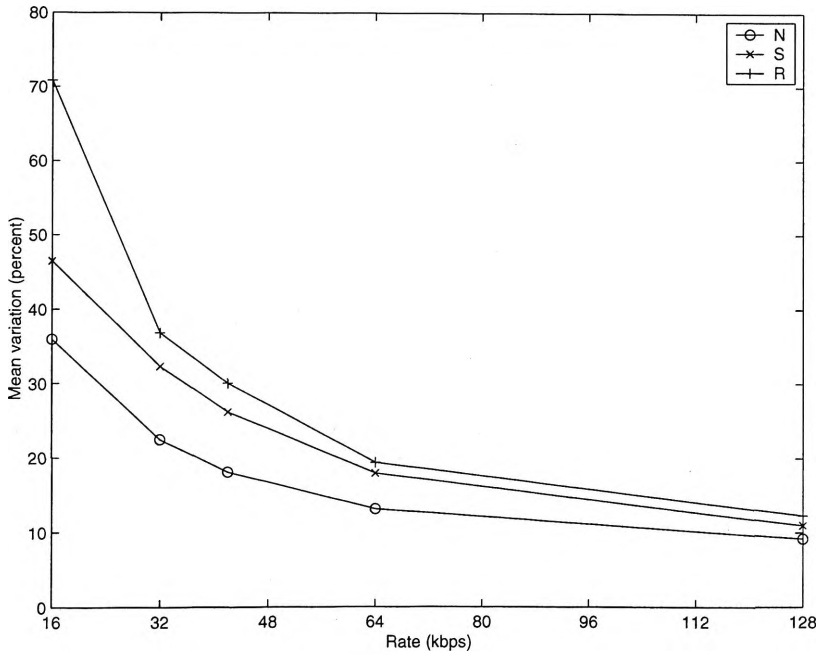

Figure 4.25: Pleasantness factors results for the scalable sinusoidal coder synthesized file x5

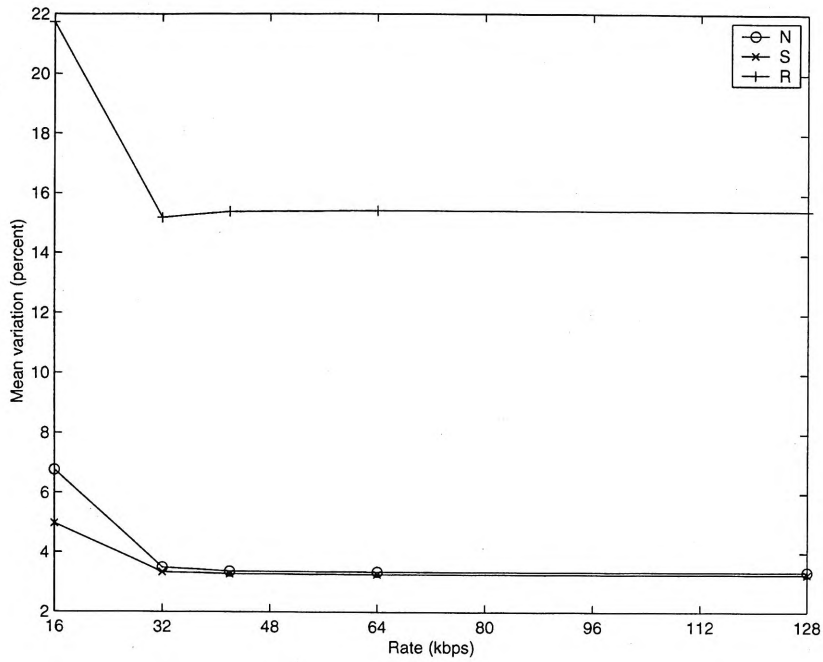

Figure 4.26: Pleasantness factors results for the scalable sinusoidal coder synthesized file x6

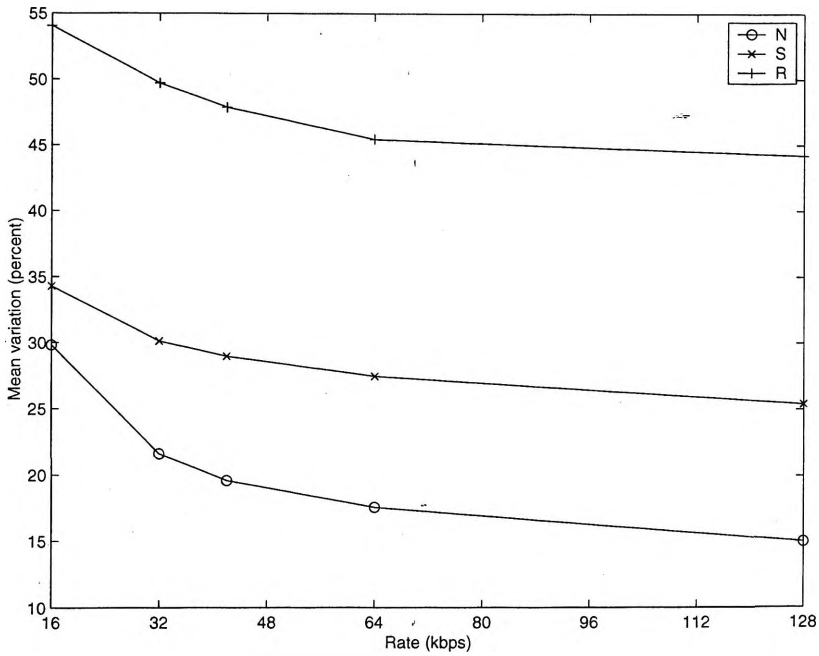

Figure 4.27: Pleasantness factors results for the scalable sinusoidal coder synthesized file x9

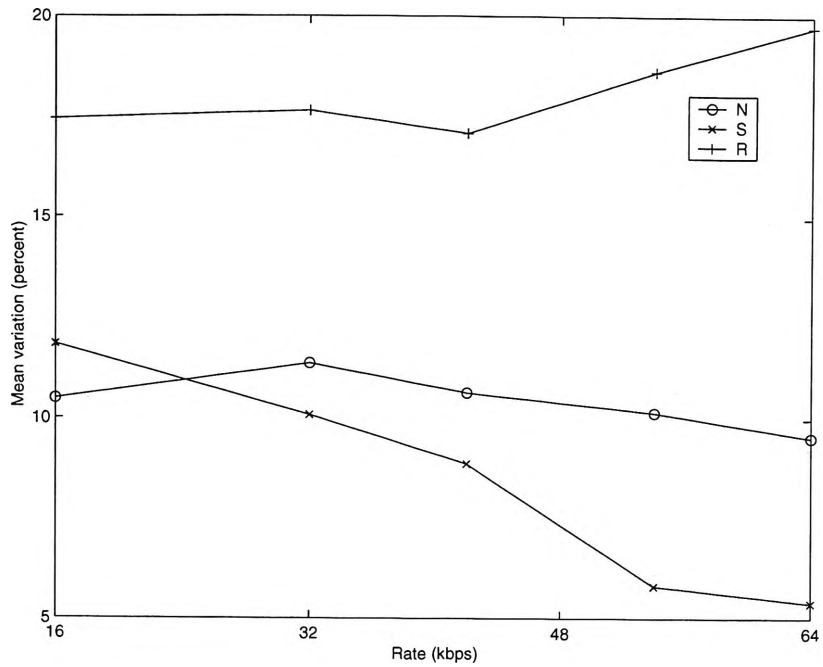

Figure 4.28: Pleasantness factors results for the MPEG AAC coder synthesized file x1

Figure 4.28 shows the psychoacoustic measures for signal x1. It can be seen that the variation is inconsistent across the bit rates, i.e., the measures neither increase nor decrease consistently as the rate increases. It is interesting to note that the loudness and roughness seem to follow the same pattern until the 42 kbps mark. As has been mentioned, both loudness and roughness are functions of the excitation pattern in the human auditory system. However, roughness has the contributing factor of modulation frequency. The divergence of the loudness and roughness indicates a different approach in compression for rates below 42 kbps to those above 42 kbps. This is not a surprising result when one considers the MPEG-4 AAC coding scheme, in which more auditory 'objects' are added as the bit rate increases, this coder was discussed in detail in Chapter 3.

Figure 4.29 shows a smooth decrease in all of the measures as the rate increases.

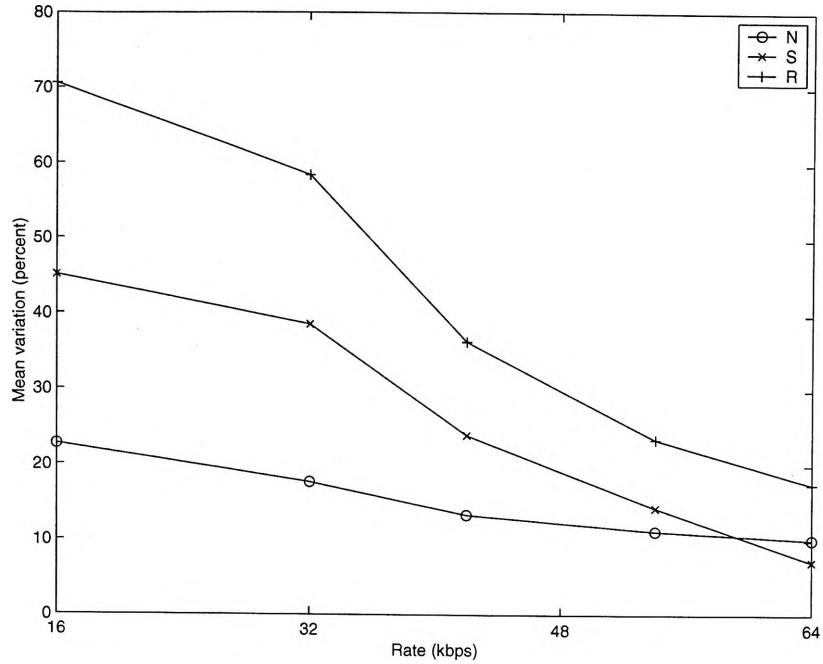

Figure 4.29: Pleasantness factors results for the MPEG AAC coder synthesized file x5

Recall that x5 is a very noise like signal and when listening to the AAC synthesized file, one can hear a very good reproduction of the original sound from low rates. This is testament to the noise substitution procedure that is applied in the AAC coder. Similarly, Figure 4.30 shows that the pleasantness factors decrease with increasing rate for file x6, however, the starting point of the factors indicates a massive variation. This was verified by listening to the file synthesized using 16 kbps. Figure 4.32 has been included to illustrate the effect that the AAC coder had on x6 at 16 kbps, the difference in the psychoacoustic factors generated for each signal becomes more understandable by studying this figure. The file that shows the oddest behavior is x9 with an increasing variation in sharpness and loudness and a decreasing variation in roughness with an increasing rate. This file is a female speech file and by listening to the coded results it is easy to pick the difference between the original and the

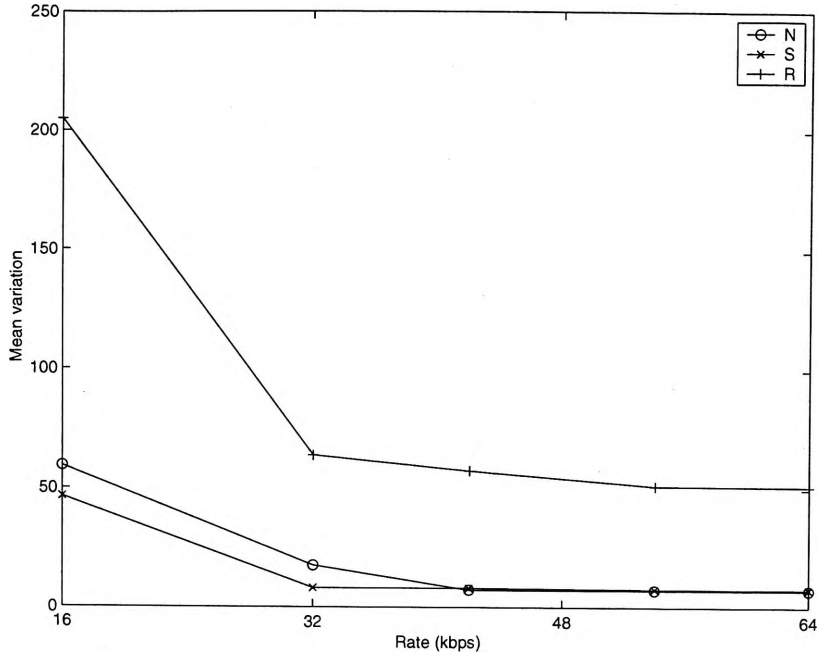

Figure 4.30: Pleasantness factors results for the MPEG AAC coder synthesized file x6

synthesized file, the better roughness performance seems to be due to more attention being paid to the envelope of the signal, this may be due to the temporal noise shaping tool that is used in the AAC coder. Although this is primarily a tool for the control of pre-echo effects, it does so by shaping the "noise" so that it better matches the envelope of the original signal.

A comment about the scalable objective results

The objective results presented have been chosen to illustrate the performance of the scalable sinusoidal coder in comparison with the AAC for files that are both difficult and easy to code for both coders. It is notable that the sinusoidal coder does scale more smoothly than the AAC coder, this can be explained by the structure of both coders and this has been the focus of the previous discussions. However, the results presented do not show whether the psychoacoustic measures are increased or

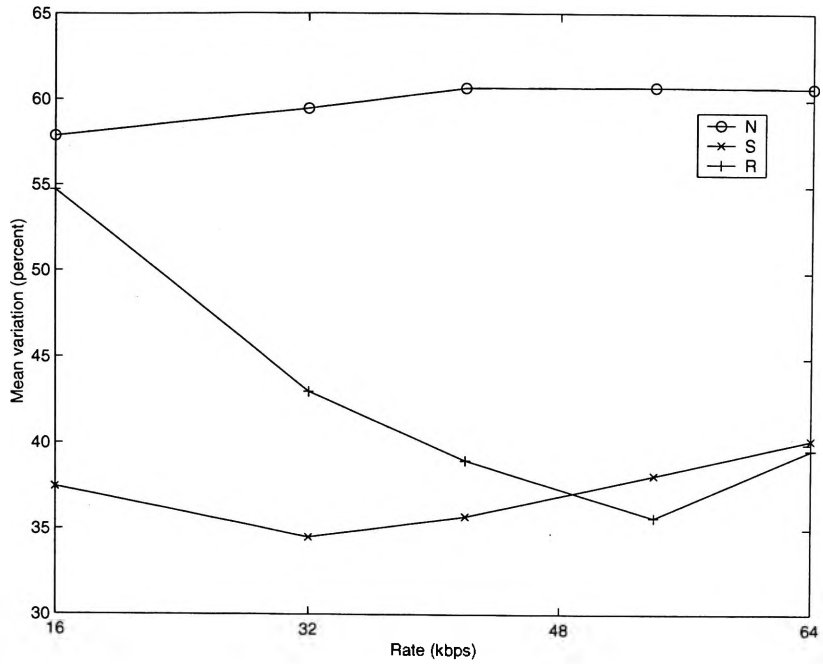

Figure 4.31: Pleasantness factors results for the MPEG AAC coder synthesized file x9

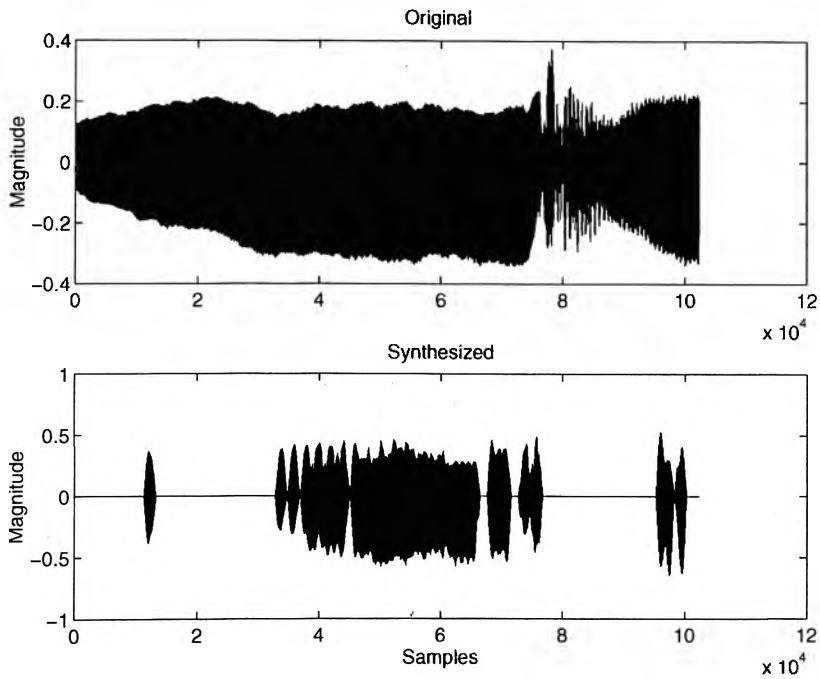

Figure 4.32: 100 frames of the coded x6 file using AAC at 16 kbps

decreased by the coding process. Instead the results have been focused on an absolute error between the original and synthesized. In the interest of completeness, Figure 4.33 is included. This figure shows the mean variation from the original x9 signal without the use of the absolute error, i.e., both positive and negative variations are taken into account. It can be seen that the loudness of the coded file is typically below that of the original, whilst the sharpness and roughness are usually above the original. Both of those observations can be heard on the coded file. It is interesting to note that both the sharpness and the roughness head towards the original sharpness and roughness indicating that the distribution between positive and negative errors becomes more equal as the rate increases, causing the absolute error to remain high, while the non-absolute error decreases on average. This result indicates a psychoacoustic correcting behavior, where the psychoacoustics of the coded file vary between higher and lower values than the original regularly. This would explain the increase in perceptual quality whilst the psychoacoustic measures continue to show a clear difference.

In comparison, Figure 4.34 shows the same type of results using the scalable sinusoidal coder. It is immediately clear that the type of distortion introduced by the sinusoidal coder is much more consistent than that introduced by the MPEG-4 AAC coder. This is not strange as the AAC coder tends to be more signal adaptive than the sinusoidal coder, however, there is a certain advantage in the consistency of an injected psychoacoustic error in that it is possible to devise techniques to counter this distortion.

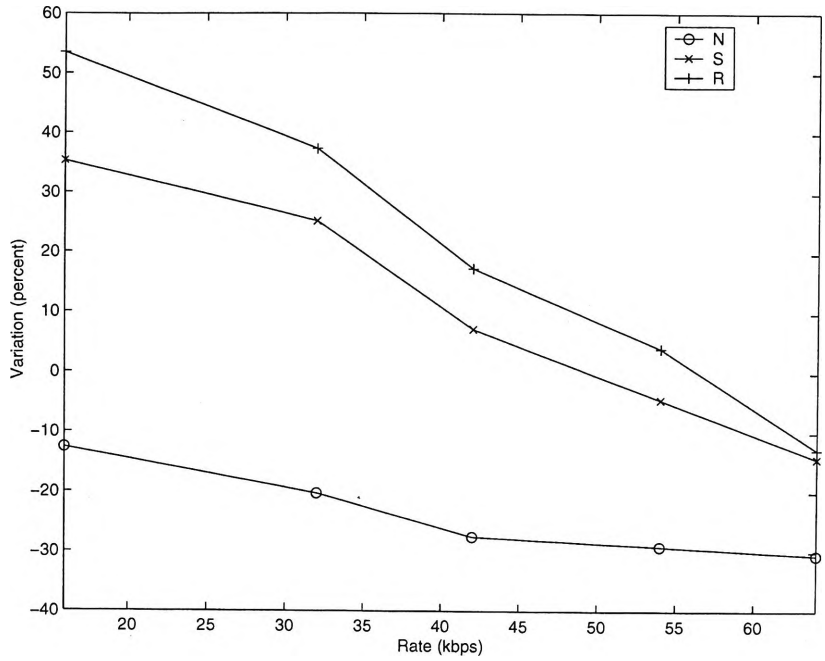

Figure 4.33: Psychoacoustic factor real mean percentage variation for AAC coded x9

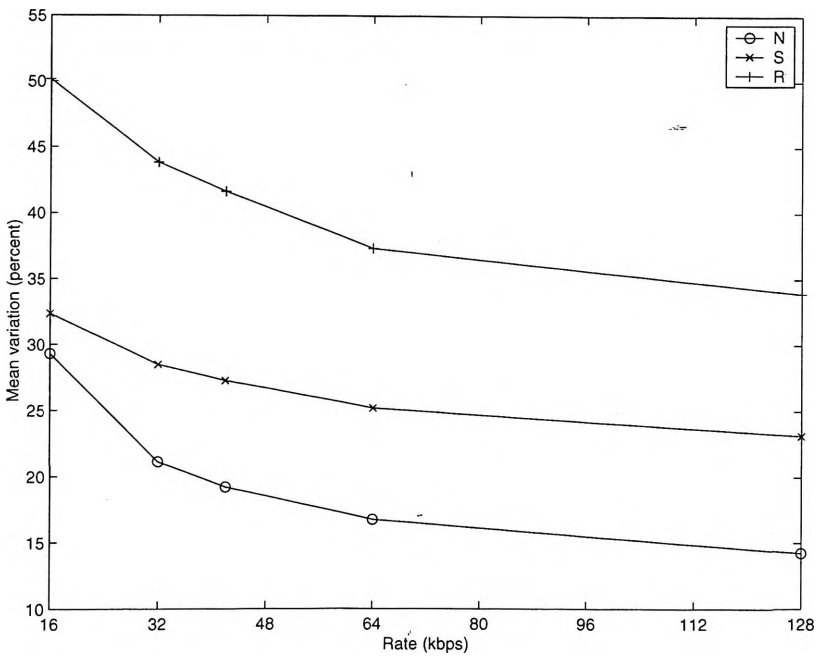

Figure 4.34: Psychoacoustic factor real mean percentage variation for the scalable sinusoidal coder coded x9

Table 4.5: Complete Relative Quality Scale

score	*Relative quality*
5	A much worse than B
4	A worse than B
3	A same as B
2	A better than B
1	A much better than B

4.4.3 Subjective scalable results

Subjective tests have been organized to compare the quality of the scalable sinusoidal coder to the MPEG-4 AAC. The files used in the tests were all those listed in Table 4.2 bar the non-English speech files. The test was such that the AAC coded files were played first but the subjects were not told of the order of the files. The scale used to compare the files was a five point scale, which is a modification of the seven point large impairment scale used to compare coders by the ITU [PS00]. The reason for the modification was one of practicality rather than test performance. Table 4.5 lists the scores and the corresponding relative quality between the two files. According to Table 4.5, a score of 3 would mean that the two sounds sounded the same, scores above 3 mean that the sinusoidal coder performed better than the AAC coder whilst scores below three mean that the opposite is true.

The group of test subjects was made up of 25 participants who were mostly in their early twenties and mostly male (as the tests were carried out at the Electrical Engineering school the choice of subjects is rather limited in terms of gender and age). The over-all mean score was found to be 2.51, that is the AAC coder slightly outperformed the sinusoidal coder across all rates and files. A break up of this mean according to each of the twelve files is shown by Figure 4.35. Figure 4.35 shows that the harmonic files (x2,x3,x4 and x6) are better coded subjectively by the scalable

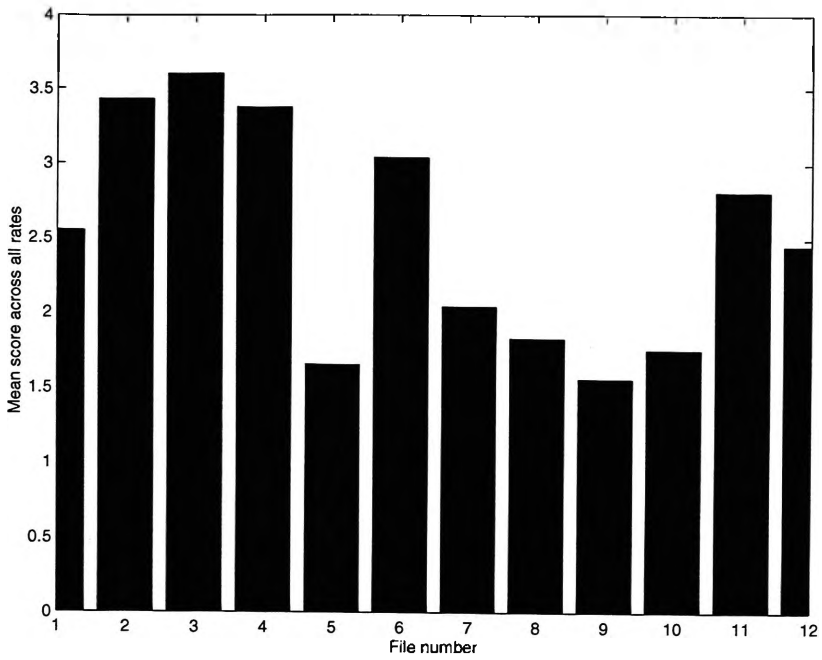

Figure 4.35: The mean score for each file across all the rates

sinusoidal coder, all the other files are better coded by the AAC coder.

