MELODIC DESCRIPTION OF AUDIO SIGNALS FOR MUSIC CONTENT PROCESSING

Emilia Gómez Gutiérrez

Research work

PhD Program Informática
i Comunicació digital

Bieni 2000-2002

Universitat Pompeu Fabra

September, 2002

Abstract

The goal of this work is to review how melodic aspects of music are represented in computer applications, and how these representations are extracted from audio signals. This review will serve to define different levels of melodic description. We will use this description scheme to represent melodic aspects of audio in a specific application context, the *Sound Palette* application, a software tool for content-based analysis and transformations. We will also implement and evaluate some techniques for extracting some of these melodic features.

Acknowledgements

First of all, I would like to thank my advisor Xavier Serra, for giving me the opportunity to work at the Music Technology Group and for supervising this research work.

In addition, I would like to thank Perfecto Herrera, who has provided many valuable ideas, feedback and support during this research period.

I am also grateful to everyone at the Music Technology Group for make the everyday work nicer and being also friends for me.

During these years it was a pleasure to collaborate with some people outside the MTG, as Anssi Klapuri, Benoit Meudic and Leandro de C.T. Gomes, on different aspects of my research.

Last, but not least, I would like to thank my family and friends, specially those that I have met in Barcelona, for helping me to discover this wonderful city and region.

"No hay una sola cosa en el mundo que no sea misteriosa,
pero ese misterio es mas evidente en determinadas cosas que en otras.
En el mar,
en el color amarillo,
en los ojos de los ancianos
y en la musica.”
J.L. Borges

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we introduce the problem we want to face and the context in which this research work has been carried out. We also bring up some questions and objectives for the present study.

1.1 Context

1.1.1 The Music Technology Group

The Music Technology Group (MTG)[1] is a research group that belongs to the Audio Visual Institute (IUA)[2] of the Universitat Pompeu Fabra (UPF)[3], in Barcelona. It was founded in 1994 by his current director, Xavier Serra, and it has more than 30 researchers.

From the initial work on spectral modeling, the MTG is dedicated to sound synthesis, audio identification, audio content analysis, description and transformations, interactive systems, and other topics related to Music Technology research and experimentation.

The research at the MTG is funded by a number private companies and various public institutions (Generalitat de Catalunya, Ministerio Español de Ciencia y Tecnología and European Commission).

MTG researchers are also giving classes in different teaching programmes within and outside the UPF: Diploma in Computer Systems, Degree in Computer Engineering, Degree in Audiovisual Communication, Doctorate in Computer Science and Digital Communication, Doctorate in Social Communication, and at the *Escuela Superior de Música de Cataluña* (ESMUC).

1.1.2 The Author

I graduated as a Telecommunication Engineer, specialized in Signal Processing, at the Escuela Superior de Ingenieros, Universidad de Sevilla[4], in 1999.

In 2000, I received a DEA (*Diplôme d'Etudes Approfondies*) in Acoustics, Signal Processing and Computer Science applied to Music (ATIAM) at the Institut de

[1] http://www.iua.upf.es/mtg
[2] http://www.iua.upf.es
[3] http://www.upf.es
[4] http://www.esi.us.es

Recherche et Coordination Acoustique Musique (IRCAM) [5], Paris. During the research period of this DEA, I was a visiting researcher at the Signal and Image Processing (TSI) group of the École National Supérieure de Télécommunications (ENST) [6], Paris. I worked on the field of digital watermarking of musical signals. Since then, I have been collaborating in this subject with this research group and with the InfoCom-Crip5 Group of the René Descartes University[7], in Paris.

From september 2000, I work as a researcher at the Music Technology Group. I am involved in the CUIDADO[8] project, an European project that aims at developing new tools for music information retrieval, and in the TABASCO project, a project with the support of the Spanish Ministry of Science and Technology, intended to develop systems for indexing and transformation of sounds using Artificial Intelligence techniques.

I am a PhD candidate of the doctoral program in Computer Science and Digital Communication, organized by the Department of Technology and the Audiovisual Institute, both of the UPF. This program reflects the research interests of the different groups of the Department of Technology and of the Audiovisual Institute of the UPF: Music Technology, Multimedia Telematics, Distributed Multimedia Applications, Image Processing, Network and Telematics and Artificial Intelligence.

I hold a doctoral grant of the Spanish Ministry of Science and Technology.

A list of muy publications is presented here:

2003

- Emilia Gómez, Anssi Klapuri and Benoit Meudic, *"Melody description and extraction in the context of music content processing"*, to appear in the Journal of New Music Research, 2003.
- Leandro de C. T. Gomes, Pedro Cano, Emilia Gómez, Madeleine Bonnet and Eloi Batlle, *"Audio watermarking and fingerprinting: for which applications?"*, to appear in the Journal of New Music Research, 2003.

2002

- Emilia Gómez, Pedro Cano, Leandro de C. T. Gomes, Eloi Batlle and Madeleine Bonnet, *Mixed Watermarking-Fingerprinting Approach for Integrity Verification of Audio Recordings*, Proceedings of IEEE International Telecommunications Symposium, Natal, Brazil.
- Pedro Cano, Emilia Gómez, Eloi Batlle, Leandro de C. T. Gomes and Madeleline Bonnet, *Audio Fingerprinting: Concepts and Applications*, Proceedings of 2002 International Conference on Fuzzy Systems Knowledge Discovery, Singapore.

2001

- Leandro de C.T. Gomes, Emilia Gómez, Madeleine Bonnet and Nicolas Moreau, *Méthodes de resynchronisation pour le tatouage audio*, Proceedings of 18th Symposium GRETSI'01 on Signal and Image Processing, Toulouse, France.

[5] http://www.ircam.fr
[6] http://www.tsi.enst.fr
[7] http://www.math-info.univ-paris5.fr/crip5/infocom
[8] http://www.cuidado.mu

- Nicolas Durand and Emilia Gómez, *Periodicity Analysis using An Harmonic Matching method and Bandwise Processing*, Proceedings of MOSART Workshop on Current Research Directions in Computer Music, Barcelona.
- Gilles Peterschmitt, Emilia Gómez and Perfecto Herrera, *Pitch-based Solo Location*, Proceedings of MOSART Workshop on Current Research Directions in Computer Music, Barcelona.
- Leandro de C. T. Gomes, Emilia Gómez and Nicolas Moreau, *Resynchronization methods for audio watermarking*, Proceedings of 111th AES Convention, New York, USA.

2000

- Emilia Gómez, *Tatouage de Signaux de Musique (Méthodes de synchronisation)*. DEA ATIAM thesis, ENST (Paris Télécom)-IRCAM (Centre Georges Pompidou).

1999

- Emilia Gómez, *Desarrollo de un software decodificador de MPEG audio sobre una plataforma DSP*, degree thesis, ESI Universidad de Sevilla.

1.2 Music Content Processing

Sometimes it happens that we are looking for a song and we remember neither its title nor its author, but only its chorus's main melody. Sometimes someone is looking for a particular passage of a musical piece, for example an harp arpeggio or a solo of violin. Finally, let's imagine that we have an audio sample and we want to transform it in order to change its tonality from minor to major, increase its tempo or replace an instrument for another one. All these taks are examples of how it would be useful to retrieve and transform music according to different aspects of its *content*.

The word *content* is defined as "*the ideas that are contained in a piece of writing, a speech or a film*" [2]. This concept applied to a piece of music can be seen as the implicit information that is related to this piece and that is represented in the piece itself. Aspects to be included inside this concept are, for example, structural aspects, rhythmic, instrumental, and melodic characteristics of the piece.

The concept of *content-analysis* is defined as the "*analysis of the manifest and latent content of a body of communicated material (as a book or film) through a classification, tabulation, and evaluation of its key symbols and themes in order to ascertain its meaning and probable effect*" [6]. Several techniques are included under the concept of *"Music-content analysis"*, as techniques for automatic transcription, rhythm and melodic characterization, instrument recognition and genre classification; that is, the techniques intended to describe any aspect related to the content of music.

Music Content Processing is a topic of research that has become very relevant in the last few years. The main reason for this is that a great amount of audio material has been made accessible to the home user through networks and other storage supports. This fact makes it necessary to develop tools intended to interact with this audio material in an easy and meaningful way. Many researchers are currently studying and developing techniques aimed at automatically describe and deal with audio data in a meaningful way. There are many disciplines involved in this issue, as signal processing, musicology, psychoacoustics, computer music, statistics and information retrieval.

1.3 Melodic Description

In the context of music content processing, melody plays a major role. *"It is melody that enables us to distinguish one work from another. It is melody that human beings are innately able to reproduce by singing, humming, and whistling. It is melody that makes music memorable: we are likely to recall a tune long after we have forgotten its text"* [103] pp. 4. The importance of melody for music perception and understanding reveals that beneath the concept of melody there are many aspects to consider, as it carries implicit information regarding harmony and rhythm. This fact complicates its automatic representation, extraction and manipulation.

Melodic description techniques are oriented toward the automatic extraction of melodic characteristics of audio. This concept comprises several disciplines. A general schema of the relationships between these disciplines is shown in Figure 1.1. Each of them refers to a different aspect of melody.

Melody together with rhythmic, harmonic, timbre and spatial location information make up the main dimensions for sound description.

1.4 Objectives and Working Thesis

The goal of this research work is to overview the concepts and techniques that are related to *Melodic Description*, i.e.:

1. study how melody has been represented in the literature,
2. review the different techniques and algorithms that have been used to extract this representation, and
3. review the application contexts in which melody description and extraction are needed.

This overview of techniques is made keeping in mind a specific application context, the Sound Palette, a tool for content-based audio edition and transformation that provides some retrieval functionalities (see section 4.6 for description). This tool is being developed within the European IST project CUIDADO (see [3]), that aims at developing content-based technologies. In this research work, we:

- define different levels and aspects of melodic description, grouping the different descriptors used in the literature. The final goal is to define a description scheme with different levels of melodic representation open enough to be adapted to any application context, and
- study the automatic extraction of the fields of this description scheme and implement some of these techniques inside a basic melodic extractor module.

Our working thesis is that we can find different levels and ways of representing melodic aspects of sound, and each application context needs to represent melody from a different point of view.

In particular, some of the questions that we will answer along this study are the following ones:

- is there any fundamental frequency estimation method appropriate for any type of sounds in any application context?

- can we propose an all-usage representation of the melodic features of a music piece?
- can the MPEG-7 standard (explained in section 2.3) be used or extended in our specific application context, the *Sound Palette* application?
- how does the use of some musical knowledge helps in melody description?

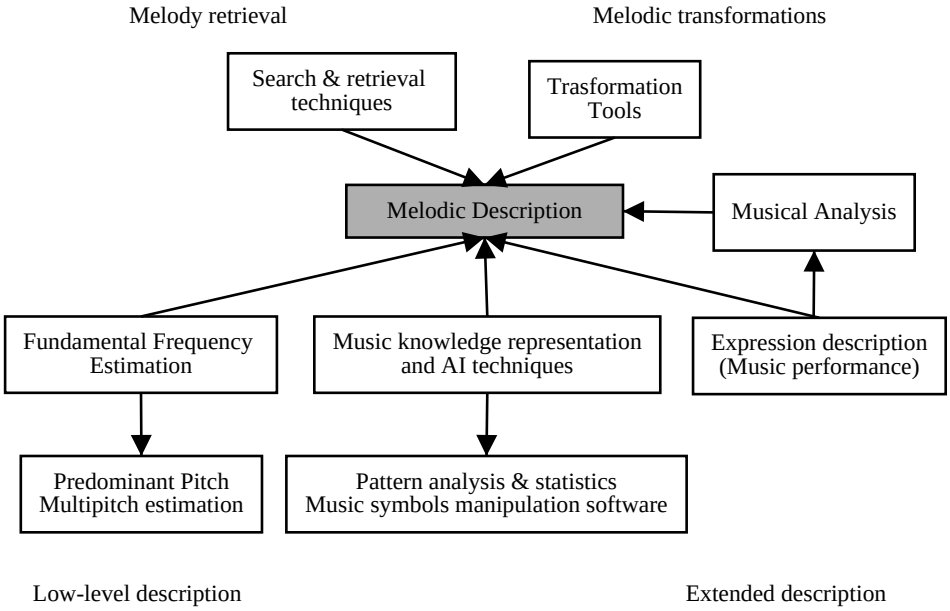


Figure 1.1: Disciplines and research fields related to Melody Description

This research work focuses on melodic description of monophonic audio signals (that is one instrument without accompaniment), although some of the presented techniques have been designed to work with polyphonic sounds.

We follow a bottom-up approach to represent melodic aspects of sound. We begin from physical and signal-related features, up to more abstract representations connected with temporal structure, musical description or cognitive aspects of melody. We divide the work in three blocks, dealing with melodic representation, melodic description extraction and applications of melodic description techniques.

This report is organized in the following way. In chapter 2, the review of the different melodic representations in the literature drives us to define different aspects of melodic description. In chapter 3, we describe the recurrent techniques used to extract this melodic description: from pitch detection to a MIDI-like representation, and toward a multi-level melodic description. Chapter 4 shows the different application contexts in which melodic analysis and representation is used. In chapter 5, we explain the work that is currently being carried out on the development of some of these techniques. Finally, both the objectives of the future research and the guiding principles to carry it out are explained, which at the same time, work as the conclusion of this research work.

Part of chapters 2, 3 and 5 is part of the following publications: [37, 91], and what will appear in [42, 41, 40].

Chapter 2

Melody Representation

2.1 Melody Definition

The goal of this section is to make a review of the different ways of defining what a melody is and how it can be described. It is very difficult to give a clear definition of melody. Several definitions can be found in the literature, considering different aspects of melodic characterization.

Melody in general has been defined as an auditory object that emerges from a series of sound transformations along the six dimensions: pitch, tempo, timbre, loudness, spatial location, and reverberant environment [58].

Most of the time the concept of melody is associated to a sequence of pitch notes. This definition can be found for example in [106]: *"a combination of a pitch series and a rhythm having a clearly defined shape"*, and on Grove Music [5]: *"pitched sounds arranged in musical time in accordance with given cultural conventions and constraints"*. Goto also considers the melody as a sequence of pitches, the most predominant detected pitches at middle and high frequency regions, in opposition to bass line, that can be found at low frequency bands [46][45].

Melody can also be represented as a set of unary features or attributes, when trying to answer the question *"which are the melodic features of this audio excerpt?"*, that characterize the melodic properties of sound. Several aspects to be considered are for example key, melodic profile, melodic density (that is the degree of melodic activity), interval distribution and tessitura (or pitch range) (see section 2.2 for a review on different descriptors related to melody). Sometimes melody is also associated with the concept of unity, considering melody as an arrangement of single tones into a meaningful sequence. This definition is close to the concept of **phrase**.

In the definition proposed in [4], melody is defined with some of these different connotations: as a *set of musical sounds in a pleasant order and arrangement*, as a **theme** (*"sequence of single tones organized rhythmically into a distinct musical phrase or theme"*), and as a song, **tune**, or phrase (*"a short musical composition containing one or a few musical ideas"*).

As shown in the previous definition, melody can be also related to the concept of tune or predominant theme, that is usually defined as something recurrent, neither too short nor too long, where all the notes are usually played by the same instrument and it is often located at the top of the register. *"Theme is an excerpt from an arbitrary portion of work (not necessarily the beginning) which enjoys the greatest melodic impor-*

tance" [103]. Other related concepts are the **prototypical melody** (if every statement of a theme is slightly varied, find the prototype behind), the **motive** (rhythmic and/or melodic fragments characteristic to a composer, improviser, or style) and the **signature** (term for motives common to two or more works of a given composer [30]). These notions show up the fact that melody, apart from being a sequential phenomenon, has also a hierarchical structure (motive, phrase, theme, etc.).

2.2 Multi-level Melody Representation

If we begin to review how to abstract melodic features of an audio file, we find different levels of representation, beginning at the signal level up to descriptors related to musical or structural aspects.

Our goal is to build a melodic representation from sound, that is, we extract this representation from the audio signal. We emphasize that many techniques for melodic description already extract the features from another level of representation (or symbolic representation), as for example methods that work with *Musical Instrument Digital Interface* (MIDI) representations. This means that, in order to use these techniques, we first have to define a MIDI-like representation of the audio signal. Methods to extract this representation from audio are presented in chapter 3.

After reviewing the features used to describe melody in the literature [42], we came up with the idea that there exist different levels of abstraction to describe melody. This idea can also be found in [33, 70, 96, 120, 121]:

Dannenberg [33] proposes to think of musical representations at different levels, from most abstract (printed music) to most concrete level (the audio signal). He mentions the need of a multiple-hierarchy scheme to represent music, that should be extensible to support new concepts and structures.

Rolland et al. [98] define what they call a *Multidescription Valued Edit Mode* model (MVEM) that establishes three levels of description: individual, local, and global. Information at the individual level concerns a single note or rest, described by features related to duration and pitch. Local descriptions contain information on a segment, or musical phrase, as, for example, the melodic profile, the contour, etc. General information about the entire piece, such as the key or the average pitch, is given at the global level. Nevertheless, this last group of features could also be associated to a single phrase or audio segment.

Widmer [120, 121] also distinguishes between note-level and multi-level parameters to characterize music performances. According to him, some aspects of performances are relevant at single note level, while others may only make sense when viewed as parts of larger, more abstract patterns.

According to Leman et al. [70, 71], different representational levels for music exist, and they provide cues for content representation of musical audio. It is pointed out the need of a conceptual architecture for melodic representation in which these different levels can be handled in a flexible way. They are currently developing this conceptual architecture within the MAMI project (explained in section 4.1.2), distinguishing three levels of concepts:

- **Low-level** concepts describe content that is close to the acoustical or sensorial properties of the signal. This group of features includes frequency, intensity as well as roughness, onset, loudness. They are typically related to temporal features of the signal and local (non-contextual) properties.

- **Mid-level** concepts involve time-space transformations and context dependencies within a time scale of the musical present. This is the level where time-space transformations may allow for the specification of the musical signal content in spatial terms (timbre, pitch, chords..) and temporal terms (beat, meter, rhythmic pattern).
- **High-level** concepts involve learning and categorization beyond the representation of the "now". This level deals with structure and interpretation, and may be related to cognitive, emotional or affective issues. The concepts may involve meanings that are not directly connected to signal features but with subjective feelings and user-dependent interpretations.

All these approaches have in common the following:

1. They distinguish descriptors associated to different types of segments, according to a temporal structure: single events, notes, phrase segment and entire piece. That means that the different levels of representation are connected in some way to **structural** aspects of music.
2. There are some descriptors related to signal attributes and other ones related to musical terms, that is, **musical knowledge** is taken into account for the definition of some of the features.
3. **Perceptual** issues are also considered, as well as context and subjective features, for music description.

Now, we are going to make a list of the descriptors that are associated to those different aspects of melodic representation.

- **Low-level descriptors**: in this group, we include signal-related descriptors, some of them being already defined in the MPEG-7 standard (see section 2.3 for details). We refer here to descriptors that are related to the physical properties of the sound. In turn, they can be divided in three different subgroups, as Leman proposes [70]:
  - Sample-level descriptors: they are descriptors related to the audio sample, as the audio waveform itself.
  - Frame-level descriptors include descriptors derived from an analysis of an audio frame, i.e. a determined number of audio samples possibly weighted by a window.
  - Model-specific descriptors, that are descriptors derived from modeling techniques of frame-level parameters. In this group, we can include parameters related to some modeling techniques related to Artificial Intelligence or Spectral Models as SMS [10]: Hidden Markov Models or Neural Networks parameters, sinusoid amplitude and frequencies, number of sinus, residual analysis parameters, etc.
- **Note descriptors** or event-related descriptors, also called individual descriptors by Rolland et al. [98]. In this group we can consider descriptors associated to a note segment, assuming that a segmentation into note has been performed. The first note feature that represents melody within a note is the pitch, that has been used in many application contexts. As an example, the system proposed by [82]

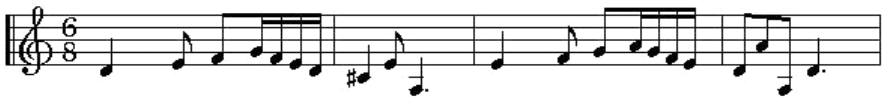
uses the note pitches as the information for melodic transcription. Other features are note duration, vibrato, attack duration, intensity, and lyrics and phonemes if considering singing voice.

- Descriptors related to **structural** aspects: in this level of description, a temporal structure is imposed to the entire audio segment. Considering the segment as a sequence of notes, they can be grouped into patterns (see section 3.5 for a review on techniques) or motives. If we introduce some musical knowledge (that is, considering also the **musical** aspects that are presented below), we can also perform segmentations having a musical meaning, as phrase segmentation (see section 3.4). These descriptors are then related to the following issues:
  - Motive analysis
  - Repetitions
  - Patterns location
  - Phrase segmentation (also related to the musical knowledge).

- **Perceptual** aspects: Human perception should be taken into account for representing melody in a meaningful way. As a consequence of studying how humans perceive the melody, many features have been defined. For instance, as a forward step from pitch information, pitch contour information has also been used in several applications as query by humming, similarity matching or melodic classification [58, 76]. A pitch contour describes a series of relative pitch transitions, an abstraction of a sequence of notes, and it has been found to be more significant to listeners in determining melodic similarity. It can also include rhythm information, as also found in [88], where both pitch and duration sequences are used to define the melody (see Figure 2.1). Different degrees of precision have been used in representing pitch intervals, varying from up/down discrimination to a semitone precision and even beyond (distinguishing enharmonic differences). The amount of other parameters in processing the melody is heterogeneous, as well. The earliest approaches disregarded timing information completely, but more recent studies have shown that durational values may sometimes out-weight pitch values in facilitating melody recognition [103, pp. 31]. Smith et al. studied the length of a melodic query sequence needed to uniquely define a melody in a database of 9,400 melodies [105]. Different pitch interval precisions (interval, approximate interval, contour) and with/without durational values were tried. As an obvious result, the more precise the representation is, the shorter the query excerpt needed is (about five notes for the most precise). More importantly, however, contour matching together with durational values were more definitive than exact pitch intervals only.

  Taking into account perception also gives sense to the definition of some statistical descriptors that are found to be more significant to humans (see statistical descriptors below).

- **Musical knowledge**. When musical knowledge is taken into account, some descriptors are relevant, as for example:
  - Key or tonality related descriptor, as the *key* descriptor defined by MPEG-7 (see section 2.3).



Pitch sequence	[293.66, 329.63, 349.23, 392, 349.23, 329.63, 293.66, 277.18, 329.63, 220, 329.63, 698.46, 783.99, 440, 783.99, 349.23, 329.63, 293.63, 440, 220, 293.66]
Duration sequence (multiples of the eighth note duration)	[2, 1, 1, 0.5, 0.5, 0.5, 0.5, 2, 1, 3, 2, 1, 1, 0.5, 0.5, 0.5, 0.5, 1, 1, 1, 3]
Interval Sequence (in semitones)	[2, 1, 2, -2, -1, -2, -1, 3, -7, 7, 1, 2, 2, -2, -2, -1, -2, 7, -1, 2, 5]

Figure 2.1: Melody Description Example

  - Scale information, that determine which type of scale is used (diatonic, chromatic, pentatonic, etc) and is also defined by the standard (see section 2.3).
  - Cadence information
  - Phrase segment definition

  Musical knowledge also gives sense to the definition of some statistical descriptors that are related to musical aspects (see Statistical descriptors below).

- **Statistical descriptors**. These are descriptors derived from a numerical analysis of other descriptors. This type of descriptors have been used in many applications, as comparative analysis [114, 38], melody retrieval [63, 64, 62, 116, 27], and algorithmic composition [115]. Some of these descriptors are computed using other features related to structural, musical or perceptual aspects of sound. Some of these descriptors are computed from note descriptors, that is, they have the need of a note segmentation algorithm (see 3.2 for review on approaches). Nevertheless, in this statistical level we can also consider descriptors that are computed as statistics from frame or sample descriptors. One example are the pitch content features proposed by Tzanetakis [116].

