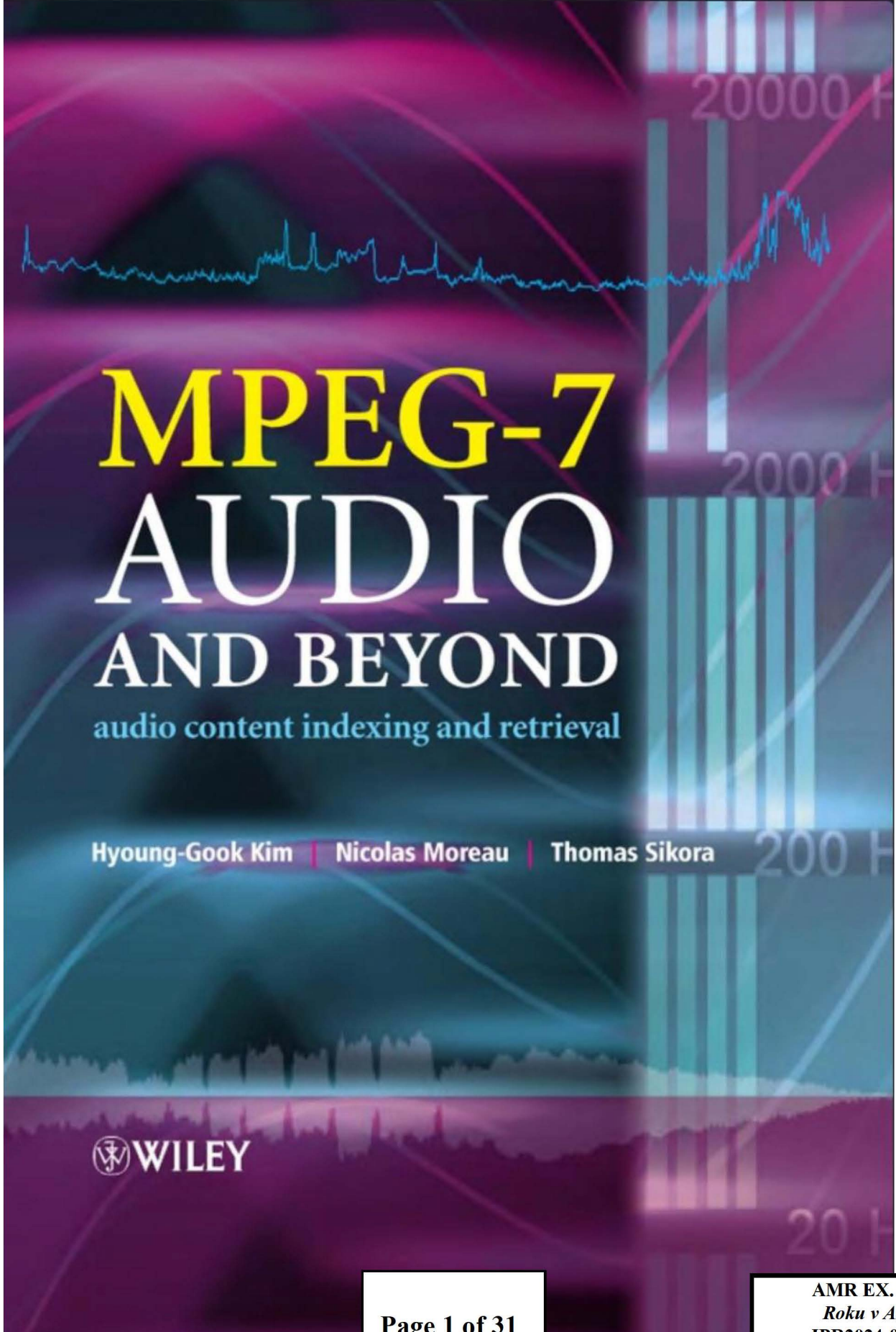




MPEG-7 AUDIO AND BEYOND

audio content indexing and retrieval

HyounG-Gook Kim | Nicolas Moreau | Thomas Sikora

 WILEY

MPEG-7 Audio and Beyond

Audio Content Indexing and Retrieval

Hyung-Gook Kim

Samsung Advanced Institute of Technology, Korea

Nicolas Moreau

Technical University of Berlin, Germany

Thomas Sikora

Communication Systems Group, Technical University of Berlin, Germany



John Wiley & Sons, Ltd

Copyright © 2005 John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester,
West Sussex PO19 8SQ, England

Telephone (+44) 1243 779777

Email (for orders and customer service enquiries): cs-books@wiley.co.uk

Visit our Home Page on www.wiley.com

All Rights Reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except under the terms of the Copyright, Designs and Patents Act 1988 or under the terms of a licence issued by the Copyright Licensing Agency Ltd, 90 Tottenham Court Road, London W1T 4LP, UK, without the permission in writing of the Publisher. Requests to the Publisher should be addressed to the Permissions Department, John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, or emailed to permreq@wiley.co.uk, or faxed to (+44) 1243 770620.

This publication is designed to provide accurate and authoritative information in regard to the subject matter covered. It is sold on the understanding that the Publisher is not engaged in rendering professional services. If professional advice or other expert assistance is required, the services of a competent professional should be sought.

Other Wiley Editorial Offices

John Wiley & Sons Inc., 111 River Street, Hoboken, NJ 07030, USA

Jossey-Bass, 989 Market Street, San Francisco, CA 94103-1741, USA

Wiley-VCH Verlag GmbH, Boschstr. 12, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 42 McDougall Street, Milton, Queensland 4064, Australia

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01, Jin Xing Distripark, Singapore 129809

John Wiley & Sons Canada Ltd, 22 Worcester Road, Etobicoke, Ontario, Canada M9W 1L1

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Library of Congress Cataloging in Publication Data

Kim, Hyoung-Gook.

Introduction to MPEG-7 audio / Hyoung-Gook Kim, Nicolas Moreau, Thomas Sikora.

p. cm.

Includes bibliographical references and index.

ISBN-13 978-0-470-09334-4 (cloth: alk. paper)

ISBN-10 0-470-09334-X (cloth: alk. paper)

1. MPEG (Video coding standard) 2. Multimedia systems. 3. Sound—Recording and reproducing—Digital techniques—Standards. I. Moreau, Nicolas. II. Sikora, Thomas. III. Title.

TK6680.5.K56 2005

006.6'96—dc22

2005011807

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

ISBN-13 978-0-470-09334-4 (HB)

ISBN-10 0-470-09334-X (HB)

Typeset in 10/12pt Times by Integra Software Services Pvt. Ltd, Pondicherry, India

Printed and bound in Great Britain by TJ International Ltd, Padstow, Cornwall

This book is printed on acid-free paper responsibly manufactured from sustainable forestry in which at least two trees are planted for each one used for paper production.

Contents

List of Acronyms	xi
List of Symbols	xv
1 Introduction	1
1.1 Audio Content Description	2
1.2 MPEG-7 Audio Content Description – An Overview	3
1.2.1 MPEG-7 Low-Level Descriptors	5
1.2.2 MPEG-7 Description Schemes	6
1.2.3 MPEG-7 Description Definition Language (DDL)	9
1.2.4 BiM (Binary Format for MPEG-7)	9
1.3 Organization of the Book	10
2 Low-Level Descriptors	13
2.1 Introduction	13
2.2 Basic Parameters and Notations	14
2.2.1 Time Domain	14
2.2.2 Frequency Domain	15
2.3 Scalable Series	17
2.3.1 Series of Scalars	18
2.3.2 Series of Vectors	20
2.3.3 Binary Series	22
2.4 Basic Descriptors	22
2.4.1 Audio Waveform	23
2.4.2 Audio Power	24
2.5 Basic Spectral Descriptors	24
2.5.1 Audio Spectrum Envelope	24
2.5.2 Audio Spectrum Centroid	27
2.5.3 Audio Spectrum Spread	29
2.5.4 Audio Spectrum Flatness	29
2.6 Basic Signal Parameters	32
2.6.1 Audio Harmonicity	33
2.6.2 Audio Fundamental Frequency	36