A finer division of the results is shown in Figure 4.36, where the mean scores of the files at each rate are given, the bars in the figure are organized such that the leftmost bar represents x1 in each grouping of bars. It is clear from this break up of the results that the harmonic files are coded well by the sinusoidal coder for all the tested rates. The performance of the sinusoidal coder as compared with the AAC varies for the other files, however one can see a trend that the scalable sinusoidal coder performs better compared to the AAC coder for rates 16 and 32 kbps than it does for the 42 and 64 kbps. This is with the exception of file x1, for which the sinusoidal coders performance clearly improves as the rate increases.

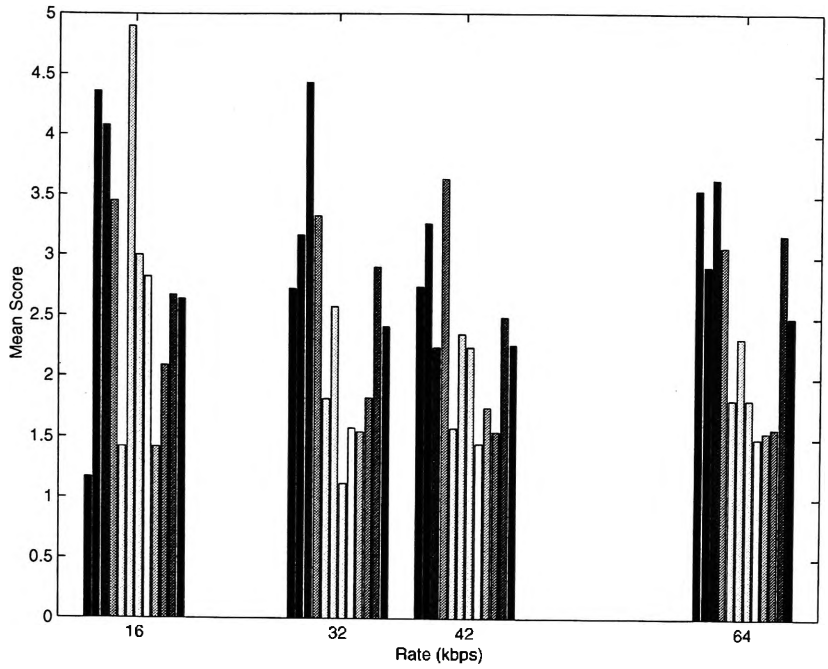

Figure 4.36: The mean score for each file at each rate

Discussion of the subjective results

The testing methodology used to compare the subjective performance of the AAC coder to the scalable sinusoidal coder can at best be described as informal. However, with the number of subjects used one can confidently state that the results do show consistency which is important in this type of test. The scale used deprived the subjects of the option "slightly" better or worse which has probably affected the results as indicated by the average score of 2.51 which lies almost exactly between scores "same" and "better". The performance of the scalable sinusoidal coder when dealing with strongly harmonic signals is clearly an advantage to this approach to scalable coding. The performance of the scalable sinusoidal coder at low rates is also encouraging.

The results presented here must be kept in perspective, the scalable sinusoidal

coder is fundamentally a single paradigm scheme whereas the AAC coder is a multi-paradigm scheme where the quality of the coded signal is contributed to by a combination of signal modelling, noise shaping and transform coding. When this is taken into consideration, the relative performance of the scalable sinusoidal scheme is definitely encouraging.

4.5 Discussion and conclusion

This chapter has presented the design of two perceptual sinusoidal coders. A variable rate coder that operated at an average rate of 38.9 kbps and a scalable rate coder. Both coders rely on the sorting of sinusoidal amplitudes to achieve high compression. It is the sorting of the perceptually relevant amplitudes that allows the changing of the variable rate coder to a scalable coder.

The scalable coder has been analyzed in more detail than the variable rate coder with subjective results suggesting that the scalable coder performs reasonably well when compared to the MPEG-4 AAC coder at the same rates. The scalable coder has been analyzed using psychoacoustic measures, as was the AAC coder. It was observed that the scalable sinusoidal coder contributes a decreasing distortion to the psychoacoustic measures used at increasing rates. This is not the case for the AAC coder. It was also noted that the sinusoidal coder's distortion tends to be of a consistent nature as opposed to the fluctuating nature of the AAC distortion. It can also be concluded that one of the major differences between the scalable sinusoidal coder and the AAC coder is the application of noise modelling and shaping in AAC. This is supported by subjective listening test results that have shown that the scalable sinusoidal coder performs well in comparison with the AAC for harmonic signals but

performs poorly for noise like signals.

Overall, the results for the scalable sinusoidal scheme have been encouraging. This scheme does have the advantage of using one paradigm across many rates. However, the scheme presented needs to have pre-defined information for the intended rates. For example, one would need to pre-define the interpolation grid for the amplitudes at each rate. Also, this scheme does not allow partial transmission of quantized parameters instead either whole parameters need to be transmitted. Finally, although the sinusoidal model is efficient in terms of producing a perceptually similar signal, it is difficult to extend the scalability towards lossless compression without paying a price at the lower rates. The exact reproduction of the original signal is of interest in this thesis.

Because of the mentioned drawbacks, this scheme is not developed any further, instead the concept of parameter perceptual sorting is taken to a new level by the adoption of the Set Partitioning in Hierarchical Trees (SPIHT) algorithm in coding schemes that are developed and presented in the following chapters.

Chapter 5

Set Partitioning In Hierarchical Trees (SPIHT) For Audio Coding

The work described in this chapter utilizes the coding and transmission algorithm known as the Set Partitioning In Hierarchical Trees (SPIHT) algorithm [SP96]. This is a popular image coding algorithm because it provides a significant improvement over the zero tree coding technique that was proposed by Shapiro [Sha93]. This chapter presents a study of the application of SPIHT to audio, it introduces a new method of defining the coefficient sets and lists that are important to the successful operation of the algorithm and it develops a perceptual scalable coding scheme that is based on SPIHT. A modification is also made to the original SPIHT algorithm to allow it to better cope with the unfamiliar situation of insignificant coefficients (due to masking) at frequencies of significant energy.

The contributions of this thesis addressed in this chapter include the proposal of a fine grain scalable perceptual compression scheme, a comprehensive study into the use and application of SPIHT to audio compression and the introduction of a perceptually inspired modification to SPIHT. These contributions are described in [RMB02c] and [RMB02b].

5.1 Introduction

The SPIHT algorithm was initially proposed as an image compression solution in [SP96] but it is general enough to have been applied to audio and electrocardiogram (ECG) signal compression, in combination with the Wavelet transform, as well [LP98][LKP00]. The SPIHT algorithm aims at performing an ordered bit plane transmission and sorts transform coefficients in an efficient manner allowing more bits to be spent on coefficients that more heavily contribute to the energy of the signal. Efficient sorting is achieved by continuously updating a predefined order of coefficients. The attractiveness of SPIHT lies in the combination of this implicit sorting with the partial bit-plane transmission.

This chapter first introduces SPIHT and describes how it can be applied to audio. This is followed by a study into the combination of SPIHT with a number of transforms, specifically the Discrete Wavelet Transform (DWT)(which is the basis of the scheme described in [LP98]), the Discrete Cosine Transform (DCT), the sinusoidal model (based on the Short Time Fourier Transform) and finally the Modulated Lapped Transform (MLT). It is shown that significant compression ratio differences are achieved depending on the transform that is used. To this end, the results presented will show that the MLT achieves the highest compression ratio. The degree of compression obtained is enhanced by the use of a masking model. Further improvements are obtained by using a modified version of the SPIHT algorithm proposed in this chapter. The modification tests for absolutely insignificant coefficients (that is zeros or coefficients below a given threshold) and removes those coefficients from the sorting and transmission process completely.

The chapter is organized such that Section 5.2 introduces SPIHT in more detail

and presents results of combining SPIHT with a number of popular transforms. Section 5.3 discusses how the degree of compression may be improved by the use of a masking model and by modifying SPIHT. A discussion of the presented results and conclusions are presented in Section 5.4.

5.2 A study of SPIHT with current audio coding transforms

5.2.1 Set Partitioning In Hierarchical Trees

The Set Partitioning In Hierarchical Trees algorithm (SPIHT) was introduced by Said and Pearlman in [SP96]. The complete algorithm is listed in [SP96] as *Algorithm II* and is presented in this chapter in Section 5.3.2 with an added modification. In this section, we consider the operation of the algorithm and its characteristics.

The algorithm is built on the principle that spectral components with more energy content should be transmitted before other components; thus the most relevant information is sent first. This is the same principle that has been applied already in this thesis in Chapter 4. SPIHT sorts the available transform coefficients and transmits both the sorted coefficients and sorting information. The algorithm is provided with an expected order of the coefficients defined in the form of trees; those coefficients closer to the roots of the trees are expected to be more significant than those at the leaves. The transmitted sorting information is used to modify this pre-defined order. The algorithm tests available coefficients and sets of coefficients (where a set of

coefficients refers to a group of related or linked coefficients)to determine if those coefficients are above a given threshold (which decreases in a base 2 logarithmic fashion). The result of the test is transmitted and the coefficients are deemed significant or insignificant relative to the current threshold. Significant coefficients are transmitted bit plane by bit plane (a B bit quantized coefficient is said to contain B bit planes).

As SPIHT includes the sorting information as part of the partial transmission of the coefficients, it produces an embedded bit stream that may be truncated at any point. This characteristic of SPIHT makes it very useful for scalable coding, as an increase in the bandwidth available will lead to an improvement in the quality of the reconstructed signal. The results presented in [SP96] stress the scalability of the algorithm. In the context of audio compression, it has been seen from the literature reviewed in Chapter 3 that currently no audio algorithm defines an enhancement layer down to one bit per frame. Because of the bit plane transmission process of SPIHT, it is possible to define the enhancement layer down to a single bit. The perceptual significance of that bit depends on the perceptual significance of the coefficient that it is apart of and how close it is to the most significant bit of that coefficient.

The pre-defined order of the coefficients is important to the performance of the algorithm; this will be demonstrated in the later sections of this chapter. In [SP96] the SPIHT algorithm used a pre-defined order that linked sub-band coefficients together in trees (with each tree being made up of a number of sets). The trees follow the natural sub-band progression of a dyadic wavelet transform having the lower frequencies located at the base of the trees [SP96]. In the audio coding related work reported in [LP98] and [LKP00], the wavelet transform was used, and so a similar way tree structure was used.as in [SP96]. The defined trees group the low frequency components

in small sets whereas the high frequency components are grouped in large sets. The reasoning being that low frequency components tend to contain more energy than the high frequency components, and so by grouping coefficients that are expected to be insignificant in large groups early, the sorting can be concentrated on a smaller set of coefficients.

In the following sub-sections the application of SPIHT to audio compression will be demonstrated by combining the SPIHT algorithm with a number of popular audio compression transforms. The aim is the determination of the transform that should be combined with SPIHT for audio compression as the work presented in [LP98] simply assumed that the wavelet transform was best suited for the operation of SPIHT. This assumption was based on the sub-band coding environment for which SPIHT was originally developed [SP96]. It will be seen, however, that this choice is not necessarily the best one.

5.2.2 Audio compression using wavelets and SPIHT

The compression scheme presented in this sub-section is based on that developed by Lu and Pearlman in [LP98]. The attractive property of the wavelet transform is the fact that the transform is implemented in a sub-band manner and so the tree structure defined in [SP96] could still be used. The filter pairs used in [LP98] are the 20-length Daubechies filter pairs.

The tree structure required by SPIHT is illustrated in Figure 5.1, which is similar to that presented in [LKP00]. The numbers in the boxes of Figure 5.1 indicate the wavelet coefficient number starting at 1. Figure 5.2 illustrates how the coefficients are related in the time frequency plane. The arrows linking the coefficients indicate the

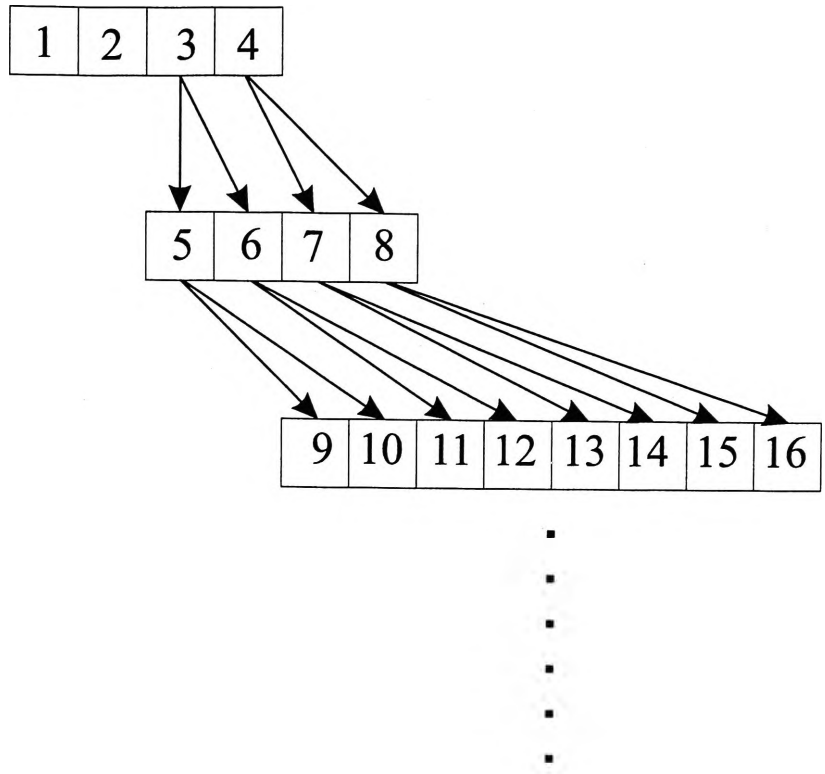

Figure 5.1: Constructing the lists

offspring of each coefficient. Coefficients without arrows emerging from them have no offspring. The set of descendants of each coefficient is obtained by linking all of the offspring obtained at every level together. The linking of the offspring for each tree node leads eventually to the definition of four sets used by SPIHT; the roots set (which in the case illustrated in Figure 5.1 would contain $\{1, 2, 3, 4\}$), the offspring set (which in the illustrated case would contain $\{5, 6\}$ for node 3), the complete descendants set (linking all of the offspring of a node together and would be $\{5, 6, 9, 10, 11, 12, ...\}$ for node 3 in this case) as well as the non-direct descendants (i.e. the descendants that are not classified as offspring). The manner in which SPIHT utilizes these sets will become clearer when the complete modified SPIHT algorithm is presented in sub-section 5.3.2.

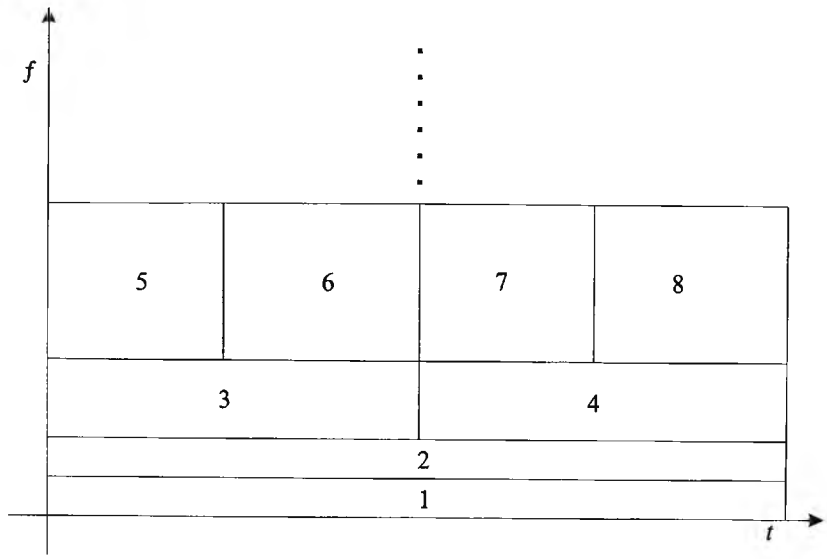

Figure 5.2: Constructing the lists in the time-frequency plane

The audio coding scheme based on the wavelet transform is diagramatically represented by Figure 5.3. In the scheme shown, the psycho-acoustic model determines the bit allocation that should be used in the quantization of the wavelet coefficients. This requires side information to be transmitted. SPIHT also needs to be modified to handle the different thresholds as each set of coefficients would be quantized at a different resolution. In the Lu/Pearlman implementation this scheme was used for the coefficients in the frequency band 0-3.4 kHz which corresponds approximately to the first 17 critical bands of the hearing spectrum [LP98]. The remaining higher frequencies were transmitted by the use of a "Reverse sorting and coding scheme" [LP98], the details of which will not be described here. The results presented by Lu and Pearlman indicated that imperceptible distortion in the synthesized signal could be obtained at bit rates between 55-66 kbps, which is competitive with current standardized coders.

The frequency band division of the spectrum was chosen as in [LP98]. This division

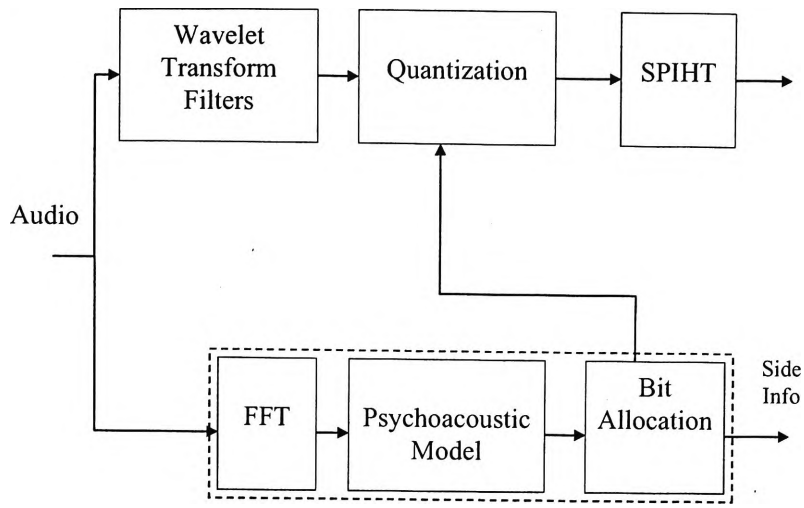

Figure 5.3: The wavelet based coding scheme

uses 29 critical bands instead of the usual 25 [Moo89][Nol97]. The MPEG model 1 (which has been described in Chapter 2) of allocating the number of bits per band was used to distribute the total number of bits in a way that ensures no more than 16 bits are allocated to a single coefficient in any band. The frame size used is 1024 samples, which corresponds to 23.2 ms when the sampling rate is 44.1 kHz.

As an indication of how SPIHT reduces the number of transmission bits required, Table 5.1 lists complete reconstruction results, where the distortion is solely due to (perceptual) quantization noise in the frequency domain (which is spread across all samples in time), for the sixteen test signals used in this work. The test signals are the same Sound Quality Assessment Material (SQAM) signals obtained from the MPEG web site [mpe] and used to analyze the performance of the scalable sinusoidal coder presented in Chapter 4. The results given are in terms of average bit rates

Table 5.1: Coding Results using the Wavelet Transform.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	46.1	167	x9	35.8	200
x2	50.9	71	x10	32.5	227
x3	46.6	180	x11	33.6	204
x4	44.4	201	x12	37.1	190
x5	31.1	227	x13	34.1	202
x6	48.0	94	x14	34	204
x7	43.2	174	x15	50.7	175
x8	43.7	162	x16	42.8	187

per frame and should be compared to 706 kbps which is the CD rate and means 16 bits per coefficient. The sound quality of the audio reconstructed with this wavelet based coder operating with the complete set of quantized coefficients is perceptually indistinguishable from the original. Some objective results are given in Table 5.1 in terms of the Segmental Signal to Noise Ratio (SegSNR).

The usefulness of the MPEG masking model becomes clearer at lower rates. This is illustrated in Figure 5.4. The figure shows the SegSNR results, as well as the mean bits used per frame for uniform quantization of the wavelet coefficients at 8, 9, and 10 bits as well as for uniform quantization using the bit distribution of the MPEG model. It can be clearly seen that far better SegSNR results may be obtained when using the 10 bit quantization without the use of the MPEG model for a lower number of bits. This means that the application of the MPEG bit allocation process is rather costly when one intends to compress the audio at a relatively high rate with the use of the complete set of coefficients. However, this result cannot be generalized to the lower rates as the MPEG bit distribution does result in better sounding audio synthesized from an incomplete set of coefficients. This means that the use of the MPEG bit

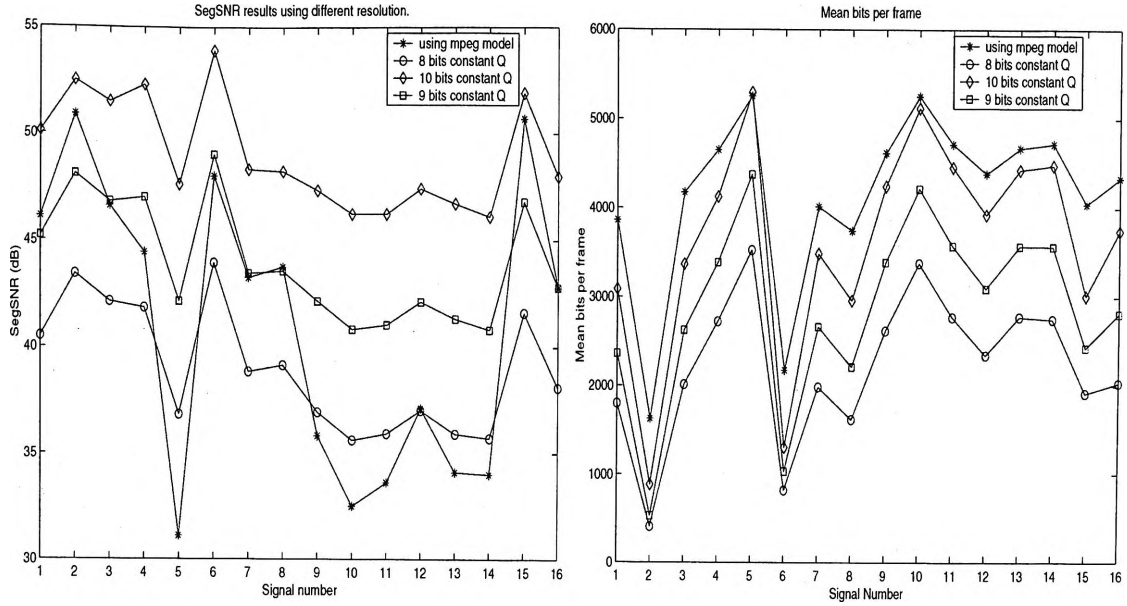

Figure 5.4: The MPEG model based perceptual quantization compared with straight scalar quantization

distribution should be weighted against bandwidth available and the computational cost of the scheme.

The significance of the results presented in Figure 5.4 will become more obvious in Chapter 6 where the dynamic range of the transform coefficients will be shown to play an important role in determining the lossless compression obtainable by the use of SPIHT.

The SQAM files were also coded at 16, 32, 42, 64 and 128 kbps, to give a demonstration of the scalability of the SPIHT algorithm. The resulting synthesized audio had almost no perceived distortion between 42 and 64 kbps which roughly coincides with the results presented by Lu and Pearlman [LP98]. The quality observed at different bit rates relied heavily on the nature of the sound being coded. As in the case of the sinusoidal coder, sounds that were narrow bandpass type signals (i.e closer to being tonal) were coded very efficiently with very few bits. This is because SPIHT

labels large sets that are insignificant as a single entity and tests for the whole sets significance. Thus, narrow bandpass type signals will have their significant coefficients located after only a few tests and refinement bits will be transmitted for the significant coefficients instead of the testing of insignificant coefficients. As pointed out in [SP96], this is the main difference between the SPIHT algorithm and the zero tree coding technique which tests each coefficient individually for significance.

Having described experimentally the use of SPIHT with the wavelet transform (i.e. in a sub-band audio compression scheme), a study of SPIHT in M-band transform based schemes would be a significant contribution to the understanding of how this set sorting algorithm should be applied to audio compression. This will ultimately determine the transform that should be combined with SPIHT for audio compression.

5.2.3 The M-Band Transform Based Codec

The general codec based on the combination of uniform short-time transforms and SPIHT is shown in Figure 5.5. In Figure 5.5, the audio signal is divided into overlapping frames and a transform is applied to each frame. The obtained coefficients are quantized and transmitted by the use of SPIHT. At the decoder, SPIHT is used to decode the bit stream received and the inverse transform is used to obtain the synthesized audio. The Figure does not explicitly show an overlap/add procedure, although one may be used (depending on the transform applied).