  - Number of notes, or number of notes per second, giving an idea of Melodic density, another relevant descriptor.
  - Pitch features:
    * Pitch variety: Measure of diversity of the pitch class set used in writing the melody:
    $$PV = \frac{NumDistinctPitch}{NumPitch} \tag{2.1}$$
    * Pitch range or tessitura.
    * Pitch Histograms: In [116] pp. 49-51, pitch histograms are used as description features for audio retrieval. These histograms are computed from the frame-level descriptors, so that note segmentation is not needed. The three dominant peaks of an enhanced autocorrelation

function (SACF) over the signal envelope are accumulated into a Pitch Histogram (PH) over the whole audio excerpt (i.e. for all the analysis frames). The frequency f corresponding to each histogram peak is converted into MIDI note number n by the equation:

$$n = \log 2\frac{f}{440} + 69 \tag{2.2}$$

Two versions of the PH are created: a *folded* (FPH) and *unfolded* (UPH) histogram. The unfolded histogram bin, or MIDI note number, holds information about the pitch range of the audio excerpt. In the folded case, all notes are mapped to a single octave using the equation:

$$c = nmod12 \tag{2.3}$$

c represents a pitch class or chroma value. Finally, this FPH is mapped to a circle of fifths histogram so that adjacent histogram bins are spaced a fifth apart (7 semitones) rather than a semitone. This mapping is achieved by the following equation:

$$c\prime = (7 \times c)mod12 \tag{2.4}$$

Some features are computed from these histograms:

- Amplitude of maximum peak of the histogram: this descriptor gives an idea of the predominant pitch class used.
- Period of the maximum peak of the unfolded histogram: this descriptor corresponds to the octave range of the dominant musical pitch.
- Period of the maximum peak of the folded histogram: this features represents the main pitch class of the audio excerpt.
- Pitch interval between the two maxima of the folded histogram: this interval codes the main tonal interval relation.
- The overall sum of the histogram: this feature measures the strength of the pitch detection.

– Tonality features:

  * Key centered: It represents the proportion of quanta (being quantum the shortest duration note) where the pitch is primary, that is either tonic or dominant, giving an indication of how strongly the melody has a sense of the key.

$$KC = \frac{NumPrimaryPitchQuanta}{NumQuanta} \tag{2.5}$$

  * Non-scale notes: Indication of how strongly tonal the melody is.

$$NonScale = \frac{NumNonScaleQuanta}{NumQuanta} \tag{2.6}$$

  * Dissonant intervals: fraction of dissonant intervals.

$$DI = \frac{NumAllIntervalDissonances}{NumNotes - 1} \tag{2.7}$$

Dissonance rating (interval in semitones):

$$interval \in [0, 9], 12\ semitones \rightarrow rating = 0.0 \quad (2.8)$$

$$interval = 10\ semitones \rightarrow rating == 0.5 \quad (2.9)$$

$$interval = 6, 1, >= 13\ semitones \rightarrow rating = 1.0 \quad (2.10)$$

  * Tonal stability: correlation between the tone profile of the melody and the tonal hierarchy profile (C major probe-tone profile).

- Contour features:
  * Melodic profile: type of profile: ascendant, descendant or constant.
  * Contour direction: defined as the overall tendency of the melody to rise or fall.

$$CD = \frac{\sum RisingIntervals}{\sum Intervals} \quad (2.11)$$

  * Contour stability: proportion of intervals for which the following interval is in the same direction. This is a measure of stability in melodic direction.

$$CS = \frac{\sum ConsecutiveIntervalsMovingInTheSameDirection}{\sum Intervals} \quad (2.12)$$

  * Movement by step: proportion of intervals that are diatonic steps. A high score indicates a smooth melodic curve with few large leaps. A diatonic step interval will be one or two semitones. Rests are ignored.

$$MbS = \frac{NumDiatonicSteps}{NumIntervals} \quad (2.13)$$

  * Leap returns: proportion of large (leap) intervals NOT followed by a return interval. A large leap is greater than or equal to 8 semitones (minor 6th). The returning interval must be at least 1 semitone but less than the leap interval preceding it.

$$LR = \frac{Num\ of\ large\ leaps\ not\ followed\ by\ a\ return\ interval}{Num\ of\ large\ leaps\ intervals} \quad (2.14)$$

  * Climax Strength: measured as the inverse of the number of times the climatic note is repeated in the melody. The highest value of 1 for this feature occurs when the climatic note is used only once. More frequent use lessens the climatic impact.

$$CS = \frac{1}{Num\ uses\ of\ climatic\ note} \quad (2.15)$$

- Note repetitions:
  * Repeated pitch: measure of the repeated pitch intervals (that is when both notes of interval are equal). Measure of melodic movement.

$$RP = \frac{Num\ of\ repeated\ pitch\ intervals}{Num\ of\ intervals} \quad (2.16)$$

    - Repeated pitch patterns of several notes: number of repeated patterns of n notes. This descriptor is derived from the structural descriptors, and is also related to the melodic movement.

$$RP(n) = \frac{Num\ of\ repeated\ note\ pitch\ sequences}{Num\ of\ notes - n - 1} \tag{2.17}$$

  - Distribution features: wide group of features where we include descriptors related to the distribution of other computed descriptors:
    - Distribution of tones: also included in the group of Pitch features (described above), as for example the mean pitch.
    - Interval distribution: as, for example, the mean interval or proximity of notes.
    - Tone duration distribution.
    - Two-tones transition distribution: registral direction.
    - Duration transition distribution.
  - Quality of successive intervals: some features intended to describe successive intervals are:
    - Mean proximity of tones
    - Registral return
    - Registral direction
    - Closure
    - Intervalic difference
    - Consonance

- **Expressivity** related descriptors: in music performances, musicians do not play exactly what is written in the score. They deviate from the score and enrich the sound with vibrato, tremolos, etc. All these aspects of sounds can be considered inside a group of descriptors related to **expressivity**. Some of these expressivity aspects are coded using low-level descriptors, note descriptors (as vibrato rate and frequency of note, articulation, attack duration and type, etc), or descriptors having a musical meaning (e.g. representing crescendos and decrescendos).

  Camurri et al. [23] define some features for expressivity representation in order to synthesize different emotions, represented in Figure 2.2 [1]. These features are:

  - Average Tempo: metronome value expressed as percentage variation from the metronomic value of a *cold* performance.
  - Note Dynamics: expressed by the average of the difference between the starting and the maximum dB values plus the difference between the maximum and the ending values.
  - Phrase Dynamics: calculated by the variance of the average of the difference between the sustain values of all the notes.
  - Articulation: local aspects of legato and staccato. Average of:

$$Art. = \frac{IOI - DR}{DR} \tag{2.18}$$

[1] this figure can be found at http://musart.dist.unige.it/sito_inglese/research/r_current/expressive.html

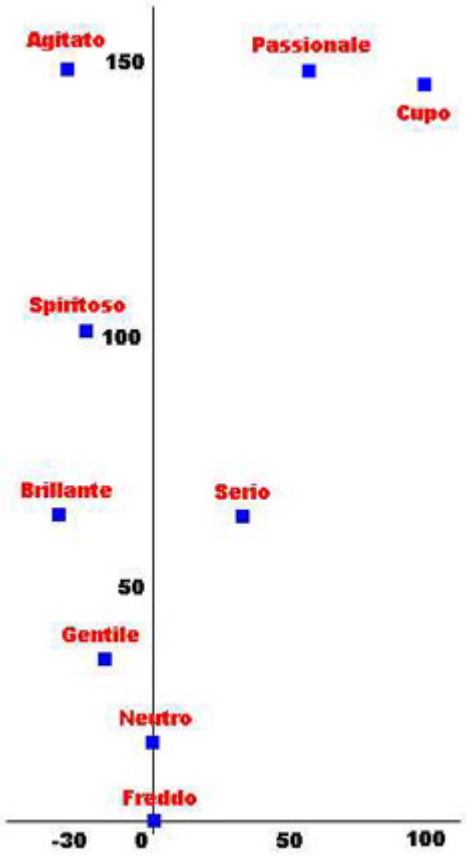


Figure 2.2: The expressiveness space

where DR is the note nominal duration, and IOI is the Inter-Onset Interval, representing the real note duration.

– Accelerando/Ritardando: local rhythmical variations in respects to the nominal duration of the current note. Average of:

$$Acc.Rit = |DR - \Delta(1+\mu)DR_n| \tag{2.19}$$

where DR_n is the current note nominal duration, Δ is the duration (in seconds) of full bar note used as reference (taken from the cold performance) and μ is the *Average Tempo* value defined previously.

– Vibrato: index taken from the average value of the amplitude vibrato effect for each note.

Some deviations from the score have been described by researchers at KTH [9] in terms of context dependent rules that are used to synthesize performances. The magnitude of the effect generated by a specific rule is determined by the rule quantity k. If the quantity is high, the effect is great and salient.

Performance rules affect duration of tones, loudness, change the exact pitch or the vibrato. Other rules create crescendos and diminuendos, change the tempo, or insert minute pauses between tones. The effects induced by a rule can be so subtle that they are very hard to perceive, or they can be so coarse that they catch the ear.

They define three different types of performance rules:

- *Differentiation Rules*: these rules enhance the differences between scale tones and between note values.
  - Duration Contrast: this rule derives from the duration contrast found between long and short note values, such as half note and eighth note. This contrast is sometimes enhanced by musicians. They play the short notes shorter and the long notes longer than nominally written on the score.
  - Melodic Charge: in traditional tonal music, there is no equality between scale tones. Some are remarkable and some are trivial. For example, the root of the chord is trivial while the augmented fourth above the root is very special. The melodic charge of a note is considered as its remarkableness. The duration, loudness and vibrato extent of the tones vary according to the melodic charge.
  - High sharp: This rule plays high tones sharp and low tones flat, just like some musicians tend to do when they play.
- *Grouping Rules*: these rules establish groups of notes at several levels: constituting melodic gestures, phrases, etc. The rules mark, by means of micropauses and lengthening of tones, the boundaries between all these tone groups. This group of rules is related to the structural-level of melodic description.
  - Punctuation: the smallest unit in the hierarchical musical structure is tone groups of up to about 7 tones, so-called melodical gestures. When playing it, it is important to mark the boundaries between such gestures. This rule automatically identifies these melodical constituents by means of a patented set of context dependent rules. The boundaries are marked micropauses.
  - Double Duration: tone groups in two note values having the ratio of 2:1 alternate are not played according to their nominal durations. Instead, the short tone is lengthened at the expense of the long note. This rule induces this effect.
  - Tuning: musicians tend to enhance differences between scales by playing the major third interval wide and the minor third interval narrow. This fact is associated to this performance rule, that produces a slight variation in the fundamental frequency of notes.
  - Phrase Arch: according to the structural-level, music has a hierarchical structure, so that small units, such as melodical gestures, join to form subphrases, which join to form phrases, etc. When musicians play, they mark the endings of these tone groups. This rule marks the phrase and subphrase endings by creating minute accelerandos and decelerandos within phrases and subphrases according to a parabolic function. Thus it increases the tempo in the beginnings and decreases it toward

the endings. The loudness is changed similarly creating crescendos and diminuendos.

  - Inégales (or swing): this rule lengthens the stressed notes in sequences of notes having the same note value. This effect is used in Baroque music and jazz. The duration relations between the stressed and unstressed notes can be changed.
  - Final Ritard: when a piece comes to its end, musicians generally slow down the tempo somewhat. The slowing down generally follows a parabolic.
  - Harmonic Charge: just as with the scale tones, the harmonies in traditional Western tonal music are not equal: there are *trivial* chords and *fantastic* chords. Harmonic charge is a concept reflecting the remarkableness of chord in its harmonic context. It is a weighted sum of the chord tones' melodic charges, using the root of the main chord of the key, that is the root of the tonic, as the reference.

- *Ensemble Rules*: these rules keep the order in ensembles. They achieve synchronization by lengthening and shortening the individual tones in the various voices according to a common overall strategy and they achieve fine tuning of successive and simultaneous intervals.

Combinations of performance rules and of their parameters can be used for synthesizing interpretations that differ in emotional quality. A proper selection of rules and rule parameters can produce a wide variety of meaningful, emotional performances, even extending the scope of the original rule definition.

Widmer et al. [120, 121] also focus in these dimensions of expressive variations: tempo and timing, dynamics and articulation to generate formal models of music expressivity.

As explained in section 1.4, we have not considered polyphonic representations, that should also take into account multiple pitches, chord and harmony representation.

2.3 The MPEG-7 Standard

A separate section has been devoted to MPEG-7, as it constitutes a relevant effort to define an unified standard for music content description. We concentrate here in describing how MPEG-7 deals with melody in different abstraction levels. Information related to MPEG-7 melody description scheme can be found in [1, 7], and is explained in [77]. Our intention is to review it here.

The Moving Picture Experts Group (MPEG) [8] is a working group of the International Standard Organization/International Electrotechnical Committee (ISO/IEC) in charge of developing standards for coded representation of digital audio and video. Established in 1988, the group has produced MPEG-1, MPEG-2, MPEG-4, MPEG-7, and now is working on the new standard MPEG-21 "Multimedia Framework" since June 2000. More details on the standards can be found in [48].

MPEG-7 is formally named “Multimedia Content Description Interface”, and it is the standard that performs multimedia content description utilities for browsing and retrieval of audio and visual content.

This standard provides normative elements, as Descriptors (also Ds), Description Schemes (also DSs) and a Description Definition Language (also DDL). The Descriptors define the syntax and the semantic of the represented features. The Description Schemes are used to group several Ds into structured and semantic relationships and units between other Ds and DSs. The Description Schemes are specified using the DDL.

The *Audio* part of the standard relies on two basic structures: the *segment*, inherited from the *Multimedia Description Scheme*, that allows to define a temporal structure of the audio signal, and the *scalable series*, a type that is inherited by all the low-level descriptors. It then distinguishes two classes of structures, the generic audio description framework and the application-related tools. The first one is defined as the basic compatibility layer upon with generic descriptions and unique applications may be built for any signal, and it includes *low-level* descriptors (LLDs), the *scalable series* scheme, and the *silence segment*. The second one includes sound recognition, instrumental timbre description, spoken content description and melody description tools, as well as tools for audio matching.

The MPEG-7 standard proposes a melodic DS that includes melody as a sequence of pitches or contour values, plus some information about scale, meter, beat and key (see Figure 2.3).

In the MPEG-7 scheme, the melodic contour uses a 5-step contour (from -2 to +2) in which intervals are quantized, and also represents basic rhythm information by storing the number of the nearest whole beat of each note, which can dramatically increase the accuracy of matches to a query. This contour has been found to be inadequate for some applications, as melodies of very different nature can be represented by similar contours. One example is the case of having a descendant chromatic melody and a descendant diatonic one. Both of them have the same contour although their melodic features are very dissimilar.

For applications requiring greater descriptive precision or reconstruction of a given melody, the Melody Description Scheme (or Melody DS) supports an expanded descriptor set and higher precision of interval encoding. Rather than quantizing to one of five levels, the precise pitch interval (with cent or greater precision) between notes is kept. Precise rhythmic information is kept by encoding the relative duration of notes defined as the logarithm of the ratio between the differential onsets, following the formula below:

$$NoteRelDuration[n] = log_2\left(\frac{Onset[n+1] - Onset[n]}{Onset[n] - Onset[n-1]}\right), \text{for } n \geq 2 \quad (2.20)$$

$$NoteRelDuration[1] = log_2\left(\frac{Onset[2] - Onset[1]}{0.5}\right) \quad (2.21)$$

Arranged around these core descriptors are a series of optional support descriptors such as lyrics, key, meter, and starting note, to be used as desired for an application.

This expanded description does not take into account silence parts that sometimes play an essential role for melodic perception. Except for lyrics and phonemes, the description could be completely extracted from the score, and no information related to the signal level would be necessary. Some examples of melodic description can be found in the MPEG schema specification documents [1] and [7].

2.4 Relationship with other description axes

In order to describe melodic aspects of music, it is important to take into account other aspects of music description that are normally closely related to melody.

Rhythm: Melody description has always been associated to rhythm description. Melodic representation usually includes features related to rhythm as note durations, tempo, beat, etc. In the MPEG-7 standard, there are only some descriptors related to rhythm description, which are *meter* and the *beat* descriptor associated to each note. Both are included in the *Melody* description scheme. This is a good example of interconnection between rhythmic and melodic characterization.

Instrument: In a first moment, the melodic features of an audio excerpt do not depend on the instrument or the instruments that are playing this audio excerpt. Nevertheless, some aspects of melodic perception are related to the instrument and melodic characterization is sometimes performed jointly with instrument analysis. The MPEG-7 standard also proposes tools to describe timbre as a perceptual phenomena. Complementary to this, one could think on defining tools for instrument labelling and categorical search. This issue arises in [41]

2.5 Representation coding

Once a description scheme is defined, another subject appears, that is the storage and management of these descriptions in an efficient way.

MPEG-7 adopted the XML syntax to store MPEG-7 descriptions, as well as the XML Schema Language with some extensions as the MPEG-7 Description Definition Language (DDL). This language provides descriptive tools by which users can create their own Description Schemes (DSs) and Descriptors (Ds).

The XML syntax is used in two different formats: a textual and a binary one. This binary format allows the compressing of MPEG-7 descriptions, because textual XML format is oriented to human reading, but it is not efficient for storage and transmission purposes. For more details, please refer to [1, 7, 48].

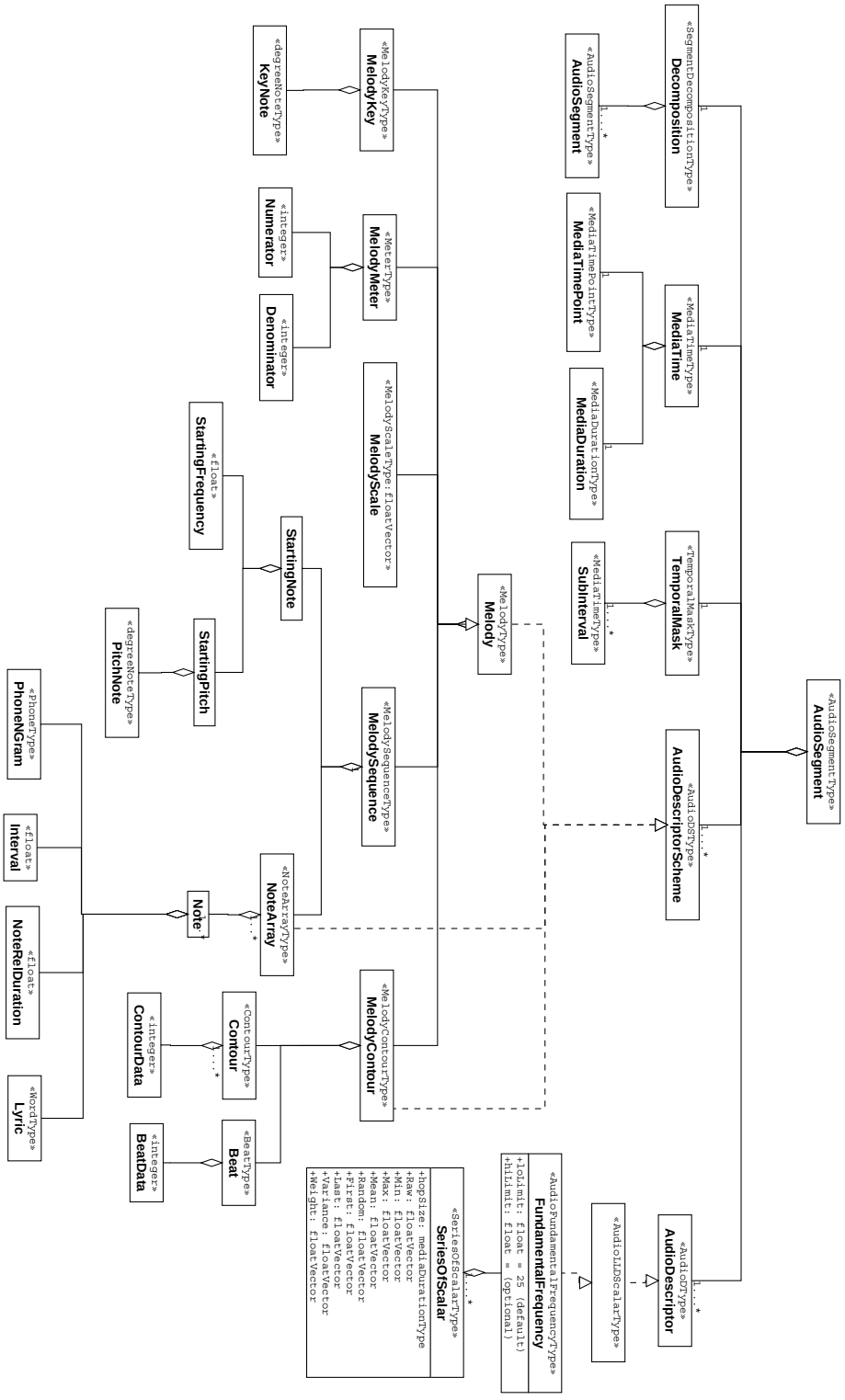


Figure 2.3: MPEG-7 melody description scheme

Chapter 3

Methods for Melody Extraction

The aim of this chapter is to describe the recurrent techniques used to extract this melodic description: from pitch detection to MIDI, and toward a multi-level melodic description.

3.1 Pitch Estimation

Fundamental frequency is the main Low-Level Descriptor to be considered when describing melody. Due to the significance of pitch detection for speech and music analysis, a lot of research has been made on this field. We can also find several surveys and evaluations of pitch detection algorithms for different purposes, as for example [52, 95, 99, 60].

3.1.1 Fundamental Frequency Estimation for Monophonic Sounds

The first solution adopted for fundamental frequency detection of musical signals was to adapt the techniques proposed for speech [52]. Later, other methods have been specifically designed for dealing with music.

There are many ways of classifying the different algorithms. One could classify them according to their processing domain. Following this rule, we can discriminate between time-domain algorithms, dealing with the signal in time domain, and frequency-domain algorithms, that use the signal in frequency domain (the spectrum of the signal). This distinction between time-domain and frequency-domain algorithms is not always so clear, as some of the algorithms can be expressed in both (time and frequency) domains, as the Autocorrelation Function (ACF) method. Another way of classifying the different methods, more adapted to the frequency domain, could be to distinguish between spectral place algorithms and spectral interval algorithms, classification proposed by Klapuri in [60]. The spectral place algorithms, as the ACF method and the cepstrum analysis (explained in this section), weight spectral components according to their spectral location. Other systems, as those that are based on envelope periodicity or spectrum autocorrelation computation, use the information corresponding to spectral intervals between components. Then, the spectrum can be arbitrary shifted without affecting the output value. These algorithms work relatively well for sounds that exhibit inharmonicity, because intervals between harmonics remain more stable than the places for the partials.

All the fundamental frequency estimation algorithms give us a measure corresponding to a portion of the signal (analysis frame). According to McKinney [52], the fundamental frequency detection process can be subdivided into three main steps that are passed through successively: the preprocessor, the basic extractor, and the post-processor (see Figure 3.1). The basic extractor performs the main task of measurement: it converts the input signal into a series of fundamental frequency estimates. The main task of the pre-processor is data reduction in order to facilitate the fundamental frequency extraction. Finally, the post-processor is a block that performs more diverse tasks, such as error detection and correction, or smoothing of an obtained contour.