2.7	Timbral Descriptors	38
2.7.1	Temporal Timbral: Requirements	39
2.7.2	Log Attack Time	40
2.7.3	Temporal Centroid	41
2.7.4	Spectral Timbral: Requirements	42
2.7.5	Harmonic Spectral Centroid	45
2.7.6	Harmonic Spectral Deviation	47
2.7.7	Harmonic Spectral Spread	47
2.7.8	Harmonic Spectral Variation	48
2.7.9	Spectral Centroid	48
2.8	Spectral Basis Representations	49
2.9	Silence Segment	50
2.10	Beyond the Scope of MPEG-7	50
2.10.1	Other Low-Level Descriptors	50
2.10.2	Mel-Frequency Cepstrum Coefficients	52
	References	55
3	Sound Classification and Similarity	59
3.1	Introduction	59
3.2	Dimensionality Reduction	61
3.2.1	Singular Value Decomposition (SVD)	61
3.2.2	Principal Component Analysis (PCA)	62
3.2.3	Independent Component Analysis (ICA)	63
3.2.4	Non-Negative Factorization (NMF)	65
3.3	Classification Methods	66
3.3.1	Gaussian Mixture Model (GMM)	66
3.3.2	Hidden Markov Model (HMM)	68
3.3.3	Neural Network (NN)	70
3.3.4	Support Vector Machine (SVM)	71
3.4	MPEG-7 Sound Classification	73
3.4.1	MPEG-7 Audio Spectrum Projection (ASP) Feature Extraction	74
3.4.2	Training Hidden Markov Models (HMMs)	77
3.4.3	Classification of Sounds	79
3.5	Comparison of MPEG-7 Audio Spectrum Projection vs. MFCC Features	79
3.6	Indexing and Similarity	84
3.6.1	Audio Retrieval Using Histogram Sum of Squared Differences	85
3.7	Simulation Results and Discussion	85
3.7.1	Plots of MPEG-7 Audio Descriptors	86
3.7.2	Parameter Selection	88
3.7.3	Results for Distinguishing Between Speech, Music and Environmental Sound	91

3.7.4	Results of Sound Classification Using Three Audio Taxonomy Methods	92
3.7.5	Results for Speaker Recognition	96
3.7.6	Results of Musical Instrument Classification	98
3.7.7	Audio Retrieval Results	99
3.8	Conclusions	100
	References	101
4	Spoken Content	103
4.1	Introduction	103
4.2	Automatic Speech Recognition	104
4.2.1	Basic Principles	104
4.2.2	Types of Speech Recognition Systems	108
4.2.3	Recognition Results	111
4.3	MPEG-7 <i>SpokenContent</i> Description	113
4.3.1	General Structure	114
4.3.2	<i>SpokenContentHeader</i>	114
4.3.3	<i>SpokenContentLattice</i>	121
4.4	Application: Spoken Document Retrieval	123
4.4.1	Basic Principles of IR and SDR	124
4.4.2	Vector Space Models	130
4.4.3	Word-Based SDR	135
4.4.4	Sub-Word-Based Vector Space Models	140
4.4.5	Sub-Word String Matching	154
4.4.6	Combining Word and Sub-Word Indexing	161
4.5	Conclusions	163
4.5.1	MPEG-7 Interoperability	163
4.5.2	MPEG-7 Flexibility	164
4.5.3	Perspectives	166
	References	167
5	Music Description Tools	171
5.1	Timbre	171
5.1.1	Introduction	171
5.1.2	<i>InstrumentTimbre</i>	173
5.1.3	<i>HarmonicInstrumentTimbre</i>	174
5.1.4	<i>PercussiveInstrumentTimbre</i>	176
5.1.5	Distance Measures	176
5.2	Melody	177
5.2.1	<i>Melody</i>	177
5.2.2	<i>Meter</i>	178
5.2.3	<i>Scale</i>	179
5.2.4	<i>Key</i>	181

5.2.5	<i>MelodyContour</i>	182
5.2.6	<i>MelodySequence</i>	185
5.3	Tempo	190
5.3.1	<i>AudioTempo</i>	192
5.3.2	<i>AudioBPM</i>	192
5.4	Application Example: Query-by-Humming	193
5.4.1	Monophonic Melody Transcription	194
5.4.2	Polyphonic Melody Transcription	196
5.4.3	Comparison of Melody Contours	200
	References	203
6	Fingerprinting and Audio Signal Quality	207
6.1	Introduction	207
6.2	Audio Signature	207
6.2.1	Generalities on Audio Fingerprinting	207
6.2.2	Fingerprint Extraction	211
6.2.3	Distance and Searching Methods	216
6.2.4	MPEG-7-Standardized <i>AudioSignature</i>	217
6.3	Audio Signal Quality	220
6.3.1	<i>AudioSignalQuality</i> Description Scheme	221
6.3.2	<i>BroadcastReady</i>	222
6.3.3	<i>IsOriginalMono</i>	222
6.3.4	<i>BackgroundNoiseLevel</i>	222
6.3.5	<i>CrossChannelCorrelation</i>	223
6.3.6	<i>RelativeDelay</i>	224
6.3.7	<i>Balance</i>	224
6.3.8	<i>DcOffset</i>	225
6.3.9	<i>Bandwidth</i>	226
6.3.10	<i>TransmissionTechnology</i>	226
6.3.11	<i>ErrorEvent</i> and <i>ErrorEventList</i>	226
	References	227
7	Application	231
7.1	Introduction	231
7.2	Automatic Audio Segmentation	234
7.2.1	Feature Extraction	235
7.2.2	Segmentation	236
7.2.3	Metric-Based Segmentation	237
7.2.4	Model-Selection-Based Segmentation	242
7.2.5	Hybrid Segmentation	243
7.2.6	Hybrid Segmentation Using MPEG-7 ASP	246
7.2.7	Segmentation Results	250

7.3	Sound Indexing and Browsing of Home Video Using Spoken Annotations	254
7.3.1	A Simple Experimental System	254
7.3.2	Retrieval Results	258
7.4	Highlights Extraction for Sport Programmes Using Audio Event Detection	259
7.4.1	Goal Event Segment Selection	261
7.4.2	System Results	262
7.5	A Spoken Document Retrieval System for Digital Photo Albums	265
	References	266
	Index	271

6

Fingerprinting and Audio Signal Quality

6.1 INTRODUCTION

This chapter is dedicated to audio fingerprinting and audio signal quality description. In general, the MPEG-7 low-level descriptors in Chapter 2 can be seen as providing a fingerprint for describing audio content. We will focus in this chapter on fingerprinting tools specifically developed for the identification of a piece of audio and for describing its quality.

6.2 AUDIO SIGNATURE

6.2.1 Generalities on Audio Fingerprinting

This section gives a general introduction to the concept of fingerprinting. The technical aspects will be detailed in Sections 6.2.2–6.2.4 quoted out of (Cano *et al.*, 2002a) and (Herre *et al.*, 2002).

6.2.1.1 Motivations

The last decades have witnessed enormous growth in digitized audio (music) content production and storage. This has made available to today's users an overwhelming amount of audio material. However, this scenario created great new challenges for search and access to audio material, turning the process of finding or identifying the desired content efficiently into a key issue in this context.

Audio fingerprinting or content-based audio identification (CBID) technologies¹ are possible and effective solutions to the aforementioned problems, providing the ability to link unlabelled audio to corresponding metadata (e.g. artist and song name), perform content-based integrity verification or watermarking support (Cano *et al.*, 2002c).

Audio *watermarking* is also another possible and much proposed solution. It is somewhat related to audio fingerprinting, but that topic is beyond the scope of this section. There are some references that explain the differences and similarities between watermarking and fingerprinting, and evaluate the applications where each technology is best suited for use (Cano *et al.*, 2002c; Gómez *et al.*, 2002; Gomes *et al.*, 2003).

The basic concept behind an audio fingerprinting system is the identification of a piece of audio content by means of a compact and unique signature extracted from it. This signature, also known as the *audio fingerprint*, can be seen as a summary or perceptual digest of the audio recording. During a training phase, those signatures are created from a set of known audio material and are then stored in a database. Unknown content, even if distorted or fragmented, should afterwards be identified by matching its signature against the ones contained in the database.

However, great difficulties arise when trying to identify audio-distorted content automatically (i.e. comparing a PCM music audio clip against the same clip compressed as MP3 audio).

Fingerprinting eliminates the direct comparison of the (typically large) digitized audio waveform as an efficient and effective approach to audio identification. Also hash methods, such as MD5 (Message Digest 5) or CRC (Cyclic Redundancy Checking), can be used to obtain a more compact representation of the audio binary file (which would allow a more efficient matching). It is difficult to achieve an acceptable robustness to compression or minimal distortions of any kind in the audio signals using hash methods, since the obtained hash values are very fragile to single bit changes.