In applying SPIHT to M-band transform coding, the trees that were used for the wavelet based coding scheme no longer describe the relationship between the transform coefficients as they did in the wavelet based implementation. Nonetheless, tree-structured sub-band based trees were used for a uniform M-band based transform

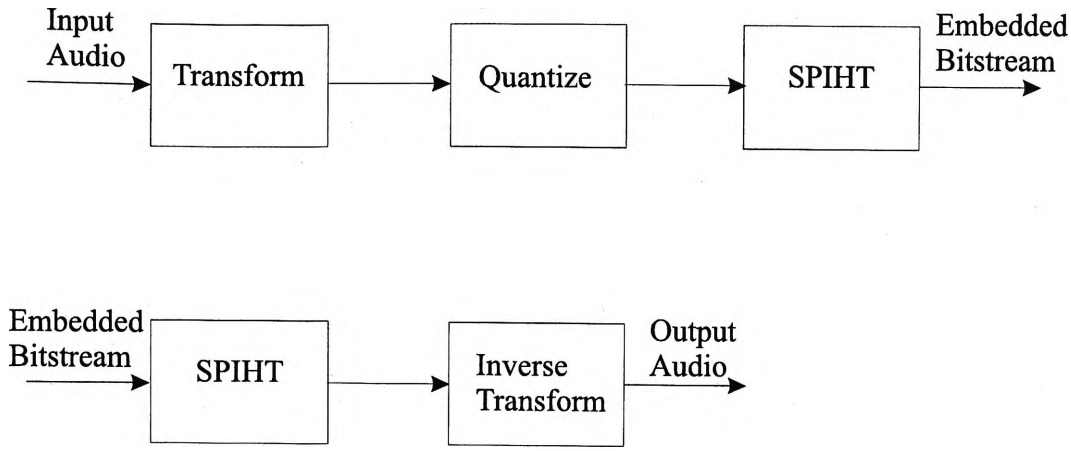

Figure 5.5: The general codec used for the M-Band Transform experiments

image compression in [TN98] with very good results. This indicates that as long as the trees define large sets of insignificant coefficients and small sets of significant coefficients, SPIHT will not use an excessive number of bits to carry out the sorting. The reason behind this result is that SPIHT sorts large sets as a starting point. The sorting then concentrates on smaller sets within those large sets. In order to reduce the cost of the sorting process as many insignificant coefficients as possible should be placed in one set, allowing all of those coefficients to be labelled insignificant with one test.

In the following new SPIHT sets that link together the frequency domain coefficients for a given frame are defined. At this point it should be realized that the sets may be defined in an arbitrary manner, i.e. there is no absolute need to define sets according to spectral content expectation for the correct operation of SPIHT. However, the compression provided by SPIHT is related to the grouping of insignificant coefficients into large sets early. In this way at each iteration of the algorithm only one bit need be used to determine the insignificance of such sets (see the algorithm

listed in sub-section 5.3.2). For example, if a set of 32 coefficients (say) had no significant coefficients (at the given SPIHT iteration) in it then not one bit of the binary representation of those 32 coefficients would be transmitted, instead a single 0 would be transmitted as the test result and all 32 coefficients ignored. On the other hand, if only a few of the 32 coefficients were significant, SPIHT would have to decompose the set into smaller sets according to the initial set definition provided and test each to determine where the significant coefficients are. It must also be remembered that SPIHT tests sets that are closer to the roots first. Knowing these facts about the operation of SPIHT, and the desire to obtain the best possible performance at the lowest rates, leads to the definition of sets in a tree structure whereby the roots contain more significant coefficients than the leaves. In our case the interest is in audio, which tends to have more significant energy at the lower end of the spectrum (partly because of the human hearing mechanism) and so it is logical to define sets that have their roots close to the low end of the audio spectrum and leaves that are at the high end of the audio spectrum. Thus, the sets we define link together coefficients in the frequency domain in an order that fits the expectation that the lower frequency coefficients should contain more energy than the higher frequency coefficients. This ordering is similar to, although not the same as, the sets defined in [SP96].

In this implementation the sets are developed by assuming that there are N roots. One of the roots is the DC-coefficient and because it is not related to any of the other coefficients in terms of multiples of frequency, it is not given any offspring. Each of the remaining $N-1$ roots are assigned N offspring. In the next step each of the offspring is assigned N offspring and so on, until the number of the available coefficients is exhausted. The offspring of any node (i) where (i) varies between 1 and $M-1$ (M

is the total number of coefficients and $i = 0$ is the DC coefficient), are defined as:

$$O(i) = iN + \{0, N-1\}. \tag{5.2.1}$$

where $\{0, N-1\}$ is the inclusive set of integers between 0 and $N-1$. Any offspring above $M-1$ are ignored. The descendants of the roots are obtained by linking the offspring together. For example, if $N = 4$, node number 1 will have offspring $\{4, 5, 6, 7\}$, node 4 will have offspring $\{16, 17, 18, 19\}$ and the descendants of node 1 will include $\{4, 5, 6, 7, 16, 17, 18, 19, ...\}$.

As part of the development of the M-band transform plus SPIHT coding system, a number of experiments were conducted to determine if the size of N affects the performance of the coder. Figure 5.6 shows the results of some of these experiments. Figure 5.6 indicates that the use of $N = 4$ is better than or equivalent to the use of any other value. This result was found to be consistent over the entire set of SQAM audio files used. Based on this result N was set to 4 for the remaining coding experiments, the results of which are presented in sub-section 5.2.4.

One may also analytically determine that, for the previously defined tree structure, $N = 4$ is a near optimal choice. To show this, assume that in a given group of M coefficients there is only one that is significant at a given bit plane. Also assume that this coefficient is in the descendant set of root j and let r be the number of iterations of SPIHT at the given bit plane. Now, for the given offspring definition, the size of the descendant set D after p offspring stages is given by:

$$size(D) = \frac{N(N^p - 1)}{N - 1} \tag{5.2.2}$$

Also, with $(N-1)$ roots, the total number of coefficients covered by all of the descendant sets in this case is $N(N^p - 1)$. Now, referring to the SPIHT algorithm

in Section 5.3 (and ignoring the modification) or to the original SPIHT algorithm in [SP96], if the algorithm was to search for a single significant coefficient then each iteration at the given bit plane would add $(2N + 1)$ bits to the cost of the search. There are also N coefficients added to the list of insignificant coefficients (LIP) which means a future cost as the algorithm continuously tests LIP coefficients. However, as N also determines the size of the descendant set covered in the search, the actual cost of the search in a given bit plane is given by :

$$R = \sum_{r=1}^{p} (2N + 1) \tag{5.2.3}$$

Table 5.2 lists the calculated cost for varying N to cover sets $M = 256$ and 512. It can be seen that $N = 4$ is the best choice for $M = 256$, however this result does not hold for $M = 512$ and because of Equation (5.2.3), as M increases the choice of $N = 2$ becomes more attractive. In our case, though, audio is being coded with a frame length of 20 ms at a sampling rate of 44.1 kHz. This means that if the MLT was used then the first 256 coefficients would cover more than 11 kHz of bandwidth. As it is highly unlikely that significant coefficients would be found at higher frequencies (especially early in the coding process), $N = 4$ appears as the best choice for the application at hand. This analysis also sheds some light on why the curves of Figure 5.6 have the illustrated shape. It must also be pointed out that the foregoing analysis is significantly simplified by not accounting for multiple significant coefficients and their relative locations in frequency. Consider, for example, 2 significant coefficients located in different descendant sets at similar positions in each set. In this case approximately twice as many bits would be required to locate both coefficients as would be for a single coefficient in the same position. The effect of the relative

Table 5.2: R as a function of N and M

N	$M = 256$			$M = 512$		
	p	LIP growth	R	p	LIP growth	R
2	7	14	35	8	16	40
4	3	12	27	4	16	36
8	2	16	34	2	16	34
16	1	16	33	2	32	66

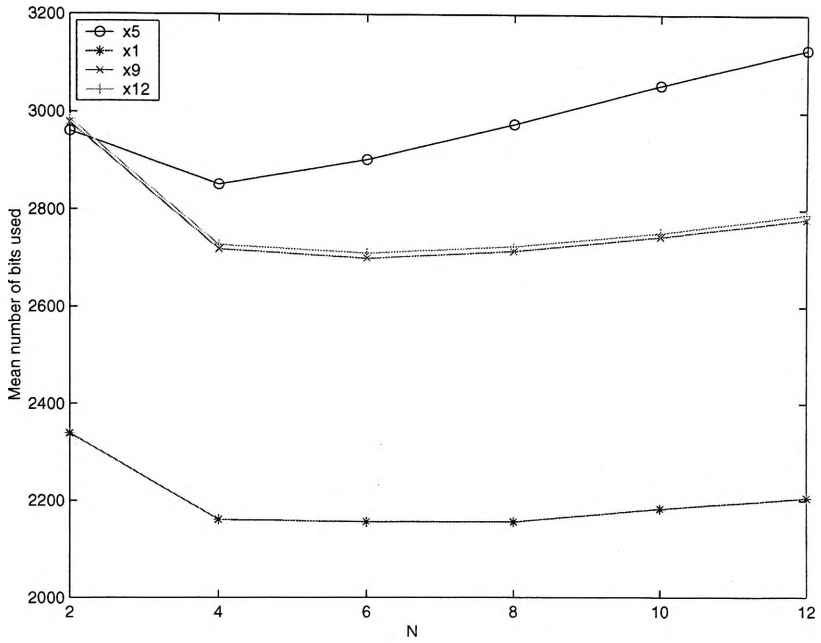

Figure 5.6: The mean number of bits required as functions of N

positions of significant coefficients can only be completely appreciated if the signal statistics are taken into account in the definition of the sets. This could mean different sets for different frames which would in turn mean overhead or side information. In this case we avoid the use of such an approach.

Table 5.3: Coding Results using the Discrete Cosine Transform.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	52.9	216	x9	50.2	271
x2	62.4	**47**	x10	48.2	331
x3	47.5	**86**	x11	45.2	267
x4	52.0	**167**	x12	52.2	273
x5	43.1	285	x13	47.9	260
x6	59.4	99	x14	43.1	273
x7	52.0	275	x15	48.9	**150**
x8	51.1	196	x16	44.9	**183**

5.2.4 M-Band Transform Coding Results

Using the DCT with SPIHT

The Discrete Cosine Transform of Type II has been defined in Chapter 2, and it is this definition of the DCT that is used in this experiment. In combining the DCT with SPIHT as illustrated in Figure 5.5, the selected frame, in this case of length 880 samples (approximately 20 ms), is windowed using a perfect reconstruction window before applying the transform. The window used is the PR sine window that has been described earlier in this thesis.

Using the lists developed with the system built around SPIHT and the DCT, complete reconstruction results were obtained, without the use of masking and are given in Table 5.3. An analysis of the results presented can be found in Section 5.2.5. The results listed in bold font indicate the files for which the DCT based codec out-performed the Wavelet based one in terms of signal compression.

Combining Sinusoidal coding with SPIHT

As the sinusoidal model is a popular audio coding model, it is important to include it in this comparative study of transforms combined with SPIHT. Although the sinusoidal model is not most accurately described as a transform, when the STFT method is used to derive it, then the parameters of the sinusoidal model (amplitudes, phases and frequencies) become another manifestation of the STFT coefficients. Here, it is assumed that unlike the traditional sinusoidal model [MQ95], the amplitudes are calculated directly from the STFT and not from an envelope approximation of the spectral amplitudes. To allow the STFT of the signal to be performed, the signal is divided into frames and each frame is windowed. In this particular implementation, the frame length is kept constant and rather than using the Hamming or Hanning windows (as is usually the case for sinusoidal coders [MQ95, GS92]), the same perfect reconstruction window used in the DCT case is also used with the sinusoidal model.

The sinusoidal model presented here requires one amplitude and one phase parameter for each sample in the time domain. Experimental work carried out and reported in [RB01] also suggests that the phase is relatively important. In [RB01] and Chapter 4 it was stated that the importance of the phase is related to the significance of the amplitude and so phase parameters that are related to small amplitudes may be quantized using coarser resolution than other phase parameters.

To combine the Sinusoidal model with SPIHT only the amplitudes of the model are transmitted partially. That is, the phase is transmitted as a scalar quantity in place of the sign bit that would be transmitted for other transforms such as Wavelet, DCT and MLT. This means that the partial transmission, prominent in SPIHT, is not used to full advantage as there are a minimum number of bits that must be transmitted, i.e.

Table 5.4: Coding Results using the Sinusoidal model.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	27.5	**111**	x9	27.7	216
x2	29.9	**24**	x10	27.1	278
x3	26.0	**65**	x11	26.1	226
x4	26.6	**90**	x12	27.9	196
x5	27.5	**222**	x13	26.8	252
x6	26.8	**50**	x14	25.7	229
x7	26.3	**128**	x15	26.5	**105**
x8	26.4	**96**	x16	27.6	**133**

the phase bits. In this implementation, 8 bits were used to represent each amplitude and 5 bits to represent each phase component, in a similar bit distribution to that used in the scalable sinusoidal coder presented in Chapter 4.

Table 5.4 lists the results of combining SPIHT with the sinusoidal model. As was the case for the DCT, the results listed are in terms of the mean bit rate required per frame for complete reconstruction, i.e. all of the parameters are coded and transmitted. As was the case in Table 5.3, Table 5.4 lists the results where the sinusoidal based codec out-performed the Wavelet based codec in terms of mean bit rates in bold font.

The Modulated Lapped Transform (MLT) combined with SPIHT

The MLT is one form of the Lapped Orthogonal Transform. In traditional block transform theory, a signal $x(n)$ is divided into blocks of length M and is transformed by the use of an orthogonal matrix of order M. On the other hand, lapped transforms take a block of length L and transform that block into M coefficients, with the condition that $L > M$ [Mal92]. In order to perform this operation there must be an

Table 5.5: Coding Results using the MLT.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	55.5	**145**	x9	52.9	**177**
x2	64.2	**31**	x10	51.5	**216**
x3	49.4	**60**	x11	50.1	**173**
x4	54.1	**110**	x12	54.9	**179**
x5	45.8	**183**	x13	50.4	**170**
x6	61.1	**68**	x14	49.4	**174**
x7	55.5	180	x15	51.7	**101**
x8	54.2	**140**	x16	47.2	**131**

overlap between consecutive blocks of $L - M$ samples [Mal92]. This means that the synthesized signal must be obtained by the use of consecutive blocks of transformed coefficients. The MLT produces one coefficient per synthesized audio sample and this has been achieved by forcing the samples synthesized to be constructed from contributions of consecutive sets of coefficients. The MLT has been described in some detail in Chapter 2.

Table 5.5 shows the obtained results for the MLT based scheme. The coding was performed using 17 bit quantized MLT coefficients (16 for the amplitude and 1 for the sign). The bold font results indicate a better performance than the Wavelet based scheme.

5.2.5 Discussion of the M-Band Transform coding results

A quick scan of Tables 5.1, 5.3, 5.4 and 5.5 shows that the lowest mean rates are achieved when the MLT is used in combination with SPIHT, this despite the fact that for most of the coefficients of the MLT the quantization resolution is greater than that used for the DWT and the sinusoidal model. The results presented thus far

are slightly biased towards the Wavelet based scheme because of the perceptual model used in the Wavelet codec. Yet even with such bias the MLT codec out-performs the Wavelet codec for the majority of the SQAM files used.

Comparing the results presented for the DCT and the Sinusoidal model codecs, the Sinusoidal model appears to have a greater degree of compression than the DCT. This degree of compression is still less than that achieved by the use of the Wavelet transform. The advantage of the Wavelet transform in this instance is that it requires one transform coefficient for each synthesized audio sample. In contrast, the DCT in this implementation requires two coefficients for a single synthesized audio sample. This is not necessarily the implementation that one would employ. For instance, a different window may be used in combination with the DCT, one that requires less overlap for PR. However, such windows may not have the frequency selectivity desirable which would result in poor synthesized audio quality when only some of the coefficients are used. Regarding the sinusoidal model, the major disadvantage encountered is the transmission of the phase parameters as scalar quantities without taking advantage of the partial transmission available with SPIHT. Despite this, the sinusoidal parameters require a lower mean rate for most files than the DWT based codec. It is interesting to note that the speech files had the worst performance when the sinusoidal model was used. This is primarily because complete reconstruction of the quantized parameters is sought in this case resulting in the transmission of all of the phase parameters.

In the following section two methods for improving the performance of the M-Band based schemes are presented. The first method incorporates masking into the coding scheme and the second modifies SPIHT in order to improve its sorting efficiency.

5.3 Improving the compression ratio of the M-Band SPIHT coder

5.3.1 The use of masking

The perceptual model and perceptual redundancy

Human hearing is based on the ear receiving sound waves and the brain interpreting those waves as sounds, as has been described in Chapter 2. The two well known techniques for determining the masked and masking components in the frequency domain are the *Johnston model* (first proposed by Johnston in [Joh88a]), and the *MPEG model 1* (as described in [PS00]). Both methods lead to the development of a masking curve for the entire spectrum of an audio signal. The major difference between the two is that the Johnston model specifies a masking value per critical band [Joh88a] whereas the MPEG model 1 specifies a masking value for each frequency bin used to describe the signal in the frequency domain (assuming that there are more frequency bins than critical bands), as has been shown through example in Chapter 2.

In the Johnston model, the simultaneous masking curve is based on the Discrete Fourier Transform (DFT). The DFT coefficients are used to calculate the power in each of 25 critical bands, as described in [Joh88a], [PS00] and [Moo89]. The calculated power in each band is modified by the use of a spreading function to include power that spreads from surrounding bands [Joh88a]. This spread power is offset by a value that is related to the content of the signal segment through a Spectral Flatness Measure (SFM) which indicates numerically how noise-like or tone-like is the signal segment.

The spread and offset power values are renormalized and the resulting masking curve is compared to the absolute threshold of hearing [Moo89], which acts as a minimum level for the masking curve. The resultant curve is the final masking curve and any spectral components that fall below this curve are deemed masked.

The MPEG model is also based on the DFT [PS00]. After the windowing, normalization and power spectral density calculation tonal and noise maskers are selected according to the criteria specified in [PS00]. The power of the tonal and noise maskers are calculated and those maskers are decimated and re-arranged at different rates in different critical bands, a detailed description of how this is achieved has been given in Chapter 2. Next, tonal and noise masker thresholds are calculated for each tone and noise masker. This involves adjusting the previously calculated power values to include a spreading function. The spreading function of the tonal maskers differs slightly to that of the noise maskers [PS00]. Finally, a global masking threshold is calculated by using the tonal and noise masker thresholds as well as the absolute threshold of hearing.

The traditional way of using the masking curves has been to provide information on how much noise may be allowed in a given frequency band [PS00][Nol97][Dav99] [Joh88a], or how accurately a given band needs to be quantized for transmission. For this purpose, a calculation of the mask-to-noise ratio in each critical band is carried out and more bits are allocated to the band with the lowest mask to noise ratio. An iterative procedure is employed where bits are assigned according to some distortion criteria [PS00]. This technique is used in the MPEG and Dolby AC transform coders [Nol97][Dav99]. This technique was also used in the wavelet based coding scheme that has been already described in this chapter.

Another way of using the masking curves is to ignore all spectral components below the curve, which is the technique used in Chapter 4 . As was mentioned in Chapter 2, our informal listening tests showed (these tests simply involved reconstructing the perceptually significant signal and comparing it to the original) that if the Johnston technique is used in this manner the audio reconstructed from non-masked components sounds the same as the original audio, which is not the case for the MPEG model 1. The masking curve produced by the MPEG model was found to be too aggressive for this type of use, as the resulting synthesized audio takes on a characteristic similar to low-pass filtering the original audio signal.

The M-band transform codecs presented in this chapter use the Johnston model to reduce the number of coefficients that must be dealt with. The structure of the M-band transform coder is modified as shown in Figure 5.7. The Wavelet based codec makes use of the MPEG model 1 in order to keep the coder's structure close to that described in [LP98]. In the following Sub-sections the results of combining the M-band transform codecs with SPIHT and masking are listed.

The M-band Transforms combined with masking and SPIHT

The combination of the masking scheme with the DCT and MLT simply requires the determination of the signal's power spectral density from those transforms as it is the power distribution in the frequency domain of the signal being processed that determines the frequency components that are masked [Joh88a]. Table 5.6 lists the results obtained when SPIHT is combined with the DCT based scheme with masking utilized. Likewise, Tables 5.7 and 5.8 list, respectively, the results of combining the sinusoidal model and the MLT with masking and SPIHT. The bold font results once

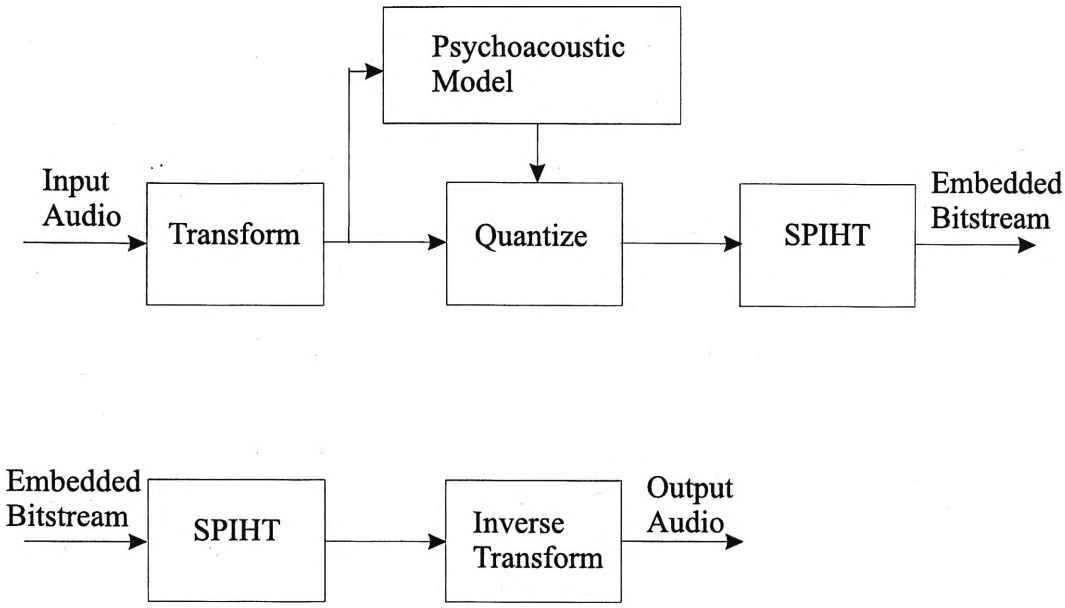

Figure 5.7: The General codec used with masking

again indicate a lower mean rate than the Wavelet based scheme and the underlined results indicate a lower mean rate than the original, relevant, M-band based scheme. A combination of bold font and underlining indicates a lower mean rate than both the Wavelet based scheme and the original M-band scheme.

Table 5.6: Coding Results using the Discrete Cosine Transform with masking.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	20.3	185	x9	22.4	214
x2	23.9	**48**	x10	22.3	242
x3	19.8	**80**	x11	19.7	**203**
x4	26.3	**155**	x12	20.9	220
x5	9.1	**223**	x13	19.9	203
x6	24.5	99	x14	20.6	207
x7	24.5	223	x15	27	**145**
x8	24.0	**162**	x16	14.9	**157**

Table 5.7: Coding Results using the Sinusoidal model with masking.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	19.2	**102**	x9	20.6	**126**
x2	19.1	**50**	x10	19.8	**141**
x3	17.1	**49**	x11	18.2	**126**
x4	21.9	**74**	x12	19.5	**141**
x5	8.9	**149**	x13	18.2	**137**
x6	21.8	**85**	x14	19.1	**131**
x7	22.0	**134**	x15	23.0	**88**
x8	21.6	**131**	x16	14.5	**121**

Table 5.8: Coding Results using the MLT with masking.

Signal	SegSNR (dB)	Mean Rate (kbps)	Signal	SegSNR (dB)	Mean Rate (kbps)
x1	16.7	**53**	x9	16.0	**55**
x2	19.2	**14**	x10	15.8	**62**
x3	17.9	**25**	x11	15.2	**54**
x4	21.8	**47**	x12	14.8	**58**
x5	7.6	**65**	x13	13.1	**54**
x6	23.3	**33**	x14	13.7	**57**
x7	20.1	**65**	x15	21.7	**44**
x8	21.4	**47**	x16	14.1	**42**

Discussion on M-band transform with masking results

The results presented in Section 5.3.1 indicate that compression gains achieved by incorporating the masking with the M-band transform coders vary according to the transform used. An examination of the results leads to the conclusion that the MLT is the most suitable transform to be used in the coding scheme tested. This may be explained by studying the properties of the MLT and SPIHT. The MLT utilizes filters that have high stop band suppression (due to the sine window employed), leading to a reduction in leakage between the bands and hence a better identification of masked and masking components. As such, when the masking model is applied to the MLT based scheme more coefficients are removed. The removal of the masked coefficients leads to more efficient SPIHT coding.

The incorporation of the masking with the sinusoidal model and the DCT also lead to significant compression gains. The disadvantage of the DCT as compared to the MLT is the redundancy in the signal transform representation. The redundancy results from the use of twice as many analysis coefficients as synthesized samples which is an implementation draw back; the solution is the use of the MLT. In contrast, the sinusoidal model does not have this disadvantage, however, the need to transmit the phase as a scalar quantity while using SPIHT is the main handicap of the sinusoidal model in the coding system used.

As the use of the MLT combined with masking produces impressive results, it is appropriate to examine the subjective quality of the MLT-SPIHT coding scheme. The following Section does so.