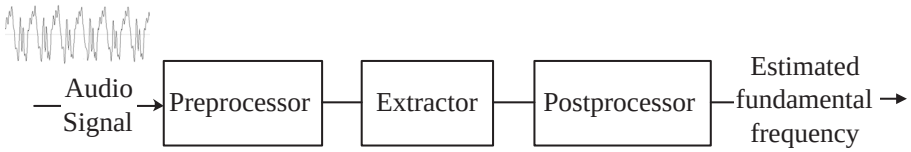


Figure 3.1: Steps of the fundamental frequency detection process

Time-domain algorithms

These algorithms try to find the periodicity of the input sound signal in the time domain.

- **Zero-crossing rate (ZCR)**: ZCR is among the first and simplest techniques toward estimating the frequency content of a signal in time domain, and consists on counting the number of times the signal crosses the 0-level reference in order to estimate the signal period. This method is very simple and inexpensive but not very accurate when dealing with noisy signals or harmonic signals where the partials are stronger than the fundamental. The value of ZCR has also been found to correlate strongly with spectral centroid, also called spectral balancing point, which is the first moment of spectral power distribution. It follows that the value of ZCR has more to do with timbre than with pitch. Due to these reasons, this method and its variants are not very much used for fundamental frequency estimation.

- **Autocorrelation (ACF)**: Time-domain Autocorrelation function based algorithms have been among the most frequently used fundamental frequency estimators. The ACF of a sequence x(n) of length K is defined as:

$$r(n) = \frac{1}{k} \sum_{k=0}^{K-n-1} x(k) \cdot x(k+n) \tag{3.1}$$

  The maximum of this function corresponds to the fundamental frequency for periodic signals. The autocorrelation can also be computed in frequency domain [60]. First, x(k) is zero-padded to twice its length and transformed into the frequency domain using a short FFT algorithm. Then, the square of the magnitude spectrum is obtained, and transformed back to the time domain. The autocorrelation function can be expressed as:

$$r(n) = \frac{1}{K} \sum_{k=0}^{K-1} \left[|X(k)|^2 \cdot \cos\left(\frac{2\pi nk}{K}\right) \right] \quad (3.2)$$

According to the classification made by Klapuri in [60] as the ACF weight spectral components according to their spectral location, systems that perform this type of fundamental frequency detection can be called spectral place type fundamental frequency estimator. In this algorithm, "twice-too low" octave errors are likely to occur, since integer multiples of the fundamental frequency $n \cdot f_0$ also have positive weights at the harmonics frequencies. "Too high" octave errors are not probable, since in that case off harmonics get a negative weight.

ACF based fundamental frequency detectors have been reported to be relatively noise immune [99] but sensitive to formants and spectral peculiarities of the analyzed sound [60].

One good example of fundamental frequency detection using correlation methods is [83], that tries to maximize the cross-correlation function over the range of feasible pitch values. This algorithm was tested on synthetic and real speech data covering a large range of speakers and a full range of pitch frequencies. A latter example of algorithms based on cross-correlation based is the Robust Algorithm for Pitch Tracking by Talkin [110]. The algorithm uses a two-phase normalized cross correlation function (NCCF) calculation between successive segments of the input signal. Through the use of cross correlation instead of autocorrelation functions, these algorithms achieve a relatively good time resolution even for low-pitched sounds.

- **Envelope periodicity**: the idea behind this model is derived from the observation that signals with more than one frequency component exhibit periodic fluctuations in its time domain amplitude envelope. The rate of these fluctuations depends on the frequency difference of each two frequency components. In the case of a harmonic sound, interval f_0 will dominate and the fundamental frequency is clearly visible in the amplitude envelope of the signal. This algorithm is spectral interval oriented and makes implicit spectral smoothing, because the amplitude envelope computation filters out single clearly higher amplitude harmonic partials [60]. It follows that smooth spectrum causes strong beating. A third property of EP models is that they are phase sensitive in summing up the harmonic partials.

  The most recent models of human pitch perception calculate envelope periodicity separately at distinct frequency bands and then combine the results across channels [84]. These methods attempt to estimate the perceived pitch, not pure physical periodicity, in acoustic signals of various kinds. The algorithm proposed by E. Terhardt represents an early and valuable model [111, 112]. Except for the simplest algorithms, that only look for signal periodicity, "perceived pitch" estimators use some knowledge about the auditory system at their different steps: when preprocessing, extracting or post processing data. Then, they could be considered as pitch estimators. However, as the psychoacoustic knowledge is only applied to better accurate the periodicity estimation and no complete model of pitch perception is applied, some auditory phenomena are not explained.

- **Parallel processing approach**: the fundamental frequency detector defined by Gold and later modified by Rabiner was designed to deal with speech signals [44,

94]. This algorithm has been successfully used in a wide variety of applications and it is based on purely time domain processing. The algorithm has three main steps:

1. The speech signal is processed so as to create a number of impulse trains that retain the periodicity of the original signal and discard features that are irrelevant to the pitch detection method. This can be considered as the pre-processing part of the algorithm.
2. Simple estimators are used to detect the period of these impulse trains.
3. All the estimates are logically combined to infer the period of the speech waveform.

The particular scheme proposed by Gold and Rabiner is presented in Figure 3.2.

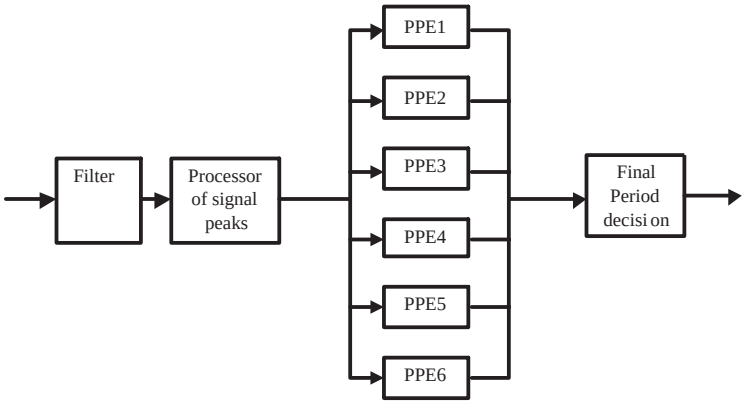


Figure 3.2: Parallel processing approach

The fundamental approach of having several processes working in parallel is unique and interesting here. This path has been little explored, but is plausible from the human perception point of view, and might be very fruitful. Bregman remarks: *"I believe that there is a great deal of redundancy in the mechanisms responsible for many phenomena in perception"* [17]. Several different processes analyze the same problem, and when one fails, the other succeeds.

This algorithm has a very low computational complexity and it gives relatively good performances.

Frequency-domain algorithms

These algorithms try to find the fundamental frequency using the spectral information of the signal, obtained by a short-time Fourier Transform or another transformation.

- **Cepstrum analysis**: Noll [89] introduced this idea for pitch determination of speech signals. The cepstrum pitch detection algorithm was the first short-term

analysis algorithm that proved realizable on a computer. This algorithm worked as a calibration standard for other, simple, algorithms. The cepstrum is the inverse Fourier transform of the power spectrum logarithm of the signal. By the logarithm operation, the source and the transfer functions are separated. Consequently, the pulse sequence originating from the periodicity source reappears in the cepstrum as a strong peak at the "quefrency" (lag) T_0, which is readily discovered by the peak-picking logic of the basic extractor. Some modifications were made to the algorithm in order to speed up the analysis. Nevertheless, these modifications elucidate the role of certain pre-processing methods, such as center clipping (see postprocessing methods section below).

Cepstrum fundamental frequency detection has close model level similarity with autocorrelation systems. The only difference to frequency domain ACF calculations is that the logarithm of the magnitude spectrum is used instead of second power. Cepstrum can also be considered as a spectral place type algorithm, with "too low" octave errors. Unlike ACF systems, cepstrum pitch estimators have been found to perform poorly in noise and to have good performances with formants and spectral peculiarities of the analyzed sounds [60, 99].

- **Spectrum autocorrelation**: the idea of these methods is derived from the observation that a periodic but non-sinusoidal signal has a periodic magnitude spectrum, the period of which is the fundamental frequency. The goal is to detect the period of the magnitude spectrum using its autocorrelation function. It can be called spectral interval type fundamental frequency estimator [60]. "Too low" octave errors are not probable since there is no spectral periodicity at half the fundamental frequency rate. But "twice too high" octave errors are likely to occur, since doubling the true spectral period picks every second harmonic of the sound. A nice implementation of this principle can be found in [66].

- **Harmonic Matching Methods**: these algorithms try to extract a period from a set of spectral maximum of the magnitude spectrum of the signal. Once these peaks in the spectrum are identified, they can be compared to the predicted harmonics for each of the possible candidate note frequencies, and a measure to fit can be developed. One of the first developed methods was [92], which tries to find the best fitting harmonic number for each component pair of the spectrum. A particular fitness measure is described on [78] as a "Two Way Mismatch" procedure. For each fundamental frequency candidate, mismatches between the harmonics generated and the measured partials frequencies are averaged over a fixed subset of the available partials. A weighting scheme is used to make the procedure robust to the presence of noise or absence of certain partials in the spectral data. The discrepancy between the measured and predicted sequences of harmonic partials is referred as the mismatch error. The harmonics and partials would "live up" for fundamental frequencies that are one or more octaves above and below the actual fundamental; thus even in the ideal case, some ambiguity occurs. In real situations, where noise and measurement uncertainty are present, the mismatch error will never be exactly zero.

  The solution presented on [78] is to employ two mismatch error calculations. The first one is based on the frequency difference between each partial in the measured sequence and its nearest neighbor in the predicted sequence (see Figure 3.3). The second is based on the mismatch between each harmonic in the predicted sequence and its nearest partial neighbor in the measured sequence.

This two-way mismatch helps to avoid octave errors by applying a penalty for partials that are present in the measured data but are not predicted, and also for partials whose presence is predicted but which do not actually appear in the measured sequence. The TWM mismatch procedure has also the benefit that the effect of any spurious components or partial missing from the measurement can be counteracted by the presence of uncorrupted partials in the same frame.

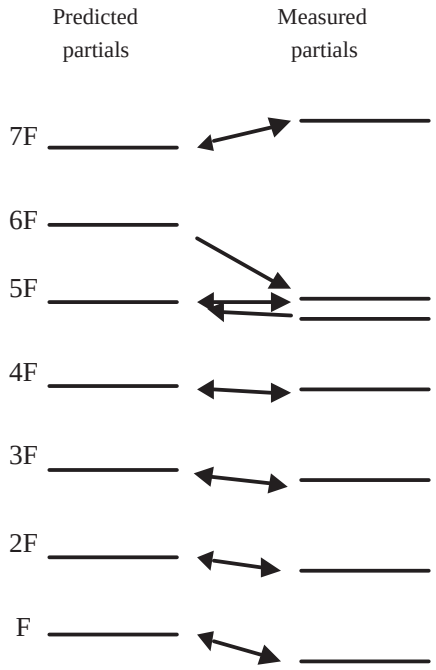


Figure 3.3: Two-Way Mismatch procedure

The two error measurements are computed as follows:

– **Predicted-to-measured mismatch error**:

$$E_{p\to m} = \sum_{n=1}^{N} E_{\omega}(\Delta f_n, f_n, a_n, A_{max}) = \tag{3.3}$$

$$= \sum_{n=1}^{N} \Delta f_n \cdot f_n^{-p} + \left(\frac{a_n}{A_{max}} \times \left[q \cdot \Delta f_n \cdot f_n^{-p} - r \right] \right) \tag{3.4}$$

where a_n, f_n correspond to the amplitude and frequency of the predicted partial number n, A_{max} is the maximum amplitude, and Δf_n is the difference between the frequency of the predicted partial and its closest measured partial.

– **Measured-to-predicted mismatch error**:

$$E_{m\to p} = \sum_{k=1}^{K} E_\omega(\Delta f_k, f_k, a_k, A_{max}) = \tag{3.5}$$

$$= \sum_{k=1}^{K} \Delta f_k \cdot f_k^{-p} + \left(\frac{a_k}{A_{max}} \times \left[q \cdot \Delta f_k \cdot f_k^{-p} - r \right] \right) \tag{3.6}$$

where a_k, f_k correspond to the amplitude and frequency of the measured partial number k, A_{max} is the maximum amplitude, and Δf_k is the difference between the frequency of the measured partial and its closest predicted partial.

The total error for the predicted fundamental frequency is then given by:

$$Err_{total} = Err_{p\to m}/N + \rho Err_{m\to p}/K \tag{3.7}$$

The parameters (p,q,r,ρ, etc) are set empirically. This is the method used in the context of Spectral Modeling Synthesis (SMS) [24] including some improvements, as having pitch dependent analysis window, a selection of spectral peaks to be used, and an optimization in the search for fundamental frequency candidates.

Another harmonic matching method is the one described in [36], based on a maximum likelihood for the fundamental frequency. After detecting the maxima of the spectrum, the probability that a maximum corresponds to a partial rather than to a noise is computed for each detected peak. To find the optimal solution, it proceeds in two steps: first, it computes the interval that contains the optimal solution, and then it obtains the precise optimal value for the fundamental frequency within this interval.

To determine which interval contains the optimal solution, the value of likelihood is computed for each of the selected intervals of the frequency axis. Then, more precise optimal values are obtained using a regression on the frequencies of the signal partials that are matched with the harmonic partials.

- **Wavelet based algorithms**: the wavelet transform (WT) is a multi-resolution, multi-scale analysis that has been shown to be very well suited for music processing because of its similarity to how the human ear processes sound. In contrast to the short-time Fourier transform, which uses a single analysis window, WT uses short windows at high frequencies and long windows for low frequencies. This is the spirit of the constant Q $\left(\frac{\Delta f}{f_c}\right)$ frequency analysis. Some wavelet based fundamental frequency algorithms have been proposed for speech analysis. They can be adapted to music analysis, as the one proposed by [56] for voice signals. The idea is to filter the signal using a wavelet with derivative properties. The output of this filter will have maxima where GCI (Glottal Closure Instant) or zero crossings happens in the input signal. After detection of these maxima, the fundamental frequency will be estimated as the distance between consecutive maxima. This filtering function will combine the bandwidth properties of the wavelet transform at different scales.

- **Bandwise processing algorithm**: following with the idea of the constant Q frequency analysis, Klapuri [60] has proposed an algorithm for periodicity analysis that calculates independent fundamental frequencies estimates at separate frequency bands. Then, these values are combined to yield a global estimate. This solves several problems, one of which is inharmonicity. In inharmonic sounds, as stretched strings, the higher harmonics may deviate from their expected spectral positions, and even the intervals between them are not constant. However, we can assume the spectral intervals to be piece-wise constant at narrow enough bands. Thus we utilize spectral intervals to calculate pitch likelihoods at separate frequency bands, and then combine the results in a manner that takes the inharmonicity into account. Another advantage of bandwise processing is that it provides robustness in the case of badly corrupted signals, where only a fragment of the whole frequency range is good enough to be used. A single fast Fourier transform is needed, after which local regions of the spectrum are separately processed. The equalized spectrum is processed in 18 logarithmically distributed bands that extend from 50Hz to 6000Hz. Each band comprises a 2/3-octave wide region of the spectrum that is subject to weighting with a triangular frequency response. Overlap between adjacent bands is 50%, which makes the overall response sum to unity. Fundamental frequency likelihoods are calculated at each band, and the values are combined taking into account that fundamental frequency can increase as a function of the band center frequency for string instruments. Some improvements were made to provide robustness in interference, where pitch is observable only at a limited band, and to adapt to signals containing a mixture of several harmonic sounds.

In Table 3.1 we can find a summary of the listed algorithms. The information about the performances has been extracted from different sources as [60, 99] or from the authors' related work.

Pitched/unpitched decision

Fundamental frequency detection algorithms suppose that a fundamental frequency is present, and it is not always the case. We could have segments where no pitch can be found, as for instance at silences, percussion solos or noise segments. A segmentation process should distinguish between voiced and unvoiced periods, so that the fundamental frequency detection will be only performed for pitched parts of the signal.

However, most of the techniques used for pitched/unpitched segmentation already use the estimated fundamental frequency to decide whether this information is valid or corresponds to an unpitched signal, in addition to other features computed from the signal.

One example appears in the SMS context (see [24]), where the portions of a sound that cannot be well represented with the harmonic model are considered as noise. There is a strict segmentation in time with routines that used the error measure of the TWM procedure along with other measures that are easily computed as part of the SMS analysis: zero crossing, energy, noisiness and harmonic distortion.

Preprocessing methods

The main task of a preprocessor is to suppress noise and to enhance the features that are useful for fundamental frequency estimation. The fundamental frequency signal

Method	**Domain**	**Spectral Place/ Interval**	**Simplicity**	**Noise**	**Inharmonicity**	**Spectral Peculiarities**
ZCR	Time	SP	Very simple			
ACF	Both	SP	Simple	Relatively immune		Sensitive
EP	Time	SI	Simple			
Rabiner	Time	SP	Relatively simple			
Cepstrum	Frequency	SP	Simple	Poor Performance		Relatively immune
Spectrum AC	Frequency	SI	Simple			
Harmonic Matching Method	Frequency	both	Quite complex	Relatively immune		Relatively immune
Wavelet based method	Frequency (WT)		Quite complex	Immune		
Bandwise EP	Time	both	Quite complex	Rather immune	Relatively immune	Relatively immune
Bandwise Klapuri	Frequency	both	Quite complex	Relatively immune	Relatively immune	Relatively immune

Table 3.1: Summary table of the different methods for fundamental frequency estimation

from a physical vibrator is usually first filtered by passing through resonance structures and the environment, and then linearly superpositioned with other co-occurring sounds and noise. The first type of interference can be called convolutive noise, and the latter type additive noise. Both should be removed to reveal the underlying fundamental frequency.

Convolutive noise suppression is usually called spectral whitening, since it aims at normalizing away all spectral peculiarities of the sound source and the environment, but leaving the spectral fine structure (fundamental frequency) intact. Inverse linear predictive filtering is a common way of performing this task. Suppression of additive noise is usually done by subtracting an estimate of the noise spectrum in power spectral domain. Additive noise spectrum can be estimated e.g. by calculating the median of filter bank output levels over time.

Some of the preprocessing methods used in speech processing are detailed in [52], as isolation of the first partial, moderate low-pass filtering to remove the influence of higher formants, inverse filtering to approximate to the excitation part, comb filtering to enhance harmonic structures, application of a filter bank, non linear processing in the spectral domain, center clipping (that destroys the formant structure without destroying the periodic structures of the signal), signal distortion by an even nonlinear function, envelope detection, instantaneous-envelop detection, and algorithmic temporal structure investigation. These algorithms can also be used for fundamental frequency detection of music signals.

Some preprocessing methods have been defined for musical signals. The method

proposed by Klapuri applies the principles of RASTA spectral processing [61, 50] in removing both additive and convolutive noise simultaneously. After obtaining the signal spectrum, a non-linear transformation is applied to the spectrum $X(k)$ to yield $X'(k) = ln\,[1 + J \cdot X(k)]$. With a proper scaling of the spectrum, additive noise goes through a linear-like transform, whereas spectral peaks, affected by convolutive noise, go through a logarithmic-like transform. By subtracting a moving average over a Bark-scale spectrum from $X'(k)$, both convolutive and additive noise are suppressed in a single operation.

Post-processing methods

The fundamental frequency contour that is the output of the different algorithms is normally noisy and sometimes badly affected with isolated errors, so different methods for correcting them have been defined.

The most usual way to smooth a function is the convolution of the input signal with the impulse response of a low-pass filter. Since the smoothing function (window) usually is of very short length, this convolution can be reduced to the weighted addition of few samples. Since the convolution is linear, we speak of linear smoothing. As it is presented in [52], the application of low pass filters removes much of the local jitter and noise, but it does not remove local gross measurements errors, and, in addition, it smears the intended discontinuities at the voiced-unvoiced transitions. Hence, some kind of non-linear smoothing might be more appropriate. In a paper by [93], median smoothing is proposed as a non-linear method. They recommended the use of a combination of median and linear smoothing in that order, because median removes short errors, and the linear smoothing removes jitter and noise.

Another approach is described in [68]. The procedure consists of storing several possible values for the fundamental frequency for each analysis frame, assigning them a score (for example, the value of the normalized autocorrelation or the TWM error) that represents the estimation goodness. The goal is to find a "track" that, following one of the estimations for each frame, will have the best score. The total score is computed using the score of the estimations considering penalizations if, for example, an abrupt fundamental change is produced. This optimal "track" can be obtained using dynamic programming techniques or using some Hidden Markov Models (as in the method proposed by Doval and Rodet). This approach minimizes the abrupt fundamental frequency changes (octave errors, for example) and gives good results in general. Its main disadvantage is that there is a delay in the estimation, as we use past and future fundamental frequency values.

Other post-processing methods are dependent on the algorithm used for sound analysis and fundamental frequency detection. One example is the approach used by SMS. Fundamental frequency is estimated using spectral peak information. Spectral peaks are computed frame by frame using windowing and FFT analysis. In the windowing procedure, window size is updated depending on the fundamental frequency. If an estimation error is produced, then the window size for the following frames will not be correctly chosen. In order to correct this type of errors, a reanalysis is made over a group of frames beginning at the last one and finishing at the first one.

Fundamental frequency history and future frames are also used to choose between candidates with the same TWM error (see [24]), having a smooth evolution with neighbor frames. The phase of the peaks can finally be useful to modify the search among fundamental frequencies candidates.

In this case, the pitch contour is smoothed according to sound properties, in opposition to median techniques.

Fundamental frequency tracking using other knowledge sources

In this section, we discuss the use of available meta-data for guiding the fundamental frequency detection and tracking process. In this point, we study the applicability of using content information for refining the process of monophonic fundamental tracking.

Content information has been used in the form of internal sound source models [57]. Martin [80] also used musical rules to transcribe four-voice polyphonic piano pieces. When some assumption about the type of signal is made or when the algorithm is adapted to some of the signal properties, we are also taking advantage of some information that can be considered as context information.

- **Timbre**: We have seen that the different algorithms work better for different sound sources in different conditions. If the algorithm chosen depends on the instrument, we could consider that some timbre (meta-data) information may be used. Also some preprocessing methods can be dependent on the instrument played. Goto discriminates melody and bass lines using different band-pass filters [46, 45].

  Another possibility could be to adapt some of the parameters of an algorithm (for example, the TWM algorithm presented in [78]) to the played instrument (e.g. considering the inharmonicity of the piano or the particularity of the spectra of some instruments). Some of these ideas are the basis of some of the multitimbre pitch detectors [46, 45, 12].

- **Rhythm**: Rhythm information has not been considered in any fundamental frequency detector. One idea could be to use information about rhythm as note duration for detecting pitch changes, provided rhythmic information could be computed first and independently.

- **Melody**: Melody description is made mostly using pitch information, so melodic information as a pitch sequence should not be available for fundamental frequency detection tracking. However, if some higher-level melodic information is present (as for example melodic profile, scale, key, etc), this could be used for fundamental tracking. For instance, key or scale information could be used to give a higher expectancy to fundamental frequency values that match the scale.

- **Genre**: Some knowledge about the genre of music that is being played or some musical knowledge could also be used. For example, "atonal" changes should not be found in most of genres, certain styles use only a restricted set of scales, etc.

3.1.2 Multipitch Estimation

It is generally admitted that single-pitch estimation methods are not appropriate as such for multipitch estimation [60], although some of the algorithms used in monophonic pitch detection can be adapted to simple polyphonic situations. In [12], it is described how some of the methods applied to monophonic fundamental frequency estimation can be adapted to polyphony. Also, the TWM procedure can be extended to duet

separation, as explained in [78], trying to find two fundamental frequencies that best explain the measured spectral peaks.