Hash methods fail to perform the desired perceptual identification of the audio content. In fact, these approaches should not be considered as content-based identification, since they do not consider the content, just the bit information in the audio binary files (Cano *et al.*, 2002a).

When compared with the direct matching of multimedia content based on waveforms, fingerprint systems present important advantages in the identification of audio contents. Fingerprints have small memory and storage requirements and perform matching efficiently. On the other hand, since perceptual irrelevancies have already been removed from the fingerprints, fingerprinting systems should be able to achieve much more robust matching results.

¹ Audio fingerprinting is also known as robust matching, robust or perceptual hashing, passive watermarking, automatic music recognition, content-based digital signatures and content-based audio identification (Cano *et al.*, 2002c).

6.2.1.2 Requirements

An audio fingerprinting system should fulfil the following basic, application-dependent requirements (Cano *et al.*, 2002a, Haitsma and Kalker, 2002):

- **Robustness:** the system should be able to identify an audio item accurately, regardless of the level of compression, distortion or interference in the transmission channel. Additionally, it should be able to deal gracefully with other sources of degradation, such as pitch shifting, time extension/compression, equalization, background noise, A/D and D/A conversion, speech and audio coding artefacts (e.g. GSM, MP3), among others. In order to achieve high robustness, the audio fingerprint should be based on features strongly invariant with respect to signal degradations, so that severely degraded audio still leads to similar fingerprints. The false negative rate (i.e. very distinct audio fingerprints corresponding to perceptually similar audio clips) is normally used to express robustness.
- **Reliability:** highly related to the robustness, this parameter is inversely related to the rate at which the system identifies an audio clip incorrectly (false positive rate). A good fingerprinting system should make very few such mismatch errors, and when faced with a very low (or below a specified threshold) identification confidence it should preferably output an “unknown” identification result. Approaches to deal with false positives have been treated for instance in (Cano *et al.*, 2001).
- **Granularity:** depending on the application, it should be able to identify whole titles from excerpts a few seconds long (this property is also known as robustness to cropping), which requires methods for dealing with shifting. This problem addresses a lack of synchronization between the extracted fingerprint and those stored in the database.
- **Efficiency:** the system should be computationally efficient. Consequently, the size of the fingerprints, the complexity of the corresponding fingerprint extraction algorithms, as well as the speed of the searching and matching algorithms, are key factors in the global efficiency of a fingerprinting system.
- **Scalability:** the algorithms used in the distinct building blocks of a fingerprinting system should scale well with the growth of the fingerprint database, so that the robustness, reliability and efficiency parameters of the system remain as specified independently of the register of new fingerprints in the database.

There is an evident interdependency between the above listed requirements. In most cases, this is when improving one parameter implies losing performance in another. A more detailed enumeration of requirements can be found in (Kalker, 2001; Cano *et al.*, 2002c).

An audio fingerprint system generally consists of two main building blocks: one responsible for the extraction of the fingerprints and another one that performs the search and matching of fingerprints. The fingerprint extraction module

should try to obtain a set of relevant perceptual features out of an audio recording, and the resultant audio fingerprint should respect the following requirements (Cano *et al.*, 2002c):

- *Discrimination power over huge numbers of other fingerprints*: a fingerprint is a perceptual digest of the recording, and so must retain the maximum of acoustically relevant information. This digest should allow discrimination over a large number of fingerprints. This may conflict with other requirements, such as efficiency and robustness.
- *Invariance to distortions*: this derives from the robustness requirement. Content-integrity applications, however, may relax this constraint for content preserving distortions in order to detect deliberate manipulations.
- *Compactness*: a small-sized representation is important for efficiency, since a large number (e.g. millions) of fingerprints need to be stored and compared. However, an excessively short representation might not be sufficient to discriminate among recordings, thus affecting robustness and reliability.
- *Computational simplicity*: for efficiency reasons, the fingerprint extraction algorithms should be computationally efficient and consequently not very time consuming.

The solutions proposed to fulfil the above requirements normally call for a trade-off between dimensionality reduction and information loss, and such a compromise is usually defined by the needs of the application in question.

6.2.1.3 General Structure of Audio Identification Systems

Independent of the specific approach to extract the content-based compact signature, a common architecture can be devised to describe the functionality of fingerprinting when used for identification (RIAA/IFPI, 2001). This general architecture is depicted in Figure 6.1. Two distinct phases can be distinguished:

- **Building the database**: off-line a memory of the audio to be recognized is created. A series of sound recordings is presented to a fingerprint generator. This generator processes audio signals in order to generate fingerprints derived uniquely from the characteristics of each sound recording. The fingerprint (e.g. the compact and unique representation) that is derived from each recording is then stored in a database and can be linked with a tag or other metadata relevant to each recording.
- **Content identification**: in the identification mode, unlabelled audio (in either streaming or file format) is presented to the input of a fingerprint generator. The fingerprint generator function processes the audio signal to produce a

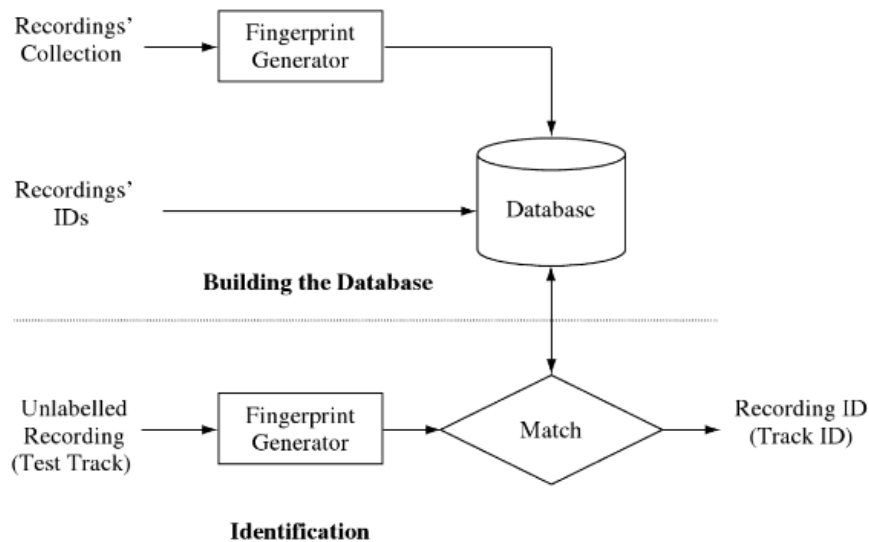


Figure 6.1 Content-based audio identification framework

fingerprint. This fingerprint is then used to query the database and is compared with the stored fingerprints. If a match is found, the resulting track identifier (Track ID) is retrieved from the database. A confidence level or proximity associated with each match may also be given.

The actual implementations of audio fingerprinting normally follow this scheme with differences on the acoustic features observed and the modelling of audio as well as in the matching and indexing algorithms.

6.2.2 Fingerprint Extraction

The overall extraction procedure is schematized in Figure 6.2. The fingerprint generator consists of a front-end and a fingerprint modelling block. These two modules are described in the following sections.

6.2.2.1 Front-End

The front-end converts an audio signal into a sequence of relevant features to feed the fingerprint model block. Several driving forces co-exist in the design of the front-end: dimensionality reduction, extraction of perceptually meaningful parameters (similar to those used by the human auditory system), design towards invariance or robustness (with respect to channel distortions, background noise, etc.), temporal correlation (systems that capture spectral dynamics).

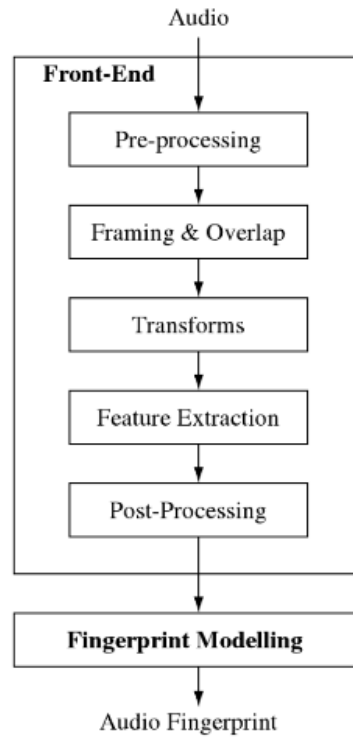


Figure 6.2 Fingerprint generator: frond-end and fingerprint modelling (Cano *et al.*, 2002a)

The front-end comprises five blocks: pre-processing, framing and overlap, transformation, feature extraction and post-processing (see Figure 6.2).