Table 5.9: Subjective test score guide [Ryd96]

score	*Sound Quality*
1	Very annoying distortion heard
2	Annoying distortion heard
3	Slightly annoying distortion
4	Some perceptible distortion heard, but its not annoying
5	No distortion can be heard

Subjective quality test results

A set of informal listening tests have been conducted to determine the subjective quality of the MLT-SPIHT coding scheme when masking is employed. The tests consisted of all of the SQAM files listed in Table 4.2 and nineteen subjects. The subjects varied in gender and age group. The subjects were asked to listen to the original signal and the synthesized signal and judge the similarity of the two signals by allocating a score between 1 and 5. The subjects were also asked to award the scores according to Table 5.9 [Ryd96].

The test results obtained showed that in 63.5% of all test cases no distortion could be heard, in other words, the score allocated was a 5. Also, in 90.1% of all test cases any distortion heard was judged to be not annoying, that is, the score allocated was either a 4 or a 5. Finally, the overall mean of the scores given for the MLT-SPIHT coding scheme with masking was 4.52. The results of the subjective test indicate that high quality audio is obtained by the combination of the MLT with masking and SPIHT.

5.3.2 Modifying SPIHT

Table 5.7 indicates that two signals required a higher mean rate when the masking was introduced in combination with the sinusoidal model; the signals being x2 and x6, both of which are very tonal in nature. The reason behind this occurrence is the labelling of some parameters insignificant when SPIHT expects those parameters to be significant. SPIHT expects the parameters closer to the roots of the trees to be more significant than those at the leaves. In the frequency domain this translates to the expectation that lower frequencies hold more significant information than higher frequency components. The introduction of the masking creates a representation whereby a number of lower frequency parameters are deemed masked and thus insignificant. This representation in turn leads to a less efficient application of SPIHT. In this Section a modification is introduced into SPIHT to account for such 'unexpected' representations.

In combining the masking model with the M-band transforms, masked coefficients are set to zero. If a masked coefficient is expected to be non-zero (through its position in the SPIHT trees) then SPIHT will test that coefficient a number of times for significance. Since a zero coefficient will never be significant and so will not be transmitted by SPIHT, a number of test bits are wasted on these significance tests. The effect of these wasted bits on the overall bit rate depends on how divergent the transform representation of the signal is from the expected representation.

As a remedy, another test was introduced into SPIHT. The new test determines if a given amplitude is significant enough that it may ultimately be included in the transmitted amplitudes. If the amplitude is significant, the algorithm proceeds as

before, otherwise that particular entry is removed from the list of insignificant amplitudes and is no longer tested for significance. Although this test adds one bit per amplitude to the cost of the algorithm, the savings made by removing insignificant amplitudes from the sorting process are usually greater when the masking model is applied. This saving, however, varies from signal to signal as the properties of the signal vary.

The following is the complete modified SPIHT algorithm. The quantities H,O,D and L are the sets of tree roots, offspring, descendants and non-immediate descendants respectively as given in [SP96]. LIS, LSP and LIP are the lists of insignificant sets, significant points and insignificant points respectively. The test for significance is given by S_n. The modification introduced in Step **2.2.1**) has been highlighted for clarity. The new test $T_n(k)$ determines if (k) is above a set threshold.

Algorithm SPIHT (modified):

1) **Initialization:** output $n = \lfloor log_2(max_i|c_i|)\rfloor$;

set the LSP as an empty list, and add the coordinates $(i) \in H$ to the LIP, and only those with descendants also to the LIS, as type A entries.

2) **Sorting Pass:**

2.1) for each entry (i) in the LIP do:

2.1.1) output $S_n(i)$;

2.1.2) if $S_n(i) = 1$ then move (i) to the LSP and output the sign of c_i;

2.2) for each entry (i) in the LIS do:

2.2.1) if the entry is of type A then

- output $S_n(D(i))$;
- if $S_n(D(i)) = 1$ then
  - * for each $(k) \in O(i)$ do:
    output $\mathbf{T_n(k)}$
    - **if $\mathbf{T_n(k) = 1}$**
      - output $S_n(k)$;
      - if $S_n(k) = 1$ then add (k) to the LSP and output the sign of c_k;
      - if $S_n(k) = 0$ then add (k) to the end of the LIP;
  - * if $L(i) \neq \emptyset$ then move (i) to the end of the LIS as an entry of type B,

and go to Step **2.2.2)**; otherwise, remove entry (i) from the LIS;

2.2.2) if the entry is of type B then

- output $S_n(L(i))$;
- if $S_n(L(i)) = 1$ then
  - add each $(k) \in O(i)$ to the end of the LIS as an entry of type A;
  - remove (i) from the LIS.

3) **Refinement Pass:** for each entry (i) in the LSP, except those included in the last sorting pass (i.e., with same n), output the nth most significant bit of $|c_i|$;

4)**Quantization-Step Update:**decrement n by 1 and go to step **2.**

The effect of this modification is depicted in Figures 5.8, 5.9 and 5.10 for the schemes using the DCT, Sinusoidal model and MLT respectively. All three figures show that both the masking and the modification lead to savings in the number of bits required to compress each of the SQAM files. It is also noteworthy that the savings vary according to the properties of the signal being compressed. The modification does not effect the quality of the synthesized audio signal as coefficients that are not received by the decoder are set to zero by default. Thus, this simple modification significantly improves the compression obtained from SPIHT.

Figure 5.10 is a clear illustration of not only the usefulness of the modification but also of the effectiveness of the MLT. Combining the MLT with simultaneous

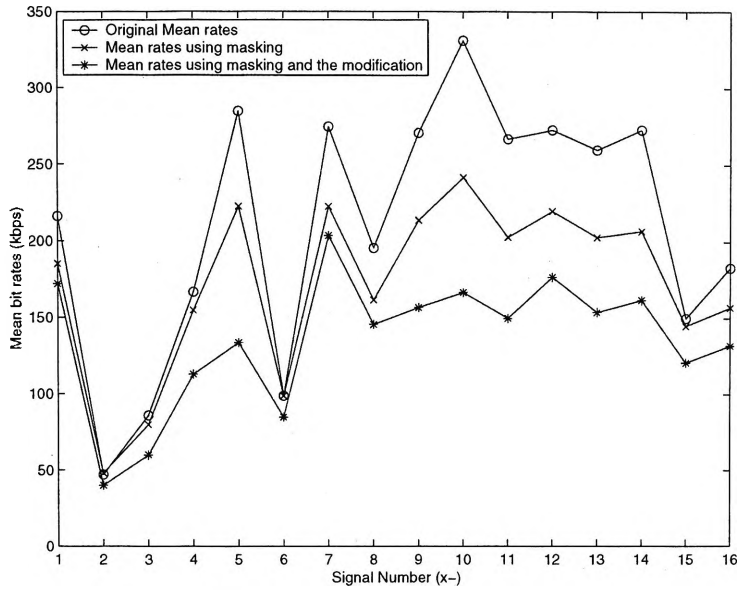

Figure 5.8: Using the DCT

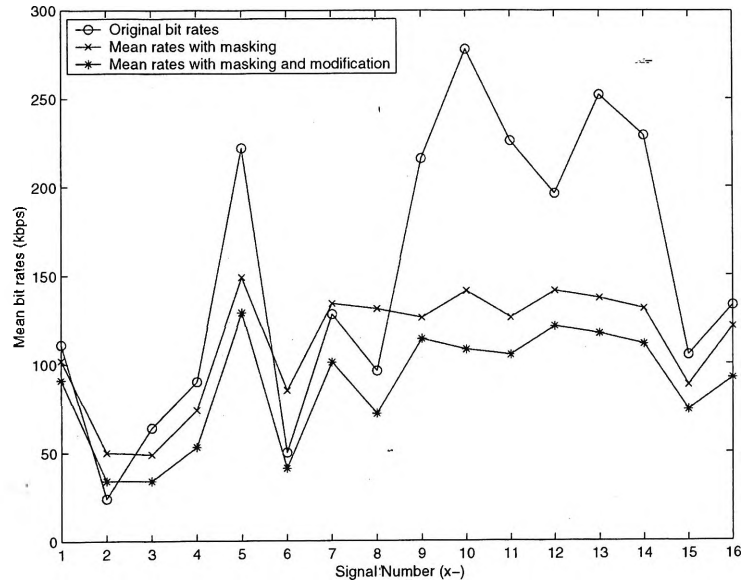

Figure 5.9: Using the Sinusoidal Model

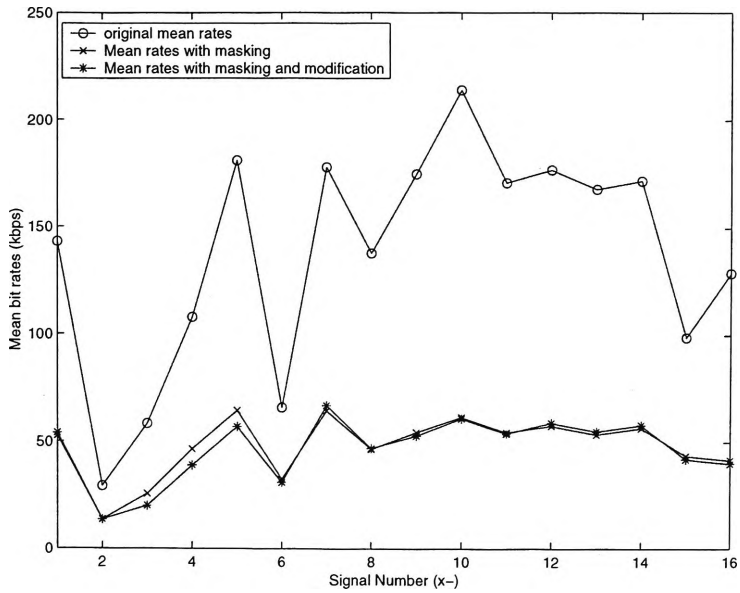

Figure 5.10: Using the MLT

masking leads to savings much more significant than obtained by using any of the other transforms. The main reason for this result is the frequency response of the filters used to obtain the MLT coefficients. The result illustrated in Figure 5.10 has its counterpart in image coding where it was shown that a general Lapped Orthogonal Transform has the potential to outperform the wavelet transform when combined with SPIHT [TN98].

This modification is specifically aimed at schemes that introduce insignificant coefficients at spectral locations where usually significant coefficients are expected and so it would not apply to the scheme based on the DWT, as that particular scheme simply used lower quantization resolution rather than reducing the number of coefficients requiring processing.

Table 5.10: Subjective Test Scores for the 64 kbps and 54 kbps codecs

Results	54 kbps	64 kbps
Overall mean	4.24	4.44
% No distortion heard	47.4	57.2
% No annoying distortion heard	80.9	88.5

5.3.3 Limited bit rate implementation of the MLT-SPIHT based coder

In the preceding sections no limit was placed on the number of bits that may be used to code any frame. Hence, the results tables list only mean bit rates. A more practical scheme is achieved when bit limits are imposed. The use of SPIHT makes this quite simple in that the first so many bits of the produced bit stream are used and the rest are discarded.

Using the MLT-SPIHT based coder with masking and the modification described in Section 5.3.2 a 54 kbps codec and a 64 kbps codec were produced. Both of these coders were tested using the same methodology described in Section 5.3.1, i.e. the synthesized audio at those rates was compared to the original audio and an indication of the distortion present determined. Table 5.10 lists the results of those subjective tests.

The results listed in Table 5.10 show that very good quality audio may be obtained by using the MLT-SPIHT based coder with masking at rates between 54 and 64 kbps. More than 80% of all test cases indicated that no annoying distortion can be heard for both the 54 and 64 kbps cases. Table 5.10 also shows that the test subjects distinguished between the two bit rates, indicating little or no saturation in terms of quality even at relatively high rates as a result of the use of a scalable coding algorithm such as SPIHT.

5.4 Conclusion

This chapter has presented a comparison of a number of compression methods built around the Set Partitioning In Hierarchical Trees algorithm. Masking models were used to increase the compression ratio achievable. This compression ratio was further improved by the introduction of a modification to SPIHT.

The Wavelet-SPIHT scheme was shown to be out-performed by the MLT-SPIHT scheme when complete reconstruction was sought after even without the use of a masking model. The other M-band transform based schemes required the introduction of a masking model to match the Wavelet based scheme.

The introduction of masking to the MLT based scheme produced significant savings in the complete reconstruction case (that is, reconstruction of all unmasked components). The subjective tests verified that the use of this scheme produced audio that was almost identical to the original. The introduction of the modification to SPIHT allowed further improvements with no loss of quality. The limited rate implementations of the MLT based scheme were shown to also perform very well in terms of subjective quality. The over all mean score of the MLT based coder at 54 kbps was above 4, indicating high quality audio being synthesized.

This chapter has demonstrated the application of SPIHT to audio compression, allowing the development of a scalable audio compression algorithm with fine grain scalability. In comparison with the sinusoidal based scheme presented in Chapter 4, the advantages are clear, that is fine grain scalability that is smooth from low to high rates as well as ease of implementation, in terms of the design of one coder that can produce a bit stream that represents the complete signal. That one bit stream can be accessed fully or partially, depending on performance considerations and on channel

capacity. It is possible to implement this scheme such that it changes rate from frame to frame, in the cases discussed here the frame lengths were 20 ms with 10 ms overlap, meaning that the rate of the coder can change every 10 ms.

Comparing the scheme to the existing scalable methods discussed in Chapter 3, again the advantages can be simply stated. Firstly, the enhancement layer of the MLT-SPIHT scheme can be considered as low as a single bit. Secondly, this scheme is not an off-line scheme, unlike the scheme presented in [Ver99] and others which have to either modify the signal to be coded or to analyze large segments of the signal resulting in considerable delay. Unlike other scalable schemes, entropy coding is not used in the MLT-SPIHT scheme and so it is not necessary to design entropy codes to insert into the compression algorithm that are audio specific. It should be noted however, that it is possible to entropy code the bit stream produced by SPIHT to further reduce the bit rate used. This approach was taken in [SP96] where arithmetic coding was used and is a popular method in image processing.

This chapter has not analyzed the synthesized audio by using the same techniques as in Chapter 4. Such analysis has been deferred to the next chapter, where a scalable to lossless scheme is designed, implemented and analyzed in some detail.

Chapter 6

Scalable To Lossless Audio Compression

In this chapter, the lossy scalable audio compression scheme described in Chapter 5 is extended to allow lossless audio compression at rates competitive with the current state of the art in lossless audio compression. The attraction of this scheme is that it allows objective scalability from low rates (such as 16 kbps) to high rates (such as 320 kbps) with a smooth increase in quality. At the highest rates, lossless audio compression is achieved. This chapter also investigates the use of integer transforms for scalable audio compression as those transforms allow the lossless reproduction of the original signal. SPIHT (as described in Chapter 5) is also modified in this chapter to produce PSPIHT or (Perceptual SPIHT), an algorithm that allows scalable perceptual compression as well as lossless compression.

. This chapter thus deals with the scalable to lossless compression contributions of this thesis, which are also described in [RMB02a] and [RMB03].

6.1 Introduction

With the introduction of third generation cellular phone systems and the possible expansion of those systems, digital cellular phone users may in the near future have access to data rates above 144 kbps [Ric00]. This is a considerable increase on the rates that the second generation of mobile telephony achieved (e.g. GSM [Ric00, EV99]), which in most implementations only provides users access to 9.6 kbps of data [Ric00]. Such an explosion in the possible bit rate, and the nature of the proposed bit streams means that multimedia compression schemes may be adjusted to allow for increased quality of the delivered product to the user. The other well known medium of multimedia delivery, the Internet, is also experiencing an increase in possibilities with the introduction of broadband technology.

The increase in bit rates means that audio compression algorithms with higher bit rates than those currently used (such as MPEG's MP3 [BKS00]) can be used to obtain higher quality. However, the new increased data rates are not necessarily constant; this is especially the case when considering the Internet. This is one of the main reasons why scalable and lossless schemes have become interesting from an application point of view.

Lossless audio compression may be viewed as an adaptation of more general lossless coding schemes such as 'Lempel Ziv' [GPS94] and 'zip' which attempt to reduce the storage capacity required for a given set of samples. However, it has been found that such algorithms only produce a small amount of compression when applied to PCM audio signals [BOvdV96]; hence it is necessary to use algorithms that take advantage of the nature of the audio signal. As was mentioned in Chapter 3 the most popular approach to lossless audio compression is to initially use a linear predictor to

produce a residual signal that has a smaller dynamic range and is more white noise like than the original. Then the predictor coefficients and the residual are coded by the use of entropy coding techniques [HS01]. The residual signal may be generated in other ways; for example, a transform coder was used in [LPN97]. A more generic scheme was proposed in [MIJM00] where a lossy coder was used to approximate the original signal and an entropy code applied to the residual, producing a lossy to lossless coding scheme. Thus, lossless compression usually takes on the form that is proposed for scalable compression in MPEG-4, namely the division of a signal into different quality layers where, in this case, the term quality refers to the waveform matching of the original signal. However, none of the schemes found to date in the literature allow scalability from lossy to lossless compression whilst allowing scalable lossy compression and a smooth transition in obtained quality between the lossy and lossless sections of the coder.

Having described the advances in the bandwidth availability for cellular telephones, and that for Internet users, it is clear that a compression scheme that allows smooth scalability from low rates to lossless rates is of interest and potential use. MPEG have started a process of standardization for such a scheme [Mor02]. SPIHT provides an important tool in the development of such a coding scheme because of its set sorting and partial transmission of coefficients or samples. However, there is the important question of how to best utilize this powerful tool for such a task. This chapter provides experimental methods and techniques that help answer this question. As a first step, the MLT-SPIHT scheme that was described in the last chapter is used to produce lossless compression. This scheme is then replaced by an integer transform combination with SPIHT as an experiment into the potential use

of these transforms in audio compression. A solution to the overall problem at hand that produces both smooth scalability and competitive lossless compression results is then proposed. This solution is extended further to allow perceptual scalability whilst maintaining the objective scalability between lossy and lossless compression. To achieve this a new algorithm is proposed, called Perceptual SPIHT (PSPIHT), which is based on the SPIHT algorithm.

6.2 Achieving lossless compression with MLT-SPIHT

As a starting point in our discussion about lossless compression it is important to clarify what is exactly meant by achieving lossless compression. Assuming that the original audio signal $x(n)$ is PCM coded and consists of a sequence of integers, it is sufficient for perfect reconstruction that a synthesized audio signal $\hat{x}(n)$ can be generated that satisfies

$$|x(n) - \hat{x}(n)| < 0.5 \tag{6.2.1}$$

for all n, because then a rounding operation allows us to recover $x(n)$ from $\hat{x}(n)$ without error. In other words, the linear synthesis part of an audio coder does not necessarily need to produce an error free reconstruction. It only needs to bring $\hat{x}(n)$ close enough to $x(n)$ so that the nonlinear rounding operation finally yields perfect reconstruction.

The MLT-SPIHT scheme was described in the last chapter as a scheme that combines the MLT with SPIHT for audio compression. It was also explained in the last chapter that SPIHT allows one to specify the accuracy to which the given

coefficients or samples are coded. It is also possible to precisely define the total bit rate that can be used for coding. When considering these facts with the knowledge that the condition for lossless representation is given by (6.2.1), it can be deduced that with a high enough coding resolution of the MLT transform coefficients one can achieve lossless compression. To show that this is indeed possible, a number of experiments were conducted where the coding resolution used (for the quantization of the MLT coefficients) was varied between 10 and 25 bits at various limiting maximum bit rates.

The frame length used in this study is 1024 samples, with 512 samples of overlap. That corresponds to 23.2 ms a frame at a sampling rate of 44.1 kHz. The maximum bit rates were set at 192 kbps, 353 kbps and 512 kbps, respectively. The similarity of the synthesized audio to the original was estimated through the calculation of the first-order entropy of the error signal $\varepsilon(n) = \tilde{x}(n) - x(n)$, where $\tilde{x}(n) = \text{round}(\hat{x}(n))$ denotes the rounded output signal of the MLT synthesis bank.

The test material that has been used in this work is the same as that used throughout this thesis. In the following results are presented that were obtained using file x1 as they are sufficient to demonstrate the conditions under which SPIHT combined with the MLT will reach lossless compression. Figure 6.1 shows the results of the experiment. There are a number of points to note from the figure: first, given a high enough rate and coding resolution the MLT-SPIHT system does produce an exact copy of the original as indicated by the entropy reaching zero. Secondly the maximum rates defined do not affect the entropy result until at least 15 bits are being used for the quantization. This illustrates how the two factors of limiting rate and quantization resolution interact to affect the quality of the synthesized signal. One

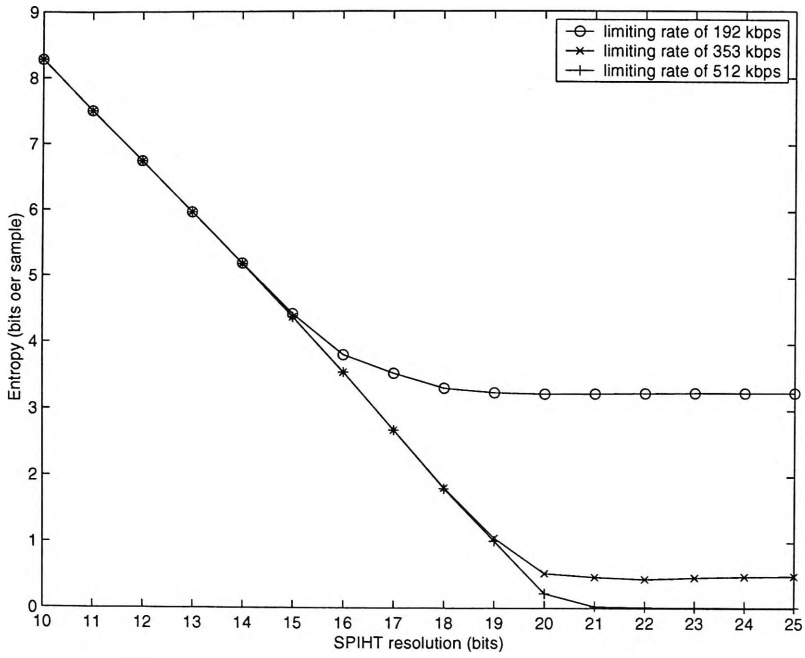

Figure 6.1: The entropy of the error signal using different SPIHT resolutions at three maximum rates.

can say that above a certain coding resolution the limiting rate is the important factor for the quality of the synthesized signal. The presented results also allow for a comment about the expected lossless rate when coding the error with an entropy code. For example, at a coding resolution of 20 bits and a lossy rate of 192 kbps the final lossless rate should be approximately 345 kbps (assuming an entropy code that codes at first-order entropy and reading from the figure that at 20 bits and 192 kbps maximum rate the entropy is approximately 3.5 bits per sample). Note that this rate is well below the 512 kbps rate which achieves approximately zero error entropy at 23 bits coding resolution using the straightforward MLT-SPIHT coder. This observation is very important for the scheme proposed here as it shows that a lossy scheme based on SPIHT combined with a lossless scheme will produce a better lossless compression ratio than the MLT-SPIHT scheme alone.

The results of the MLT-SPIHT scheme applied to lossless compression lead towards one potential solution to the scalability to lossless. However, in answering the question stated in Section 6.1 one must explore other possible avenues for the application of SPIHT. One of these avenues is the combination of SPIHT with integer to integer transforms. The reasoning is simple, given a PCM coded signal with integer samples then an integer transform will produce integer coefficients. These coefficients can then be transmitted without loss by the use of SPIHT. The next section explains integer transforms and explores the use of two integer transforms with SPIHT.

6.3 Integer to Integer Transforms

Transform coding, as first introduced in [HS63], is suboptimal to vector quantization in terms of representing a given signal with a limited number of bits [Goy00]. This sub-optimality originates from the reliance of transform coders on scalar entropy quantization schemes. However, if one considers the ideal transform coding scenario, that is the transform decorrelates the input signal perfectly then scalar entropy coders become just as efficient as vector entropy coders with much less complexity [Goy00]. This much reduced complexity is a major attraction of transform coding.

The integer to integer transform coding scheme is shown in Figure 6.2, and to allow a comparison with the standard transform coding scheme, the standard transform coding scheme is repeated in Figure 6.3. The difference between the two schemes is that in the integer to integer case, the quantization is carried out before the transformation. The use of an invertible transform leads to lossless reconstruction of the signal. It is shown in [Goy00] that the real benefit of transform coding comes about from its decorrelation of the input signal. This can be derived by assuming very high

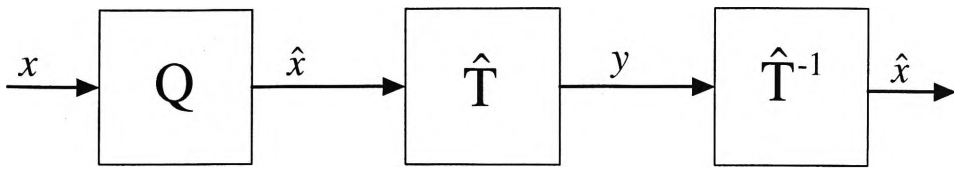

Figure 6.2: Integer to Integer Transform Coding

bit rate coding and comparing the scalar entropy of the standard transform coding scheme and the integer to integer coding scheme.