Multipitch estimation is oriented toward auditory scene analysis and sound separation: if an algorithm can find the pitch of a sound and not get confused by other co-occurring sounds, the pitch information can be used to separate the partials of the sound from the mixture. Indeed, the most successful multipitch estimation methods have applied the principles known from human auditory organization.

Kashino et al. implemented these principles in a Bayesian probability network, where bottom-up signal analysis could be integrated with temporal and musical predictions [57]. A recent example following the same principles is that of Wamsley & Godsill [119], who use the Bayesian probabilistic framework in estimating the harmonic model parameters jointly for a certain number of frames. Godsmark and Brown have developed a model that is able to resolve melodic lines from polyphonic music through the integration of diverse knowledge [43]. The system proposed by Goto [46, 45] is more application-oriented, and is able to detect melody and bass lines in real-world polyphonic recordings by making the assumption that these two are placed in different frequency regions.

Other methods are listed in [60], and a system is described following an iterative method with a separation approach. This algorithm operates reasonably accurately for polyphonies at a wide fundamental frequency range and for a variety of sound sources.

The state-of-the-art multipitch estimators operate reasonably accurately for clean signals, the frame-level error rates progressively increasing from two percent in two-voice signals up to about twenty percent error rates in five-voice polyphonies. However, the performance decreases significantly in the presence of noise, and the number of concurrent voices is often underestimated. Also, reliable multipitch estimation requires significantly longer time frames (around 100 ms) than single-pitch estimation [60].

3.2 Note Segmentation

Once the fundamental frequency has been estimated, we need to delimitate the note boundaries, in order to extract a sequence of notes and the descriptors associated to note segments.

There are several methods that have been designed to perform note segmentation. If we consider that the fundamental frequency has been correctly estimated and post-processed to eliminate isolated errors, the simplest approach is to use this information to estimate the note limits. The audio excerpt is then segmented according to the fundamental frequency variations. One limitation of this method is the difficulty to differenciate two consecutive notes with the same pitch. The use of energy, in addition to fundamental frequency, solves this problem. These two low-level features has been used for note segmentation by McNab et al. in [82].

An important issue to consider when segmenting notes is the presence of amplitude and fundamental frequency modulations. When using fundamental frequency, we have to take into account the presence of vibrato, in order to distinguish note transitions from frequency modulations. Rossignol, in [100] chapter 1, proposes to eliminate vibrato from the fundamental frequency trajectory as a preprocessing step before note segmentation. He defines a set of observation functions related to fundamental frequency, energy, voicing coefficient and spectral "flux" (related to the variation on the spectrum magnitude between two consecutive analysis frames). A decision function is defined for each of these observation functions, by detecting the peaks correspond-

ing to note transitions. Finally, an overall decision function is defined, based on the individual ones.

As a forward step, some techniques have evolved toward bandwise processing, that is, the use of features computed in different frequency bands. One relevant example is the system proposed by Klapuri [59] for detecting onsets. The results of the different frequency bands are combined using a psychoacustical model of intensity coding. This seems to be more adapted to the human auditory system.

The methods mentioned above belong are considered as feature-based segmentation techniques, according to the classification made in [51] pp. 3. We refer to this document for a complete review of audio segmentation techniques.

3.3 Extracting Melody from Note Sequences

Considering melody as a pitch sequence, several cases make the automatic extraction of melody very difficult [88]:

- A single line played by a single instrument or voice may be formed by movement between two or more melodic or accompaniment strands.
- Two or more contrapuntal lines may have equal claim as "the melody".
- The melodic line may move from one voice to another, possibly with overlap.
- There may be passages of figuration not properly considered as melody.

This problematic also appears in [103] section 1.1.3.

We have presented above several methods that aimed at tracking pitches. As output, the algorithms provided pitch sequences. In this part, we will present some approaches, which attempt to identify the notes of the pitch sequences that are likely to correspond to the main melody.This task can be considered not only for polyphonic sounds, but also for monophonic sounds. Indeed, monophony could have notes that do not belong to the melody (as for example grace notes, passing notes or the case that we had several interleaved voices in a monophony).

In a first step, in order to simplify the issue, one should try to detect note groupings. This would provide heuristics that could be taken as hypothesis in the melody extraction task. For instance, experiments have been done on the way the listener achieves melodic groupings in order to separate the different voices (see [102] p.131 and [81]).

Other approaches can also simplify the melody extraction task by making assumptions and restrictions on the type of music which is analyzed. Indeed, melody extraction depends not only on the melody definition, but also on the type of music from which we want to extract the melody. For instance, methods can be different according to the complexity of the music (monophonic or polyphonic music), the genre (classical with melodic ornamentations, jazz with singing voice, etc) or the representation of the music (audio, midi etc).

Uitdenbogerd [117] has worked on MIDI files containing channels information. She has evaluated some algorithms that extract from a MIDI file a sequence of notes that could be called the melody. The first algorithm considers the top notes of the sequence as the melody. The others combine both entropy information and structure of the MIDI file. However, the first algorithm gives the best results.

According to Uitdenbogerd [117], "there does not appear to have been much research on the problem of extracting a melody from a piece of music". Indeed, it appears

that very few researches focus on this area in comparison to the interest that is granted to other tasks such as melody matching and pattern induction.

3.4 Melodic Segmentation

The goal of melodic segmentation techniques is to establish a temporal structure on a sequence of notes. This structure is specified by a set of pair of boundaries that divides the audio excerpt in a serie of audio segments. It may involve different levels of hierarchy, as those defined by Lerdahl and Jackendoff [73], and may include overlapping segments and unclassified areas that do not belong to any segment.

One relevant method proposed by Cambouropoulos [19] and refined in [21] is the *Local Boundary Detection Model* (LBDM). This model computes the transition strength of each interval of a melodic surface according to local discontinuities. The peaks of this function, which gives a measure of the transition probability, are considered as segment boundaries. This method is based on two rules: the *Change Rule* (hence CR) and the *Proximity Rule* (hence PR). The CR is proportional to the degree of change between two consecutive intervals, that is, for an idential succession of intervals the boundary strength is zero. This rule can be implemented by a *degree-of-change* function. Regarding the PR, if two consecutive intervals are different, the boundary introduced on the larger interval is proportionally stronger. This means that each boundary is weighted according to the size of its absolute interval, so that segment boundaries are located at larger intervals. This rule can be implemented by multiplying the *degree-of-change* function with the absolute value of the interval.

In [21], Cambouropoulos uses not only pitch, but also temporal (inter-onset intervals) and *rest* intervals (between current onset and previous offset). He compared this algorithm with the punctuation rules defined by Friberg et al. from KTH [9], getting coherent results. The LBDM has beeen used by Melucci and Orio [86, 85] for content-based retrieval of melodies.

Another approach is found in the *Grouper* [1] module of the *Melisma* music analyzer, implemented by Temperley and Sleator (see section 4.2). This module uses three criteria to select the note boundaries. The first one considers the *gap score* for each pair of notes, that is, the sum of the inter-onset interval (or IOI) and the offset-to-onset interval (OOI), that corresponds to the *rest* parameters of the LBDM. Phrases receive a weight proportional to the *gap score* between the notes at the boundary. The second one considers an optimal phrase length in number of notes. The third one is related to the metrical position of the phrase beginnig, relative to the metrical position of the previous phrase's beginning. Metrical position is defined as *the number of beats between the previous level 3 beat and the beat of the phrase beginning*.

Spevak et al. [108, 113] have compared several algorithms for melodic segmentation: maintly the LBDM, the Melisma Grouper, and a memory-based approach, the Data-Oriented Parsing (DOP) from Bod [16]. They also describe other approaches to melodic segmentation. According to Spevak et al. [108, 113], *melodic segmentation is an ambiguous affair. For a given melody, it is typically not possible to determine one 'correct' segmentation, because the process is influenced by a rich and varied set of context, where local structure (gestalt principles), higher-level structure (e.g. recurring motives, harmony, melodic parallelism) and style-dependent norms and the breaking of these norms all impact what is perceived as salient*. To explore this issue,

[1] http://www.link.cs.cmu.edu/music-analysis/grouper.html

they have compared manual segmentation of different melodic excerpts, where participants were asked to markup these scores, identifying between which pairs of notes they thought that a phrase and/or subphrase boundary occurred. In [108], results from manual segmentation and the different algorithms are compared in four experiments. Many aspects have to be taken into account, as the algorithm stability, its flexibility, the ambigüity of the segmentation task (related to the agreement between experts), and the algorithm parameters. We refer to [108] for a detalied comparison according to these different aspects.

Melodic segmentation is very much connected to pattern extraction. An overview of approaches is made below, in section 3.5.

3.5 Melody Pattern Extraction

This section addresses a very general issue that is encountered in any application manipulating data sequences (music, text, video, data coding, etc). It consists in comparing and measuring the similarity between two different sequences of events. In the following sections, we will associate the concept of melody matching to the notion of pattern induction (or extraction). Indeed, comparing two melodies and extracting a musical pattern from a sequence have common aspects: in both tasks, one needs to perform similarity measurements.

We will propose some approaches that could be particularly interesting in the context of melody analysis.

Non music-oriented approaches

Several non music-oriented disciplines deal with pattern induction (extraction) or pattern matching. However, most of the algorithms that are employed cannot be directly used in a musical context. Indeed, musical patterns must be determined considering various musical dimensions, for example temporal, cognitive, and contextual ones.

For instance, we can consider a data compression algorithm, the Lempel-Ziv algorithm [122], which has already proved to have musical applications [69]. The Lempel-Ziv algorithm proceeds as follows: a sequence of elements is read from the beginning to the end element by element. At each step, the current element, if different from the previous ones, is placed in a database. If the element is already in the database, then the pattern constituted by the element and its following is considered, etc. The method seems interesting as it filters some patterns among all the possible patterns. The question is: are the detected patterns linked with a musical structure?

One could answer that the motivic and melodic structures are not preserved. Indeed, once a pattern is detected, it cannot be changed by the new information given by the following elements of the sequence, so the positions of the patterns in the sequence are arbitrarily determined by the order in which the sequence is analyzed. Moreover, possible hierarchical relations between the patterns are not taken into account, which seems contradictory with the musical organization of motives. However, one advantage is that the method is incremental and analyzes each event of the sequence by considering the only already analyzed ones, which may be linked with the conditions in which we listen to music.

This illustrates the fact that the specifics of music (temporal aspect, polyphonic aspect, hierarchical aspect, etc.) must be considered before adapting any algorithm to a musical purpose. Several studies have been done on this subject. For instance,

Cambouropoulos [20] discusses some of the issues relative to the application of string processing techniques on musical sequences, paying special attention to pattern extraction.

Music-oriented approaches

According to [101], p. 168, "Music is composed, to an important degree, of patterns that are repeated and transformed. Patterns occur in all of music's constituent elements, including melody, rhythm, harmony and texture". Pattern induction means learning to recognize sequential structures from repeated exposures, followed by matching new input against the learned sequences [101].

Melodic pattern matching has several uses in interactive music systems and in music content analysis. For instance, Cope has used patterns to define the stylistic characteristics of a particular composer, and to reproduce algorithmic compositions that resemble the works of a certain composer [30]. Cope employs the term signature, used to refer to melodic and rhythmic patterns found repeatedly in a composer's body of work. Rolland and Ganascia have also used pattern induction to analyze the jazz improvisations of Charlie Parker [97]. In their system, patterns in a musical corpus are clustered according to similarity, and each cluster is associated with one prototypical pattern.

In automatic music transcription or musical interaction, the predictive power of recognized musical patterns as originators of expectations would be highly valuable.

The basic problem in musical pattern induction is that the patterns most often appear in varying and transformed forms. In music, it is natural that a melody may be transposed to a different pitch register, or played with slightly different tempo. Also, members of the melodic sequence may be deleted, inserted, substituted, or rhythmically displaced. Several elements may be replaced for a single one (consolidation), or vice versa (fragmentation).

Matching varied patterns calls for a proper definition of similarity. The trivial assumption that two patterns are similar if they have identical pitches is usually not appropriate. Relaxation of this assumption can be approached by allowing some different notes to appear in the sequences, or by requiring only that the profiles of the two melodies are similar. Most often it is important to consider the note durations along with the pitch values. Sometimes, other features such as dynamics, timbre or note density can also play a role in the recognition of similarities. More complex features such as harmony or structural features (for instance, how are the motives of the melody organized in time) should also be taken into account. Some studies have attempted to distinguish the features that are more relevant in our perception of similarity.

Lamont et al. [67] suggest that complex features and more simple features such as pitches or dynamics play different roles in the similarity judgments of listeners. [34] insists on the fact that a training phase plays an important role for listeners in the recognizing of similarities between motives.

Before defining what is similar, a first step consists on selecting note sequences that could be possible patterns. For this task, a brute force matching of all patterns of all lengths would be too computationally demanding. [96] proposes a dynamic programming-based approach using structural, local and global descriptors for representing melodies. The algorithm first builds a graph of similarity from the musical sequences and then extracts patterns from the graph. Another approach consists in filtering all the possible patterns of notes by considering musically advised grouping of melodic streams into phrases.

Cooper and Meyer [29] have proposed a set of rhythmic note grouping rules that could be used for this task. The rules are based on harmonic, dynamic, or melodic features (for instance, two pitches belonging to the same harmony or having the same frequency or duration will often be grouped). In the psychoacoustics area, McAdams [81] has performed experiments that aimed at showing that notes could be perceptually grouped together according to various criteria see also [17] and [104] for literature on perceptual grouping). Other methods propose to optimize pattern finding. For instance, [54], locate patterns using a correlative matrix approach. To make the process of finding repeating patterns more efficient, a matrix is used to keep the intermediate results of substring matching.

Once patterns have been located, similarity between patterns has to be quantified. We can find two different approaches when trying to measure similarity.

One elegant way to solve this problem is to describe the patterns in a way that musical pattern matching will turn into a generic (not music specific) pattern matching problem. Musical similarity is in that case implicitly coded into the melody representation (example with pitch contour is given below). Doing this, the musical specificities of the considered features will not have to be taken into account in the execution of the similarity measuring algorithms, and we will be able to use generic algorithms found in other scientific areas.

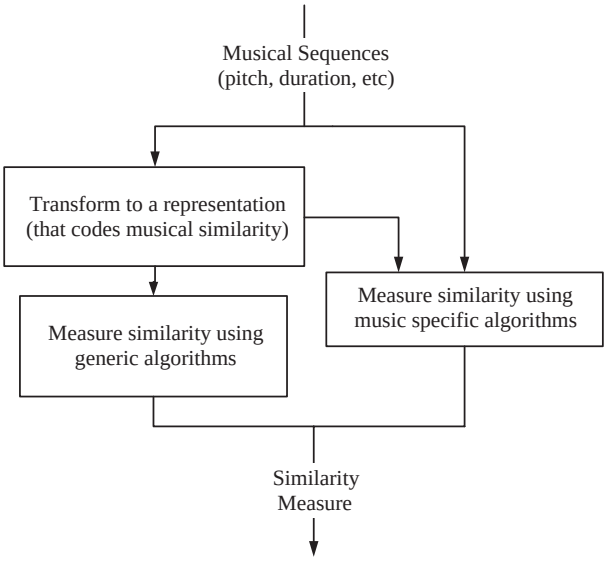


Figure 3.4: Two approaches for measuring similarity

For instance, if the similarity is based on the pitch contour, then the pitch contour representation should be used to code the melody. This representation will then be treated as a numeric sequence and not as a musical sequence. If the similarity is based on the rhythm (which is invariant by scaling), then proportional representation should be used. One can also represent rhythm using a relative time base through the use of duration ratios instead of absolute time values [109, 1]. Duration ratios are calculated by dividing the duration of a note by the duration of the previous one. If the similarity is based on the invariance by transposition, inversion or melodic ornamentation, then a new representation format could be defined.

A problem with the intervalic representation is that substituting one pitch value for another affects the intervals on both sides. Similarly, rhythmic disposition of one note destroys two duration ratios. This can be solved by calculating a cumulative sum over the intervalic representations in the course of the comparison. In this case, substitution or displacement of a single note affects only one interval [101].

Assuming that the feature representations can be considered as numerical sequences abstracted from their musical context, usual similarity measuring algorithms can be used. The fact is that many algorithms that perform similarity measure can be found in other information retrieval areas than music, such as speech recognition (for instance the algorithm proposed in [87]). Other areas such as mathematics and particularly statistics could provide interesting algorithms. Some algorithms used in pattern recognition could also be used. Proposed algorithms in the MPEG-7 format should also be considered, as for instance algorithms that compare sounds from a *SoundModelType* description and algorithms that calculate similarity metric. Uitdenbogerd [117] has tested several techniques for measuring the similarity: local alignment, longest common subsequence, an n-gram counting technique that include the frequency of occurrence of each n-gram, and the Ukkonen n-gram measure. The conclusion was that local alignment gives the best results, but is a quite slow technique, thus faster n-gram based techniques were investigated. Results can be found in [118].

Suppose now that we consider not only one feature, but several. This poses the problem of how to evaluate the interaction of the dimensions on the similarity measure.

A solution could be to consider the dimensions separately, and then the global similarity could be calculated as a mean of the several measures. However, the dimensions should sometimes be considered simultaneously. For instance, two musical sequences could have a common melodic pattern on a defined period, alternatively with a rhythmic one the rest of the time. Considering the two dimensions (pitch and rhythm) separately, the similarity would be quite low between the two sequences. Considering the two dimensions simultaneously, the similarity measure would be higher.

Considering several features simultaneously, Li proposes to constitute a database of audio segment classes that will be used as reference for the similarity measure [75]. Each class can be seen as a generalization of its components. Then, an input melody will be divided in segments that will be compared to the various classes of the database using the NFL (nearest feature line) algorithm. Li also evaluates the confidence in the similarity measure. In that way, all the features are considered simultaneously. However, specific high-level musical considerations, as invariance by contour, by scaling or by inversion, are not taken into account neither in the features representation by classes nor in the NFL algorithm. However, it could be interesting to apply the methodology to high-level musical representations.

Another approach to solve the problem of measuring similarity is to define specific algorithms adapted to musical specificities (for instance, an algorithm which states that two melodies, apparently different, are similar because they have the same structure ababa, can be said to be adapted to the structural specificity of music). If musically advised representations, which would turn musical pattern matching into a generic pattern-matching problem, are not used, the musical similarity information must be encoded to the matching algorithms.

There are some examples of this approach. Buteau and Mazzola propose a distance measure algorithm that takes into account a set of mathematical transformations on melodic patterns, as inversion or symmetric transformations [18]. Clausen et al. [28] and [31] propose algorithms to measure the similarity between melodies containing note gaps. Cambouropoulos [22, 20] and Rolland [96] also propose algorithms being

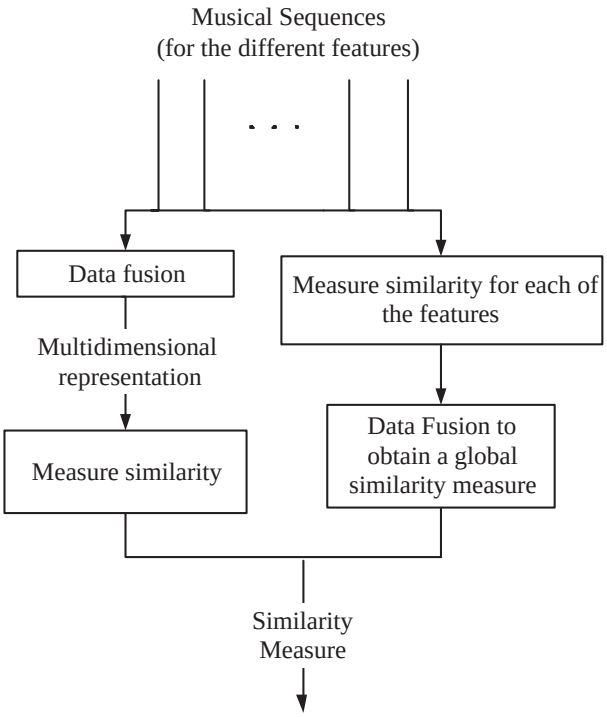


Figure 3.5: Two approaches for measuring similarity between music sequences taking into account several features

sensitive to approximate musical repetitions.

Damerau [32] and Levenshtein [74] have proposed an algorithm that measures a metric distance between two strings. It is based on several editing rules that transform one melody to the other. A cost can be associated to the transformations in order to give preference to some of them. Orpen and Huron [90] notice that this algorithm is an approach to characterize the quantitative similarity for non-quantitative data (data which is not represented by numerical sequence, for instance a string-sequence). He has performed several experimentations of the algorithm (implemented at the *simil* command of the Humdrum Toolkit [55]), and he has concluded that it provides a promising way of characterizing the quantitative similarity between musical passages.

Dynamic programming has been successfully used to handle the cases of insertion, deletion, and substitution in melodic fragments. Dannenberg was among the first to apply dynamic programming to music. Together with Bloch, they later designed a program for real-time accompaniment of keyboard performances [15]. Computer accompaniment is closely related to score following that, in turn, requires pattern matching. The approach was later significantly developed by Stammen and Pennycook [109] who included durational values for rhythm, dynamic time warping with global constraints, automatic segmentation into phrases through the implementation of the grouping rules proposed by Lerdahl and Jackendoff in [73]. Subsequent improvements have included a real-time pattern induction and recognition approach of [101] and inclusion of metrical stress and temporal weighting [79].

3.6 Mid and High-level Melodic Features

In section 2.2, we divided the melodic features into different levels of abstraction and a list of related descriptors was attached to each level of description. These descriptors have been used in the literature in different application contexts.

Inside the features that take into account musical knowledge, tonality characterization has been a very important subject of study. Krumhansl defines a key-finding algorithm based on tonal hierarchies [65]. The idea of this algorithm is that the tonal hierarchies function as a kind of template against which the tones of the musical selection are matched. This algorithm with some modifications is used by the *Melisma* music analyzer (see 4.2).

We refer to section 2.2 for formulas of computation of some *statistical-level* features.

Other higher-level description features are:

- **Harmonic features**: Several features related to harmony can provide information on the melody as the key, the scale, and the position of the cadences. However, the automatic extraction of these features is a very difficult task that requires musical knowledge and presents some ambiguities. For example, in order to detect the scale, we first need to identify which notes belong to the scale and which ones are out of it. Then, there can be more than one possible harmony (key, scale, etc) for the same set of pitches. Some work on harmony analysis and references can be found in [13].

- **Rhythmic features**: Rhythm is an important descriptor for melody. The rhythm extraction issue has been addressed in several papers (references can be found in [35, 102, 47]).

- **Expressivity features**: in section 2.2, we listed the descriptors that are currently used to characterize expressivity in music. They are mainly related to tempo and timing, dynamics and articulation. Its automatic extraction is linked to the techniques explained below, as they belong to different abstraction levels (see section 2.2). There are also some systems that construct models to characterize performances using **A**rtificial **I**ntelligence techniques. One example is found in Widmer et al. [120, 121], that apply machine learning and data mining techniques to study expressive music performance. Using these techniques, they build formal models of expressive performance from real performances by human musicians.

3.7 Melodic Transformations

Once we have a melody description scheme, we can define some transformations based on these melodic features. The needed transformations depend on the application.

In [11] a set of operations for sound transformation is described, as well as a review of some approaches coming from the artificial intelligence area. High-level music transformations can be sorted in different groups:

- **Harmonic transformations**:
  - Modulation: It consists on changing the tonality of the melody, and it requires the knowledge of key and scale information.

  - Reduction: It consists on simplifying the harmonic sequence (therefore the melody) by cutting some harmonies, which do not play an important role in the sequence. For instance, one would keep the cadences and the modulations.
  - Harmonization: the goal is to associate a chord sequence with a melody. Several chord sequences can be associated to a melody, which then appears each time differently.

  Depending on the melody, the last two tasks can be very difficult to be performed automatically at the signal level.