Pre-Processing

Most of the front-ends for audio fingerprinting start with a pre-processing step, where the audio is digitized (if necessary) and converted to a general digital audio format (e.g. 16-bit PCM, 5–44.1 kHz, mono). The signal may also be subjected to other types of processing, such as GSM encoding/decoding in a mobile phone system, pre-emphasis, amplitude normalization (bounding the dynamic range to $[-1, 1]$), among others. In the training phase (i.e. when adding a new fingerprint to the database), the fingerprint is usually extracted from an audio source with the best quality possible, trying to minimize interference, distortion or unnecessary processing of the original audio recording.

Framing and Overlap

The audio signal is then divided into frames (whose size should be chosen so that measurements can be assumed to be a stationary signal), windowed (in order to minimize discontinuities) and finally overlapped (this assures some robustness to shifting). The choice of the frame size, window type and overlap factor is again a trade-off between the rate of change in the spectrum and system complexity.

Typical values for frame size and overlap factor are in the ranges 10–500 ms and 50–80%, respectively.

Linear Transforms: Spectral Estimates

In order to achieve the desired reduction of redundancy in the audio signal, the audio frames are first submitted to a suitably chosen linear transformation. The most common transformation used in audio fingerprinting systems is the fast Fourier transform (FFT), but some other transforms have also been proposed: the discrete cosine transform (DCT), the Haar transform or the Walsh–Hadamard transform (Subramanya *et al.*, 1997), and the modulated complex lapped transform (MCLT) (Mihcak and Venkatesan, 2001; Burges *et al.*, 2002). Richly *et al.* did a comparison of the DFT and the Walsh–Hadamard transform that revealed that the DFT is generally less sensitive to shifting (Richly *et al.*, 2000). The MCLT exhibits approximate shift invariance properties.

There are optimal transforms in the sense of information packing and decorrelation properties, like Karhunen–Loève (KL) or singular value decomposition (SVD) (Theodoris and Koutroumbas, 1998). These transforms, however, are signal dependent and computationally complex. For that reason, lower-complexity transforms using fixed basis vectors are more common. Most content-based audio identification methods therefore use standard transforms to facilitate efficient compression, noise removal and subsequent processing (Lourens, 1990; Kurth *et al.*, 2002).

Feature Extraction

Once a time–frequency representation of the audio signal has been transformed, additional transformations are applied to the audio frames to generate the final acoustic vectors. The purpose is again to reduce the dimensionality, and at the same time to increase the invariance to distortions. A large number of algorithms have been proposed to generate the final feature vectors, and many of them extract several features by means of a critical-band analysis of the spectrum.

It is very common to include knowledge of the human auditory system to extract more perceptually meaningful parameters. Therefore, many systems extract several features performing a critical-band analysis of the spectrum. In (Papaodysseus *et al.*, 2001; Cano *et al.*, 2002a), mel-frequency cepstrum coefficients (MFCCs) are used. In (Allamanche *et al.*, 2001), the choice is the spectral flatness measure (SFM), which is an estimation of the tone-like or noise-like quality for a band in the spectrum. Papaodysseus *et al.* proposed a solution based on “band representative vectors”, which are an ordered list of indexes of bands with prominent tones (Papaodysseus *et al.*, 2001). Kimura *et al.* use the energy of each frequency band (Kimura *et al.*, 2001), and Haitsma and Kalker propose the use of the energies of 33 bark-scaled bands to obtain a “hash string”, which is the sign of the energy band differences, in both the time and in frequency axes (Haitsma and Kalker, 2002).

Sukittanon and Atlas claim that spectral estimates and related features are only inadequate when audio channel distortion occurs (Sukittanon and Atlas, 2002). They proposed modulation frequency analysis to characterize the time varying behaviour of audio signals. In this case, features correspond to the geometric mean of the modulation frequency estimation of the energy of 19 bark-spaced band filters.

Approaches for music information retrieval include features that have proved valid for comparing sound: harmonicity, bandwidth, loudness, ZCR, etc. (Blum *et al.*, 1999).

(Burges *et al.*, 2002) point out that the features commonly used are heuristic, and as such may not be optimal. For that reason they use a modified Karhunen–Loève (KL) transform, namely oriented principal component analysis (OPCA), to find the optimal features in an unsupervised way. If KL (which is also known as principal component analysis, PCA) finds a set of orthogonal directions which maximize the signal variance, OPCA obtains a set of possible non-orthogonal directions which take some predefined distortions into account.

Post-Processing

Most of the above-mentioned features are absolute measurements. In order to characterize better the temporal variations in the signal, higher-order time derivatives are added to the feature vector. In (Batlle *et al.*, 2002; Cano *et al.*, 2002a), the feature vector is the concatenation of MFCCs, their derivative (Δ) and the acceleration ($\Delta\Delta$), as well as the Δ and $\Delta\Delta$ of the energy. In order to minimize the tendency of derivative features to amplify noise, (Batlle *et al.*, 2002) use cepstrum mean normalization (CMN) to reduce slowly varying channel distortions. They both use transforms (e.g. PCA) to compact the feature vector representation. Some other systems use only the derivative of the features, discarding their absolute values (Allamanche *et al.*, 2001; Kurth *et al.*, 2002). It is quite common to apply a very low-resolution quantization to the features: ternary (Richly *et al.*, 2000) or binary (Haitsma and Kalker, 2002; Kurth *et al.*, 2002). This allows the gaining of robustness against distortions (Haitsma and Kalker, 2002; Kurth *et al.*, 2002), normalizes (Richly *et al.*, 2000), eases hardware implementations and reduces memory requirements. Binary sequences are required to extract error correcting words. In (Mihcak and Venkatesan, 2001), the discretization is designed to increase randomness in order to minimize fingerprint collision probability.

6.2.2.2 Fingerprint Modelling

The fingerprint modelling block usually receives a sequence of feature vectors calculated on a frame-by-frame basis. This is a sequence of vectors where redundancies may be exploited in the frame time vicinity, inside a recording and across the whole database to reduce the fingerprint size further. The type of model chosen conditions the distance metric and also the design of indexing algorithms for fast retrieval.

A very concise form of fingerprint is achieved by summarizing the multi-dimensional vector sequences of a whole song (or a fragment of it) in a single vector.

In Etantrum,¹ the fingerprint was calculated from the means and variances of the 16 bank-filtered energies corresponding to 30 s of audio encoding with a signature of up to 512 bits, which along with information on the original audio format was sent to a server for identification (Cano *et al.*, 2002a). MusicBrainz's² TRM audio fingerprint is composed of the average zero crossing rate feature, the estimated beats per minute (BPM), an average representation of the spectrum and some more features of a piece of audio. This fingerprint model proved to be computationally efficient and compact, addressing the requirements of the application for which this system was designed: linking MP3 files to metadata (title, artist, etc.). This application gives priority to low complexity (on both the client and server side) to the detriment of robustness.

Fingerprints can also be simple sequences of features. In (Haitsma and Kalker, 2002) and (Papaodysseus *et al.*, 2001), the fingerprint, which consists of a sequence of band representative vectors, is binary encoded for memory efficiency. Some systems include high-level musically meaningful attributes, like rhythm (BPM) or prominent pitch (Blum *et al.*, 1999).

Following the reasoning on the possible sub-optimality of heuristic features, (Burgess *et al.*, 2002) employs several layers of OPCA to decrease the local statistical redundancy of feature vectors with respect to time, reducing dimensionality and achieving better robustness to shifting and pitching.

(Allamanche *et al.*, 2001) propose the exploration of "global redundancies" within an audio piece, assuming that the audio features of a given audio item are similar among them. From this assumption, a compact representation can be generated by clustering the feature vectors, thus approximating the sequence of feature vectors by a much lower number of representative code vectors, a codebook. However, the temporal evolution of the extracted features is lost with such an approximation. In order to achieve higher recognition results and faster matching, short-time statistics are collected over regions of time, which allows taking into account certain temporal dependencies and shortening the length of each sequence.