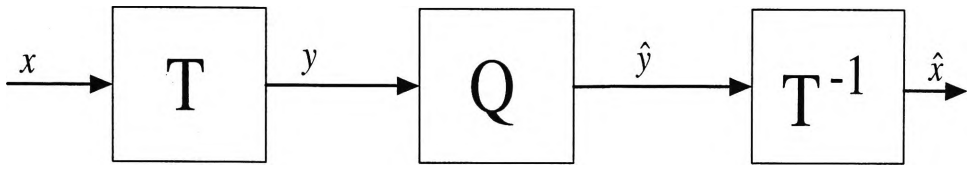

Figure 6.3: The Transform Coding Scheme

Integer to integer transforms map integers to integers. The function of these transforms is the same as other transforms; namely, to map a set of correlated coefficients to a set of less correlated coefficients [Beu84]. The new set of coefficients should ideally contain only a small number of large coefficients and many small coefficients, allowing the use of only a few coefficients to approximate the original signal [Beu84].

There are two broad categories of integer to integer transforms; integer to integer transforms with non-integer coefficients that map integers to integers but do not have integer coefficients only in the basis functions, and integer integer to integer transforms that have only integers as coefficients in the basis functions, these are referred to in this thesis as integer transforms. The work presented here is only concerned with two integer transforms, the Walsh-Hadamard transform [Beu84] and the integer cosine transform as presented in [Cha89].

6.3.1 The Walsh-Hadamard Transform

The transform matrix of the order 8 Walsh-Hadamard transform is given by (6.3.1). The Walsh transform is actually an ordered version of the Hadamard transform. The ordering is done in terms of increasing sequency [Beu84]. Sequency is defined as the number of times a function crosses zero divided by two, it is a more general version of frequency and allows non periodic functions to be described in a manner similar to frequency. Thus the ordering in terms of increasing sequency means that the rows in W are organized such that the number of sign changes increases along the rows.

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \tag{6.3.1}$$

The Walsh-Hadamard transform has a number of important properties, the most useful in the case of compression being the invertability of the transform, that is

$$\mathbf{W}_M{}^T \times \mathbf{W}_M = MI \tag{6.3.2}$$

The Walsh-Hadamard transform can be used for numerous other applications [Beu84], [YH97], including the development of fast implementations of other transforms, such as the Discrete Fourier Transform (DFT) [YH97]. However, in the context of signal compression, transforms are primarily chosen with good frequency selectivity

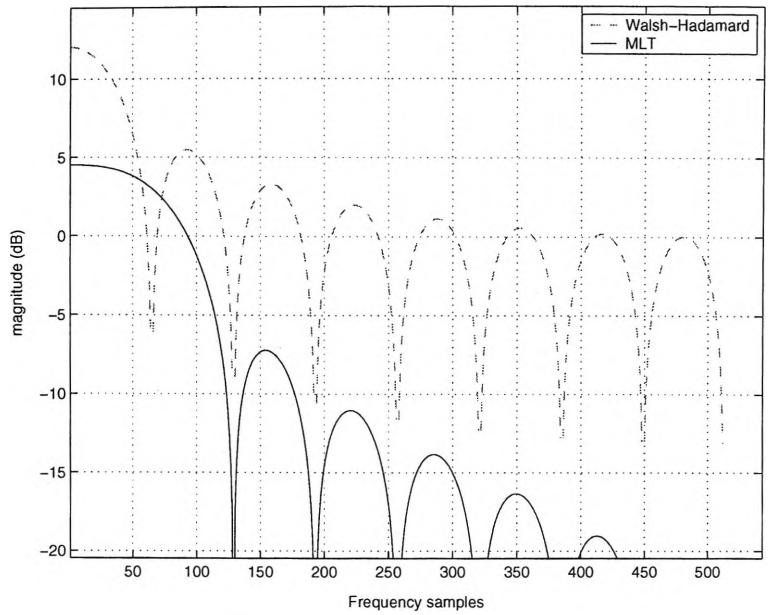

Figure 6.4: Comparison of the Walsh-Hadamard lowpass filter with the MLT low pass filter, both of order 16

as a prime objective [Mal92]. This allows the reduction of leakage between frequency bands and hence a more accurate representation of the signal in the frequency domain. The use of the Walsh-Hadamard transform results in a poorer frequency selectivity than other, more popular, transforms in signal compression such as the MLT. The difference in frequency selectivity may be observed from Figure 6.4 which illustrates the frequency response of the low pass filters of the MLT of order 16 as compared with the frequency response of the equivalent filter in the Walsh-Hadamard transform. It is observable from Figure 6.4 that the first side lobe attenuation of the MLT is approximately double that of the Walsh-Hadamard transform in decibels.

Yet there is another important consideration in lossless signal compression, that of dynamic range. The use of an integer transform on integer signals results in transform coefficients that have a greater dynamic range than the input coefficients.

This means that the transform coefficients require a larger number of bits to be quantized in a scalar manner. The increase in the dynamic range is directly related to the size of the coefficients and the length of the transform filters. The increase in the dynamic range leads to an increase in the code book size required to represent the transform coefficients [GTA00], which in turn presents difficulties for compression. Hence, integer transforms that do not lead to a large increase in the dynamic range are attractive for compression purposes. Since the Walsh-Hadamard transform only utilizes positive and negative ones it can also be deduced that the maximum possible increase in the dynamic range, in bits, is (where M is the order of the transform)

$$\Delta = \lceil \log_2 M \rceil \tag{6.3.3}$$

6.3.2 The Integer Cosine Transform

The Integer Cosine Transform (ICT) originally developed by Cham [Cha89] has integer basis functions based on the DCT. There have been a number of schemes proposed since for the design of sinusoidal transforms with integer coefficients [PD00]. The attraction of using sinusoidal based transforms lies in the nature of the signals being transformed. Signals that tend to have a pseudo-sinusoidal behavior are more appropriately transformed with transforms that have sinusoidal basis functions, where as signals that are more 'square' in nature are better modelled using a sequency transform such as the Walsh-Hadamard transform [Beu84]. To illustrate such an effect, consider Figures 6.5 and 6.6. Figure 6.5 shows the obtained DCT coefficients (of order 64) when a 50 Hz sinusoid is transformed. Figure 6.6 shows the equivalent Walsh-Hadamard coefficients. Note how the energy of the signal is localized in the frequency

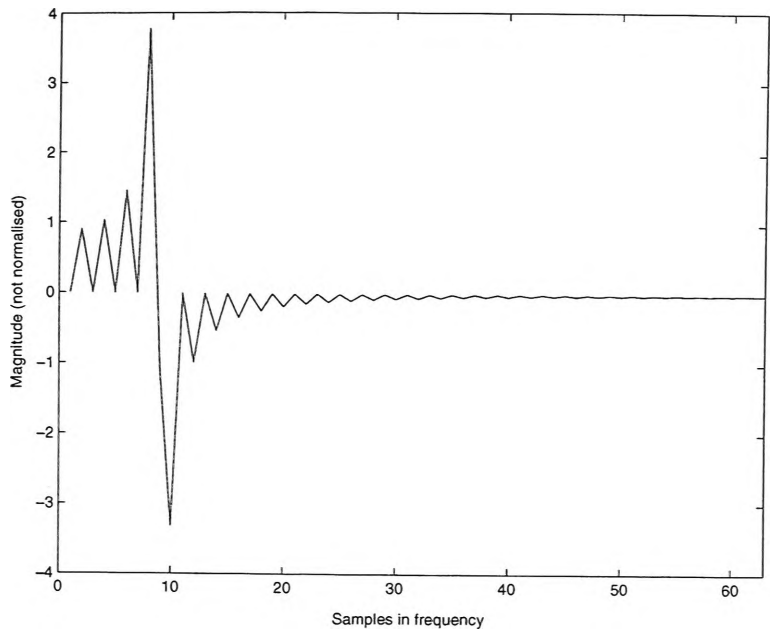

Figure 6.5: DCT coefficients for a sinusoidal input

domain but is spread in the sequency domain. In terms of signal compression, it is of importance that the transform coefficients be highly localized.

The ICT, as developed in [Cha89] has a transform matrix built on the property of 'dyadic symmetry' from the DCT. It is argued that an infinite number of ICTs may be generated through this technique, and as such Cham proposes the use of the following matrix for the design of order 8 ICTs:

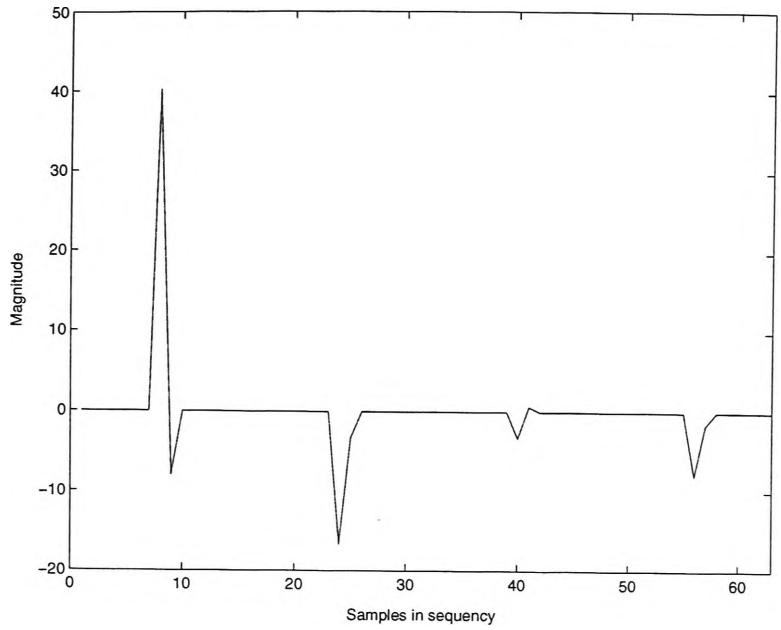

Figure 6.6: Walsh-Hadamard Transform coefficients for a sinusoidal input

$$\mathbf{C_8}^T = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a & b & c & d & -d & -c & -b & -a \\ e & f & -f & -e & -e & -f & f & e \\ b & -d & -a & -c & c & a & d & -b \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ c & -a & d & b & -b & -d & a & -c \\ f & -e & e & -f & -f & e & -e & f \\ d & -c & b & -a & a & -b & c & -d \end{pmatrix} \tag{6.3.4}$$

As the integer cosine transform is actually defined by the use of the set $\{a, b, c, d, e, f\}$, each different ICT will be referred to as $\mathbf{C}\{a, b, c, d, e, f\}$. Cham [Cha89] describes a technique of developing order M ICTs from the order 8 ICT recursively, it is included here for completeness. The method generates an order $2M$ transform matrix from an order M transform matrix.

For i = [0,...,M-1] basis vectors of $\mathbf{C}_{2M}(i,j)$

$$\mathbf{C}_{2M}(i,2j) = \mathbf{C}_M(i,j)$$

$$\mathbf{C}_{2M}(i,2j+1) = \mathbf{C}_M(i,j)$$

$$\text{j} \in [0,\ldots,M\text{-}1]$$

For i = [M,...,$2M$]

$$\mathbf{C}_{2M}(i+M,2j) = \mathbf{C}_M(i,j)$$

$$\mathbf{C}_{2M}(i+M,2j+1) = -\mathbf{C}_M(i,j)$$

$$\text{j} \in [0,2,4,\ldots,M\text{-}2]$$

$$\mathbf{C}_{2M}(i+M,2j) = -\mathbf{C}_M(i,j)$$

$$\mathbf{C}_{2M}(i+M,2j+1) = \mathbf{C}_M(i,j)$$

$$\text{j} \in [1,3,5,\ldots,M\text{-}1]$$

Figure 6.7 displays the obtained coefficients for the order 64 ICT with integer set $\{3,2,1,1,3,1\}$ for the same sinusoidal input as used to obtain the DCT coefficients of Figure 6.5. Note that the transform coefficients energy is slightly more localized than those of the Walsh-Hadamard case. The integer set used for this example is actually the worst performing one in terms of limited coefficient set reconstruction presented in [Cha89], however, it is the set that results in the least increase in dynamic range. The dynamic range set increase for the order 8 C{3,2,1,1,3,1} is given by:

$$\Delta = \lceil \log_2(2 + \sum \{3,2,1,1,3,1\}) \rceil = 4 \quad \text{bits}$$

That is, one bit more than the equivalent Walsh-Hadamard Transform. The frequency selectivity of the transform improves with the use of larger integers [Cha89]. Thus, the use of the ICT is quite varied depending on the cost that one is willing to pay in terms of dynamic range increase for the improvement in frequency selectivity and implementation complexity.

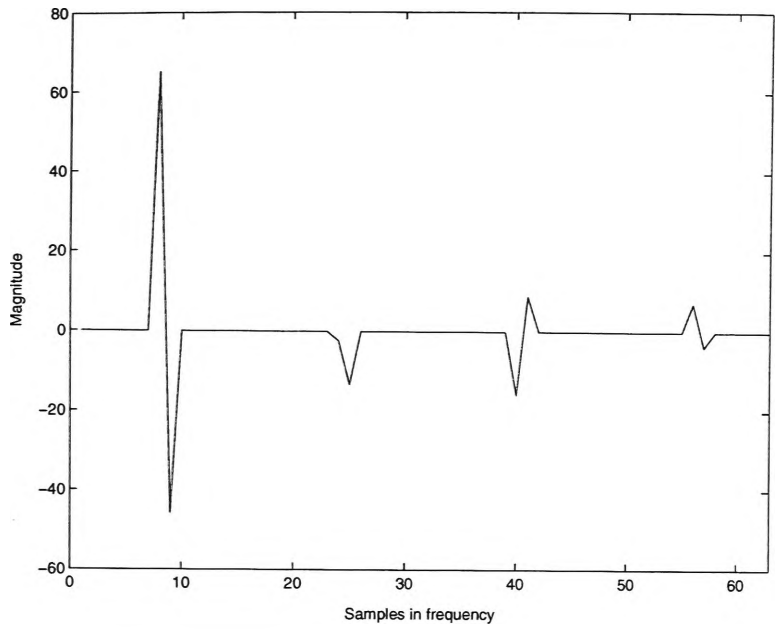

Figure 6.7: C(3,2,1,1,3,1) coefficients for a sinusoidal input

6.4 The Integer to integer system and results

The two transforms described previously were combined with SPIHT in a similar way to the MLT. The use of such transforms means that the increase in the dynamic range must be taken into account. As explained earlier, a large dynamic range means a larger code book is required to represent the signal when entropy coding is being used. SPIHT acts very much like an entropy coder in that as the dynamic range of the input increases so does the number of bits required to code the input. To clarify this point, Figure 6.8 has been included. The Figure shows the required number of bits to transmit the complete set of MLT coefficients as related to the quantization level of the coefficients. The hyperbolic increase in the mean bit rate clearly shows the detrimental effect of increasing the code book size, this effect has been similarly noted in [GTA00].

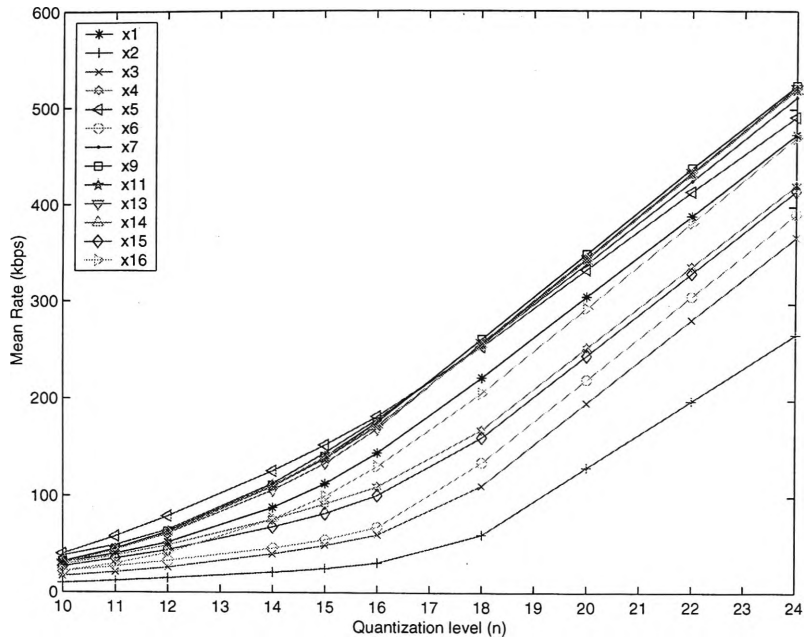

Figure 6.8: Mean Bit Rates vs the Quantization Resolution used in SPIHT

There is no overlap in the experimental system, overlap is quite often used in audio compression algorithms because modulated filter banks are used for the compression [Shl97]. In this case the filter banks being used, that is the transforms, are not modulated. Although a proper modulation would improve the frequency selectivity of the transform, the introduction of the overlap would cause redundancy unless a lapped transform was being used which is not the case here. Also, as the aim is the study of how well these transforms perform in a lossless coder, edge effects become irrelevant ultimately as one seeks a perfect reconstruction of the original signal.

In implementing this system a choice must be made with regards to the frame length that will be used. As mentioned a number of times previously, the longer the transform used, the greater the increase in the dynamic range. However, the shorter the transform the lower the frequency resolution and the greater the distribution of perceptual error for a given quantization noise. Hence, there is a trade off between

Table 6.1: Lossless Compression Results for different frame lengths

Frame length (samples)	Duration (ms)	Mean Rate (kbps)
64	1.45	582
128	2.90	595
256	5.8	610
512	11.6	629
1024	23.2	649

lossless compression and quality at the lower rates. Table 6.1 shows the compression results for the different frame lengths used. The results were generated by compressing 12.44 seconds of signal x1 using the basic system outlined and the varying frame lengths. The signal is a monotone uniform PCM signal (16 bit quantization) sampled at 44.1 kHz, hence the original bit rate is 706 kpbs .

The effect of the frame length is clear from Table 6.1. Using those results, the frame length was set at 64 samples and results obtained for lossless as well as scalable compression. Table 6.2 lists the results of the lossless compression. Table 6.2 shows that the maximum compression ratio obtained is 1.74. This compression is not competitive with the state of the art in lossless audio compression. One of the main reasons for this is that with the use of the Walsh-Hadamard Transform very few zero coefficients are obtained. As SPIHT is a sorting algorithm that transmits all of the significant bits of any non-zero coefficient, and so does not transmit any zero coefficients, the lack of zero coefficients means that a large number of bits must be transmitted. This problem may be reduced by the use of a more efficient transform (in terms of grouping the energy of the input signal into a few coefficients).

Similarly, Table 6.3 shows the lossless compression results for ICT{3.2.1.1.3.1}. The results show very little compression. This is the result of increasing the dynamic range without any real gain in terms of efficiency. In [Cha89] ICT{3.2.1.1.3.1} is the

Table 6.2: Lossless Compression Results For the SQAM Files

Signal	Mean Rate (kbps)	Compression Ratio	Signal	Mean Rate (kbps)	Compression Ratio
x1	570	1.24	x9	542	1.30
x2	405	1.74	x10	559	1.27
x3	550	1.28	x11	518	1.36
x4	598	1.18	x12	552	1.28
x5	502	1.41	x13	519	1.36
x6	557	1.27	x14	517	1.37
x7	582	1.21	x15	548	1.29
x8	563	1.25	x16	518	1.36

Table 6.3: Lossless Compression Results For the SQAM Files Using ICT{3,2,1,1,3,1}

Signal	Mean Rate (kbps)	Compression Ratio	Signal	Mean Rate (kbps)	Compression Ratio
x1	667	1.06	x9	640	1.1
x2	547	1.29	x10	644	1.1
x3	609	1.16	x11	609	1.16
x4	653	1.08	x12	658	1.07
x5	578	1.22	x13	614	1.15
x6	697	1.01	x14	605	1.17
x7	675	1.05	x15	636	1.11
x8	666	1.06	x16	618	1.14

worst performing ICT in terms of Mean Square Error (MSE), yet it is the version presented that will introduce the lowest dynamic range increase, because of the values of the coefficients. One could attempt to improve the performance of the ICT based scheme, however, the initial results were not encouraging and have simply illustrated the detrimental effects of increasing the dynamic range.

In terms of perceptual quality, by simply listening to the synthesized audio produced by the described schemes it was observed that the scheme (using either transform) is perceptually indistinguishable from the original only at around the 256 kbps

Table 6.4: Experimental Results for The Overall Rate Given Various Base Rates

Base Rate (kbps)	Error Rate (kbps)	First Order Entropy (bits per sample)	Total Rate Expected (kbps)	Actual Total Rate (kbps)
64	344	7.02	374	408
80	329	6.48	366	409
96	305	6.01	362	401
128	273	5.24	359	401
192	213	3.72	356	405
256	170	2.56	369	426

mark. The lower rates display a buzziness in the synthesized sound that could be attributed to a short frame length being used (as in lossy coding, a short frame length will lead to noise being spread across greater sections of the spectrum).

In conclusion to this section it can be said that whilst the use of the integer transforms utilized here may result in implementation advantages, the rates obtained for lossless compression are not much better than the MLT-SPIHT scheme, and both schemes sub-perform the state of the art. The results presented thus far dictate that the lossy to lossless scheme be built of a lossy scalable section and a scalable lossless section.

6.5 The SPIHT scalable-to-lossless scheme

It has been mentioned in the previous section that if an entropy code for the residual error was to be combined with the lossy MLT-SPIHT scheme, then a good overall lossless compression ratio may be expected. It has also been outlined that one of the factors that generally influence the performance of SPIHT is the dynamic range of the input data. Hence, it is possible to reason that if the dynamic range of the

Table 6.5: Results for The Lossless SPIHT Coder

Signal	Rate (kbps)	Compression Ratio	Bits/Sample
x1	318	2.22	7.20
x2	134	5.27	3.03
x3	206	3.43	4.65
x4	266	2.65	6.01
x5	346	2.04	7.84
x6	232	3.04	5.23
x7	354	1.99	8.01
x8	317	2.23	7.18
x9	366	1.93	8.28
x10	405	1.74	9.17
x11	362	1.95	8.19
x12	368	1.92	8.33
x13	360	1.96	8.15
x14	360	1.96	8.15
x15	255	2.77	5.75
x16	306	2.31	6.93

synthesis error was sufficiently small then SPIHT could still be used to code the error signal at an acceptable rate, whilst maintaining the scalability of the coder in terms of waveform matching, until the lossless condition is met.

An example of the difference in dynamic range between the original audio signal $x(n)$ and the error signal $\varepsilon(n)$ is shown in Figure 6.9 where $x(n)$ is coded at 64 kbps. It can be seen from the figure that the reduction in dynamic range is significant. It is also important to determine the statistical properties of the error signal, particularly the similarity between the error signal and white noise. This is important as it determines if there would be any gain in the use of a transform to further decorrelate the error signal. As expected, an analysis shows that the more bits that are spent on the compression of the original signal the more white-noise like is the error signal, and the less benefit one can expect from transforming the error signal. To illustrate

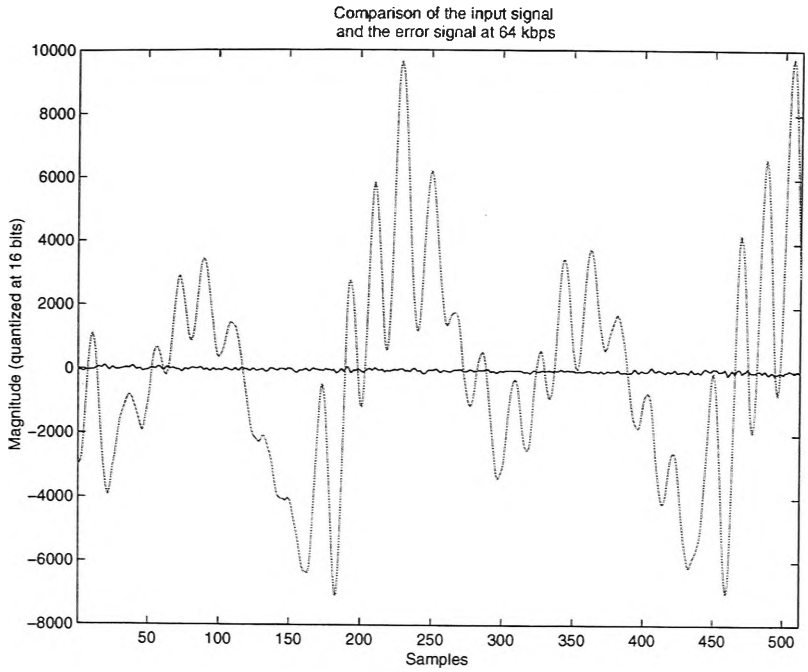

Figure 6.9: The difference in the dynamic range between the error signal and the original signal when the lossy coder is operating at 64 kbps (the smaller signal is the error).

this, Figure 6.10 shows the Power Spectral Densities (PSDs) of two versions of the error signal for a coded frame of audio at rates of 64 kbps and 128 kbps, respectively.