- **Melodic (horizontal) transformations**:

  - Transposition (or pitch shifting): This consists on augmenting or diminishing the pitch values of the melody by a given interval. In a given key, this could be done with the constraint to preserve the scale and, of course, relevant properties as timbre and rhythm.
  - Symmetries: It consists on performing axial or central symmetries on the melody. Musical terms are inversion, retrograde or contrary movement.
  - Ornamentation/reduction: It consists on adding/deleting notes to the melody without changing its structure.

- **Rhythmic transformations**:

  - Time compression and dilatation (time stretch): It consists on changing segment durations without modifying the rhythmic structure. Although it cannot be considered as a rhythmic transformation (as it does not transform the rhythmic features), it is related to the temporal domain.
  - Various symmetries: It consists in performing axial or central symmetries on the rhythmic sequence.

  These transformations are usually performed on a midi-like music sequence, so that we assume that at least pitch and rhythm information are available. Some literature on musical transformation can be found in [49]. Hofmann-Engl [53] also formulates general melodic transformations as transposition and inversion using reflection and translation matrices, in order to measure melodic similarity based on these transformations.

Chapter 4

Software and Application Contexts

The goal of this chapter is to review the different application contexts in which melodic analysis and representation is used. This chapter is not thought to be an exhaustive list of software packages and systems because this would go out of the limits of this research project. We will try to give an idea of the different fields where all the reviewed techniques are useful, giving examples of some implemented systems.

In the last section, we will focus on the *Sound Palette* tool, that has been the application context in which this research work has been carried out.

4.1 Content-based Navigation

4.1.1 Query by humming

The general block diagram for a Query by Humming system is presented in Figure 4.1. In this type of systems, a hummed input is described, and then compared to the descriptions on a database in order to find the most similar one, that is, to identify the sung query.

This application context is one of the most important ones. Here, we will present some of the already implemented systems that perform query by humming.

MELDEX: New Zealand Digital Library MElody inDEX.

Author: Rodger J. McNab, Lloyd A. Smith, David Bainbridge and Ian H. Witten, Department of Computer Science, University of Waikato [82].

Description: MELDEX is a query by humming system accessible through a web page. It allows to find tunes in a database using a combination of an uploaded sung sample and text query. The test database is a collection of 9.400 international folk tunes. In Figure 4.2 we can see the demonstration page, where searchers can be performed either at the beginning or anywhere in a tune. Figure 4.3 shows an example of the results of a melodic query.

web address: See *http://www.dlib.org/dlib/may97/meldex/05witten.html* for a complete description of the system. A demonstration page is found in *http://nzdl2.cs.waikato.ac.nz/cgi-bin/gwmm?mt=music&c=meldex&a=page&p=query&aq=0*.

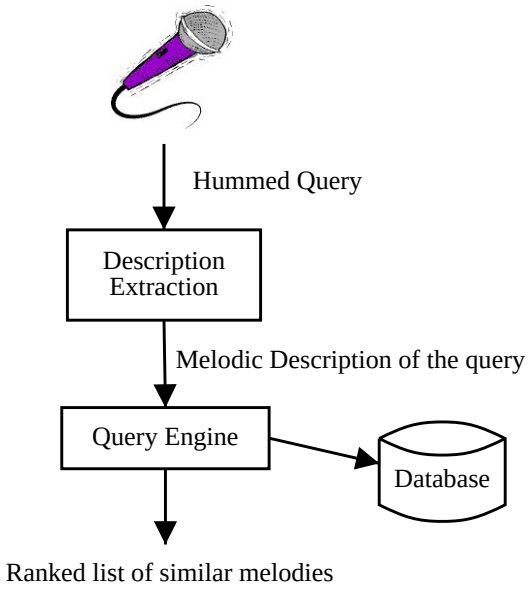


Figure 4.1: Query by humming system architecture

QBH Client

Author: developed by Wei Chai, MIT Media Lab;

Description: QBH Client is a query by humming client that tracks the pitch contour and timing information of the user's query. Then, it sends the query to the Query By Humming (QBH) server via CGI, receives the result and playback. It runs under windows 98/2000, and it is freely distributed. There are some configurable options. See Figure 4.4 for an example of query results.

web address: http://web.media.mit.edu/ chaiwei/qbhweb/

ECHO: A WWW-Melody Retrieval System

Author: Tomonari Sonoda, Masataka Goto and Yoichi Murakoa, from Waseda University.

Description: query by humming software that represents the audio query using different contours (not only an up/down representation) and also includes rhythm contour [107]. The system records the user's voice (see Figure 4.5 a)) and send it to the server, that performs the query. The query results are then presented to the user (see Figure 4.5 b)).

web address: *http://www.sonoda.net/echo/index.html*

Melodiscov

Author: Pierre Yves Rolland, Gailius Raskini and Jean-Gabriel Ganascia, Paris VI University.

Description: query by humming software [98], following the principle *What you hum is what you get*. The search is performed in two phases: first, we get a sym-

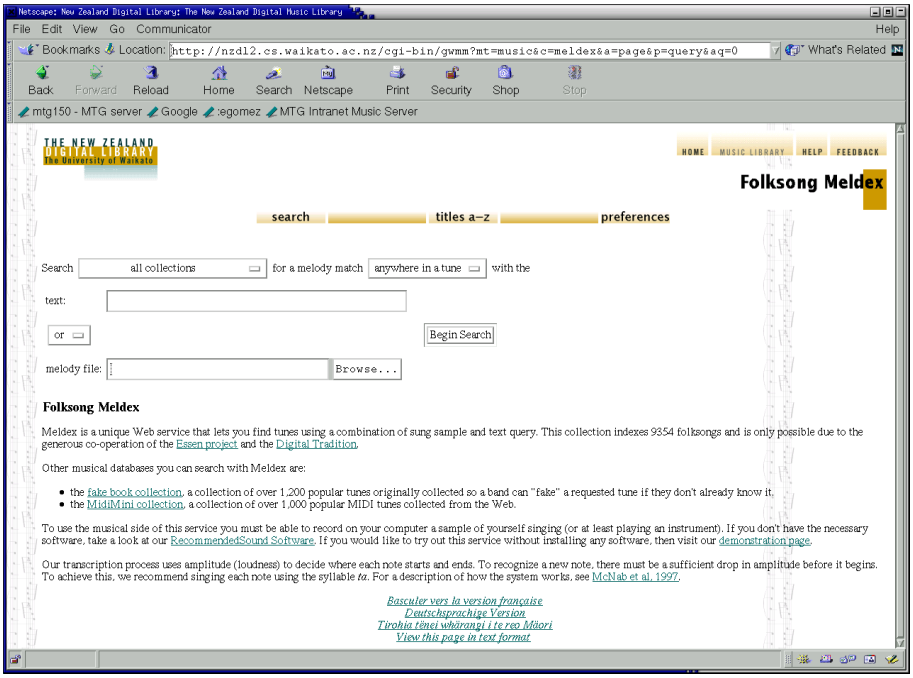


Figure 4.2: MELDEX Demonstration web page

bolic representation of the hummed phrase, and then a pattern matching algorithm is performed. In this system, the audio query is transcribed also when singing lyrics.

To represent a musical phrase, they use the MVEM (Multidescription valued edit mode) Model (see section 2.2 for details in this model).

Lemström and Laine

Author: Kjell Lemström and Pauli Laine, from the University of Finland.

Description: query by humming software [72], focusing on the retrieval process. They derive a representation for musical data, the inner representation, converted and established from MIDI files. A two-dimensional relative code, derived from the inner representation, was used for the matching and was independent from the original key. This representation was composed of interval and duration sequences. An efficient indexing structure, the suffix-tree, was used for the matching phase.

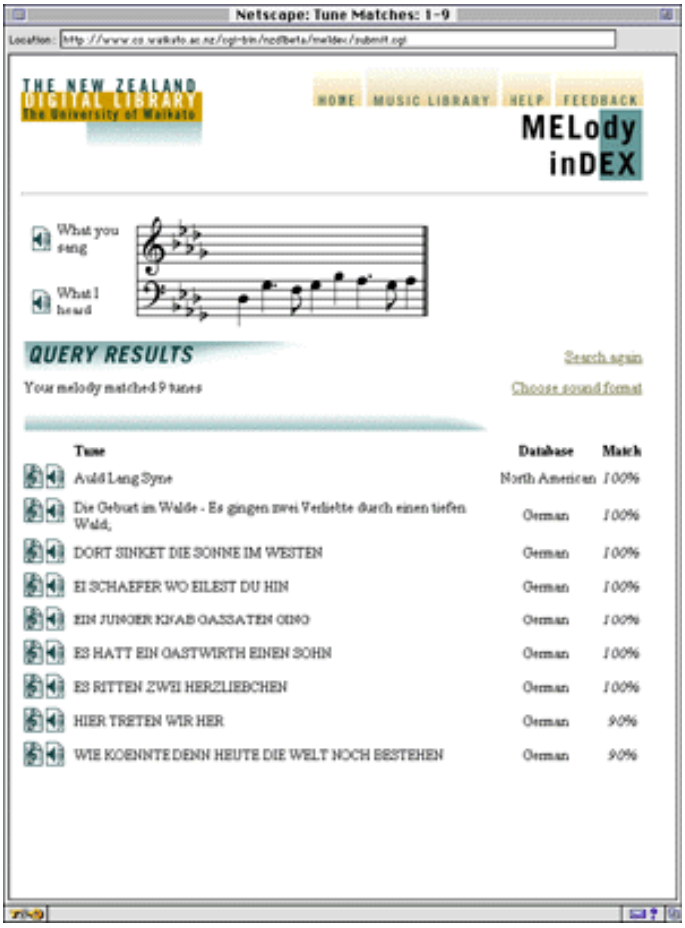


Figure 4.3: MELDEX query results example

Figure 4.4: QBH client query results example

a) Voice recording b) Query result

Figure 4.5: ECHO user interface

4.1.2 Melody retrieval

Query by humming systems can be considered as a subclass of melody retrieval systems where the user interacts with the system by a hummed query. Some of the implemented systems can be accessed either through querying by humming or by some features as pitch classes, contour, etc. In this case, they can be considered as melody retrieval systems.

MAMI: Musical Audio-Mining

Author: developed by different research laboratories of the Ghent University.

Description: This approach is based on the *Query by Humming* paradigm. It is a data-mining project for audio recognition that investigates ways of searching an audio archive in a database. It focuses on audio signals, including all musical styles. This project tries to develop an integrated system for audio description using different levels of representation, trying to be compliant with the MPEG-7 standard (see section 2.2 for details on the different levels of description that they consider). More information about this project is also found in [70, 71].

web address: *http://www.ipem.rug.ac.be/mami*

Query by Pitch Dynamics (QPD)

Author: melody retrieval system developed by the Link Group, Carnegie-Mellon University.

Description: QPD is a software for melody retrieval. As said in [14], they include more than just a tool to retrieve a MIDI file that contains a musical query. They also wanted to provide similarity measures between songs and extent it to automatic clustering and navigation through similar songs.

They represent a musical score by removing pitches that are not used. Each of the used pitches is then given a rank. R-trees are used for storage and retrieval of representations. The web interface also supported searches by artist or title and by MIDI instrument content.

Catfind

Author: developed by Yip Chi Lap, from the University of Hong Kong.

Description: Software for melody retrieval that performs textual searches by melodic contour or by melodic interval sequences at the web, using the melody contour as feature.

web address: *http://zodiac.csis.hku.hk:8192/catfind/Music/ContentSearch.html* Last update: 1999/10/13

MARSYAS

Author: Georges Tzanetakis, from Princeton University.

Description: *MusicAl Research SYstem for Analysis and Synthesis* or *Manipulation Analysis, and Retrieval SYstems for Audio and Signals*. In this PhD Thesis [116] of Princeton University, Georges Tzanetakis develops a set of tools for the analysis, manipulation and retrieval of audio signals. All the proposed algorithms and interfaces are integrated in a free software tool designed for rapid prototyping of computer audition research.

There are other relevant systems for melody retrieval. Some of them are described in [14]: the *MusiFind* project, developed by Stephen Downie, *MUSIR*, developed by Salosaari and Jarvelin, the system proposed by Ghias et al., *Themefinder* (web address *http://densmore.themefinder.org*), developed by Kornstadt from Stanford University, *Tuneserver*, by Lutz Prechelt and Rainer Typke from the University of Karlsruhe (web address *http://name-this-tune.com, http://tuneserver.de*), and *Vega*, designed by Liu and Chen for both video and music. We also cite the *MiDiLiB* project by the University of Bonn (*http://www-mmdb.iai.uni-bonn.de/ forschungprojekte/ midilib/english*).

When trying to evaluate and compare performances of different music retrieval systems, it is important to have access to a standarized test collection, task and evaluation metric. The definition of a formal framework for meaningful evaluation of Music Information Retrieval (from now MIR) systems is one of the main objectives of the MIR initiative [1] research community. This initiative has started the *Music Information Retrieval (MIR) and Music Digital Library (MDL) Evaluation Project*[2] to handle this problem.

4.2 Music Analysis

In this section, we include the software used for different purposes inside Music Research: melodic classification, comparative analysis, musical analysis, etc.

Humdrum

Author: David Huron [3]

Description: *Humdrum* is a software developed to support Music Research in general. This software has been used by musicologists to answer a wide variety of questions, some of them related to comparative analysis by pattern and motivic analysis. Free distribution.

web address: *http://www.music-cog.ohio-state.edu/Humdrum/index.html*

There are two different tools:

Humdrum Syntax: grammar for music representation: symbolic textual representation of content. pitch, duration, key.

Humdrum Toolkit: software manipulating humdrum syntax: pattern location (*pattern*), interval computation (*mint*), transpose (*trans*), estimate the key (*key*). These commands will generate a transformed humdrum representation.

The Melisma Music Analyzer

Authors: Daniel Sleator and Davy Temperley, from the School of Computer Science, Carnegie Mellon University.

Description: The Melisma Music Analyzer is a tool for analyzing music. It takes a MIDI like representation of a musical piece (as an "event list"–a list of notes, with pitch, on-time, and off-time), and it extracts information about meter, phrase structure, contrapuntal structure (the grouping of notes into melodic lines), harmony, pitch-spelling, and key.

It works under LINUX, it is open source and it is freely distributed.

[1] http://music-ir.org

[2] http://music-ir.org/evaluation

[3] http://www.music-cog.ohio-state.edu/Huron/Huron.html

web address: http://www.link.cs.cmu.edu/music-analysis/

Department of Music, University of Jyväskylä

Authors: Petri Toiviainen, Tuomas Eerola, Jukka Luoivuori et al. from the Department of music, University of Jyväskylä.

Description: They have worked with statistical parameters in order to perform an automatic classification of folk melodies [114] and to perform similarity measure [38].

Music composition system using genetic algorithm

Authors: Michael Towsey, Andrew Brown, Susan Wright and Joachim Diederich, from Queensland University of Technology.

Description: Melodic continuation system that uses genetic algorithms (GA) [115]. They use 21 melodic features as the bases for a GA fitness function and for mutation procedures.

4.3 Expressivity Analysis

Director Musices

Authors: researchers of the KTH Music Acoustics Group.

Description: This research group have defined some rules for music performance in order to generate and analyze expressive performances (see 2.2 for explanation of the performance rules). Most of them are implemented in this program. Its manual has references to the relevant papers for each rule. It works under both Macintosh and Windows.

web page: *http://www.speech.kth.se/music/performance*
http://www.speech.kth.se/music/performance/download

There are other researchers that have worked on expressivity and performance analysis (see section 2.2 for details).

EyesWeb

Authors: InfoMus - Laboratorio di Informatica Musicale, University of Genova.

Description: EyesWeb is an open software platform for the development of real-time music and multimedia applications. It includes a hardware and software platform to support the user (i) in the development and experimenting of computational models of expressive content communication and of gesture mapping strategies, (ii) in fast development and experiment cycles of interactive performance setups. An intuitive visual programming language allows to map at different levels movement and audio into integrated music, visual, and mobile scenery. EyesWeb has been designed with a special focus on the analysis of expressive content in movement, midi, audio, and music signals.

web page: *http://musart.dist.unige.it/Eywindex.html*

4.4 Automatic Transcription Systems

Music transcription is the act of converting an audio excerpt into its music notation. Music transcription systems take a sound signal as its input and attempts to recognize

onset, duration and pitch of notes. That is, it tries to find the symbolic representation of this audio excerpt. In this section, we will include also Wave to MIDI software. The difference between Automatic Transcription Systems and Wave to MIDI software is that in automatic transcription, we do not attempt to reproduce a perceptually close "copy" of the original performance. In this sense, automatic music transcription is similar to the task of speech recognition. Automatic music transcription still remains an unsolved scientific problem, mainly for polyphonic sounds.

There are several automatic transcription systems that have been developed in different context: instrument, conditions, monophony/polyphony, etc.

Signal Processing Laboratory, Tampere University of Technology

Author: Anssi Klapuri and other researchers from the Signal Processing Laboratory of the Tampere University of Technology.

Description: One example of Automatic Transcription System, is the system developed by Anssi Klapuri and other researchers from the Signal Processing Laboratory of the Tampere University of Technology [61]. Transients are first detected, followed by a multipitch estimation algorithm. Bandwise processing is applied.

Recognisoft

Description: Wave to MIDI software and music notation software for Windows. It achieves high accuracy in extracting sequences of notes out of the audio records of solo performances. A screen-shot is presented in Figure 4.6. It can not handle polyphonic performances.

web address: *http://www.recognisoft.com*

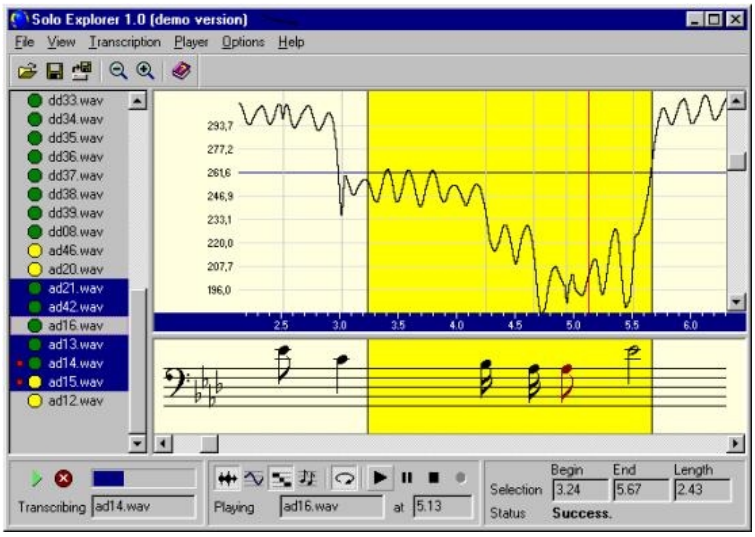


Figure 4.6: Recognisoft screen-shot

A list of wave to MIDI software in the Internet is listed on the Recognisoft web page [4].

[4] http://www.recognisoft.com/links.htm

4.5 Transformation Tools

Melodyne

Description: Commercial software from Celemony. It performs Melody analysis and transformation: pitch shifting, formant correction. Change in intonation is possible by an increase or decrease in phrasing or vibrato. Time stretching is also possible. Finally, for monophonic sounds, it is also provided a beat estimator. See Figure 4.7 for screen-shot.

web address: *http://www.celemony.com/melodyne/*

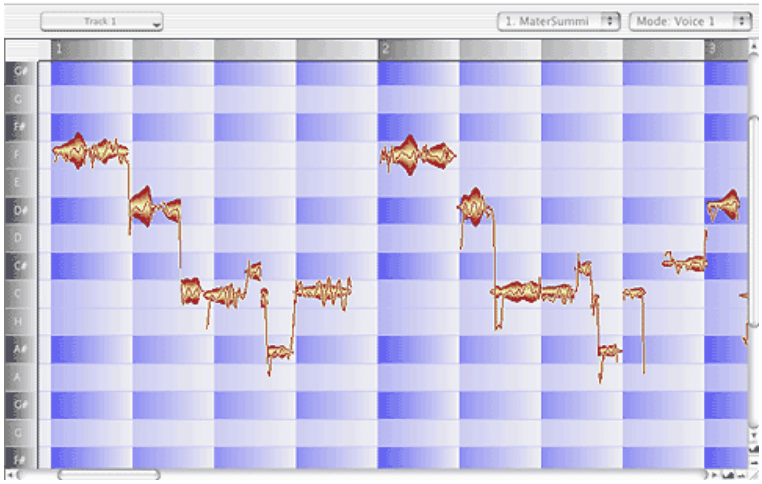


Figure 4.7: Melodyne screen-shot

Pitch & Time

Description: Commercial software by Serato, pitch shift and time stretch system. Serato Pitch 'n Time offers time compression and expansion from a ratio of half speed up to double speed independent of pitch, and pitch shifting of up to 12 semitones independent of tempo. See Figure 4.8 for screen-shot.

web address: *http://www.serato.com/products/pnt/*

4.6 The Sound Palette

The *Sound Palette* application has been thought as an editing, processing and mixing tool for sound designer and professional musicians. The difference between the *Sound Palette* and other similar tools is that it provides a way to interact with the content of the audio, by offering audio sample management, editing and transformation features based on sound content description.

Two versions of the *Sound Palette* are being developed in parallel: an "online" and an "offline" version. We concentrate on the second one, which is restricted to the management of samples on the user's PC and includes sound edition and transforma-

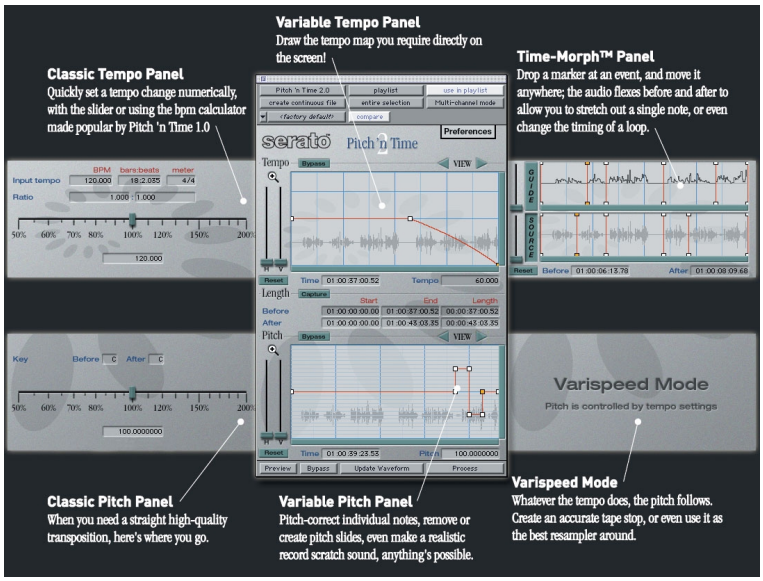


Figure 4.8: Serato Pitch 'n Time screen-shot

tion funcionalities. This tool generates content descriptions of sound files, so thatthe following functionalities can be provided related to this content description:

- Content edition
- Content-based navigation and retrieval among the database of sounds described previously with this tool.
- Content-based transformations

The *Sound Palette* works with monophonic sounds, drum loops, and mixtures, only if they have been created using this tool. Two versions are being developed, one stand-alone (the *Sound Palette* off-line) and other on-line.

It mainly consists of the following modules:

- **Content editor**: the content editor is a tool that generates content description of sound files. For a new file, the content editor automatically describes the audio file and stores this description on a database. If the sound file has already been described, the editor reads the description. Once the description is read, it graphically shows the information to the user. The main descriptors categories are related to melody, rhythm and instrument. The user can refine these descriptions.
- **Instrument assistant**: the instrument assistant is a tool for content-based organization and management of sound libraries. It will automatically classify the "samples" available to the user and will search for enhancing the current sound palette. It will retrieve sounds using similarity criteria, and it will graphically show the position of sounds in relevant perceptual or feature spaces. It will also allow the user to build a self-defined labelling system. Descriptors to be used

Figure 4.9: Melodic content editing and transformation section of the Offline Sound Palette

are not only the perceptual ones but also of other kind (for allowing user-defined descriptors, among other reasons).