(Cano *et al.*, 2002b) and (Batlle *et al.*, 2002) use a fingerprint model much inspired by speech research, and which further exploits global redundancy. They view a corpus of music as sentences constructed of concatenating sound classes of a finite alphabet (e.g. "perceptually equivalent" drum sounds occur in a great number of pop songs). This approximation yields a fingerprint which consists of sequences of indexes to a set of sound classes representative of a collection of audio items. The sound classes are estimated via unsupervised clustering and

¹ Available on-line at: <http://www.freshmeat.net/project/songprint>.

² MusicBrainz: <http://www.musicbrainz.org>.

modelled with hidden Markov models (HMMs). This fingerprint representation retains the information on the evolution of audio through time.

In (Mihcak and Venkatesan, 2001), discrete sequences are mapped to a dictionary of error correcting words. In (Kurth *et al.*, 2002), the error correcting codes form the basis of their indexing method.

6.2.3 Distance and Searching Methods

After an audio database has been indexed with the extracted fingerprints of individual recordings, it can be searched by the fingerprint identification module as depicted in Figure 6.1.

In order to compare fingerprints, it is necessary to define some sort of metric. Distance metrics are closely related to the type of model chosen. Correlation is a common option, but the Euclidean distance, or variations of it that deal with sequences of different lengths, are used for instance in (Blum *et al.*, 1999). In (Sukittanon and Atlas, 2002), the classification is nearest neighbour using a cross-entropy estimation. In systems where the vector feature sequences are quantized, a Manhattan distance (or Hamming distance when the quantization is binary) is common (Richly *et al.*, 2000; Haitsma and Kalker, 2002). Another error metric called the *exponential pseudo norm* (EPM), suggested by (Mihcak and Venkatesan, 2001), could be more appropriate to distinguish between close and distant values with an emphasis stronger than linear.

So far, the presented identification frameworks are based on a template matching paradigm (Theodoris and Koutroumbas, 1998): both the reference fingerprints (the ones stored in the database) and the test fingerprints (the ones extracted from the unknown audio) are in the same format and are thus compared according to some distance metric, e.g. Hamming distance, a correlation and so on.

However, in some systems only the reference items are actually fingerprints, compactly modelled as a codebook or a sequence of indexes to HMMs (Allamanche *et al.*, 2001; Batlle *et al.*, 2002), and in these cases the distances are computed directly between the feature sequence extracted from the unknown audio and the reference audio fingerprints stored in the repository. In (Allamanche *et al.*, 2001), the feature vector sequence is matched to the different codebooks using a distance metric. For each codebook, the errors are accumulated. The unknown items are then assigned to the class which yields the lowest accumulated error. In (Batlle *et al.*, 2002), the feature sequence is run against the fingerprints (a concatenation of indexes pointing at HMM sound classes) using the Viterbi algorithm and the most likely passage in the database is selected.

Besides the definition of a distance metric for a fingerprint comparison, a fundamental issue for the usability of such a system is how the comparison of the unknown audio will in fact be made efficiently against all the possibly millions of fingerprints, and this depends heavily on the fingerprint model being used. Vector spaces allow the use of efficient, existing spatial access methods (Baeza-Yates

and Ribeiro-Neto 1999). The general goal is to build a data structure, an index, to reduce the number of distance evaluations when a query is presented. As stated by (Chávez *et al.*, 2001), most indexing algorithms for proximity searching build sets of equivalence classes, discard some classes and exhaustively search the rest. They use a simpler distance to eliminate many hypotheses quickly, and the use of indexing methods to overcome the brute-force exhaustive matching with a more expensive distance function is found in the content-based audio identification literature, e.g. in (Kenyon, 1999). Haitsma and Kalker proposed an efficient search algorithm: instead of following a brute-force approach where the distance to every position in the database would be calculated, the distance is calculated for a few candidate positions only (Haitsma and Kalker, 2002). These candidates contain, with very high probability, the best-matching position in the database. In (Cano *et al.*, 2002a), heuristics similar to those used in computational biology for the comparison of DNA are used to speed up a search in a system where the fingerprints are sequences of symbols. (Kurth *et al.*, 2002) presents an index that uses code words extracted from binary sequences representing the audio. These approaches, although very fast, make assumptions on the errors permitted in the words used to build the index. This could result in false dismissals; the simple coarse distance used for discarding unpromising hypotheses must lower bound the more expensive fine distance.

The final step in an audio fingerprinting system is to decide whether the query is present or not in the repository. The result of the comparison of the extracted fingerprint with the database of fingerprints is a score which results from the calculated distances. The decision for a correct identification must be based on a score that is beyond a certain threshold. However, this threshold is not trivial to define, and dependent on the fingerprint model, on the discriminative information of the query, on the similarity of the fingerprints in the database and on the database size. The larger the database, the higher the probability of returning a false positive, and consequently the lower the reliability of the system.

6.2.4 MPEG-7-Standardized *AudioSignature*

The MPEG-7 audio standard provides a generic framework for the descriptive annotation of audio data. The *AudioSignature* high-level tool is a condensed representation of an audio signal, designed to provide a unique content identifier for the purpose of robust automatic identification. It is a compact-sized audio signature which can be used as a fingerprint. This tool also provides an example of how to use the low-level MPEG-7 framework.

6.2.4.1 Description Scheme

The *AudioSignature* description essentially consists of a statistical summarization of *AudioSpectrumFlatness* low-level descriptors (LLDs) over a period of

time. These *AudioSpectrumFlatness* descriptors (see Chapter 2) are extracted on a frame-by-frame basis. The spectral flatness LLDs are stored in the unique attribute of the *AudioSignature* description scheme, which is called *Flatness*.

There are some restrictions regarding the instantiations of the *AudioSpectrumFlatness* descriptors. In order to constitute a valid *AudioSignature* description, the following requirements have to be satisfied:

- The *AudioSpectrumFlatness* descriptions contained by the *Flatness* attribute must be stored as a *SeriesOfVectorBinary* description.
- Both the *Mean* and the *Variance* fields of the *SeriesOfVectorBinary* series containing the *Flatness* features have to be instantiated.
- The scaling ratio (i.e. the *ratio* attribute of the *SeriesOfVectorBinary*) must range between 2 and 128. The default value is 32.
- The *loEdge* attribute of the *AudioSpectrumFlatness* descriptor is fixed at 250 Hz.
- The *hiEdge* attribute of the *AudioSpectrumFlatness* descriptor must be at least 500 Hz. The default value is 4000 Hz.

6.2.4.2 Scalability

The *AudioSignature* description scheme instantiates the *AudioSpectrumFlatness* LLDs in such a way that an interoperable hierarchy of scalable audio signatures can be established with regard to the following parameters:

- Temporal scope of the audio signatures.
- Temporal resolution of the audio signatures.
- Spectral coverage/bandwidth of the audio signatures.

The temporal scope of the audio signatures represents a first degree of freedom and relates to the start position and the length of the audio item for which the feature extraction is carried out. The signal segment used for signature generation can be chosen freely and depends on the type of application envisaged.

The temporal resolution of the fingerprint is an important parameter which can be used to control the trade-off between fingerprint compactness and its descriptive power (i.e. its ability to discriminate between many different audio signals). This temporal scalability is obtained by using the mean and variance of the LLD values, as provided by the generic *SeriesOfVectorBinary* construct, with selectable degrees of decimation, and thus temporal resolution. Consequently, signatures may be rescaled (scaled down) in their temporal resolution according to the standard *SeriesOfVectorBinary* scaling procedures as desired, e.g. in order to achieve a compatible temporal resolution between two signatures.

A second dimension of *AudioSignature* scalability resides in its spectral coverage/bandwidth. The *AudioSpectrumFlatness* descriptor provides a vector of feature values, each value corresponding to a specific quarter-octave frequency

band. The number of frequency bands above a fixed base frequency (250 Hz) can be selected as a further parameter of scalability. While signatures may provide different numbers of frequency bands, a meaningful comparison between them is always possible for the bands common to the compared signatures, since these relate to common fixed band definitions.

6.2.4.3 Use in Fingerprinting Applications

The main application for the *AudioSignature* description scheme is the automatic identification of an unknown piece of audio based on a database of registered audio items. An *AudioSignature* is extracted from the item to be identified and then matched to all previously registered *AudioSignature* descriptions in a database. The best-matching reference *AudioSignature* description is the most likely candidate to correspond to the unknown signal.