The structure of the coder proposed in this paper is depicted in Figure 6.11. It consists of the combination of the lossy coder described earlier, which is based on the Modulated Lapped Transform (MLT) and SPIHT, and a lossless coder for transmitting the error made by the lossy part. The lossy part is given by the right half of the structure in Figure 6.11, and the error coding (if present) takes place in the left half. Note that both parts of the coder are based on the SPIHT algorithm. The input signal is transformed using the MLT where floating point calculations are used. The transform coefficients are encoded using SPIHT, and the bitstream is transmitted to the decoder. We will refer to this bitstream as bit stream one ($bst1$). Bitstream one

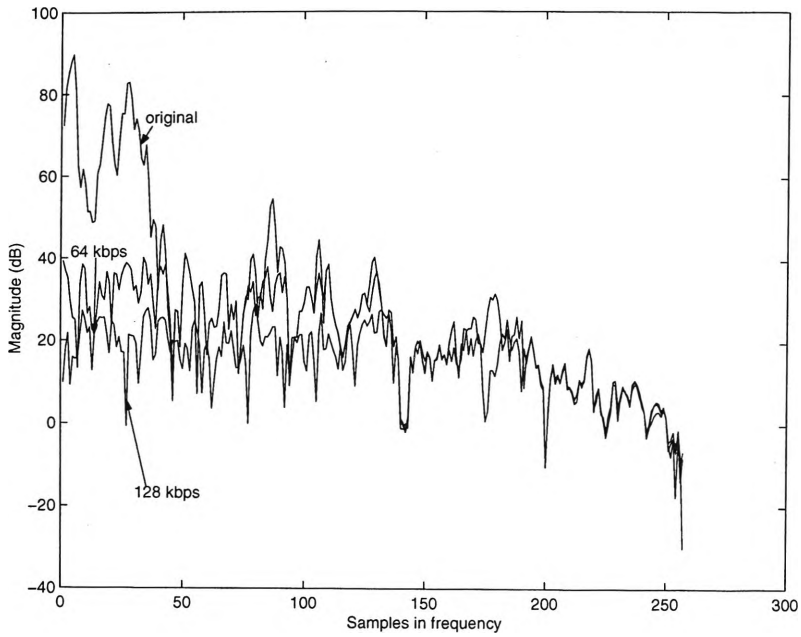

Figure 6.10: PSD of the error signal at 64 kbps and 128 kbps as compared to the original.

is decoded at the encoder and the synthesized audio is subtracted from the original audio to obtain the output error. Here integer operations are used, so that the error output is integer and, as has been discussed, usually has a dynamic range that is less than that of the original integer signal. The time-domain error signal is then encoded into bitstream two (*bst*2), using a second SPIHT encoder. At the decoder, both bitstreams are received as one global bitstream, with bitstream one making up the first part of the total bitstream. The decoder may decode up to any rate desired. If bitstream one containing the transform coefficients is used up, then the decoder recognizes that the remaining bitstream is for the time-domain error signal, which it reconstructs and adds to the synthesized signal.

To produce good lossless performance, the maximum lossy rate should be determined. This is the rate at which the scheme stops transmitting frequency domain

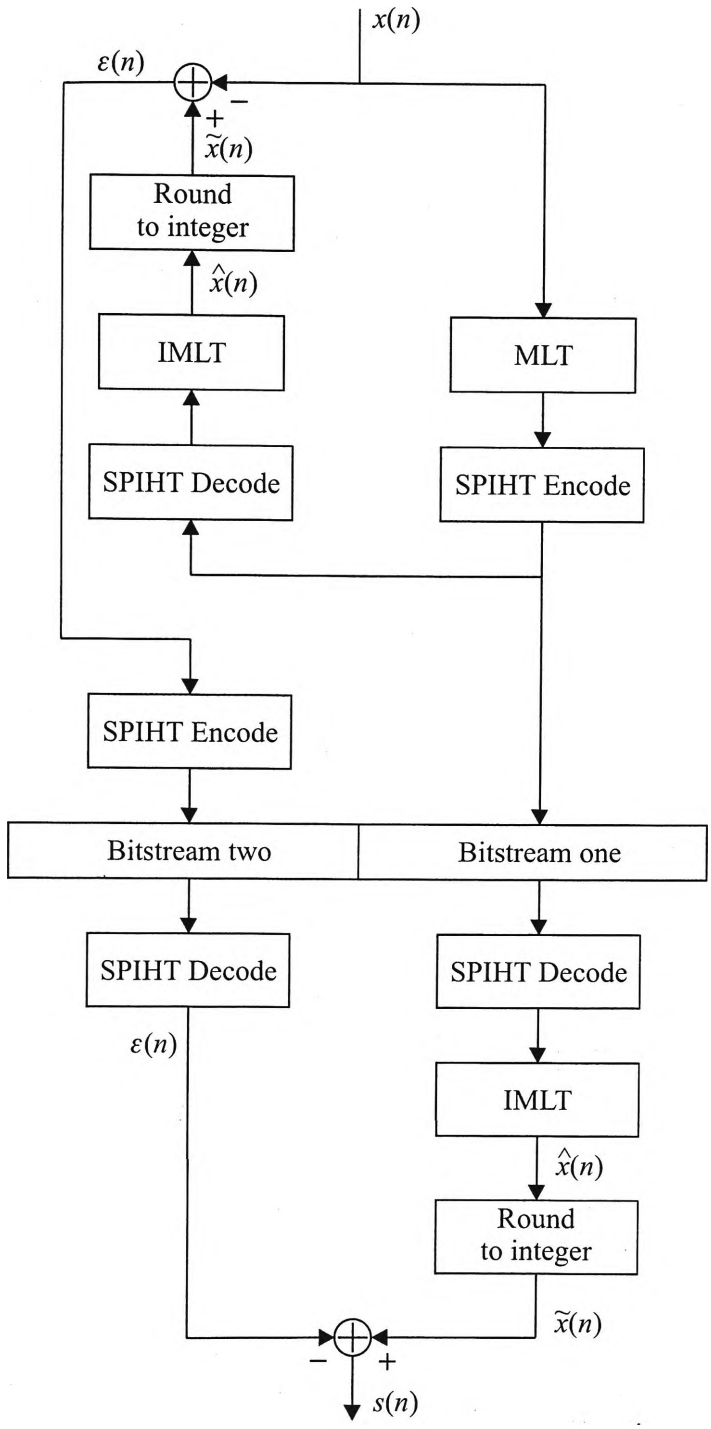

Figure 6.11: The scalable-to-lossless scheme based on SPIHT

coefficients and starts transmitting time domain error samples. That is, its the maximum rate allocated to bitstream one.

6.5.1 Determining the maximum lossy rate

A number of experiments were conducted to determine what rate the lossy scalable component of the coder should be set at. Figure 6.12 shows the results of one such experiment where the lossy maximum rate was set to values between 16 kbps and 256 kbps (inclusive) in 16 kbps intervals. Here 18 bits for SPIHT coding resolution was used. At each maximum lossy rate the entropy of the error has been calculated and used to determine the lower bound for the rate required to achieve lossless compression if an entropy code was to be used to code the error. Two of the three curves on the graph describe the expected rate in different situations, and one gives the collected rate with the proposed coder. The top curve (i.e. the one with the worst performance) describes the expected lossless rate if lossy rate reservation was used, that is if bitstream one was allocated the maximum lossy rate all the time. SPIHT does not require such reservation of bitstream space. The second curve from the top takes this into account and does not assume that bitstream one is allocated the maximum rate all the time, instead it utilizes the actual rate required by SPIHT for a complete reconstruction of the coefficients up to the coding resolution that is hard coded at both encoder and decoder. This curve continues to decrease with the decreasing entropy of the error signal and finally at 192 kbps crosses the lowest curve in the figure. The lowest curve in the figure is the actual rate collected for the proposed coder. It is noticeable that the SPIHT scheme outperforms the lossy-plus-entropy code scheme until the 192 kbps mark for the maximum lossy rate. The reason behind this is that

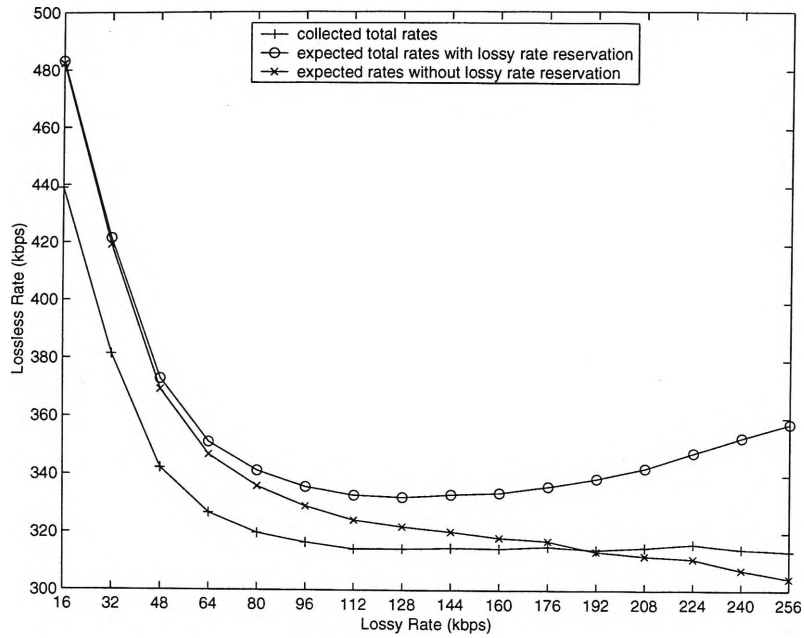

Figure 6.12: Mean lossless rates collected, compared with the lossless rates expected.

SPIHT transmits only the significant bits of the coefficients, and importantly, for zero coefficients or samples the algorithm does not transmit a single bit. An entropy code must transmit at least one bit (and in most cases two) per coefficient, even if that coefficient is zero. SPIHT avoids these extra bits by recognizing large sets of zero coefficients or samples and treating them collectively in the sorting process.

6.5.2 Psychoacoustic analysis of the lossy component

Having analyzed the results in terms of lossless compression, the performance of the coder has to be analyzed for its subjective effects on the synthesized audio at different lossy rates. A psychoacoustic analysis of the lossy scalable component of the coder was performed to add a perceptual dimension to the choice of the maximum lossy scalable rate. The analysis determined the mean variation between the pleasantness parameters of the original signal and the synthesized signal at different maximum

lossy rates.

Figure 6.13 shows the results obtained at different maximum lossy rates for signal x1. The curves show the mean percentage variation in the psychoacoustic parameters, denoted as E_v and calculated by the use of the equation:

$$E_v = E\Big(\frac{|p - p_0|}{p_0}\Big) \times 100\% \tag{6.5.1}$$

where p_0 is the value of a pleasantness factor calculated for the original signal $x(n)$, p is the pleasantness factor calculated for the reconstructed signal, and $E(\cdot)$ denotes the expectation operation. It can be seen that the mean variation decreases with the increasing rate, however it can also be seen that the variation is not significant at any rate, starting at near the 10% mark for sharpness and roughness and near the 3% mark for loudness. The low variation of loudness is expected as at 32 kbps, the lowest rate used, SPIHT would have transmitted good approximations of the most significant spectral components, leading to a loudness level that is similar to the original one. Sharpness is influenced by the center frequency of the signal and the distribution of spectral components, which should also be well approximated at 32 kbps. A similar line of reasoning follows for the roughness result. Thus the variation is expected to be small, the important property is how the variation is reduced. That is, at what rate does the reduction in variation saturate. The presented figure shows that the percentage variation reaches a knee point at around the 96 to 128 kbps marks. Similar results were obtained for other signals tested. The knee point position was found to depend on the spectral content of the signal being used, which is expected, with highly tonal signals reaching the knee point at lower rates than more noise-like signals. Using the psychoacoustic results and the lossless rate versus lossy rate results presented in Section 6.5.1, it is safe to conclude that any lossy rate between 128 kbps

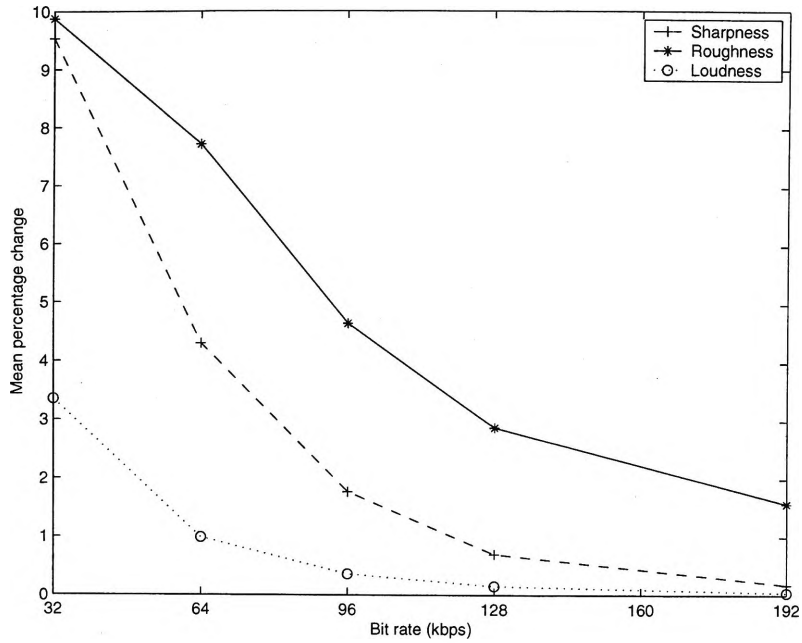

Figure 6.13: Sharpness, roughness and loudness variations at different lossy rates for x1.

and 192 kbps will produce good scalable-to-lossless performance.

6.6 Results

Two sets of results are presented here: the lossless compression results and the objective scalable lossy results of the MLT-SPIHT coder. First we consider the lossless compression results.

6.6.1 The lossless compression results

Using the experiments described in Sections 6.5.1 and 6.5.2, it was determined that a lossy maximum rate of 192 kbps should be used in combination with a coding resolution of 18 bits per spectral coefficient and 16 bits (PCM) per time domain error coefficient. Table 6.6 shows the results for the lossless compression of the SQAM

files of Table 4.2. Most of the files show a compression ratio that is above 2, which is competitive with the current state of the art in lossless compression [HS01]. The lowest compression ratio was 1.74 for female French speech, whilst the greatest ratio obtained was 5.27 for an electronic tune. The average compression ratio obtained was 2.46. As with other current schemes, the compression ratio depends strongly on the content of the signal [HS01]. In most current schemes, the compression ratio is higher for highly predictable signals that can be very well modelled by the use of a linear predictor. In this case, and because of the scalability capability, the more concentrated the energy of the signal is in the frequency domain the higher the compression ratio. The reason being that a signal with concentrated energy in the frequency domain is coded very well in the first part of the coder and so a very small, highly uncorrelated, error signal is produced leading to a high lossless compression ratio overall.

Table 6.6: Results for the Lossless SPIHT Coder.

Signal	Mean Rate (kbps)	Compression Ratio	Bits/Sample
x1	318	2.22	7.20
x2	134	5.27	3.03
x3	206	3.43	4.65
x4	266	2.65	6.01
x5	346	2.04	7.84
x6	232	3.04	5.23
x7	354	1.99	8.01
x8	317	2.23	7.18
x9	366	1.93	8.28
x10	405	1.74	9.17
x11	362	1.95	8.19
x12	368	1.92	8.33
x13	360	1.96	8.15
x14	360	1.96	8.15
x15	255	2.77	5.75
x16	306	2.31	6.93

6.6.2 Objective Results for the scalable-to-lossless and lossy coders

Figure 6.14 shows the Segmental Signal-to-Noise Ratio (SegSNR) results for a lossy coded version of signal x1 (up to 240 kbps) as well as the performance of the scalable-to-lossless scheme described earlier up to and including 320 kbps. The SegSNR values are calculated using frames that are 17.5 ms long and not overlapping (note that this does not match the frame selection in the coding scheme). It can be seen from Figure 6.14 that there is a knee point for the coder at around the 64 kbps mark. It can also be seen that the SegSNR is above the 40 dB mark at 64 kbps indicating that a high quality signal has been synthesized. As expected, the lossy coder saturates at the high bit rates. In contrast, the scalable-to-lossless scheme continues to improve the SegSNR of the synthesized signal. It is important to note that the values presented in the figure are calculated across frames that have not been perfectly reconstructed. At 320 kbps there were 530 frames (from a total of 1426) that were coded losslessly. The remaining error in the other frames is clearly very small. Note also that the rates listed in Table 6.6 are average rates, while the rates shown in Figure 6.14 are the maximum rates that the coder is permitted to use.

6.7 The Perceptual SPIHT algorithm

The scheme described in Section 6.5 achieves the objective of smooth objective scalability (as can be seen from Figure 6.14). However, the scheme does not take into consideration perceptual issues. Specifically it does not give preference to perceptually significant coefficients over perceptually insignificant coefficients to the detriment

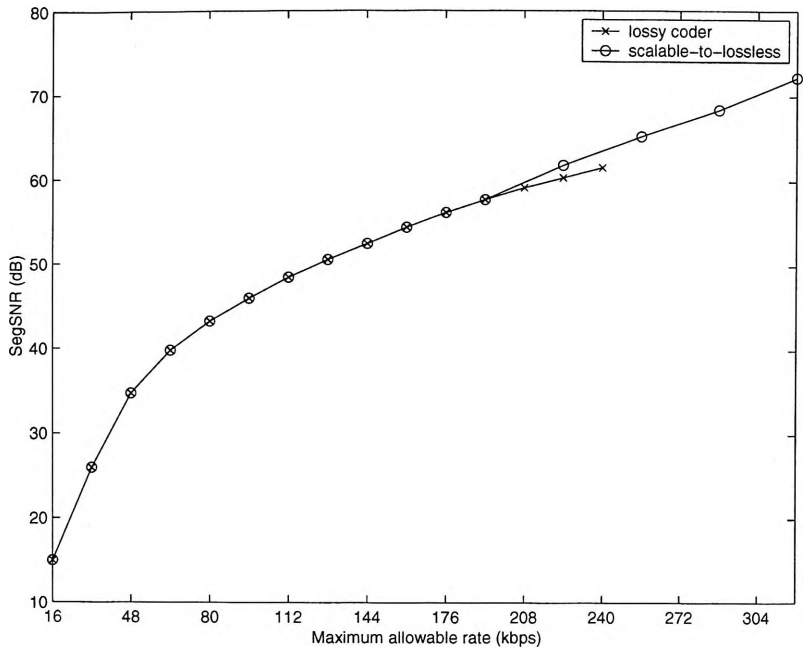

Figure 6.14: Objective results for the lossy MLT-SPIHT coder and the scalable-to-lossless coder.

of the coder performance at lower rates. In previous chapters, the masking model was used to determine coefficients or signal components that are perceptually insignificant to remove them from the coding process. In many current audio coders the masking model is used to modify the scalar quantizers used in each band. In this case because the aim is the scaling to lossless both uses of masking have the disadvantage of distorting frequency domain signal components in a way that will increase the noise dynamic range and so reduce the ability of an algorithm such as SPIHT to produce competitive lossless compression results. An example of this effect has been given in Chapter 4 where the SegSNR of the perceptually equivalent signal was shown to be quite low.

In order to prevent the above described effect, SPIHT itself was modified to allow it to transmit the perceptually significant coefficients first whilst maintaining the

quantization resolution of the perceptually insignificant components at such a level that, when transmitted, the synthesis error signal is still small. To achieve this, the output bitstream of SPIHT itself must again be divided into the perceptually significant bit stream and the perceptually insignificant bitstream. The perceptually significant bitstream is transmitted first.

To generate the required bitstream a few new definitions are required. Let v_{pe} be a binary vector with perceptual significance information for all the coefficients. That is, if $v_{pe}(n) = 1$ then coefficient n is perceptually significant otherwise it is perceptually insignificant. Also let $LPISP$ be the list of perceptually insignificant energy significant points. That is, $LPISP$ contains pointers to coefficients that are significant in terms of energy but lie in a spectral band that contains other more significant coefficients which have masked this coefficient. Finally, name the perceptually significant bitstream $bst1a$ and the perceptually insignificant bitstream $bst1b$.

The complete algorithm is listed below. As in SPIHT, the first stage is an initialization stage, in this case however $LPISP$ is also set to empty. In the sorting pass, the energy significance test is maintained as the first test. Sorting bits are sent to $bst1a$ until an energy significant coefficient is encountered. This coefficient is tested for perceptual significance by checking the corresponding entry is vpe, if the coefficient is found to be significant (and $bst1a$ is not full) then the sign bit and further refinement bits are sent to $bst1a$, otherwise these bits are sent to $bst1b$. The perceptual significance test is only applied to individual coefficients and not to whole sets as the energy significance test is. The same process is followed at the decoder as the encoder resulting in equivalent operation at both ends of the coder. Note that the major task of the algorithm is to re-arrange the bitstream produced in a more

perceptually accurate manner. Some extra overhead is encountered in the bitstream formatting as a pointer must also be transmitted indicating the length of $bst1a$, this is necessary for the decoder to be able to divide the total bitstream correctly and to allow $bst1a$ to be less than its hard-coded maximum length should the signal contain fewer significant components than expected. Although the listed algorithm outputs perceptual significance information it does so only for energy significant components and even then only when there is space in $bst1a$, hence it would be rare to encounter a situation where all of v_{pe} is transmitted.

Algorithm PSPIHT:

1) **Initialization:** output $n = \lfloor log_2(max_i|c_i|)\rfloor$;

set the LSP as an empty list, and add the coordinates

$(i, j) \in H$ to the LIP, and only those with descendants

also to the LIS, as type A entries.

Set LPISP as empty set.

2)**Sorting Pass:**

2.1) for each entry (i) in the LIP do:

If $bst1a$ is not full

2.1.1) output $S_n(i)$; to $bst1a$

Else

2.1.1) output $S_n(i)$; to $bst1b$

2.1.2) if $S_n(i) = 1$ and $bst1a$ is not full then output $v_{pe}(i)$ to $bst1a$

else if $bst1a$ is full

do not output $v_{pe}(i)$ move (i) to LPISP and output

the sign of c_i to $bst1b$;

If $v_{pe}(i) = 1$ then move (i) to the LSP

If $bst1a$ is not full output the sign of c_i to $bst1a$;

Else output the sign of c_i to $bst1b$;

Else move (i) to the LPISP and output the sign of c_i to $bst1b$

2.2) for each entry (i) in the LIS do:

2.2.1) if the entry is of type A then

If $bst1a$ is not full

- output $S_n(D(i))$; to $bst1a$

Else

bullet output $S_n(D(i))$; to $bst1b$

- if $S_n(D(i)) = 1$ then

  * for each $(k) \in O(i)$ do:

    If $bst1a$ is not full

    - output $S_n(k)$; to $bst1a$

    Else

    - output $S_n(k)$; to $bst1b$

    - if $S_n(k) = 1$

      If $bst1a$ is not full output $v_{pe}(k)$ to $bst1a$

      Else do not output $v_{pe}(k)$, move (k) to the LPISP

      and output the sign of c_i to $bst1b$

      If $v_{pe}(k) = 1$ then add (k) to the LSP

      If $bst1a$ is not full output the sign of c_k to $bst1a$

      Else output the sign of c_k to $bst1b$

      Else add (k) to the LPISP and output the sign of c_k to $bst1b$

    - if $S_n(k) = 0$ then add (k) to the

      end of the LIP;

  * if $L(i) \neq \emptyset$ then move (i) to the

    end of the LIS as an entry of type B,

    and go to Step **2.2.2);** otherwise, remove

    entry (i) from the LIS;

2.2.2) if the entry is of type B then

If $bst1a$ is not full

- output $S_n(L(i))$ to $bst1a$
- output $S_n(L(i))$ to $bst1b$

- if $S_n(L(i)) = 1$ then
  - add each $(k) \in O(i)$ to the end of the LIS as an entry of type A;
  - remove (i) from the LIS.