- **Transformation assistant**: It is a tool for content-based transformation and synthesis of sounds. It will transform an original recording using content descriptors, and it will allow the generation of new tracks by transforming the existing ones. The Transformation Assistant will take inputs from the other assistants (Instrument and Mixer Assistant) in order to generate the desired result.
- **Mixing assistant**: It is a tool for preparing rough mixes using content information.

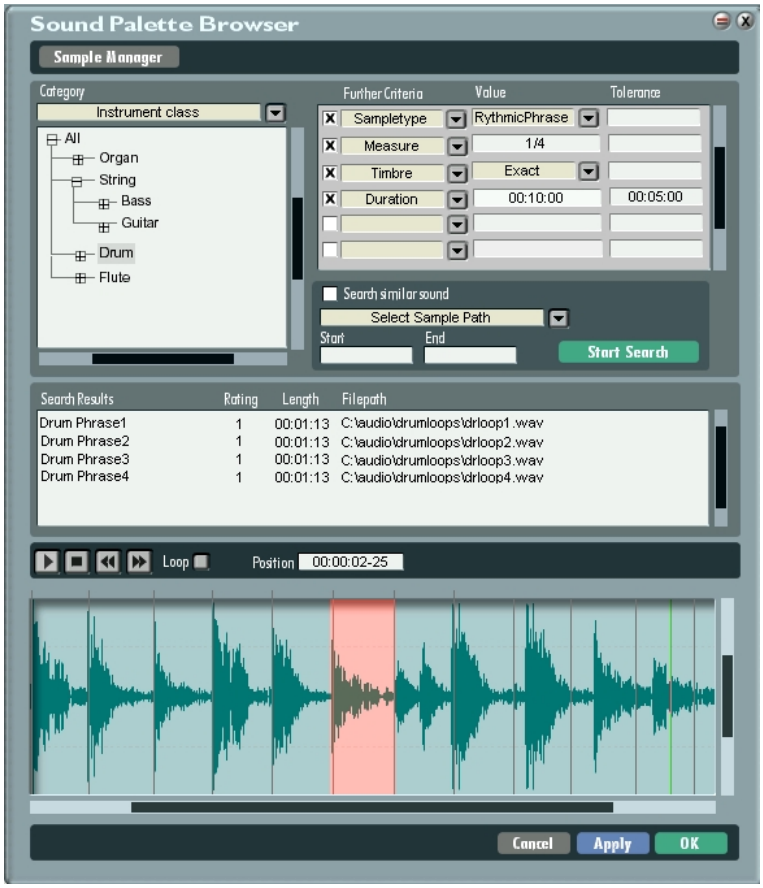


Figure 4.10: Browsing and retrieval section of the Offline Sound Palette

Chapter 5

Selected Approach and Conclusions

In previous chapters, we have reviewed the different ways of representing melodic features of an audio excerpt and the techniques used to automatically extract these melodic descriptors.

As a continuation of this overview, we explain the work that is currently being carried out on the development of some of these techniques. First of all, we study the performance of two different approaches for fundamental frequency estimation, a harmonic matching model and a bandwise processing approach. As a forward step, once note limits are known, we obtain a basic melodic description by a sequence of notes. This approach is oriented toward automatic transcription, without quantization of durations and pitches.

In this chapter, we first see the implementation work that is being carried out. Then, we arrive to some conclusions after the overview and the implementation. Finally, the objectives of the future research are explained, which at the same time, work as the conclusion of this research work.

All the techniques seen in this chapter have been developed in the context of the *Sound Palette* application, explained in section 4.6.

5.1 System Architecture

Figure 5.3 presents the overall structure of the system that is implemented inside this application context. The audio files are analyzed and described, and their descriptions are stored in a database. XML format is used for communication between the database and the description modules.

The system provides some retrieval functionalities, so that the user can search for sounds according to their content features.

Finally, the system also supplies some transformation capabilities to the user, that transform the sound according to the extracted and stored features.

Three categories of sounds have been identified: samples (or isolated notes), monophonic pitched audio phrases and drum loops. In the case of drum loops, melodic description is not provided.

The present work is limited to the description extraction block and centered on melodic features. Its input is an audio file, and its output is a melodic description

stored in XML format. A complementary work is being performed for rhythm and instrument description.

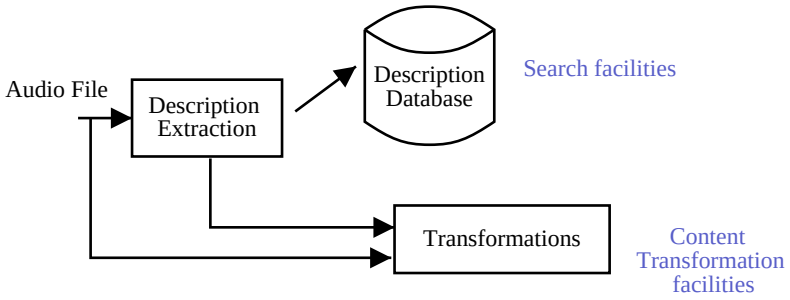


Figure 5.1: Overall system architecture

5.2 Melodic Description scheme

In this section, we detail the melodic aspects of the description scheme that has been used, keeping in mind the MPEG-7 standard. We only go into details on the melodic aspects of this description scheme, although other description axes (rhythm and instrument) are also represented (see [41] in appendix for details on rhythmic and instrument description scheme).

For the first implementation of the description scheme, the objective has been to store as much descriptors as possible. Nevertheless, we will define different levels of data simplification, according to a temporal structure of the audio excerpt (e.g. fundamental frequency values assigned to a frame or a note have different meanings).

On MPEG-7 descriptions

MPEG-7, as shown in section 2.3, is a worthy effort to define an all-purpose melody description scheme. As we already saw, this scheme provides two levels of melodic description:

- a melody contour

  or

- an expanded melodic descriptor,

as well as some information about lyric, scale and meter that is arranged around these core descriptors. We refer to Figure 2.3 on page 26 for a detailed schema.

All these features are included in the *mpeg7:Melody* description scheme (DS). According to Lindsay and Herre [77], the Melody Contour DS has been defined as a compact representation for melodic information, and has been thought for efficient melodic similarity matching, as for example, in query by humming. For applications requiring greater descriptive precision or reconstruction, the *mpeg7:Melody* DS supports an expanded descriptor set and high precision of interval encoding. For our purposes, we also have found some limitations when using this expanded descriptor that we will explain below.

Comparing to a MIDI-like representation, we can find that some needed information is simplified or left out the standard. Regarding the *mpeg7:Note* representation, included in the *mpeg7:Melody* DS, can not store some relevant signal-related features as intensity, intra-note segments articulation or vibrato, that are also important to characterize an audio phrase. The *Note* type includes only quantized note pitch and note relative duration information. It does not store fundamental frequency and exact duration information, that are important, for example, for expressivity characterization or performance analysis. It does neither take into account silences, which play an essential role for melodic perception. Some of these features are already coded by the MIDI representation.

MPEG-7 also includes some optional descriptors related to key, scale and meter. These unary descriptors are associated to an audio segment, and they have been found to be difficult to compute automatically. We also need to store other unary descriptors computed automatically from the low-level descriptors (see 2.2) and attached to the audio excerpt.

Description Scheme Specification

In this first implementation, the idea has been to define different type of segments, according to the temporal structure of the audio signal. Each of the segment types accounts for different description schemes. At this moment, we have distinguished two type of segments:

- **Segment**: this general segment type represents the whole audio excerpt, e.g. composed by a sequence of notes, that in principle can be either monophonic or polyphonic. This segment has its associated descriptors and description schemes and can be decomposed in other segment (for example, a polyphonic segment could be decomposed in a collection of monophonic segments) and in a sequence of *Note* segments. This segment type has an associated description scheme differing from that of the Note, as well as the low-level descriptors (hence LLDs), associated to analysis frames.

- **Note**: this segment type represents a note event. The note has an associated description scheme, accounting for melodic, rhythmic and instrument descriptors.

The **Frame** is a storage entity that has been defined on the context of the Spectral Modeling System (hence SMS) [10]. It is associated to the spectral analysis frame and it is used to represent an event in time. The frame corresponds to the minimal unit of signal that is used for spectral analysis. The frame duration and the hop time between frames are parameters of the analysis algorithm. Each frame has attached a set of low-level spectral descriptors that corresponds to the MPEG-7 LLDs, and a set of descriptors associated to the SMS analysis system. An array of frames is attached to the segment that represents the audio excerpt.

The fact that justifies the definition of two different segment types is that a note has different melodic, rhythmic and instrumental features than a musical phrase or general audio excerpt, and there are some attributes that do not have any sense associated to a note (for example, the *mpeg7:Melody* descriptor). Nevertheless, a Note is in fact a segment with some specific descriptors.

We will now specify which set of features describes each of the segment types.

- **Description Scheme associated to Note**:

The melodic descriptors associated to a note segment are:

- *Temporal location*: the exact temporal location of a note is described by two descriptors: *Begin* and *End*. These two descriptors specify a point in the time axe. We have also added a note duration descriptor.
- The quantization of the duration information defines a *Symbolic Duration* (quarter of note, etc). This descriptor has been introduced in order to allow also the encoding of score representations, as well as audio.
- *Fundamental frequency*: this feature is defined as the representative note fundamental frequency computed without quantization (see Section 5.3.2 for details on computation).
- *Pitch Note*: quantized value of the fundamental frequency. It can be expressed either as a couple of pitch and octave values (as *mpeg7: degreeNoteType*) or using the MIDI-Note representation, establishing a direct mapping between melodic description and MIDI representation.
- *Intensity*: this features represents the note dynamic, and is defined by a floating point value indicating its intensity. This descriptor is necessary to analyze phrasing and expressivity (crescendo, diminuendo, etc) of a melodic phrase, and it can be linked to the *energy* low-level descriptor.
- *Vibrato*: vibrato is also important when trying to characterize how the musical phrase has been performed, it is defined by the vibrato frequency and amplitude.
- *Intra-note segments* features: as explained in previous sections, it is important for some applications to characterize articulation. Intra-note related features, as attack or release characteristics, are related to articulation. We have chosen to use some descriptors indicating the duration and type of the intra-note segments (attack, sustain and release). The definition of these descriptors are under study.

- **Description Scheme associated to Segment**:

  The melodic descriptors associated to a general segment that we are currently using are:

  - *Temporal Location*: as well as for the Note, the exact temporal location is coded by two descriptors: *Begin* and *End*. These two descriptors specify a point in the time axe.
  - *Note Array*: array of Note segments. In order to remain compatible with the standard, the array of Note should be attached to the Melody DS. Nevertheless, we are also considering the possibility of having it directly attached to the segment, by using a segment decomposition structure.
  - *Contour*: this descriptor is designed as an extension of the mpeg7:Contour type, where the contour values are not restricted to the range $[-2, +2]$. We are currently testing different levels of interval quantization, beginning by the simplest Up/Down/Repeat contour (see Section 5.3.3).
  - *Unary Descriptors*: we have incorporated some "unary" descriptors, derived from the low-level descriptors, duration, and pitch sequences, that model some aspects as tessitura, melodic density or interval distribution.

These features provide a way to characterize a melody without explicitly giving the pitch sequence and should be included in this description scheme. This is an open data container to which more descriptors can be easily added.

We have grouped these descriptors into a general container for Melodic descriptors, the *Melody* description scheme, as also found in the standard.

This description scheme has been implemented by a set of C++ classes designed to store the descriptor values. The low-level descriptors are already included in the CLAM C++ development framework [1] used in the development of most of the techniques that have been implemented within this research work. The rest of them has been implemented and added to this library. An UML diagram of the description scheme is shown in Figure 5.2.

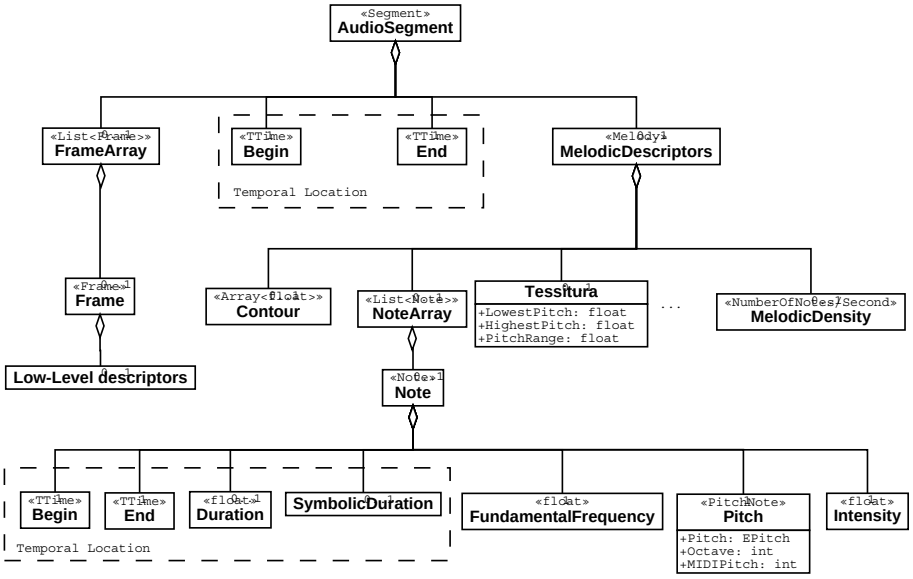


Figure 5.2: Class diagram of melodic descriptors

5.3 Work in progress on melodic description extraction techniques

In this section we describe the algorithms in which we are currently working to obtain melodic features of an audio excerpt. We begin with the lower level melodic description, that is the frame-based fundamental frequency, followed by a Note description, up to melodic descriptors associated to the whole audio file.

Figure 5.3 represents the steps that are performed to obtain a melodic description from audio. First, the audio signal is divided into analysis frames, and the set of low-level descriptors are computed for each analysis frame. These low-level descriptors

[1] http://www.iua.upf.es/mtg/clam

are used by the note segmentation algorithm, as well as in a preprocessing step of the fundamental frequency algorithm.

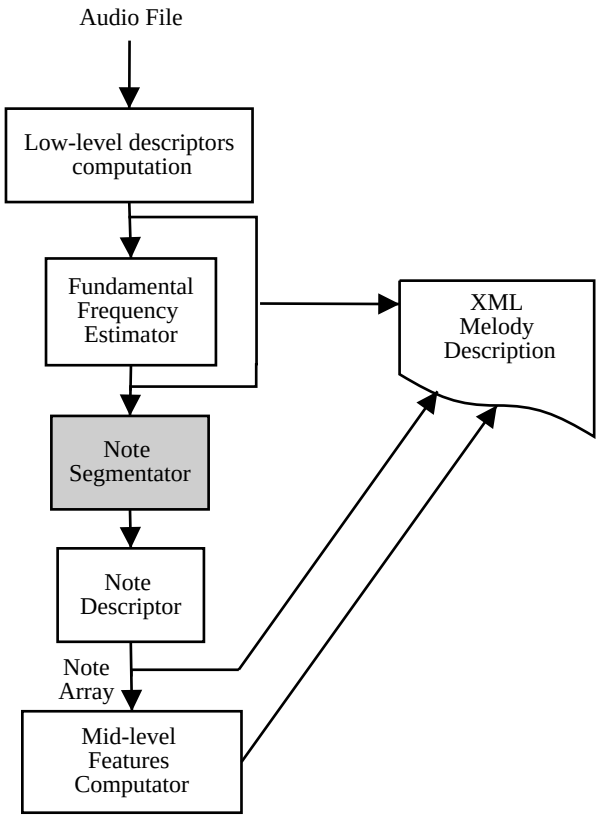


Figure 5.3: Block diagram of the melody descriptor

The fundamental frequency detector outputs an estimates for each analysis frame. Using these values and the low-level descriptors, the note segmentator block detects the note boundaries. Once the note boundaries are known, the note descriptors are computed from the low-level and the fundamental frequency values.

Finally, note and low-level descriptors are combined to compute the global descriptors, associated to the whole music segment.

Al these descriptors are stored into an XML document according to this description scheme. An example of the structure of a XML description for a monophonic saxophone phrase is presented below.

```
<AudioSegment>
 <BeginTime>0</BeginTime>
 <EndTime>10.658</EndTime>
 <FramesArray>
    <Frame>
       <CenterTime>0.01161</CenterTime>
       <Duration>0.02322</Duration>
       <Spectrum>
          <Scale>Log</Scale>
```

```
          <SpectralRange>22050</SpectralRange>
          <prSize>1025</prSize>
          <MagBuffer>-82.2844 -83.0799 ... </MagBuffer>
          <PhaseBuffer>0 -0.224169 ... </PhaseBuffer>
          ...
         <Descriptors>
             <Energy>4.02654e-08</Energy>
             <Centroid>264.543</Centroid>
             <Kurtosis>61.4152</Kurtosis>
             <MFCC>-32.3263 -0.462856 ... </MFCC>
             <MaxMagFreq>0</MaxMagFreq>
             <LowFreqEnergyRelation>0.393396</LowFreqEnergyRelation>
             <Skewness>6.6301</Skewness>
             <Rolloff>0</Rolloff>
         </Descriptors>
       </Spectrum>
       <SpectralPeakArray>
         <Scale>Log</Scale>
         <nPeaks>10</nPeaks>
         <nMaxPeaks>100</nMaxPeaks>
         <MagBuffer> ... </MagBuffer>
         <FreqBuffer> ... </FreqBuffer>
         <PhaseBuffer> ... </PhaseBuffer>
         <BinPosBuffer> ... </BinPosBuffer>
         <BinWidthBuffer> ... </BinWidthBuffer>
       </SpectralPeakArray>
      <Fundamental>
        <nMaxCandidates>1</nMaxCandidates>
        <nCandidates>1</nCandidates>
        <CandidatesFreq>25</CandidatesFreq>
        <CandidatesErr>100</CandidatesErr>
    </Fundamental>
  </Frame>
  <Frame>
   ...
  </Frame>
   ...
 <Descriptors>
   <Melody>
    <NoteArray>
     <Note>
      <Time>
        <Begin>0.325</Begin>
        <End>2.449</End>
      </Time>
      <PitchNote>
        <Pitch>D</Pitch>
        <Octave>5</Octave>
      </PitchNote>
      <MIDINote>74</MIDINote>
      <FundFreq>591.693</FundFreq>
      <Energy>0.0337453</Energy>
     </Note>
     <Note>
```

```
        <Time>
          <Begin>2.542</Begin>
          <End>2.67</End>
        </Time>
        <PitchNote>
          <Pitch>C</Pitch>
          <Octave>5</Octave>
        </PitchNote>
        <MIDINote>72</MIDINote>
        <FundFreq>527.423</FundFreq>
        <Energy>0.0101042</Energy>
       </Note>
       ...
      </NoteArray>
      <NumberOfNotes>16</NumberOfNotes>
      <Tessitura>14000</Tessitura>
      <Contour>-1 -1 1 1 1 0 0 0 0 1 -1 -1 1 -1 -1</Contour>
       ...
    </Melody>
  </Descriptors>
</AudioSegment>
```

5.3.1 Fundamental frequency estimation

In the fundamental frequency estimation block, two different approaches are being tested, improved and compared. The first one implements a harmonic-matching technique, and the other one estimates the predominant fundamental frequency by dividing the spectrum in different frequency band (i.e. performing a bandwise processing). This predominant fundamental frequency estimator is also thought to work for polyphonic sounds. We are also testing it with simple polyphonies and inharmonic sounds.

The first algorithm is used in the SMS context [10]), and it computes a fundamental frequency estimate from a set of spectral peaks. This algorithm does not take into account inharmonicities, that are present in some type of sounds. This is supposed to be an advantage of the use of a bandwise processing, that is the resistance against inharmonicity and the robustness against band distortions.

In this application context, there are a list of properties that the fundamental frequency algorithm should verify:

- *Generality*: the algorithm should work well for different instrument types, therefore for sounds with varied spectral shapes and spectral peak distributions.
- *Robustness* to noise: this is not a main requirement, because we are working with sounds recorded in studio, but the algorithm should also work when some noise is present.
- *Robustness* to inharmonicity: but the algorithm should also work for sounds presenting a small inharmonicity coefficient.
- *Predominant* estimation for polyphonic sounds: the algorithm ought to behave well for monophonic sounds, although we will also consider simple polyphonies as a more complex problem that noisy monophonies. We only consider monotimbral sounds or solo parts, where there is a clear predominant pitch in the spectrum.

Harmonic Matching model approach

Figure 5.4 shows the block diagram for the fundamental frequency estimator following a harmonic-matching approach. This algorithm is based on the one proposed in [78], and is detailed in section 3.1.1.

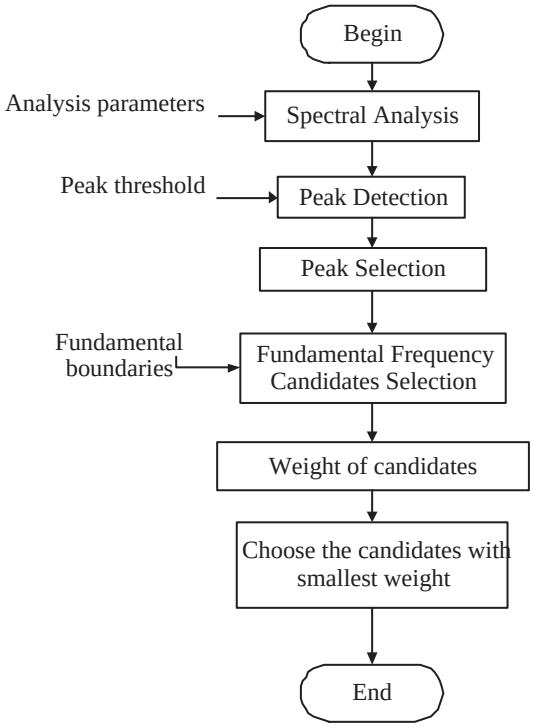


Figure 5.4: Flow diagram of the TWM algorithm

First of all, we perform a spectral analysis of a portion of sound, called analysis frame, whose size is a parameter of the algorithm. This spectral analysis lies in multiplying the audio frame with an appropriate analysis window and performing a Discrete Fourier Transform (DFT) to obtain its spectrum.

Secondly, the prominent spectral peaks of the spectrum are detected from the spectrum magnitude. These spectral peaks of the spectrum are defined as the local maxima of the spectrum which magnitude is greater than a threshold. This magnitude threshold is also a parameter of the algorithm.

These spectral peaks are compared to a harmonic series and an two-way mismatch (TWM) error is computed for each fundamental frequency candidates. The candidate with the minimum error is chosen to be the fundamental frequency estimate.

The implementation of this algorithm has been made in C++.

After a first test of this implementation, we identified some situations where the spectrum is not exactly harmonic:

- High frequency sounds: for sounds with a high fundamental frequency, there are some spectral peaks that appears before the maximum magnitude peak and

between the harmonics. We also can find doubled harmonic peaks, as e.g. for violin high frequency samples.

- Inharmonicity: for string instruments, we also remark that the harmonics are not equally spaced, i.e, the spectrum presents some inharmonicities.
- Noise: noise peaks can have strong amplitudes and cause errors in the fundamental frequency estimation.
- Transitions: there were some estimation errors in isolated corrupted spectrums, that are frames where the sound is not harmonic usually located at transitions or silence parts.