In a wider sense, the *AudioSignature* structure may be used to identify corresponding MPEG-7 descriptions for audio items which are delivered without any other descriptive data. To this end, the MPEG-7 descriptions available at some server have to include an *AudioSignature* description of each described item.

An example of an MPEG-7-based audio identification system is depicted in Figure 6.3 (Herre *et al.*, 2002). It consists mainly of the two basic extraction and identification modes of operation already introduced in Figure 6.1.

The first steps in the signal processing chain are the same for training and classification: the audio signal is converted to a standard format (e.g. monophonic PCM) at the pre-processing stage of the feature extractor. This stage is followed by the extraction of the *AudioSpectrumFlatness* LLD features.

A feature processor is then used to decrease the description size by means of statistical data summarization. In the MPEG-7 framework, this is done by applying the appropriate rescaling operation to the MPEG-7 *ScalableSeries* of

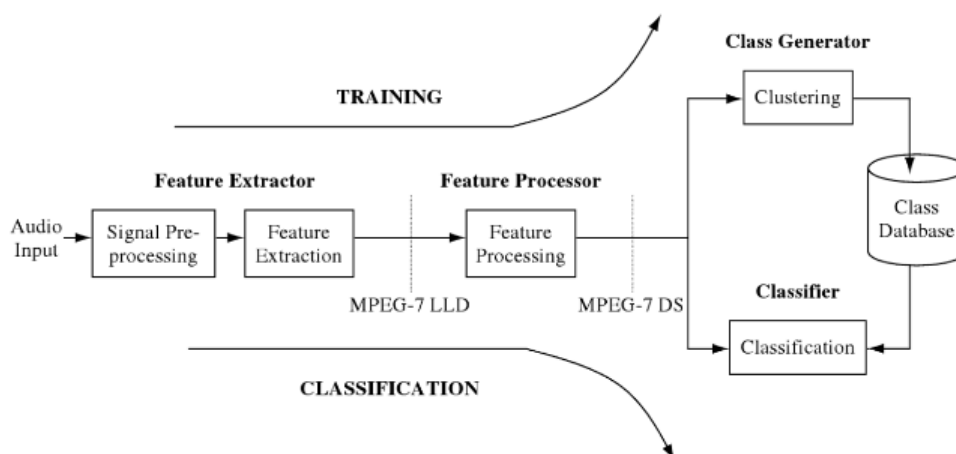


Figure 6.3 MPEG-7 standard audio identification system (Herre *et al.*, 2002)

vectors containing the *AudioSpectrumFlatness* features. The final result is an MPEG-7 *AudioSignature* description that can be interpreted as a fingerprint of the original piece of audio.

Based on this representation, matching between fingerprints can be done in numerous different ways. Since the choice of the matching approach or distance metric does not affect the interoperability of different applications using such a fingerprint, this choice is beyond the scope of the MPEG-7 standardization and left up to the individual application.

(Herre *et al.*, 2002) proposes a simple matching of MPEG-7 *AudioSignature* descriptions that is performed based on a vector quantization (VQ) and nearest-neighbour approach. During the training phase, the class generator in Figure 6.3 performs clustering (e.g. LBG vector quantization (Linde *et al.*, 1980)) using a set of training items. The resulting reference codebooks are then stored in the system's class database. During the classification phase, the signal at the output of the feature processor is compared by the classifier with the codebooks stored in the reference database. The item with the shortest distance (matching error) is presented at the system's output as a result. More sophisticated techniques can be used in the MPEG-7 *AudioSignature* framework to increase both matching robustness and speed.

As mentioned above, the scalability of the MPEG-7-based fingerprinting framework comes from the ability to vary some extraction parameters, such as temporal scope, temporal resolution and number of spectral bands. In this way, a flexible trade-off between the compactness of the fingerprint and its recognition robustness can be achieved.

From an application point of view, this is a powerful concept which helps satisfy the need for a wide range of applications by a single framework. More importantly, the fingerprint representation still maintains interoperability so that fingerprints extracted for one application can still be compared with a fingerprint database set up for a different purpose.

The specification for the *AudioSignature* extraction method guarantees worldwide compatibility between all standards-compliant applications. Numerous different applications, such as broadcast monitoring, Internet services or mobile audio identification services using cellular phones, are currently under development.

6.3 AUDIO SIGNAL QUALITY

The description of the “objective” or “subjective” quality of an audio signal is of great interest for many applications. In the following we describe the MPEG-7 tools developed for this purpose.

The MPEG-7 *AudioSignalQuality* descriptor contains several features reflecting the quality of a signal stored in an *AudioSegment* descriptor. The *AudioSignalQuality* features are often extracted without any perceptual or

psycho-acoustical considerations and may not describe the subjective sound quality of audio signals.

The quality information enclosed in *AudioSignalQuality* descriptions could be used, for example, to select the files that should be downloaded among a list of audio documents retrieved on the Internet. More generally, this information helps to decide if a file is of sufficient quality to be used for a particular purpose. The *AudioSignalQuality* can also be used to guide the retrieval of audio documents in a database, based on the quality information.

6.3.1 *AudioSignalQuality* Description Scheme

The *AudioSignalQuality* description scheme is a set of descriptors that have been designed to handle and describe audio signal quality information. In particular the handling of single error events in audio streams is considered. An *AudioSignalQuality* description scheme comprises the following attributes:

- *Operator*: designates the person who is responsible for the audio quality information.
- *UsedTool*: designates the system that was used by the *Operator* to create the quality information. *UsedTool* is stored in a *CreationTool* descriptor.
- *BroadcastReady*: describes whether or not the sound material is ready for broadcasting. *BroadcastReady* is a Boolean parameter (false or true).
- *IsOriginalMono*: describes if a signal was originally mono if it presently has more than one channel. *IsOriginalMono* is a Boolean parameter (false or true).
- *BackgroundNoiseLevel*: contains the estimations of the noise levels in the different channels of a stereo signal.
- *CrossChannelCorrelation*: describes the correlations between the channels of a stereo signal.
- *RelativeDelay*: describes the relative delays between the channels of a stereo signal.
- *Balance*: describes the relative level between the channels of a stereo signal.
- *DcOffset*: describes the mean relative to the maximum of each channel of a stereo signal.
- *Bandwidth*: describes the upper limit of the signal's bandwidth for each channel.
- *TransmissionTechnology*: describes the technology with which the audio file was transmitted or recorded using a predefined set of categories.
- *ErrorEventList*: contains different *ErrorEvent* descriptors. An *ErrorEvent* describes the event time of a specified error type in the signal. The type of error is labelled according to a predefined set of categories.

These quality attributes are detailed in the following sections.

6.3.2 *BroadcastReady*

BroadcastReady indicates if the sound material is ready for broadcasting (value set to “true”) or not (value set to “false”). Its value should generally result from the subjective evaluation of an operator, according to the context of the application. For example, the quality of a piece of audio may be quite bad, but it may be ready for broadcasting (e.g. a historical piece of audio, or a news report recorded in adverse conditions).

6.3.3 *IsOriginalMono*

IsOriginalMono describes whether or not a stereo signal (with N_{CH} channels) was originally recorded as a mono signal. The extraction of the *IsOriginalMono* descriptor is not normative. The standard recommends a method based on the calculation of normalized cross-correlations between the N_{CH} channels. If any of the derived coefficients is greater than a threshold reflecting correlation between the channels, *IsOriginalMono* is set to “true”. Otherwise it is set to “false”.

6.3.4 *BackgroundNoiseLevel*

The *BackgroundNoiseLevel* attribute indicates the noise level within an *AudioSegment*. A noise-level feature is computed separately in each channel of the signal. These values, expressed in dB, are in the range $[-\infty, 0]$. The extraction of the *BackgroundNoiseLevel* for an N_{CH} -channel signal is not standardized. A simple method consists of the following steps:

1. The absolute maximum amplitude $A_{dBmax}(i)$ is computed in dB, for each channel i ($1 \leq i \leq N_{CH}$), as:

$$A_{dBmax}(i) = 20 \log_{10} \left(\max_n |s_i(n)| \right) \quad (1 \leq i \leq N_{CH}), \quad (6.1)$$

where $s_i(n)$ represents the digital signal in the i th channel.