3) **Refinement Pass:** for each entry (i) in the LSP,
except those included in the last sorting pass (i.e., with
same n), output the nth most significant bit of $|c_i|$ to $bst1a$ if it is not full,
otherwise output the nth most significant bit of $|c_i|$ to $bst1b$
for each entry (i) in the LPISP,
except those included in the last sorting pass (i.e., with
same n), output the nth most significant bit of $|c_i|$ to $bst1b$

4)**Quantization-Step Update:**decrement n by 1 and go
to step **2.**

6.8 Results for the PSPIHT based scheme

The PSPIHT algorithm was inserted into the lossy section of the scalable to lossless system shown in Figure 6.11. By listening to the synthesized audio it could be heard that the perceptual quality of the scheme at 64 kbps is quite similar to that produced by the MLT-SPIHT scheme with masking described in Chapter 5. This

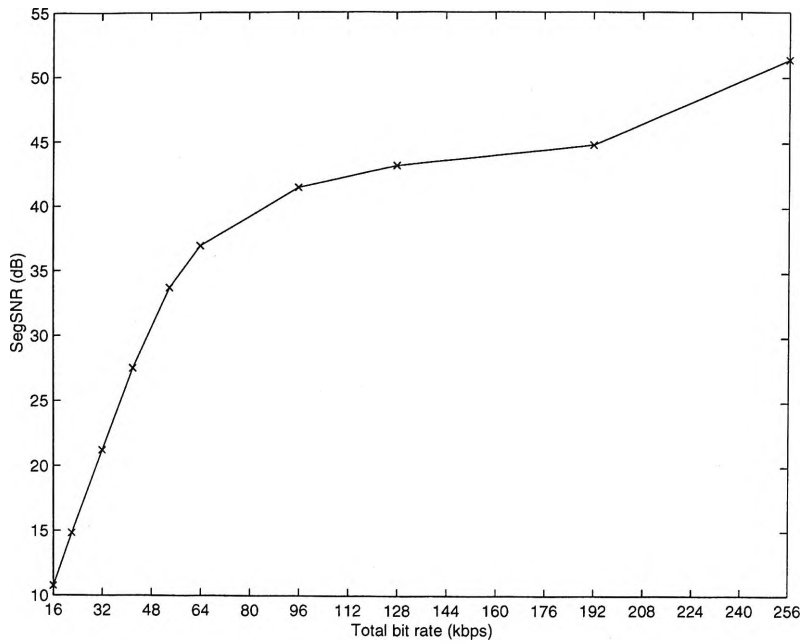

Figure 6.15: Objective results for the PSPIHT algorithm using x1 and a maximum lossy rate of 192 kbps

was the motivation behind developing this algorithm, i.e., obtaining a perceptually similar sound to the original signal at a lower rate than the basic MLT-SPIHT scheme (without masking). Subjective listening tests have also been performed to determine the quality of the PSPIHT-MLT scalable scheme. The objective results of the scheme are presented here first.

6.8.1 Objective PSPIHT-MLT results

Figure 6.15 shows the SegSNR results for the processed signal x1 using PSPIHT at rates from 16 kbps to 192 kbps. The result at 256 kbps is for the whole scheme using a maximum lossy rate of 192 kbps. Comparing Figure 6.15 with Figure 6.14 it can be observed that the SEGSNR of the PSPIHT synthesized signal is lower than that of the SPIHT synthesized signal. This is an expected result as extra bits are being spent

by PSPIHT to inform the decoder of the perceptual significance of frequency domain coefficients. The advantage of PSIHT is that perceptually significant coefficients are transmitted first without resorting to distorting the perceptually insignificant coefficients, effectively a delay is being enforced on the transmission of coefficients that contribute to waveform matching the original signal but do not contribute to the perceptual matching of the original signal. Figure 6.15 also shows the continuous scalability of the proposed scheme is maintained when using PSPIHT in the lossy side of the coder.

The effect of using the PSPIHT algorithm on the lossless rate is given by Table 6.7. The results shown are for the proposed scheme using a maximum lossy rate of 96 kbps. The reason for changing the maximum lossy rate is to minimize the detrimental effect of using PSPIHT on the lossless rate. The reason for the increase in the lossless rate can be deduced from Figure 6.15. The delay that PSPIHT imposes on perceptually insignificant coefficients means that the SegSNR of the synthesized signal is lower which in turn means that the dynamic range of the error is greater resulting in the reduced performance of SPIHT. To counter this effect, some tuning of the maximum lossy rate was carried out and it was determined that a maximum lossy rate of 96 kbps should be used. Table 6.7 shows that an average increase of 27 kbps can be expected, which is an acceptable cost when one considers that, potentially, a maximum of 44.1 kbps can be spent on perceptual significance information.

The increase in the lossless rate varied according to the content of the signal being compressed. Harmonic signals did not suffer as great an increase as inharmonic signals. This can be explained by reflecting on the spectral representation of each type of signal. Harmonic signals have few significant components, both perceptually and

Table 6.7: Lossless compression performance using PSPIHT

Signal	Mean Rate (kbps)	Compression Ratio	Bits per sample	Δ Rate (kbps)
x1	349	2.02	7.9	31
x2	134	5.27	3.0	0
x3	224	3.15	5.1	18
x4	305	2.31	6.9	39
x5	393	1.80	8.9	47
x6	244	2.89	5.5	12
x7	392	1.80	8.9	38
x8	347	2.03	7.9	30
x9	405	1.74	9.2	39
x10	422	1.67	9.6	17
x11	385	1.83	8.7	23
x12	396	1.78	9.0	28
x13	376	1.88	8.5	16
x14	387	1.82	8.8	27
x15	296	2.39	6.7	41
x16	331	2.13	7.5	25

in terms of energy. Noise like signals have significantly less perceptually significant components than energy significant components. This conclusion may be drawn from results presented in Chapter 4. As such, the residual signal produced for inharmonic signals maintains some significant energy components resulting in a higher dynamic range and more bits for lossless compression. However, the results presented still show a mean compression ratio of 2.29:1 and a compression ratio for most files which is competitive with the state of the art.

6.8.2 Subjective PSPIHT-MLT results

Informal listening tests have been conducted to compare the proposed scheme to the MPEG-4 AAC coder as well as to obtain an idea of the perceptual scalability of the PSPIHT-MLT scheme. The tests involved a total of 39 subjects. The subjects were

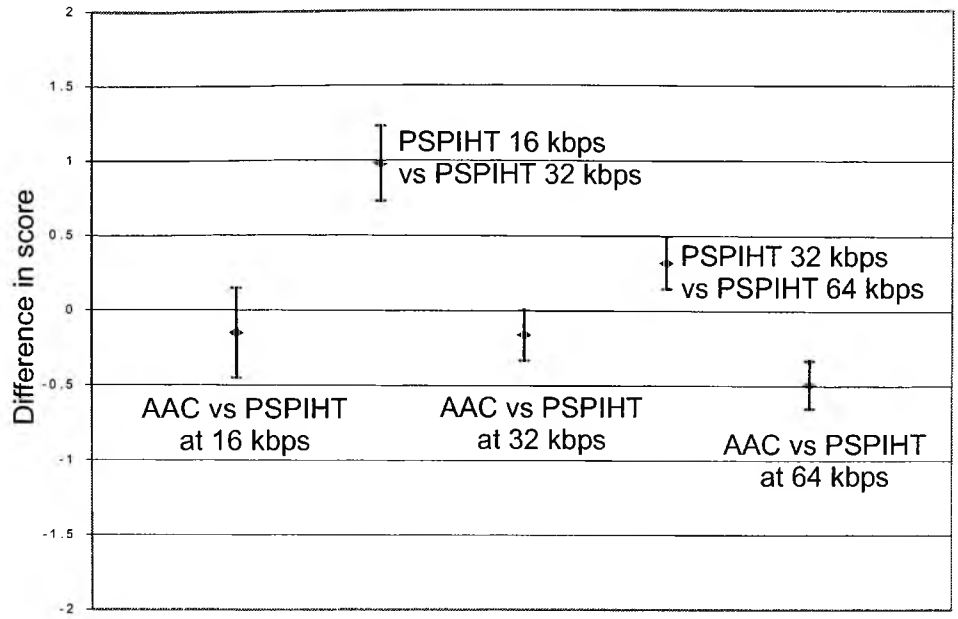

Figure 6.16: Listening test results

asked to listen to two differently coded versions of the same sound (labelled A and B) and indicate which version was more preferable. A scale of 1 to 5 was used as indicated in Table 6.8. The comparison tests involved comparing the AAC coder and the PSPIHT coder at 16, 32 and 64 kbps. Also tested was the PSPIHT coder at lower rates against the PSPIHT coder at higher rates. Specifically, the PSPIHT coder at 16 kbps has been compared to the PSPIHT coder at 32 kbps which in turn has been compared to the PSPIHT coder at 64 kbps.

The mean scores for all the comparisons have been obtained as well as the difference between the mean scores and the value 3. Fig. 6.16 shows the results with their 95% confidence intervals. In the results shown in Fig. 6.16 a score that is below 0 indicates that the first coder mentioned in the label is better than the second one according to the scale used in the test. For example, the first data point plotted is below 0 indicating that, on average, the AAC coder is better than the PSPIHT coder

Table 6.8: The subjective comparison scale used

Score	Subjective opinion
1	A much better than B
2	A better than B
3	A and B are the same
4	A worse than B
5	A much worse than B

(both at 16 kbps) within the shown 95% confidence interval.

The subjective results shown in Fig. 6.16 indicate that the AAC coder and the proposed PSPIHT-MLT coder perform quite similarly at 16 and 32 kbps (with 95% confidence). The AAC outperforms the PSPIHT-MLT scheme at 64 kbps by a margin of 0.49 indicating a small overall difference in quality. The perceptual scalability of the PSPIHT-MLT scheme is clearly shown by the presented results with a clear difference between the 16 kbps coder and the 32 kbps coder, as well as a clear (although reduced) difference between the 64 kbps coder and the 32 kbps coder.

6.9 Conclusions

This chapter has presented a study into extending the MLT-SPIHT compression scheme into a scalable to lossless compression scheme. This led to the proposal of a scalable to lossless scheme that applies SPIHT twice. This chapter also introduced a new perceptually based set sorting algorithm, PSPIHT. The investigation into lossless compression showed that the basic MLT-SPIHT can achieve lossless compression, however, the required bandwidth is not competitive with the state of the art. It was shown that the combination of the MLT-SPIHT scheme with an entropy code would produce competitive lossless results, however, the entropy scheme has to be scalable

to achieve the set objectives.

The use of integer transforms has also been investigated. It was shown that whilst integer transforms can lead to lossless compression, the rates achieved were also not competitive with the state of the art. Other drawbacks of the integer transforms, such as reduced frequency selectivity, also compromised the performance of the algorithm at low rates. There is potential for the application of such transforms in audio compression because of the complexity gains that can be made, however, in this particular application, the increase in dynamic range that these transforms introduced and the reduced frequency selectivity meant that the integer transforms did not perform well.

The proposed scalable to lossless MLT-SPIHT scheme has delivered good lossless compression results. The scalability of the scheme has been shown to be objective in nature. This is because perceptual considerations were not accounted for. In order to account for such perceptual considerations, PSPIHT has been proposed. This algorithm allows the delay in transmission of frequency components that have zero perceptual entropy but are significant in terms of energy. This algorithm increases the lossless compression cost slightly but does introduce the advantage of scaling smoothly from a perceptually lossless representation to an objectively lossless representation. Subjective listening tests have indicated that the PSPIHT-MLT scheme performs comparably with the AAC coder at 16, 32 and 64 kbps. The advantage of the PSPIHT-MLT and SPIHT-MLT scheme is that given a higher rate bit stream any lower rate bit stream can be extracted from that bit stream. For example, the 16 kbps and 32 kbps bit streams can both be extracted from the 64 kbps bit stream. This is certainly advantageous when a variable rate channel is being used, as is the case in Internet applications.

Chapter 7

Conclusions and future work

7.1 Conclusions

This thesis has been aimed at the study of scalable and perceptual audio compression. A number of schemes and algorithms have been presented which aim to achieve scalable perceptual compression. The development of such schemes has given rise to a number of conclusions, these conclusions will be considered in respect of each of the chapters.

Chapter 2 focused on the signal processing theory and psychoacoustic concepts that have been applied or have been found of some relevance to audio compression. The focus in that chapter was on concepts that were related to the work presented in later chapters. Also, in Chapter 2, a new technique for analyzing audio and speech coders was presented. This technique is based on the relative roughness, loudness, sharpness and tonality of the synthesized signal to the original signal. In a way this is similar to the approach that recent objective measures such as PEAQ have taken. The use of these psychoacoustic parameters was demonstrated by example to provide some insight into the operation of the coder being analyzed. Such an analysis scheme

aids in determining how to counter the perceptual distortion that has been introduced by the coder. For example, if a coding scheme produced a coded signal that was not as rough as the original but had similar loudness one may conclude that the envelope of the synthesized signal should have a higher modulation frequency.

Chapter 3 summarized related literature about perceptual and scalable audio compression. The field of perceptual audio compression was seen to have advanced considerably in the past one and a half decades with a number of significant audio compression standards. It can also be concluded from the presented material that there are a number of areas of research in audio compression that still require some attention. Effective scalability is one of the issues that still requires some work; testament to this is the standardization by MPEG of a number of coding schemes as objects in a scalable coder (MPEG-4). Signal modelling is also still a rich area of research, even though current and recent models have produced good results. The majority of the literature in perceptual audio compression has been found to be based on transform coding concepts with parametric coding schemes becoming more popular of late because of the push for lower rate audio. However, it was also noted that high rate compression for lossless applications is drawing increased attention. Fundamentally, whilst compression will continue to be commercially important, the increases in bandwidth availability across the Internet and for wireless communications has meant that high rate coders are once again being considered but this time these would be used to produce an exact copy of the original signal.

The first compression scheme proposed in this thesis appears in Chapter 4. This coding scheme was based on the paradigm of sorting sinusoidal parameters to obtain both perceptual and scalable compression. The sorting of the sinusoids allowed the

development of novel quantization schemes such as spline interpolation between transmitted amplitudes and weighted phase quantization. Phase components were found to be of limited significance. Specifically, it was found that phase components belonging to highly significant sinusoidal components should be quantized whilst other phase components may be ignored. It is acknowledged that it may not be necessary to explicitly quantize the phase, rather implicit quantization such as the inclusion of the envelope of the signal will provide equivalent results. The sinusoidal coder presented was implemented as both a variable rate compression scheme and a scalable compression scheme. Both schemes provided promising results at around the 40 kbps mark.

Further, the scalable sinusoidal scheme developed was compared to the MPEG-4 AAC coder. The comparison showed that whilst the MPEG-4 AAC scheme slightly outperformed the proposed coder overall, the sinusoidal scheme clearly performed better for harmonic signals. This conclusion is drawn from the subjective test results obtained and presented in Chapter 4. The objective psychoacoustic results obtained showed that the scalable sinusoidal scheme scales in terms of differences in loudness, sharpness and roughness whereas the MPEG-4 AAC scheme does not show the same scalable behavior. This is most likely the result of coding objects, such as noise shaping, that are employed in the MPEG-4 scheme. This result also raises the potential of adopting a systematic approach to reducing the psychoacoustic pleasantness factor error (for each of the factors used) in the scalable sinusoidal coder. Also, this result adds weight to the psychoacoustic analysis technique that was proposed as it gives a different perspective on modelling an audio signal by the use of both harmonic and noise components.

The main disadvantage of the scalable sinusoidal scheme was found to be the limitation in the granularity of the scheme. Since the definition of the scalable rates must be carried out off-line, it is difficult to envisage defining the scalable rates such that there is only a single bit of difference. Because of this limitation, the work presented shifted to incorporating SPIHT into audio coding schemes. SPIHT allows fine grain scalability without many off-line definitions and so Chapter 5 focused on studying how SPIHT should be used to achieve scalable and perceptual compression. The results presented in that chapter showed that the MLT combined with the SPIHT and a masking model produces significant compression with good audio quality. The subjective tests conducted and reported can be used to make three conclusions. Firstly, it was shown that ignoring signal components with zero perceptual entropy introduced, at worst, negligible perceptual distortion. Secondly it was shown that good quality audio synthesis can be obtained by the use of the MLT-SPIHT scheme (with the proposed set definitions) and, finally, that the scalability of the scheme was clear perceptually at the rates tested. The introduction of the masking model, and the way that it was used, resulted in a need to modify SPIHT to maintain its compression performance. The modification introduced was shown to help reduce the mean bit rate.

The MLT-SPIHT coder has a number of advantages. It is a low delay scalable scheme that allows scalability down to a single bit per frame. The scheme allows the variation in bit rate from frame to frame, this is very useful in applications such as internet transmission where a variable bit rate is the norm. The scheme is also scalable in terms of the psychoacoustic measures already discussed, as shown by the results presented in Chapter 6.

Chapter 6 extended the MLT-SPIHT scheme so that it could achieve lossless compression. The scheme adopted is actually a double application of SPIHT in that a residual signal is produced that is also transmitted by the use of SPIHT. The results presented showed that this is a more appropriate approach than attempting to reach lossless levels without coding an error signal. It was, however, also shown that the MLT-SPIHT scheme is capable of reaching lossless compression given a high enough bit rate (that is lower than the original rate) and coding resolution of the MLT coefficients. Chapter 6 also presented a study of the application of integer to integer transforms to lossless compression. The results presented showed that the trade-off between dynamic range increase and frequency selectivity is very difficult to advantageously balance. An increase in an integer transforms' frequency selectivity will mean an increase in the dynamic range of the transformed sequence; this hinders the ability of SPIHT to compress the audio signal in a lossless manner.

Whilst the scalable to lossless scheme presented produced good results, it did not incorporate masking in any form. This observation led to the development of the Perceptual SPIHT algorithm which introduces more bitstream formatting into the application of SPIHT. This allows the transmission of perceptually significant coefficients first whilst not corrupting the other coefficients to a point where the recovery of the original signal losslessly becomes more difficult. The difficulty in obtaining good lossless compression results when the lossy coder used only contains perceptually significant coefficients can be appreciated more when the results of Chapter 4 are taken into consideration. In that chapter the SegSNR of the perceptually identical signal to the original was shown to barely reach 30 dB, indicating a significant amount of spectral color in the residual signal. This makes the compression of the residual signal

more difficult and so reduces the level of compression achieved. The PSPIHT-MLT scheme resulted in similar perceptual quality to the MLT-SPIHT with masking, and maintained the ability to scale smoothly to lossless compression. The compression ratios achieved were, as expected, not as high as the compression ratios of the MLT-SPIHT scheme without masking with an average increase of approximately 27 kbps. It is notable that the increase in bit rate depended on the content of the signal with tonal signals suffering a much lower increase than noise-like signals. This is to be expected as tonal signals have perceptually significant components that are also energy significant whilst noise-like signals tend to have energy significant components that are not perceptually significant.

Overall, this thesis has provided a number of promising solutions to scalable perceptual compression and a novel scalable to lossless scheme that has produced good results.

7.2 Future work

There are a number of directions in which the presented work may be taken. As a starting point, lets consider the analysis of audio coders by the use of the mentioned psychoacoustic measures. This work can be developed to provide a reliable perceptual error criteria that allows the selection of signal components that minimize this global psychoacoustic error. That is, a weighting function would be developed (from psychoacoustic data) for the relative significance of each measure, signal components would be extracted in an iterative manner continuously decreasing the global error criteria. Although there have been approaches to audio compression along these lines in the past, this effort would be directed towards the development of more effective

signal models.

The development of more effective signal models is inherently related to the scalable sinusoidal coder, which has been observed to introduce a consistent form of psychoacoustic distortion for a given audio signal. A natural extension to this work would be the addition of psychoacoustic distortion reduction techniques that would result in increased perceptual quality at lower rates. The sinusoidal extraction can also be improved through the extraction of sinusoids according a more sophisticated error criterion.

With regards to the MLT-SPIHT coding scheme, better quality at lower rates remains an illusive aim. The difficulty arises from the trade-off that must be made with regards to the fine granular scalability and quality. The most obvious direction of extending this work is to incorporate a noise model into the coder whilst maintaining the scalability. The challenge is to maintain the gain that each bit contributes to the quality of the synthesized audio. It is possible that this problem can be approached from a hierarchical perceptual perspective, where perceptually significant sets are defined rather than harmonically related sets. This would lead to perceptual set sorting where the most perceptually significant set would be given preference over the other sets. The hurdle to overcome here is the definition of the sets on a frame to frame basis, as perception is very much related to the signal statistics which tend to change over short periods of time. The solution may lie in a backward adaptive approach where the sets are backward adapted at both encoder and decoder to better match the perceptual content of the signal. The implementation of such a scheme should provide a better SPIHT performance.

The scalable to lossless work can also be improved upon in terms of lossless compression and better performance at the lower rates. This work can be expanded to include scalability between different sampling rates. The solution to the sampling rate (and thus bandwidth) scalability can be developed in a similar manner to the MPEG-4 scalable solution, namely the coding of consecutive base and residual layers. Scalability can be achieved by coding the difference between the interpolated base layer signal and the higher bandwidth sampled signal. As a side issue, integer transforms should be more comprehensively studied for audio applications as they provide significant complexity reduction, freeing system resources for the implementation of much more complex compression algorithms.

As a final future direction of work stemming from this thesis, the overall scalability issue can be enhanced to include multichannel and scene creation scalability. That is scalable audio compression should progress from this point towards the development of a multichannel compression algorithm that produces an embedded bitstream allowing a smooth transition from basic scene recreation to complex scene recreation. The MLT-SPIHT scheme can be used as a starting point to the development of such complex compression systems.

Bibliography

[ACRG99] A. Aggarwal, V. Cuperman, K. Rose, and A. Gersho. *Perceptual zerotrees for scalable wavelet coding of wideband audio.* PROCEEDINGS OF 1999 IEEE WORKSHOP ON SPEECH CODING, pages 16–18, June 1999.

[AK00] M.D. Adams and F. Kossentini. *Reversible Integer to integer wavelet transforms for image compression: Performance evaluation and analysis.* IEEE TRANSACTIONS ON IMAGE PROCESSING, 9(6):1010–24, June 2000.

[AKY+99] K. Akagiri, M. Katakura, H. Yamouchi, E. Saito, M. Kokut, M. Nishiguchi, and K. Tsutsui. *Sony systems.* In V. K. Madisetti and D. B. Williams, editors, DIGITAL SIGNAL PROCESSING HANDBOOK, chapter 43. CRC Press, Boca Raton, 1999.

[and] *Australian National Database of Spoken Langauge* (ANDOSL). CD-ROM.

[AR91] H. Anton and C. Rorres. ELEMENTARY LINEAR ALGEBRA. Wiley, Singapore, 6 edition, 1991.

[BBT96] R. J. Beaton, M. Beerends, J. G. Kehyl, and W. C. Treurniet. *Objective Perceptual Measurement of Audio Quality.* In N. Gilchrist and

C. Grewin, editors, COLLECTED PAPERS ON DIGITAL AUDIO BIT-RATE REDUCTION, pages 126–152. AES, USA, 1996.

[BD98] S. Boland and M. Deriche. *Hybrid LPC and discrete wavelet transform audio coding with a novel bit allocation algorithm.* PROCEEDINGS OF ICASSP-98, pages 3657–3660, May 1998.

[Beu84] K.G. Beuchamp. APPLICATIONS OF WALSH AND RELATED FUNCTIONS. Academic Press, London, UK, 1984.

[BHJ+91] K. Brandenburg, J. Herre, J. D. Johnston, Y. Mahieux, and E. Schroeder. *ASPEC: adaptive spectral entropy coding of high quality music signals.* In PROCEEDINGS OF THE 90TH AES CONVENTION, *AES preprint 3011*, February 1991.

[BJ90] K. Brandenburg and J. D. Johnston. *Second generation perceptual audio coding: the hybrid coder.* PROCEEDINGS OF THE 88TH AES CONVENTION, *AES preprint 2937*, March 1990.

[BKS00] K Brandenburg, O Kunz, and A Sugiyama. MPEG-4 *natural audio coding.* SIGNAL PROCESSING: IMAGE COMMUNICATION, 15(4):423–444, January 2000.

[BOvdV96] A.A.M.L. Bruekers, W.J. Oomen, and R.J. van der Vleuten. *Lossless coding for* DVD *audio.* AES 101st convention, AES preprint 4358, November 1996.

[CDSY96] A.R. Calderbank, I. Daubechies, W. Sweldens, and B-L. Yeo. *Wavelet transforms that map integers to integers.* Bell-labs technical report, August 1996.

[CG90] W. Y. Chan and A. Gersho. *High fidelity audio transform coding with vector quantization.* PROCEEDINGS OF ICASSP-90, pages 1109–1112, May 1990.

[Cha89] W-k. Cham. *Development of integer cosine transforms by the principle of dyadic symmetry.* IEE PROCEEDINGS, 136(4):276–282, August 1989.

[Cho73] J. Chowing. *The synthesis of complex audio spectra by means of frequency modulation.* JOURNAL OF THE AES, 21(7):526–529, September 1973.

[CL97] P.G. Craven and M.J. Law. *Lossless compression using* IIR *prediction filters.* AES 102nd convention, AES preprint 4415, March 1997.

[CL01] P. Chang and J. Lin. *Scalable embedded zero tree wavelet packet audio coding.* PROCEEDINGS OF 2001 IEEE THIRD WORKSHOP ON SIGNAL PROCESSING ADVANCES IN WIRELESS COMMUNICATIONS, pages 384–387, March 2001.

[Cox95] R. V. Cox. *Speech Coding Standards.* In W.B. Kleijn and K. K. Paliwal, editors, SPEECH CODING AND SYNTHESIS, pages 49–78, Netherlands, 1995. Elsevier Science.

[CT91] T.M. Cover and J.A. Thomas. ELEMENTS OF INFORMATION THEORY. Wiley, USA, 1991.

[CW96] W. Chang and C. Want. *A masking threshold adapted weighting filter for excitation search.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 4(2):124–132, March 1996.

[Dav95] P. Davis. *A tutorial on MPEG/Audio compression.* IEEE MULTIMEDIA MAGAZINE, 2(2):60–74, Summer 1995.

[Dav99] G.A. Davidson. Digital Audio Coding: Dolby AC-3, chapter 41. CRC Press LLC, 1999.

[DLU91] Y. F. Dehery, M. Lever, and P. Urcun. *A MUSICAM source codec for digital audio broadcasting and storage.* PROCEEDINGS OF ICASSP-91, pages 3605–3608, May 1991.