Some improvements to the original algorithm proposed in [78] are under test to deal with these situations:

- Peak selection: after locating the spectral peaks, a peak selection routine has been added in order to eliminate spectral peaks corresponding to noise. The peak selection is done according to a masking threshold around each of the maximum magnitude peaks. The form of the masking threshold depends on the peak amplitude, and uses there different slopes depending on the frequency distance to the peak frequency.
- Context awareness: we have introduced the notion of context awareness to take into account previous values of the fundamental frequency estimation (history) and instrument dependencies to fix the algorithm parameters and to obtain a more adapted result.
- Previous classification: in order to adapt the algorithm to different types of signals, a rough classification using low-level signal descriptors is being tested.
- Noise gate: in the preprocessing step, a noise gate based on some low-level signal descriptor is applied to detect silences, so that the estimation is only performed in non-silences segments of the sound.

Some evaluation work is done for the fundamental frequency estimator module, using a rather regulated subset of samples from the IRCAM Sound On Line database [2].

The main difficulty in this application context is that the algorithm has to work well for any instrument source, so that it has to be flexible enough, and at the same time it should work in a non-supervised way (that is, its parameters must be fixed). The previous classification step, which uses low-level signal descriptors, can work as a preprocessing step to fix some of these analysis parameters.

Bandwise processing approach

This seconds approach splits the signal spectrum into different frequency bands. A fundamental frequency estimate is computed for each of the frequency band, and then the results for all the bands are combined to yield a global fundamental frequency estimate. We have adapted a techniques that was proposed by Anssi Klapuri. Details about this algorithm are found in section 3.1.1.

[2] http://www.ircam.fr/produits/technologies/sol

A block diagram of this approach is shown in Figure 5.5. First, we perform a spectral analysis to the audio signal, that consists in windowing the audio frame and perform a Fast Fourier Transform (FFT) to obtain its spectrum. As a preprocessing step before fundamental frequency estimation, an equalization block is designed. This equalization block is intended to eliminate both additive and convolutive noise. Then, we divide the spectrum into 21 constant-Q bands:

$$Q = \frac{central frequency}{bandwidth} \tag{5.1}$$

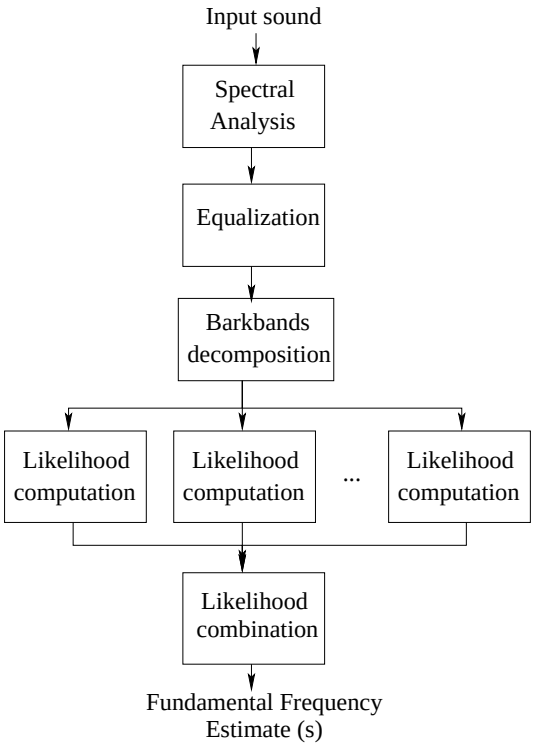


Figure 5.5: Flow diagram of the bandwise approach

To do so, we construct a filter bank using triangular windows for each frequency band, as can be seen in Figure 5.6. We are also testing with other window types in order to evaluate the influence of the window shape for fundamental frequency estimation and choose the optimal one.

For the output of each bandpass filter, we compute a predominant vector. This predominant vector represents, for each frequency value, its probability of being the fundamental frequency when taking into account only the filter's frequency band. the peaks of the predominant vectors are then the most probable fundamental frequency candidates when considering this frequency band. Each of the fundamental frequency candidates can be assigned a weight determined by the normalized value of the predominant vector for this frequency.

In the last step of the algorithm, the results of the 21 bands are combined to compute a global fundamental frequency estimate.

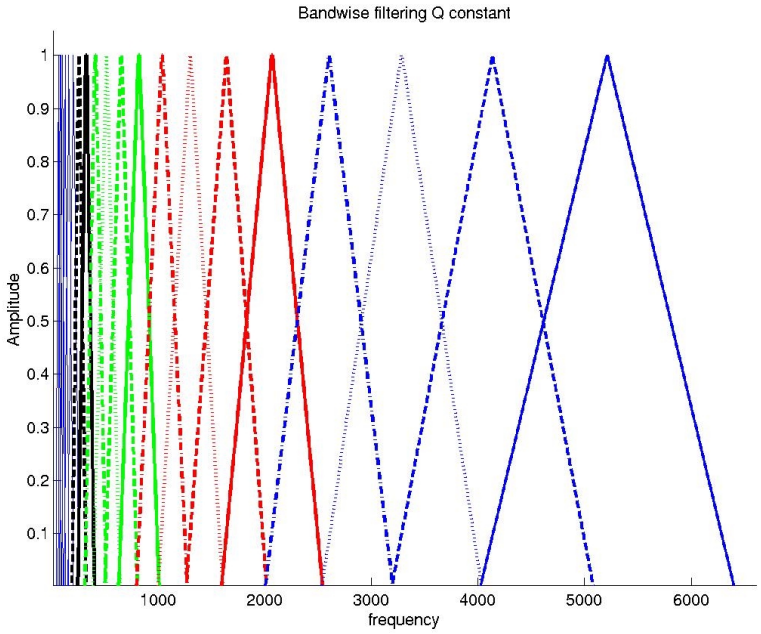


Figure 5.6: Filterbank used for bandwise processing

This algorithm was implemented using C++ classes and MATLAB code.

Qualitative comparison between algorithms

We look at the output of both algorithms in order to decide if the correct pitch has been detected, in stable part of notes. We are working with sounds of different natures:

- Pseudo-harmonic sounds, as e.g., wind instruments (saxophone, trumpet, clarinet, etc).
- Sounds presenting inharmonicities, i.e., sounds whose harmonics are not equally spaced.
- Noisy sounds, which let us test the algorithm performance against noise.
- Spectra where a frequency band is corrupted, so that we can remark the differences of using a bandwise robustness.
- Simple polyphony: we will only deal with a set of simple polyphonies and solo parts, considered as a most complex problem than noisy spectral where there is always a predominant fundamental frequency that presents the clearest harmonicity. The advantages of using a bandwise processing are evaluated when dealing with this type of sounds, because the predominant frequency may be clear only in a small frequency band.
- Sounds of a wide frequency range are used for the tests, to verify the performance for very low and very high fundamental frequency samples.

Some qualitative preliminary results are the following ones:

- For harmonic sounds, the performance of both algorithms are quite similar. Nevertheless, the TWM procedure achieves a better frequency resolution, due to the frequency interpolation that is made in the spectral peak picking and in the search of the minimum of the Two-Way error function.
- For sounds presenting inharmonicities, some differences are found. The bandwise algorithm takes into account inharmonicity and performs an estimation of the inharmonicity factor, so that some estimation errors are corrected.
- For noisy sounds, it appears that the bandwise processing is much more efficient, that is because of a preprocessing step that is applied in order to reduce noise (see [50] for details). This preprocessing efficiently cleans the spectrum. Fundamental frequency is thus tracked nearly as long as the note is heardable in the original signal.
- Polyphonic sounds allow a comparison of the algorithm performance in another kind of noisy environment, that is the presence of other sounds. In this kind of environment, the Two-Way Mismatch proved to be weaker than the bandwise processing.

More details on the tests and examples of performances are presented in appendix A [37]. Figure 5.7 shows an example of visualization of some of the computed low-level descriptors, frame fundamental frequency, energy and spectral centroid, for a saxophone phrase.

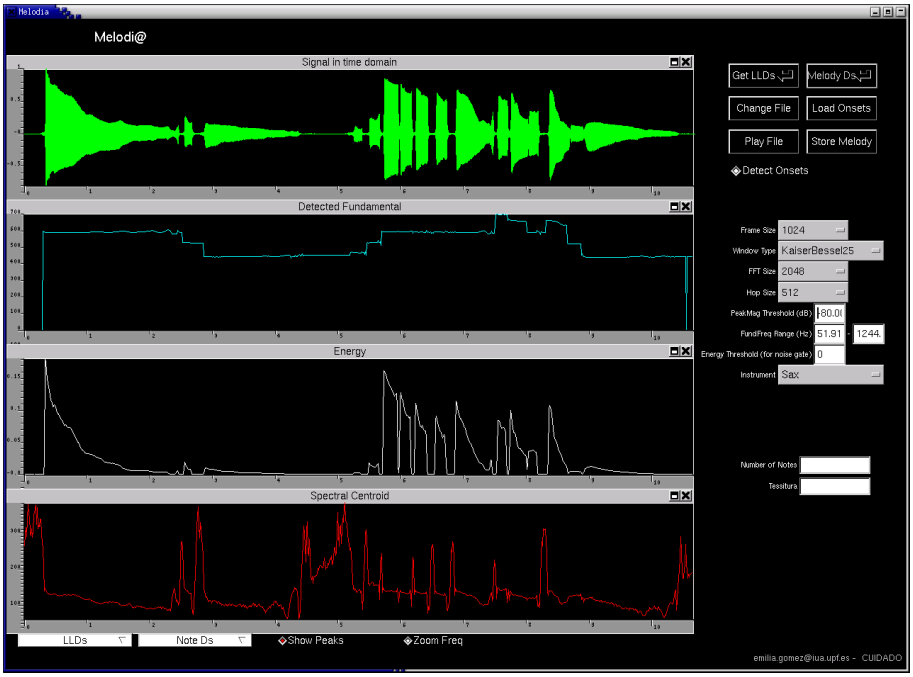


Figure 5.7: Class diagram of melodic descriptors

5.3.2 Note features computation

Once we have estimated a fundamental frequency value for each analysis frame, and we know the note boundaries, we compute note descriptors using this estimation and the low-level descriptors values.

The low-level descriptors associated to a note segment (as e.g. energy, centroid, spectral flatness, etc) are computed by averaging the frame values within this note segment.

Pitch histograms have been used to compute the pitch note and the fundamental frequency that represents each note segment, as found in [82]. This is done to avoid taking into account mistaken frames in the fundamental frequency mean computation. The procedure to assign a pitch and fundamental frequency value to a note segment is the following one, used by [82].

First, frequency values are converted into cents, by the following formula:

$$c = 1200 \cdot \frac{\log\left(\frac{f}{f_{ref}}\right)}{log2} \tag{5.2}$$

where $f_{ref} = 8.176$.

Then, we define histograms with bins of $100\ cents$ and hop size of $5\ cents$ and we compute the maximum of the histogram to identify the note pitch. Finally, we compute the frequency mean for all the points that belong to the histogram. The MIDI pitch and the "textual" pitch note (pitch value and octave) is computed by quantization of this fundamental frequency mean.

No quantization is being performed neither in the fundamental frequency value nor in the temporal note information for melodic description.

Figure 5.8 shows an example of visualization of some note descriptors for a saxophone audio phrase.

5.3.3 Features derived from a note array

Some descriptors related to whole audio excerpt (represented by a Segment) are now under test. Our first goal is to verify its utility for simple melodic search and retrieval and for simple melodic characterization on the context of melodic transformations, and see how statistical descriptors can be linked to musical or descriptive concepts. We are using mid-level descriptors based on mathematical and statistical computations of the pitch and duration sequences and on the fundamental frequency values of each frame. Some of them are:

- Number of notes, number of notes per second and number of distinct notes per second: these descriptors are related to melodic density and variation of the audio excerpt.
- Tessitura or fundamental frequency range: lowest and highest pitches and pitch range.
- Melody Contour: *Parson Code*: this is the most used melodic descriptor in the literature for search and retrieval purposes. We first simplify to the simpler up/down/constant (U D R) contour defined by Denys Parsons in 1975 [39], where each interval is represented by a value $i/i \in \{+1, 0, -1\}$ if the interval is ascendant, descendant or constant.

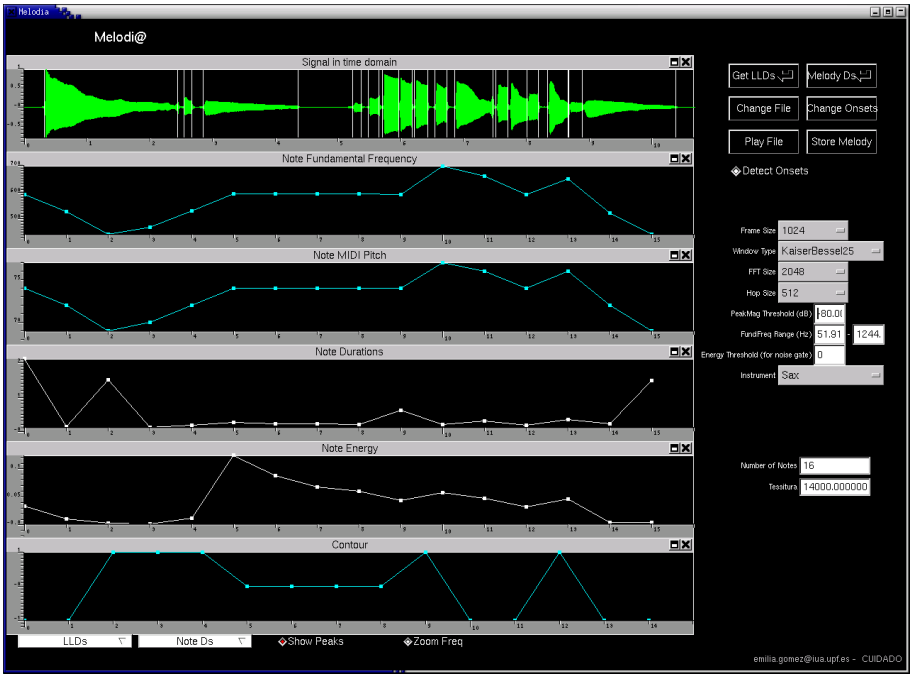


Figure 5.8: Class diagram of melodic descriptors

- Interval Distribution : type of intervals used on the melody. For the moment, we are using an histogram that represents which type of intervals are used.
- Fundamental frequency histograms: some ideas of pitch histograms from Tzanetakis (see section 2.2 and [116]) are also under test.

5.4 Database issues

For the test of fundamental frequency estimation techniques, we can easily find sounds from different sources. We mainly use samples (i.e. isolated notes) from different instruments found at the IRCAM Sound On Line database (see footnote 2). The tests are being performed with 342 samples of the following instruments in different play modes, represented in Table 5.1.

When testing the overall features representing the whole audio excerpt, one of the main encountered problems was to build a database of melodies. It has been difficult to find monophonic melodies from different instruments. In the author's opinion, the availability of MIDI melody databases in the Internet may be one of the reasons why a lot of research is performed using symbolic data. We have collected the following melodic phrases (a total of 146 phrases with different lengths and number of notes) recorded from different instruments. Some of them are arpeggios and scales, being also useful for testing purposes.

Instrument	Number of samples
Accordion	21
Bassoon	12
Double bass	33
Clarinet	10
French Horn	14
Flute	14
Guitar	28
Harp	17
Oboe	18
Sax	11
Trombone	21
Trumpet	21
Tuba	13
Cello	36
Violin	35
Viola	39

Table 5.1: Monophonic samples used for testing the fundamental frequency algorithms (source: see footnote 2)

Instrument	Number of samples
Accordion	2
Banjo	1
Bass	12
Bassoon	3
Celesta	1
Cello	8
Clarinet	9
Contrabasson	1
Double bass	3
Electronic	1
Flute	8
Glockenspiel	1
Guitar	9
Harp	2
Harpsichord	1
Horn	8
Mandolin	1
Oboe	6
Piano	5
Piccolo	2
Sax	20
Trombone	2
Trumpet	8
Tuba	7
Vibraphone	1
Viola	5
Violin	11
Voice	8

Table 5.2: Monophonic phrases used for testing the melodic descriptors

5.5 Discussion

The main objectives of the present research work have been the following ones: first of all, to establish a taxonomy of the different aspects and levels of melodic description that are used in the literature; secondly, to review the approaches that are being used for the automatic extraction of these descriptors; and finally, to see in which application contexts melodic description is needed.

We have discovered that it is a wide topic, where there is a lot of research that is being made and many methods are under study aiming at describing and processing audio according to its melodic features. In a broad melody processing system, different stages can be distinguished. The extraction of a melody description from the audio signal is one of the most important ones. We have reviewed some of the methods used to describe melodic features of an audio excerpt.

We can divide the different approaches in two tendencies:

- **Automatic transcription systems**, that are those whose goal is to perform a PCM to MIDI conversion, with or without quantization, of the audio excerpt. In this type of systems, we find the difficulty to perform accurate pitch tracking and precise note segmentation even for monophonic sounds. Many methods have been reviewed at section 3.1 that deal with different type of sounds in different contexts. The use of context information as

  - Instrument type or spectral features of sound
  - Number of concurrent voices
  - Recording Conditions (noise, reverb, etc)
  - Location of the stable part of the note

  to adapt the tracking algorithm has helped to solve this problem for monophonic sounds. In the author's opinion, it still remains difficult to find a fundamental frequency algorithm that works well for any frequency range, instrument and conditions.

  If we consider polyphony, some attempts at melody transcription have been made, but we are still far away from having a polyphonic transcriptor working for real audio recordings (i.e. what we call "real sounds" or songs).

- Some **statistical or model-based techniques** and procedures do not aim at computing the musical score or the MIDI transcription, but they extract model-based parameters (see 2.2) to describe sound in an abstract, that is not musical, way. They are usually used for search, retrieval and identification purposes. In this group of approaches, description is connected to **identification** (see [25, 26] for a general introduction to audio recognition techniques).

If we think of sharing descriptions and communicating between applications and systems, it is necessary to define a common melody description scheme. This scheme ought to be valid for any application in any usage situation, which is a very hard requirement. It might be more practical to try to devise a set of specific application contexts and define a description scheme open enough to be easily adapted to each of these target applications. An attempt to such a description scheme is the MPEG-7 standard (see section 2.3 and 5.2).

In the author's opinion, when this description scheme will be used and adapted to the different usage scenario's needs, some improvements will be thought according to these different application contexts. We have to work toward multi-layered descriptions which treat musical information on a number of levels and regarding different aspects, as described in section 2.2.

The fact that justifies the definition of multiple levels of description is that each application context focuses on a different aspect of melody. For example, melody retrieval systems do not need exact accuracy and description of the audio material. It could be enough to have some rough features or low-level parameters that will give a measure of similarity between melodies. These features would not necessarily have a musical or structural meaning.

On the other hand, comparative analysis or composition tools need to introduce musical knowledge to the description. It is also necessary to have a very accurate transcription of the audio excerpt when comparing performances of the same excerpt and when describing expressivity.

Finally, once we have a melody description at different levels, another important issue is to transform the audio following these melodic attributes. Techniques intended to transform the audio following content information are now under research, and these techniques will give the possibility to transform the content by using attributes at different levels: according to signal attributes, to musical attributes, textual attributes, etc.

We have seen that there are many fields to consider when trying to represent melodic features of audio. The reason is that the concept of "melody" is a complex concept that is related to different aspects of sound as perception, musical knowledge or structural hierarchy.

5.6 Future research

This section is devoted to the definition of the PhD objectives and goals. Here we will define the main axes of research that will be followed during the PhD thesis.

First, we have to delimit the type of sounds we are going to address. We have focused the state of the art in monophonic audio phrases (one instrument, without accompaniment). Nevertheless, we will continue keeping in mind polyphonies, and we will also try to test the implemented algorithms with polyphonic sounds.

Our first objective is to extract automatically a set of low and mid-level objective descriptors related to melodic attributes from monophonic audio phrases.

We also need to decide what to do with these melodic descriptors. One possibility is to use them for melodic retrieval and content-based navigation, or use them to perform content-based transformations. In this PhD thesis, we will CONCENTRATE ON DESCRIPTION, although keeping in mind these two uses, covered by the *Sound Palette* tool.

For both application contexts, it would be desirable to EXTENT THIS SET OF DESCRIPTORS WITH SOME HIGH-LEVEL CONCEPTS. This idea appears in [70]. The important question here is how subjective concepts are defined and how they can be connected to objective descriptors.

One important issue is to concentrate in a specific user profile. We concentrate on home users, neither musical experts nor people familiar with this field of research. The problem is restricted then to know how a home user describes a monophonic melody using textual labels, and validate that these labels can be linked to low and mid-level descriptors computed automatically from audio signals.

The goal of the PhD thesis will be to PERFORM A BOTTOM-UP DERIVATION OF SUBJECTIVE DESCRIPTORS FOR MELODIC PHRASES, BRIDGING THE GAP BETWEEN THE DIFFERENT LEVELS OF MELODIC DESCRIPTION. This problem can be divided into different tasks:

- Description Scheme specification: to specify a flexible structure or melodic description scheme including these different aspects of melodic description: objective (automatically computed) and subjective (textual labels) descriptors.
- Database construction: to construct a test database of monophonic melodies. This database must be representative of the different melodic features that we want to analyze. All the samples of the database should be of the same instrument in order to eliminate the influence of timbre.
- Descriptor extraction: to complete the implementation and testing of the methods to automatically extract the low and mid-level descriptors from audio. Test the algorithms with the database and with the set of sounds that we have used for the first tests, in order to get accurate algorithms.
- High-level descriptors definition: to delimit the set of labels or textual descriptors that we want to connect with the descriptors that are extracted automatically from the audio signal. We will take into account previous research on melodic characterization (some examples are shown in section 2.2).
- Users: to define a group of home users that will manually attach each of the audio phrases to labels.
- Test the validity of this labels and define a strategy to connect the extracted low and mid-level descriptors with the high-level labels.
- Validate the connection rules.

Bibliography

[1] Mpeg working documents, 2001. http://www.cselt.it/mpeg/working_documents.htm.

[2] Cambridge international dictionary of english online, 2002. http://dictionary.cambridge.org.

[3] Cuidado project web site, 2002. http://www.cuidado.mu.

[4] Educational dictionary-thesaurus. online, 2002. http://www.wordsmyth.net.

[5] Grove's integrated music resource on the web. on line, 2002. http://www.grovemusic.com.

[6] Merrian-webster's collegiate dictionary, 2002. http://www.m-w.com.

[7] Mpeg-7 schema and description examples, 2002. http://pmedia.i2.ibm.com:8000/mpeg7/schema/.

[8] Mpeg home page, 2002. http://mpeg.telecomitalialab.com.

[9] The science of music performance, 2002. http://www.speech.kth.se/music/performance_rules.html.

[10] Spectral modeling synthesis home page, 2002. http://www.iua.upf.es/sms.

[11] X. Amatriain, J. Bonada, A. Loscos, J. L. Arcos, and V. Verfaille. Adressing the content level in audio and music transformations. *Journal of New Music Research*, to appear, 2003. http://www.iua.upf.es/mtg.

[12] E. J. Anderson. Limitations of short-time fourier transforms in polyphonic pitch recognition. Technical Report Ph.D. qualifying project report, Department of Computer Science and Engineering in the Graduate School of the University of Washington, 14-5-1997. http://www.cs.washington.edu/homes/eric/Publications.html.

[13] J. Barthélemy and A. Bonardi. Figured bass and tonalities recognition. In *International Symposium on Music Information Retrieval*, Indiana University, Plymouth, Massachusetts, 2001.

[14] S. Blackburn. *Content based retrieval and navigation of music using melodic pitch contour*. Phd thesis, University of Southampton, 2000. http://www.ecs.soton.ac.uk/ sgb97r/mm/.