2. The signal is divided into blocks (5 ms long, typically), in which the mean power is estimated as:

$$P_i(j) = \frac{1}{L_B} \sum_{n=(j-1)L_B}^{jL_B-1} s_i^2(n) \quad (1 \leq j \leq \text{number of blocks}),$$

$$(1 \leq i \leq N_{CH}) \quad (6.2)$$

where j is the block index, L_B is the length of a block (in number of samples), and $P_i(j)$ is the mean power of the j th block in the i th channel signal $s_i(n)$.

3. Then, the minimum block power of each channel is computed in dB:

$$P_{dBmin}(i) = 10 \log_{10} \left(\min_j |P_i(j)| \right) \quad (1 \leq i \leq N_{CH}). \quad (6.3)$$

4. Finally, the *BackgroundNoiseLevel* feature of the i th channel is defined by the difference:

$$BNL(i) = P_{dBmin}(i) - A_{dBmax}(i) \quad (1 \leq i \leq N_{CH}). \quad (6.4)$$

The noise level features should be normalized in each channel by the maximum amplitude of the signal, in order to make the descriptor independent of the recording level. Finally, the extraction process yields values, which are stored in a vector, as a summary of the noise level in one *AudioSegment*.

6.3.5 CrossChannelCorrelation

The *CrossChannelCorrelation* attribute describes the correlation between the channels of a signal stored in an *AudioSegment*. It is a measurement of the relationship between the first channel and the $N_{CH} - 1$ other channels of a multi-channel signal, independently of their levels. The extraction of the *CrossChannelCorrelation* features from an N_{CH} -channel signal is not standardized. A possible method consists of the following steps:

1. Each channel is normalized to its maximum value.
2. The cross-correlations $\Gamma_{s1,si}$ between the first channel and the $N_{CH} - 1$ other channels ($2 \leq i \leq N_{CH}$) are estimated.
3. Each correlation $\Gamma_{s1,si}$ is normalized by the geometric mean of the first channel's autocorrelation $\Gamma_{s1,s1}$ and the i th channel's autocorrelation $\Gamma_{si,si}$.
4. Finally, the $(i - 1)$ th *CrossChannelCorrelation* feature is defined as the middle coefficient of the i th channel's normalized cross correlation:

$$Cor(i - 1) = \frac{\Gamma_{s1,si}(0)}{\sqrt{\Gamma_{s1,s1}(0)\Gamma_{si,si}(0)}} \quad (2 \leq i \leq N_{CH}). \quad (6.5)$$

This procedure yields a vector of $N_{CH} - 1$ *CrossChannelCorrelation* features $Cor(i)$ ($1 \leq i \leq N_{CH} - 1$), used to describe the audio segment. Each *CrossChannelCorrelation* feature Cor ranges between -1 and $+1$:

- $Cor = +1$: the channels are completely correlated.
- $Cor = 0$: the channels are uncorrelated.
- $Cor = -1$: the channels are out of phase.

In the case of two sine signals ($N_{CH} = 2$), the *CrossChannelCorrelation* is defined by a unique feature $Cor = \cos(\varphi)$, where φ is the phase shift between the two channels.

6.3.6 RelativeDelay

The *RelativeDelay* attribute describes the relative delays between two or more channels of a stereo signal. The delay values are expressed in milliseconds. They are restricted to the range $[-0.5 \text{ ms}, +0.5 \text{ ms}]$, in order to prevent ambiguity with pitch or other correlations in the signal. The extraction of the *RelativeDelay* features from an N_{CH} -channel signal is not standardized. A possible method consists of the following steps:

1. An *unscaled* cross-correlation function UCC_{s_1, s_i} between the first channel and each of the $N_{CH} - 1$ other channels is estimated as:

$$UCC_{s_1, s_i}(m) = \begin{cases} \sum_{n=0}^{L_S-m-1} s_1(n)s_i(n+m) & (m \geq 0) \\ UCC_{s_1, s_i}(-m) & (m < 0) \end{cases} \quad (2 \leq i \leq N_{CH}), \quad (6.6)$$

where L_S is the length of the input signal in number of samples. The UCC_{s_1, s_i} cross-correlation functions have $2L_S - 1$ coefficients.

2. The extraction system searches the position $m_{max}(i)$ of the maximum of UCC_{s_1, s_i} in a search region corresponding to $\pm 0.5 \text{ ms}$ (defined according to the sampling frequency).
3. The $(i - 1)$ th *RelativeDelay* feature is then estimated for the i th channel by taking the difference between the position of the maximum $m = m_{max}(i)$ and the position of the middle coefficient $m = 0$. This time interval is converted to ms:

$$RD(i - 1) = \frac{m_{max}(i)}{F_s} \quad (2 \leq i \leq N_{CH}), \quad (6.7)$$

where F_s is the sample rate of the input signal.

This procedure yields a vector of $N_{CH} - 1$ *RelativeDelay* features $RD(i) (1 \leq i \leq N_{CH} - 1)$ for the whole audio segment. For a mono signal, a single *RelativeDelay* value is set to 0.

6.3.7 Balance

The *Balance* attribute describes the relative level between two or more channels of an *AudioSegment*. The *Balance* features are expressed in dB within a

$[-100 \text{ dB}, 100 \text{ dB}]$ range. The extraction of the *Balance* features from an N_{CH} -channel signal is not standardized. A possible method consists of the following steps:

1. The mean power is calculated in each channel as:

$$P_i = \frac{1}{L_S} \sum_{n=0}^{L_S-1} s_i^2(n) \quad (1 \leq i \leq N_{CH}), \quad (6.8)$$

where L_S is the length of the input signal.

2. The $(i-1)$ th *Balance* feature is defined by the ratio (in dB) between the first channel's power and the i th channel's power:

$$Bal(i-1) = 10 \log_{10} \left(\frac{P_1}{P_i} \right) \quad (2 \leq i \leq N_{CH}). \quad (6.9)$$

The extraction procedure yields a vector of $N_{CH} - 1$ *Balance* features $Bal(i) (1 \leq i \leq N_{CH} - 1)$ for the whole audio segment. For a mono signal, a single *Balance* value is set to 0.

6.3.8 DcOffset

The *DcOffset* attribute describes the mean relative to the maximum of each channel of an *AudioSegment*. As audio signals should have a zero mean, a DC offset may indicate a bad analogue–digital conversion. The *DcOffset* features take their values in the $[-1, 1]$ range. The extraction of the *DcOffset* features from an N_{CH} -channel signal is not standardized. A possible method consists of the following steps:

1. The mean amplitude is first calculated within each channel:

$$\bar{s}_i = \frac{1}{L_S} \sum_{n=0}^{L_S-1} s_i(n) \quad (1 \leq i \leq N_{CH}), \quad (6.10)$$

where L_S is the length of the input signal.

2. The *DcOffset* features are obtained by normalizing these values by the maximum of the absolute magnitude value in each channel:

$$DC(i) = \frac{\bar{s}_i}{\max_n |s_i(n)|} \quad (1 \leq i \leq N_{CH}). \quad (6.11)$$

The extraction procedure yields a vector of N_{CH} *DcOffset* features $DC(i) (1 \leq i \leq N_{CH})$ for the whole audio segment.

6.3.9 Bandwidth

Bandwidth describes the upper limit of the signal's bandwidth for each channel. The *Bandwidth* features are expressed in Hz, and take their values within the range $[0 \text{ Hz}, F_s/2]$, where F_s is the sample rate of the input signal. These features give an estimation of the original signal bandwidth in each channel. This gives an indication of the technical quality of the original recording.

To extract the *Bandwidth* description from an N_{CH} -channel signal, the following method is proposed. First, the local power spectra of the signal are calculated from successive overlapping frames (e.g. 30 ms frames starting every 10 ms) within each channel. A maximum filter is then used over the local spectra to get a maximum power spectrum $MPS_i(k)$ for each channel. A logarithmic maximum power spectrum (LMPS) is defined in each channel as:

$$LMPS_i(k) = 10 \log_{10}(MPS_i(k)) \quad (1 \leq i \leq N_{CH}). \quad (6.12)$$

A boundary is used to find the edge of the bandwidth of the LMPS of each channel. The maximum value LMP_{max} and minimum value LMP_{min} of each LMPS are calculated. The boundary LMP_{bound} for the upper limit of the bandwidth is set to 70% of $(LMP_{max} - LMP_{min})$ below LMP_{max} . The upper edge of the bandwidth is the frequency BW above which the power spectrum falls below LMP_{bound} . The extraction procedure yields a vector of N_{CH} *Bandwidth* features $BW(i)$ ($1 \leq i \leq N_{CH}$) for the whole audio segment.