[DPH93] J. R. Deller, J. G. Proakis, and J. H. L. Hansen. DISCRETE TIME PROCESSING OF SPEECH SIGNALS. Macmillan publishing company, USA, 1993.

[DS97] I. Daubechies and W. Sweldens. *Factoring wavelet transforms into lifting steps.* Bell labs, Lucent tech. Technical report, November 1997.

[DWJ00] H. Dongmei, G. Wen, and W. Jiangqin. *Complexity scalable audio coding algorithm based on wavelet packet decomposition.* PROCEEDINGS OF WCC2000-ICSP2000, pages 659–665, August 2000.

[EEM+97] G. W. Elko, S. J. Elliot, S. Makino, J. M. Kates, M. Bosi, J. O. Smith III, and M. Kahrs. *The Past, Present, and Future of Audio Signal Processing.* IEEE SIGNAL PROCESSING MAGAZINE, 14(5):30–57, September 1997.

[EP98] B. Edler and H. Purnhagen. *Concepts of hybrid audio coding schemes based on parametric techniques.* PROCEEDINGS OF THE 105TH AES CONVENTION, *AES preprint 4808*, 1998.

[EP00] B. Edler and H. Purnhagen. *Parametric Audio Coding.* In PROCEEDINGS OF THE FIFTH INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING, volume 1, pages 21–24, 2000.

[EV99] J. Eberspacher and H.-J. Vogel. GSM SWITCHING, SERVICES AND PROTOCOLS. Wiley, Chichester, 1999.

[FBD+96] L. D. Fielder, M. Bosi, G. Davidson, M. Davis, C. Todd, and S. Vernon. *AC-2 and AC-3: Low complexity transform based audio coding.* In N. Gilchrist and C. Grewin, editors, COLLECTED PAPERS ON DIGITAL AUDIO BIT-RATE REDUCTION, pages 54–72. AES, USA, 1996.

[GG92] A. Gersho and R. M. Gray. VECTOR QUANTIZATION AND SIGNAL COMPRESSION. Kluwer Academic publishers, USA, 1992.

[GHKB02] R. Geiger, J. Herre, J. Koller, and K. Brandenburg. *INTMDCT - A link between perceptual and lossless audio coding.* PROCEEDINGS OF ICASSP-2002, 2:1813–1816, May 2002.

[Goo97] M. M. Goodwin. ADAPTIVE SIGNAL MODELS: THEORY, ALGORITHMS, AND AUDIO APPLICATIONS. PhD thesis, University of California, Berkeley, California, Fall 1997.

[Goy00] V.K. Goyal. *Transform coding with integer-to-integer transforms.* IEEE TRANSACTIONS ON INFORMATION THEORY, 46(2):465–473, March 2000.

[GPS94] S.W. Golomb, R.E. Peile, and R.A. Scholtz. BASIC CONCEPTS IN INFORMATION THEORY AND CODING. Plenum press, NY, 1994.

[GS92] E.B. George and J.T. Smith. *Analysis-by-Synthesis/Overlap-add sinusoidal coding applied to the analysis and synthesis of musical tones.* JOURNAL OF THE AUDIO ENGINEERING SOCEITY, 40(6):497–516, June 1992.

[GTA00] C.D. Giurcaneanu, I. Tabus, and J. Astola. *Adaptive context-based sequential prediction for lossless audio compression.* SIGNAL PROCESSING, 80(11):2283–2294, November 2000.

[Hah96] S.L. Hahn. *Hilbert transforms.* In A. D. Poularikas, editor, THE TRANSFORMS AND APPLICATIONS HANDBOOK, chapter 7. CRC Press, USA, 1996.

[Hal99] J.L. Hall. *Auditory Psychophysics for coding applications.* In V. K. Madisetti and D. B. Williams, editors, DIGITAL SIGNAL PROCESSING HANDBOOK, chapter 39. CRC Press, Boca Raton, 1999.

[Har98] W.M. Hartman. SIGNALS, SOUND, AND SENSATION. Springer, New York, USA, 1998.

[HAT96] K. Hamdy, M. Ali, and A. Tewfik. *Low bit rate high quality audio coding with combined harmonic and wavelet representations.* PROCEEDINGS OF ICASSP-96, pages 1045–1048, May 1996.

[HJ97] J. Herre and J.D. Johnston. *Continuosly signal adaptive filterbank for high quality perceptual audio coding.* PROCEEDINGS OF IEEE ASSP WORKSHOP ON APPLICATIONS OF SIGNAL PROCESSING TO AUDIO AND ACOUSTICS, pages 4–7, 1997.

[HK00] M. Hansen and B. Kollmeier. *Objective modeling of speech quality with a psychoacoustically validated auditory model.* JOURNAL OF THE AUDIO ENGINEERING SOCIETY, 48(5):395–409, May 2000.

[HLK96] A. Harma, U. Laine, and M. Karjalainen. *Warped Linear Prediction (WLP) in audio coding.* PROCEEDINGS OF NORSIG-96, pages 367–370, September 1996.

[HS63] J.J.Y. Huang and P.M. Schultheiss. *Block quantization of correlated gaussian random variables.* IEEE TRANSACTIONS ON COMMUNICATIONS SYSTEMS, 11(3):289–296, September 1963.

[HS01] M. Hans and R.W. Schafer. *Lossless compression of digital audio.* IEEE SIGNAL PROCESSING MAGAZINE, 18(4):21–32, July 2001.

[IMM95] N. Iwakami, T. Moriya, and S. Miki. *High-quality audio coding at less than 64 kbps by using transform domain interleave vector quatization* (TWINVQ). PROCEEDINGS OF ICASSP-95, pages 3095–3098, May 1995.

[ITN02] M. Ikehara, T.D. Tran, and T.Q. Nguyen. *A family of lapped regular transforms with integer coefficients.* IEEE TRANSACTIONS ON SIGNAL PROCESSING, 50(4):834–841, April 2002.

[JK00a] A. Jbira and A. Kondoz. *Multi-layer scalable LPC audio format.* PROCEEDINGS OF ISCAS-2000, pages 209–212, MAY 2000.

[JK00b] A. Jbira and A. Kondoz. *Multiple frequency harmonics analysis and synthesis of audio signals.* PROCEEDINGS OF ICASSP-2000, pages 873–876, June 2000.

[JMIM01] A. Jin, T. Moriya, N. Iwakami, and S. Miki. *Scalable audio coding based on hierarchical transform coding modules (translated version).* ELECTRONICS AND COMMUNICATIONS IN JAPAN, 84(8):34–45, 2001.

[JMN+99] A. Jin, T. Moriya, T. Norimatsu, M. Tsushima, and T. Ishikawa. *Scalable audio coder based on quantizer units of* MDCT *coefficients.* In PROCEEDINGS OF ICASSP-99, pages 897–900, March 1999.

[Joh88a] J. D. Johnston. *Transform Coding of Audio Signals Using Perceptual Noise Criteria.* IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, 6(2):314–323, February 1988.

[Joh88b] J.D. Johnston. *Estimation of perceptual entropy using noise masking criteria.* PROCEEDINGS OF ICASSP-88, 5:2524–2527, 1988.

[Joh89] J.D. Johnston. *Perecptual transform coding of wideband stereo signals.* PROCEEDINGS OF ICASSP-89, 3:1993–1996, 1989.

[JPC89] T. E. Tremain J. P. Campbell, V. C. Welch. *An expandable error protected 4800bps* CELP *coder (US Federal Standard 4800 bps voice coder).* In PROCEEDINGS OF ICASSP 89, volume 2, pages 735–738, USA, 1989.

[JSDQ96] D. Johnston, D. Sinha, S. Dorward, and S. R. Quackenbush. AT&T *perceptual audio coding (pac).* In N. Gilchrist and C. Grewin, editors, COLLECTED PAPERS ON DIGITAL AUDIO BIT-RATE REDUCTION, pages 73–82. AES, USA, 1996.

[KCG00] K. Koishida, V. Cuperman, and A. Gersho. *A 16 kbit/s bandwidth scalable audio coder based on the G.729 standard.* PROCEEDINGS OF ICASSP-2000, pages 149–152, June 2000.

[KP95] W.B. Kleijn and K. K. Paliwal. *An introduction to speech coding.* In W.B. Kleijn and K. K. Paliwal, editors, SPEECH CODING HANDBOOK, chapter 1. Elsevier, Netherlands, 1995.

[KS96] P. E. Kudumakis and Sandler. *Wavelet packet based scalable audio coding.* PROCEEDINGS OF ICASSP-96, pages 41–44, May 1996.

[Lev98] S.N. Levine. AUDIO REPRESENTATIONS FOR DATA COMPRESSION AND COMPRESSED DOMAIN PROCESSING. PhD thesis, Department of electrical engineering, Stanford university, December 1998.

[LJ02] J. Li and J. D. Johnston. *A progressive to lossless embedded audio coder (*PLEAC*) with multiple factorization reversible transform. ISO/IEC JTC1/SC29/WG11 M9136,* December 2002.

[LKP00] Z. Lu, D. Y. Kim, and W. A. Pearlman. *Wavelet Compression of ECG Signals by the SET Partitioning in Hierarchical Trees Algorithm.* IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, 47(7):849–856, July 2000.

[LP98] Z. Lu and W. A. Pearlman. *An Efficient, Low-Complexity Audio Coder Delivering Multiple Levels of Quality For Interactive Applications.* In 1998 IEEE SECOND WORKSHOP ON MULTIMEDIA SIGNAL PROCESSING, pages 529–534, 1998.

[LPN97] T. Liebchen, M. Purat, and P. Noll. *Lossless Transform coding of audio signals.* PROCEEDINGS OF THE 102ND AES CONVENTION, *AES preprint 4414*, March 1997.

[Mal92] H. S. Malvar. SIGNAL PROCESSING WITH LAPPED TRANSFORMS. Artec House, Inc., Boston, 1992.

[MC89] J. Mahieux, Y. Petit and A. Charbonnier. *Transform coding of audio signals using correlation between successive transform blocks.* PROCEEDINGS OF ICASSP-89, pages 2021–2024, May 1989.

[Mer99] A. Mertins. SIGNAL ANALYSIS. Wiley, England, 1999.

[MH88] T. Moriya and M. Honda. *Transform coding of speech using a weighted vector quantizer.* IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, 6(2):425–431, February 1988.

[MIJM00] T. Moriya, N. Iwakami, A. Jin, and T. Mori. *A design of lossy and lossless audio coding.* PROCEEDINGS OF ICASSP-2000, pages 889–892, June 2000.

[Moo89] B. C. J. Moore. AN INTRODUCTION TO THE PSYCHOLOGY OF HEARING. Academic Press, San Deigo, third edition, 1989.

[Moo96] B. C. J. Moore. *Masking in the human auditory system.* In N. Gilchrist and C. Grewin, editors, COLLECTED PAPERS ON DIGITAL AUDIO BIT-RATE REDUCTION, pages 9–19. AES, USA, 1996.

[Moo97] B. C. J. Moore. AN INTRODUCTION TO THE PSYCHOLOGY OF HEARING. Academic press, London, 4 edition, 1997.

[Mor02] T. Moriya. *Report of* AHG *on Issues in lossless audio coding.* ISO/IEC JTC1/SC29/WG11 M7955, March 2002.

[mpe] Mpeg web site at http://www.tnt.uni-hannover.de/project/mpeg/audio.

[MQ95] R.J. McAulay and T.F. Quatieri. *Sinusoidal coding.* In W.B. Kleijn and K. K. Paliwal, editors, SPEECH CODING AND SYNTHESIS, chapter 4. Elsevier publishing, Netherlands, 1995.

[ND01] D. Ning and M. Deriche. *Wideband audio compression using a combined wavelet and* WLPC *representation.* PROCEEDINGS OF ISSPA-2001, pages 711–714, August 2001.

[NHD98] J. Nieuwenhuijse, R. Heusdens, and E.F. Deprettere. *Robust exponential modeling of audio signals.* PROCEEDINGS OF ICASSP-98, pages 3581–3584, May 1998.

[NJ84] P. Noll and N. S. Jayant. DIGITAL CODING OF WAVEFORMS. Prentice-Hall, USA, 1984.

[NK00] H. Najafzadeh and P. Kabal. *Perceptual bit allocation for low rate coding of narrowband audio.* PROCEEDINGS OF ICASSP-2000, pages 893–896, June 2000.

[Nol97] P. Noll. MPEG *digital audio coding.* IEEE SIGNAL PROCESSING MAGAZINE, 14(5):59–81, September 1997.

[PD00] S-C. Pei and J-J Ding. *The integer transforms analogous to discrete trigonometric transforms.* IEEE TRANSACTIONS ON SIGNAL PROCESSING, 48(12):3345–3364, December 2000.

[PEF98] H. Purnhagen, B. Edler, and C. Ferekidis. *Object based analysis/synthesis audio coder for very low bit rates.* PROCEEDINGS OF THE 104TH AES CONVENTION, *AES preprint 4747*, May 1998.

[PJ95] J. Princen and J.D. Johnston. *Audio coding with signal adaptive filterbanks.* PROCEEDINGS OF ICASSP-95, 5:3071–3074, 1995.

[PM95] M. Paraskevas and J. Mourjopoulos. *A differential perceptual audio coding method with reduced bit rate requirements.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 3(6):490–503, November 1995.

[PM00] H. Purnhagen and N. Miene. HILN - *the* MPEG-*4 parametric audio coding tools.* In PROCEEDINGS OF ISCAS 2000, volume 3, pages 201–204, 2000.

[PN96] M. Purat and P. Noll. *Audio coding with a dynamic wavelet packet decomposition based on frequency varying modulated lapped transforms.* PROCEEDINGS OF ICASSP-96, pages 1021–1024, May 1996.

[Pri95] J. Prieto. *Demonstration of Telecommunication Industry Speech Compression Algorithms.* website at www.lincom-asg.com, October 1995.

[PS00] T. Painter and A. Spanias. *Perceptual Coding of Digital Audio.* PROCEEDINGS OF THE IEEE, 88(4):451–513, April 2000.

[PS01] T. Painter and A. Spanias. *Perceptual component selection in sinusoidal coding of audio.* PROCEEDINGS OF 2001 IEEE WORKSHOP ON MULTIMEDIA SIGNAL PROCESSING, pages 187–192, October 2001.

[PSP96] A. Pena, C. Serantes, and N. Prelcic. *ARCO (Adaptive Resolution Codec): A hybrid approach to perceptual audio coding.* PROCEEDINGS OF THE 100TH AES CONVENTION, *AES preprint 4178*, May 1996.

[PWS96] W. J. Pielemeier, G. H. Wakefield, and M. H. Simoni. *Time Frequency analysis of musical signals.* PROCEEDINGS OF THE IEEE, 84(9):1216–1230, September 1996.

[QJ97] S.R. Quackenbush and J.D. Johnston. *Noiseless coding of quantized spectral components in* MPEG-2 *advanced audio coding.* PROCEEDINGS OF IEEE WASSP, pages 1–4, 1997.

[Qui01] T. Qui. *Lossless audio coding based on high order context modeling.* PROCEEDINGS OF 2001 IEEE FOURTH WORKSHOP ON MULTIMEDIA SIGNAL PROCESSING, pages 575–580, October 2001.

[Ram95] R.P. Ramachandran. The Use of pitch prediction in speech coding, chapter 1, pages 3–22. Kluwer Academic Publishers, Boston, USA, 1995.

[Ram99] R.P. Ramachandran. *Quantization of discrete time signals.* In V. K. Madisetti and D. B. Williams, editors, DIGITAL SIGNAL PROCESSING HANDBOOK, chapter 6. CRC Press, Boca Raton, 1999.

[RB01] M. Raad and I. Burnett. *Audio coding using sorted sinusoidal parameters.* In PROCEEDINGS OF ISCAS2001, volume 2, pages 401–404, 2001.

[RBM01] M. Raad, I. Burnett, and A. Mertins. *Scalable Audio Coding Employing Sorted Sinusoidal Parameters.* In PROCEEDINGS OF THE SIXTH INTERNATIONAL SYMPOSIUM ON SIGNAL PROCESSING AND ITS APPLICATIONS 2001 (ISSPA2001), volume 1, pages 174–177, 2001.

[Ric00] K.W. Richardson. UMTS *overview.* ELECTRONICS AND COMMUNICATION ENGINEERING JOURNAL, 12(3):93–100, June 2000.

[RMB02a] M. Raad, A. Mertins, and I. Burnett. *A scalable to lossless audio compression scheme.* In PROCEEDINGS OF WITSP2002, pages 167–172, 2002.

[RMB02b] M. Raad, A. Mertins, and I. Burnett. *Audio Coding Based on the Modulated Lapped Transform (MLT) and Set Partitioning In Hierarchical Trees.* In PROCEEDINGS OF SCI2002, volume 3, pages 303–306, 2002.

[RMB02c] M. Raad, A. Mertins, and I. Burnett. *Audio Compression Using the* MLT *and* SPIHT. PROCEEDINGS OF DSPCS 02, pages 128–132, 2002.

[RMB03] M. Raad, A. Mertins, and I. Burnett. *Scalable to lossless audio compression based on perceptual set partitioning in hierarchical trees (PSPIHT).* In PROCEEDINGS OF ICASSP03, volume 5, pages 624–627, May 2003.

[Ron00] Y. Rongshan. *Subband audio coding using a perceptually hybrid vector-scalar quantization.* PROCEEDINGS OF ICASSP-2000, pages 827–830, August 2000.

[RRBM02] M. Raad, C. Ritz, I. Burnett, and A. Mertins. *The Analysis of Speech Codecs Using Psycho-acoustic Measures.* In PROCEEDINGS OF SCW02, pages 108–110, October 2002.

[RS78] L. R. Rabiner and R. W. Schafer. DIGITAL PROCESSING OF SPEECH SIGNALS. Prentice-Hall, USA, 1978.

[RT00] J.M. Rodrigues and A. M. Tome. *On the use of backward adaptation in a perceptual audio coder.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 8(4):488–490, July 2000.

[RV91] O. Rioul and M. Vetterli. *Wavelets and Signal Processing.* IEEE SIGNAL PROCESSING MAGAZINE, 8(4):14–38, October 1991.

[RY90] K. R. Rao and P. Yip. DISCRETE COSINE TRANSFORM ALGORITHMS, ADVANTAGES, APPLICATIONS. Academic Press, Inc., London, 1990.

[Ryd96] T. Ryden. *Using listening tests to assess audio codecs.* In Neil Gilchrist and Christer Grewin, editors, COLLECTED PAPERS ON DIGITAL AUDIO BIT RATE REDUCTION, pages 115–125, USA, 1996. Audio Engineering Society, Inc.

[SA99] J. O. Smith and J. S. Abel. *Bark and ERB bilinear transforms.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 7(6):697–708, November 1999.

[Sch96] D. Schulz. *Improving audio codecs by noise substitution.* JOURNAL OF THE AES, 44(7/8):593–598, July/August 1996.

[Sha93] J. M. Shapiro. *Embedded image coding using zero trees of wavelet coefficients.* IEEE TRANSACTIONS ON SIGNAL PROCESSING, 41(12):3445–3462, December 1993.

[Shl97] S. Shlien. *The Modulated Lapped Transform.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 5(4):359–366, July 1997.

[Sho01] Y. Shoham. *Variable size vector entropy coding of speech and audio.* PROCEEDINGS OF ICASSP-2001, pages 769–772, May 2001.

[Sin90] S. Singhal. *High quality audio coding using multi-pulse LPC.* PROCEEDINGS OF ICASSP-90, pages 1101–1104, May 1990.

[SJ96] D. Sinha and J.D. Johnston. *Audio compression at low bit rates using a signal adaptive switched filter bank.* PROCEEDINGS OF ICASSP-96, 2:1053–1056, 1996.

[SJ98] P. Srinivasen and L. Jamieson. *High quality audio compression using an adaptive wavelet packet decomposition and psychoacoustic modeling.* IEEE TRANSACTIONS ON SIGNAL PROCESSING, 46(4):1085–1093, April 1998.

[SJDQ99] D. Sinha, D. Johnston, S. Dorward, and S. R. Quackenbush. *The Perceptual Audio Coder (PAC).* In V. K. Madisetti and D. B. Williams, editors, DIGITAL SIGNAL PROCESSING HANDBOOK, chapter 42. CRC Press, Boca Raton, 1999.

[SP96] A. Said and W. A. Pearlman. *A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees.* IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, 6(3):243–250, June 1996.

[ST93a] D. Sinha and A. Tewfik. *Low bit rate transparent audi compression using a dynamic dictionary and optimized wavelets.* PROCEEDINGS OF ICASSP-93, 1:197–200, May 1993.

[ST93b] D. Sinha and A.H. Tewfik. *Low bit rate transparent audio compression using adapted wavelets.* IEEE TRANSACTIONS ON SIGNAL PROCESSING, 41(12):3463–3479, December 1993.

[St98] R. Salami *et al. Design and description of* CS-ACELP*: a toll quality 8 kb/s speech coder.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 6(2):116–130, march 1998.

[Swe96] W. Sweldens. *The lifting scheme: A custom-Design construction of the biorthognal wavelets.* APPLIED AND COMPUTATIONAL HARMONIC ANALYSIS, 3(2):186–200, April 1996.

[TN98] T.D. Tran and T.Q. Nguyen. *A Lapped Transform Progressive Image Coder.* In PROCEEDINGS OF ISCAS 1998, volume 4, pages 1–4, 1998.

[TS00] W. C. Treurniet and G. A. Souldore. *Evaluation of the ITU-R objective audio quality measurement method.* JOURNAL OF THE AUDIO ENGINEERING SOCIETY, 48(3):164–173, March 2000.

[TSL87] G. Theile, G. Stoll, and M. Link. *Low bit-rate coding of high quality audio signals.* PROCEEDINGS OF THE 82ND AES CONVENTION*, preprint 2432*, March 1987.

[TSS+96] K. Tsutsui, H. Suzuki, O. Shinoyoskis, M. Sorohana, K. Akagiri, and R.M. Heddle. ATRAC*: adaptive transform acoustic coding for minidisc.* In N. Gilchrist and C. Grewin, editors, COLLECTED PAPERS ON DIGITAL AUDIO BIT-RATE REDUCTION, pages 95–101. AES, USA, 1996.

[Tt00] T. Thiede *et al.* PEAQ*- the* ITU *standard for objective measurement of perceived audio quality.* JOURNAL OF THE AUDIO ENGINEERING SOCIETY, 48(1/2):3–29, January/February 2000.

[VA01] M. S. Vinton and L. E. Atlas. *A scalable and progressive audio codec.* PROCEEDINGS OF ICASSP-2001, pages 3277–3280, May 2001.

[VDk01] W. Verhelst and D. De koning. *Noise shaping filter design for minimally audible signal requantization.* PROCEEDINGS OF 2001 IEEE WORKSHOP ON THE APPLICATIONS OF SIGNAL PROCESSING TO AUDIO AND ACOUSTICS, pages 147–150, October 2001.

[Vel92] R.N.J Veldhuis. *Bit rates in audio source coding.* IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, 10(1):86–96, January 1992.

[Ver99] T.S. Verma. A PERCEPTUALLY BASED AUDIO SIGNAL MODEL WITH APPLICATION TO SCALABLE AUDIO COMPRESSION. PhD thesis, Department of Electrical Engineering, Stanford university, October 1999.

[VHK01] R. Vafin, R. Heusden, and W.B. Kleijn. *Modifying transients for efficient coding of audio.* PROCEEDINGS OF ICASSP-2001, pages 3285–3288, May 2001.

[VHvdPK01] R. Vafin, R. Heusden, S. van de Par, and W.B. Kleijn. *Improved modeling of audio signals by modifying transient locations.* PROCEEDINGS OF 2001 IEEE WORKSHOP ON THE APPLICATIONS OF SIGNAL PROCESSING TO AUDIO AND ACOUSTICS, pages 143–146, October 2001.

[VK95] M. Vetterli and J. Kovacevic. WAVELETS AND SUBBAND CODING. Prentice Hall, PTR., New Jersey, 1995.

[VM00] J. S. Verma and T. H. Y. Meng. *A 6 kbps to 85 kbps scalable audio coder.* PROCEEDINGS OF ICASSP-2000, pages 877–880, June 2000.

[WNC87] I. H. Witten, R. M. Neal, and J. G. Cleary. *Arithmetic coding for data compression.* COMMUNICATIONS OF THE ACM, 30(6):520–540, June 1987.

[YH97] R.K. Yarlagadda and J.E. Hershey. HADAMARD MATRIX ANALYSIS AND SYNTHESIS. Kluwer Academic Publishers, U.S.A, 1997.

[Yip96] P. Yip. *Sine and cosine transforms.* In A.D. Poularikas, editor, THE TRANSFORMS AND APPLICATIONS HANDBOOK, chapter 3. CRC Press, USA, 1996.

[YK02] R. Yu and C. C. Ko. *A warped linear prediction based subband audio coding algorithm.* IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, 10(1):1–8, January 2002.

[ZF99] E. Zwicker and H. Fastel. PSYCHOACOUSTICS. Springer-Verlag, Berlin, second edition, 1999.

Betta Bookbinding & Printing
M & D Morrisey 02 42612998
26 Fields St
Kanabooka NSW 2530