6.3.10 TransmissionTechnology

The *TransmissionTechnology* attribute describes the technology in which the audio file was transmitted or recorded. The description uses a predefined set of categories describing different possible transmission and recording technologies.

The extraction of *TransmissionTechnology* has to be made manually by a human operator. The sound can be labelled with 10 categories defined by the standard. The operator has to be familiar with the different transmission or recording technologies in order to choose a proper category. Some categories may pack different types of signals together, which share similar acoustic qualities. For instance, the *Category 6*, as defined by the standard, stands for two distinct types of bad-quality recordings: speech over telephones with a [50 Hz–8 kHz] bandwidth and vinyl before 1960.

6.3.11 ErrorEvent and ErrorEventList

The *ErrorEventList* description contains a list of *ErrorEvent* descriptors. An *ErrorEvent* descriptor is used to describe a type of error occurring in the input audio signal. It consists of the following attributes:

- *ErrorClass*: describes the error type using a predefined set of categories. The standard defines 12 error categories: *Click* (a high-frequency burst of short duration), *ClickSegment* (a segment containing many clicks), *DropOut* (absence of high frequencies for a short period), *Pop* (a low-frequency burst), *DigitalClip* (distortion occurring when a digital signal is clipped), *AnalogClip* (distortion occurring when an analogue signal is clipped), *SampleHold* (click at start and end, short muting of signal), *BlockRepeating* (repetition of a short block), *Jitter* (single sample click), *MissingBlock* (click at the transition caused by missing blocks), *DigitalZero* (click at the transition caused by zero-valued samples) and *Other* (any other error).
- *ChannelNo*: specifies the channel in which the error occurs.
- *TimeStamp*: specifies the temporal location of the error.
- *Relevance*: is the degree of relevance of the error. The possible integer values range from 0 (relevance not specified) to 7 (high relevance). An error with low relevance (e.g. *Relevance* = 1) is hardly audible. An error with high relevance (e.g. *Relevance* = 7) is very disturbing.
- *DetectionProcess*: describes the process of detection: *Manual* or *Automatic*.
- *Status*: describes the current status of the error. This label is set automatically or by a listener. Five labels are possible: *Undefined* (default), *checked* (the error has been checked), *needs restoration* (the error needs to be restored), *restored* (the error has been restored) and *deleted* (the detected error was a false alarm).
- *Comment*: contains any comment about the detected error.

The *ErrorEvent* is used to describe typical errors that occur in audio data, in particular those resulting from an analogue–digital conversion. The *ErrorClass* category may be set manually by a human listener, or automatically extracted from the input signal, for instance through a click detection algorithm. A given audio segment can be indexed with different *ErrorEvent* descriptors due to the *ErrorEventList* attribute.

REFERENCES

- Allamanche E., Herre J., Helmuth O., Fröba B., Kasten T. and Cremer M. (2001) “Content-based Identification of Audio Material Using MPEG-7 Low Level Description”, *International Symposium on Music Information Retrieval*, Bloomington, IN, USA, October.
- Baeza-Yates R. and Ribeiro-Neto B. (1999) *Modern Information Retrieval*, Addison-Wesley, Reading, MA.
- Battle E., Masip J. and Gaus E. (2002) “Automatic Song Identification in Noisy Broadcast Audio”, *International Conference on Signal and Image Processing (SIP 2002)*, Kauai, HI, USA, August.

- Blum T., Keislar D., Wheaton J. and Wold E. (1999) "Method and Article of Manufacture for Content-Based Analysis, Storage, Retrieval and Segmentation of Audio Information", US Patent 5918.223.
- Burges C., Platt J. and Jana S. (2002) "Extracting Noise-Robust Features from Audio Data", *ICASSP 2002*, Orlando, FL, USA, May.
- Cano P., Kaltenbrunner M., Mayor O. and Batlle E. (2001) "Statistical Significance in Song-Spotting in Audio", *International Symposium on Music Information Retrieval (MUSIC IR 2001)*, Bloomington, IN, USA, October.
- Cano P., Batlle E., Kalker T. and Haitsma J. (2002a) "A Review of Algorithms for Audio Fingerprinting", *International Workshop on Multimedia Signal Processing (MMSP 2002)*, St Thomas, Virgin Islands, December.
- Cano P., Batlle E., Mayer H. and Neuschmied H. (2002b) "Robust Sound Modeling for Song Detection in Broadcast Audio", *AES 112th International Convention*, Munich, Germany, May.
- Cano P., Gómez E., Batlle E., Gomes L. and Bonnet M. (2002c) "Audio Fingerprinting: Concepts and Applications", *International Conference on Fuzzy Systems Knowledge Discovery (FSKD'02)*, Singapore, November.
- Chávez E., Navarro G., Baeza-Yates R. A. and Marroquín J. L. (2001) "Searching in Metric Spaces", *ACM Computing Surveys*, vol. 23, no. 3, pp. 273–321.
- Gomes L., Cano P., Gómez E., Bonnet M. and Batlle E. (2003) "Audio Watermarking and Fingerprinting: For Which Applications?", *Journal of New Music Research*, vol. 32, no. 1, pp. 65–81.
- Gómez E., Cano P., Gomes L., Batlle E. and Bonnet M. (2002) "Mixed Watermarking-Fingerprinting Approach for Integrity Verification of Audio Recordings", *International Telecommunications Symposium (ITS 2002)*, Natal, Brazil, September.
- Haitsma J. and Kalker T. (2002) "A Highly Robust Audio Fingerprinting System", *3rd International Conference on Music Information Retrieval (ISMIR2002)*, Paris, France, October.
- Herre J., Hellmuth O. and Cremer M. (2002) "Scalable Robust Audio Fingerprinting Using MPEG-7 Content Description", *IEEE Workshop on Multimedia Signal Processing (MMSP 2002)*, Virgin Islands, December.
- Kalker T. (2001) "Applications and Challenges for Audio Fingerprinting", *111th AES Convention*, New York, USA, December.
- Kenyon S. (1999) "Signal Recognition System and Method", US Patent 5.210.820.
- Kimura A., Kashino K., Kurozumi T. and Murase H. (2001) "Very Quick Audio Searching: Introducing Global Pruning to the Time-Series Active Search", *ICASSP'01*, vol. 3, pp. 1429–1432, Salt Lake City, UT, USA, May.
- Kurth F., Ribbrock A. and Clausen M. (2002) "Identification of Highly Distorted Audio Material for Querying Large Scale Databases", *112th AES International Convention*, Munich, Germany, May.
- Linde Y., Buzo A. and Gray R. M. (1980) "An Algorithm for Vector Quantizer Design", *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95.
- Lourens J. G. (1990) "Detecting and Logging Advertisements Using its Sound", *IEEE Transactions on Broadcasting*, vol. 36, no. 3, pp. 231–233.
- Mihcak M. K. and Venkatesan R. (2001) "A Perceptual Audio Hashing Algorithm: A Tool for Robust Audio Identification and Information Hiding", *4th Workshop on Information Hiding*, Pittsburgh, PA, USA, April.

- Papaodysseus C., Roussopoulos G., Fragoulis D. and Alexiou C. (2001) "A New Approach to the Automatic Recognition of Musical Recordings", *Journal of the AES*, vol. 49, no. 1/2, pp. 23–35.
- RIAA/IFPI (2001) "Request for Information on Audio Fingerprinting Technologies", available at <http://www.ifpi.org/site-content/press/20010615.html>.
- Richly G., Varga L., Kovács F. and Hosszú G. (2000) "Short-term Sound Stream Characterization for Reliable, Real-Time Occurrence Monitoring of Given Sound-Prints", *10th IEEE Mediterranean Electrotechnical Conference (MELECON 2000)*, pp. 29–31, Cyprus, May.
- Subramanya S., Simba R., Narahari B. and Youssef A. (1997) "Transform-Based Indexing of Audio Data for Multimedia Databases", *IEEE International Conference on Multimedia Computing and Systems (ICMCS '97)*, pp. 211–218, Ottawa, Canada, June.
- Sukittanon S. and Atlas L. (2002) "Modulation Frequency Features for Audio Fingerprinting", *ICASSP 2002*, Orlando, FL, USA, May.
- Theodoris S. and Koutroumbas K. (1998) *Pattern Recognition*, Academic Press, San Diego, CA.