[15] J. Bloch and R. Dannenberg. Real-time computer accompaniment of keyboard performances. In *International Computer Music Conference*, San Francisco, 1985.

[16] R. Bod. Memory-based models of melodic analysis: challenging the gestalt principles. *Journal of New Music Research*, 30(3), 2001.

[17] A. S. Bregman. Psychological data and computational auditory scene analysis. In D Rosenthal and H. G. Okuno, editors, *Computational auditory scene analysis*. Lawrence Erlbaum Associates, Inc., 1998.

[18] C. Buteau and G. Mazzola. From contour similarity to motivic topologies. *Musicae Scientiae*, 4(2):125–149, 2000.

[19] E. Cambouropoulos. A formal theory for the discovery of local boundaries in a melodic surface. In *Journees d' Informatique Musicale*, Caen, 1996.

[20] E. Cambouropoulos. Extracting 'significant' patterns from musical strings: Some interesting problems. In *London String Days 2000 workshop*, King's College London and Coty University, London, 2000.

[21] E. Cambouropoulos. The local boundary detection model and its application in the study of expressive timing. In *International Computer Music Conference*, La Havana, Cuba, 2001. http://www.ai.univie.ac.at/ emilios/pub.html.

[22] E. Cambouropoulos, M. Chrochemore, S. C. Lliopoulos, L. Mouchard, and Y. J. Pinzon. Algorithms for computing approximate repetitions in musical sequences. In *10th Australasian Workshop On Combinatorial Algorithms*, 1999. http://citeseer.nj.nec.com/cambouropoulos99algorithms.html.

[23] A. Camurri, R. Dillon, and A. Saron. An experiment on analysis and synthesis of musical expressivity. In *XIII Colloquium on Musical Informatics (CIM)*, L'Aquila, Italy, 2000. http://musart.dist.unige.it/sito_inglese/research/r_current/expressive.html.

[24] P. Cano. Fundamental frequency estimation in the sms analysis. In *COSTG6 Conference on Digital Audio Effects (DAFX)*, 1998. http://www.iua.upf.es/mtg.

[25] P. Cano, E. Batlle, E. Gómez, L. de C. T. Gomes, and M. Bonnet. Audio fingerprinting: concepts and applications. In *1st International Conference on Fuzzy Systems and Knowledge Discovery, Singapore*, November 2002. http://www.iua.upf.es/mtg.

[26] P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A review of algorithms for audio fingerprinting. In *International Workshop on Multimedia Signal Processing, US Virgin Islands*, December 2002. http://www.iua.upf.es/mtg.

[27] Y. Chi-Lap and B. Kao. A study of musical features for melody databases. In *10th International Conference and Workshop on Database and Expert Systems Applications, DEXA*, pages 724–733, Florence, Italy, 1999.

[28] M. Clausen, R. Engelbrecht, D. Meyer, and J. Schmitz. Proms: A web-based tool for searching in polyphonic music. In *International Symposium on Music Information Retrieval*, Indiana University, Plymouth, Massachusetts, 2000.

[29] G. W. Cooper and L. B. Meyer. Rhythms on lower architectonic levels. In *The rhythmic structure of music*. University of Chicago Press, Chicago, Illinois, 1960. http://www.music.indiana.edu/som/courses/rhythm/annotations/Cooper60.html.

[30] D. Cope. *Signatures and Earmarks: Computer Recognition of Patterns in Music*. Melodic Similarity: Concepts, Procedures, and Applications. MIT Press, Cambridge, Massachusetts, 1998.

[31] M. Crochemore, C. S. Iliopoulos, Y. J. Pinzon, and W. Rytter. Finding motifs with gaps. In *International Symposium on Music Information Retrieval, poster*, Indiana University, Plymouth, Massachusetts, 2000.

[32] F. J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[33] R. B. Dannenberg. A brief survey of music representation issues, techniques and systems. *Computer Music Journal*, 17(3):20–30, 1993. http://www-2.cs.cmu.edu/ music/papers.html.

[34] I. Deliège. Prototype effects in music listening: an empirical approach to the notion of imprint. *Music Perception*, 18(3), 2002.

[35] P. Desain and L. Windsor. *Rhythm perception and production*. Lisse: Swets & Zeitlinger, 2000. http://www.nici.kun.nl/mmm/papers/rpp-book/rpp-book.html.

[36] B. Doval and X. Rodet. Fundamental frequency estimation using a new harmonic matching method. In *International Computer Music Conference*, pages 555–558, 1991.

[37] N. Durand and E. Gómez. Periodicity analysis using a harmonic matching method and bandwise processing. In *MOSART Workshop on Current Research Directions in Computer Music*, Barcelona, 2001. http://www.iua.upf.es/mtg/publications/mosart2001-durand.pdf.

[38] T. Eerola, T. Järvinen, J. Louhivuori, and P. Toiviainen. Statistical features and perceived similarity of folk melodies. *Music Perception*, 18(3):275–296, 2001.

[39] N. Fogwall. The search for a notation index. web publication, April 2001. http://www.af.lu.se/ fogwall/notation.html.

[40] E. Gómez, F. Gouyon, P. Herrera, and X. Amatriain. Music description schemes for enhancing the current mpeg-7 standard. In *in preparation*, 2002.

[41] E. Gómez, F. Gouyon, P. Herrera, and X. Amatriain. Using the mpeg-7 standard for the description of musical content. *in preparation*, 2002.

[42] E. Gómez, A. Klapuri, and B. Meudic. Melody description and extraction in the context of music content processing. *Journal of New Music Research*, 2002. to appear.

[43] D. Godsmark and G. J. Brown. A blackboard architecture for computational auditory scene analysis. *Speech Communication*, 27:351–366, 1999.

[44] B. Gold and L. Rabiner. Parallel processing techniques for estimating pitch periods of speech in the time domain. *Journal of the Acoustic Society of America*, 46:442–448, 1969.

[45] M. Goto. A real-time music scene description system: Detecting melody and bass lines in audio signals. In *IJCAI Workshop on Computational Auditory Scene Analysis*, pages 31–40, 1999. http://www.etl.go.jp/ goto/PROJ/f0.html.

[46] M. Goto. A robust predominant-f0 estimation method for real-time detection of melody and bass lines in cd recordings. In *IEEE International Conference on Acoustics Speech and Signal Processing*, pages 757–760, 2000. http://www.etl.go.jp/ goto/PROJ/f0.html.

[47] F. Gouyon and B. Meudic. Towards rhythmic content processing of musical signals - fostering complementary approaches. *Journal of New Music Research*, to appear.

[48] W. Haas and H. Mayer. Mpeg and its relevance for content-based multimedia retrieval. *Journal of Universal Computer Science*, 7(6):530–547, 2001.

[49] B. Hammel. An essay on patterns in musical composition transformations, mathematical groups, and the nature of musical substance., 2000. http://graham.main.nc.us/ bhammel/MUSIC/compose.html.

[50] H. Hermansky, N. Morgan, and H. G. Hirsch. Recognition of speech in additive and convolutional noise based on rasta spectral processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 83–86, Minneapolis, Minnesota, 1993.

[51] P. Herrera and E. Gómez. Study report on audio segmentation. Technical report, CUIDADO internal report, Music Technology Group, Pompeu Fabra University, 12-11-2001.

[52] W. Hess. *Pitch Determination of Speech Signals. Algorithms and Devices.* Springer Series in Information Sciences. Springer-Verlag, Berlin, New York, Tokyo, springer-verlag edition, 1983.

[53] L. Hofmann-Engl. Towards a cognitive model of melodic similarity. In *International Symposium on Music Information Retrieval*, Indiana University, Plymouth, Massachusetts, 2001. http://www.chameleongroup.org.uk/research/model.html.

[54] J. L. Hsu, L. Chih-Chin, and A. L. P. Chen. Efficient repeating pattern finding in music databases. In *7th ACM international conference on information and knowledge management*, 1998.

[55] D. Huron. The humdrum toolkit: Software for music research, 1999. http://www.music-cog.ohio-state.edu/Humdrum/index.html.

[56] T. Jehan. *Musical Signal Parameter Estimation*. PhD thesis, CNMAT; IFSIC, 1997. http://www.cnmat.berkeley.edu/ tristan/Report/Report.html.

[57] K. Kashino, T. Kinoshita, and H. Tanaka. Organization of hierarchical perceptual sounds: music scene analysis with autonomous processing modules and a quantitative information integration mechanism. In *International Joint Conference On Artificial Intelligence*, Montreal, 1995.

[58] Y. E. Kim, W. Chai, R. Garcia, and B. Vercoe. Analysis of a contour-based representation for melody. In *International Symposium on Music Information Retrieval*, Indiana University, Plymouth, Massachusetts, 2000. http://www.media.mit.edu/ chaiwei/papers.html.

[59] A. Klapuri. Sound onset detection by applying psychoacousticknowledge. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1999.

[60] A. Klapuri. Qualitative and quantitative aspects in the design of periodicity estimation algorithms. In *European Signal Processing Conference*, 2000.

[61] A. Klapuri, T. Virtanen, A. Eronen, and J. Seppänen. Automatic transcription of musical recordings. In *Consistent & Reliable Acoustic Cues Workshop*, 2001.

[62] B. Kostek. Computer-based recognition of musica phrases using the rough-set approach. *Information Sciences*, 104:15–30, 1998.

[63] B. Kostek and M. Szczerba. Midi database for the automatic recognition of musical phrases. In *100th AES Convention*, Copenhagen, 1996.

[64] B. Kostek and M. Szczerba. Application of algorithms dealing with time domain uncertinty for the automatic recognition of musical phrases. In *102nd AES Convention*, Munich, 1997.

[65] C. L. Krumhansl. *Cognitive foundations of musical pitch*. Oxford University Press, New York, 1990.

[66] A. Lahat, R. J. Niederjohn, and D. A. Krubsack. A spectral autocorrelation method for measurement of the fundamental frequency of noise-corrupted speech. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(6):741–750, 1987.

[67] A. Lamont and N. Dibben. Motivic structure and the perception of similarity. *Music Perception*, 18(3), 2001.

[68] J. Laroche. Traitement des signaux audio-fréquences. Technical report, Ecole National Supérieure de Télécommunications, 1995.

[69] O. Lartillot, S. Dubnov, G. Assayag, and G. Bejerano. Automatic modeling of musical style. In *International Computer Music Conference*, La Havana, Cuba, 2001.

[70] M. Leman. Musical audio mining. In *Dealing with the data flood Symposium*, Rotterdam, 2002.

[71] M. Leman, L.P. Clarisse, B. De Baets, H. De Meyer, M. Lesaffre, G. Martens, J.P. Martens, and D. Van Steelant. Tendencies, perspectives, and opportunities of musical audio-mining. In *Forum Acusticum, session SS-MUS-01*, Sevilla, 2002.

[72] K. Lemström and P. Laine. Musical information retrieval using musical parameters. In *International Computer Music Conference*, Ann Arbour, 1998.

[73] F. Lerdahl and R. Jackendoff. *A generative theory of tonal music*. MIT Press, Cambridge, Massachusetts, 1983.

[74] V. I. Levenshtein. Binary codes capable of correcting deletions, insertion and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

[75] S. Z. Li. Content-based classification and retrieval of audio using the nearest feature line method. *IEEE transactions on speech and content based processings*, 2000.

[76] A. T. Lindsay. Using contour as a mid-level representation of melody. Technical report, Master of Science in Media Arts and Sciences Thesis, Massachusetts Institute of Technology, 1996.

[77] A. T. Lindsay and J Herre. Mpeg-7 and mpeg-7 audio - an overview. *Journal of the Audio Engineering Society*, 49(7/8):589–594, 2001.

[78] R. C. Maher and J. W. Beauchamp. Fundamental frequency estimation of musical signals using a two-way mismatch procedure. *Journal of the Acoustic Society of America*, 95:2254–2263, 1993.

[79] D. O Maidin. A geometrical algorithm for melodic difference. In W. B. Hewlett and E Selfridge-Field, editors, *Melodic Similarity: Concepts, Procedures, and Applications*. MIT Press, 1998.

[80] K. D. Martin. Automatic transcription of simple polyphonic music: Robust front end processing. In *Third Joint Meeting of the Acoustical Societies of America and Japan*, 1996.

[81] S. McAdams. Audition: physiologie, perception et cognition. In M. Robert J. Requin and M. Richelle, editors, *Traité de psychologie expérimentale*, pages 283–344. Presses Universitaires de France, 1994.

[82] R. J. McNab, L. A. Smith, and I. H. Witten. Signal processing for melody transcription. *SIG, Working paper*, 95(22), 1996.

[83] J. Medan, E. Yair, and D. Chazan. Super resolution pitch determination of speech signals. *IEEE Transactions on Signal Processing*, 39(1), 1991.

[84] R. Meddis and M. J. Hewitt. Virtual pitch and phase sensitivity of a computer model of the auditory periphery. i: Pitch identification. *JASA*, 89(6):2866–2882, 1991.

[85] M. Melucci and N. Orio. Musical information retrieval using melodic surface. In *4th ACM Conference on Digital Libraries*, Berkeley, CA, 1999.

[86] M. Melucci and N. Orio. The use of melodic segmentation for content-based retrieval of musical data. In *International Computer Music Conference*, Bejing, 1999.

[87] M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24:161–175, 1990.

[88] N. Nettheim. On the spectral analysis of melody. *Journal of New Music Research*, 21:135–148, 1992. http://users.bigpond.net.au/nettheim/specmel/specmel.htm.

[89] A. M. Noll. Cepstrum pitch determination. *Journal of the Acoustic Society of America*, 41:293–309, 1967.

[90] K. S. Orpen and D. Huron. Measurement of similarity in music : a quantitative approach for non-parametric representations. *Computers in music research*, 4:1–44, 1992. http://www.music-cog.ohio-state.edu/Huron/Publications/orpen.similarity.text.htm.

[91] G. Petterschmitt, E. Gómez, and P. Herrera. Pitch-based solo location. In *MOSART Workshop on Current Research Directions in Computer Music*, Barcelona, 2001.

[92] M. Piszczalski and B. A. Galler. Predicting musical pitch from component frequency ratios. *Journal of the Acoustic Society of America*, 66:710–720, 1979.

[93] L. R. Rabiner, M. R. Sambur, and C. E. Schmidt. Applications of a nonlinear smoothing algorithm to speech processing. *IEEE Trans. ASSP*, 23(6), 1975.

[94] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals*. Prentice-Hall, 1978.

[95] C. Road. Pitch and rhythm recognition in midi systems. In *The Computer Music Tutorial*, pages 503–531. The MIT Press, 1996.

[96] P. Y. Rolland. Discovering patterns in musical sequences. *Journal of New Music Research*, pages 334–350, 1999.

[97] P. Y. Rolland and J. Ganascia. Automated motive-oriented analysis of musical corpuses: A jazz case study. In *International Computer Music Conference*, San Francisco, 1996.

[98] P. Y. Rolland, G. Raskinis, and J. G. Ganascia. Musical content-based retrieval: an overview of the melodiscov approach and system. In *ACM International Multimedia Conference*, pages 81–84, 1999. http://www-poleia.lip6.fr/ rolland/#publications.

[99] J. Romero and S. Cerdá. Uso del análisis multirresolución para calcular el pitch de señales en presencia de ruido. *Revista de Acústica (SEA)*, 28, 1997. http://www.ia.csic.es/Sea/.

[100] S. Rossignol. *Segmentation et indexation des signaux sonores musicaux*. Phd thesis, Université Paris VI-IRCAM, 2000. http://sound.media.mit.edu/eds/thesis/.

[101] R. Rowe. *Machine Musicianship*. MIT Press, Cambridge, Massachusetts, 2001.

[102] E. D. Scheirer. *Music listening systems*. Phd thesis, Program in Arts and Sciences, MIT, 2000. http://sound.media.mit.edu/eds/thesis/.

[103] E. Selfridge-Field. *Conceptual and Representational Issues in Melodic Comparison*. Melodic Similarity: Concepts, Procedures, and Applications. MIT Press, Cambridge, Massachusetts, 1998.

[104] P. Smaragdis. *Redundancy reduction for computational audition, a unifying approach*. Phd, Massachusetts Institute of Technology, Media Laboratory, 2001.

[105] L. A. Smith, R. J. Mc Nab, and I. H. Witten. Sequence-based melodic comparison: a dynamic programming approach. In W. B. Hewlett and E. Selfridge-Field, editors, *Melodic Similarity: Concepts, Procedures, and Applications*. MIT Press, 1998.

[106] L. Solomon. Terms, 1996. http://www.azstarnet.com/ solo/glossary.htm.

[107] T. Sonoda, M. Goto, and Y. Muraoka. A www-based melody retrieval system. In *International Computer Music Conference*, 1998.

[108] C. Spevak, B. Thom, and K. Höthker. Evaluating melodic segmentation. In *Second International Conference ICMAI*, 2002. http://i11www.ira.uka.de/ musik/segmentation.

[109] D. Stammen and B. Pennycook. Real-time recognition of melodic fragments using the dynamic timewarp algorithm. In *International Computer Music Conference*, pages 232–235, San Francisco, 1993.

[110] D. Talkin. Robust algorithm for pitch tracking. In W. B. Kleijn and K. K. Paliwal, editors, *Speech Coding and Synthesis*. Elsevier Science B. V., 1995.

[111] E. Terhardt. Calculating virtual pitch. *Hearing Research*, 1:155–182, 1979.

[112] E. Terhardt, G. Stoll, and M. Seewann. Algorithm for extraction of pitch and pitch salience from complex tonal signals. *Journal of the Acoustic Society of America*, 71:679–688, 1981.

[113] B. Thom, C. Spevak, and K. Höthker. Melodic segmentation: evaluating the performance of algorithms and musical experts. In *International Computer Music Conference*, Göteborg, 2002. http://i11www.ira.uka.de/ musik/segmentation.

[114] P. Toiviainen and T. Eerola. A method for comparative analysis of folk music based on musical feature extraction and neural networks. In *VII International Symposium on Systematic and Comparative Musicology and III International Conference on Cognitive Musicology*, University of Jyväskylä, Finland, August 16-19 2001. http://www.cc.jyu.fi/ ptee/publications.html.

[115] M. Towsey, A. Brown, S. Wright, and J. Diederich. Towards melodic extension using genetic algorithms. *Educational Technology & Society*, 4(2), 2001. http://ifets.ieee.org/periodical/vol_2_2001/towsey.html.

[116] G. Tzanetakis. *Manipulation, analysis and retrieval systems for audio signals*. Phd thesis, Computer Science Department, Princeton University, June 2002. http://www.cs.princeton.edu/ gtzan/publications.html.

[117] A. L. Uitdenbogerd and J. Zobel. Manipulation of music for melody matching. In *ACM Multimedia*, 1998.

[118] A. L. Uitdenbogerd and J. Zobel. Melodic matching techniques for large music databases. In *ACM International Multimedia Conference*, Orlando, Florida, 1999.

[119] P. J. Walmsley, S. J. Godsill, and P. J. W. Rayner. Bayesian graphical models for polyphonic pitch tracking. In *Diderot Forum*, Vienna, 1999.

[120] G. Widmer. Using artificial intelligence and machine learning to study expressive music performance: Project survey and first report. *AI Communications*, 14(3):149–162, 2001.

[121] G. Widmer. In search of the horowitz factor: Interim report on a musical discovery project. invited paper. In *5th International Conference on Discovery Science (DS'02)*, Lübeck, Germany, 2002.

[122] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, pages 337–343, 1977.

Appendix A

Related publications

- Nicolas Durand and Emilia Gómez, *"Periodicity analysis using an Harmonic Matching method and Bandwise Processing"*, Proceedings of MOSART Workshop on Current Research Directions in Computer Music, Barcelona, 2001.
- Gilles Peterschmitt, Emilia Gómez and Pefecto Herrera, *"Pitch-based solo location"*, Proceedings of MOSART Workshop on Current Research Directions in Computer Music, Barcelona, 2001.
- Emilia Gómez, Fabien Gouyon, Perfecto Herrera and Xavier Amatriain, *"Music content description schemes and the MPEG-7 standard"*, in preparation.
- Emilia Gómez, Fabien Gouyon, Perfecto Herrera and Xavier Amatriain, *"Using the MPEG-7 standard for the description of musical content"*, in preparation.

Periodicity Analysis using a Harmonic Matching Method and Bandwise Processing

Nicolas Durand, Emilia Gómez
Music Technology Group, Pompeu Fabra University
{nicolas.durand, emilia.gomez}@iua.upf.es, http://www.iua.upf.es/mtg

Abstract

In this paper, two methods for fundamental frequency estimation are compared in order to study the advantages of using a bandwise processing for periodicity analysis. The first one is a harmonic matching algorithm that tries to match the peaks of the magnitude spectrum to a harmonic series. The second one splits the signal in separate frequency bands and computes an estimate for each band. Finally, the result is combined to obtain a global estimate.

1 Introduction

Pitch detection has always been an important field of research in the scope of speech and audio processing. There are a hundred of different methods that have been proposed and that work well for different types of sounds in different conditions.

The goal of this paper is to study the advantages of using a bandwise processing in a fundamental frequency estimator.

To do so, two algorithms for fundamental frequency estimation will be compared. The first one is a harmonic matching method that deals with a single band, and the second one processes separately different frequency bands.

2 Harmonic Matching Method

The Two-Way mismatch algorithm is a method that tries to find the harmonic series that best corresponds to the spectral peaks. This algorithm is presented at [5] and has been adapted to the SMS context (see [1]).

Once the peaks of the magnitude spectrum are identified, they can be compared to the predicted harmonics for each of the possible candidate note frequencies, and a measure to fit can be developed. A particular fitness measure is described in [5] as a Two-Way Mismatch procedure.

For each fundamental frequency candidate, mismatches between the harmonics generated and the measured partials frequencies are averaged over a fixed subset of the available partials. A weighting scheme is used to make the procedure robust to the presence of noise or absence of certain partials in the spectral data. The discrepancy between the measured and predicted sequences of harmonic partials is referred as the *mismatch error*. The harmonics and partials would “live up” for fundamental frequencies that are one or more octaves above and below the actual fundamental; thus even in the ideal case, some ambiguity occurs. In real situations, where noise and measurement uncertainty are present, the mismatch error will never be exactly zero.

The solution presented is to employ two mismatch error calculations. The first one is based on the frequency difference between each partial in the measured sequence and its nearest neighbor in the predicted sequence (see figure 1). The second is based on the mismatch between each harmonic in the predicted sequence and its nearest partial neighbor in the measured sequence.

This two-way mismatch helps avoid octave errors by applying a penalty for partials that are present in the measured data but are not predicted, and also for partials whose presence in the measured data is predicted but do not actually appear in the measured sequence. The TWM procedure has also the benefit that the effect of any spurious components or partial missing from the measurement can be counteracted by the presence of uncorrupted partials in the same frame.

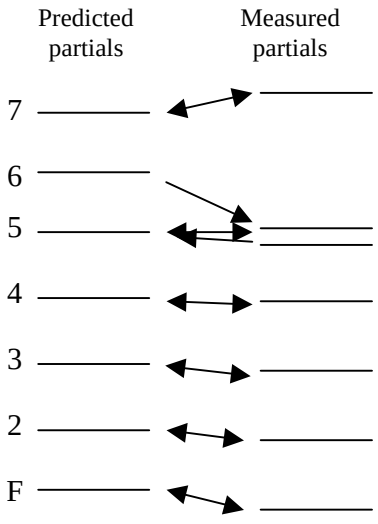


Figure 1: TWM procedure

The two error measurements are computed as following:

- Predicted-to-measured mismatch error

$$Err_{p\to m} = \sum_{n=1}^{N} E_w(\Delta f_n,\ f_n, a_n, A_{\max}) \quad (1)$$

$$= \sum_{n=1}^{N} \Delta f_n \cdot (f_n)^{-p} + (\frac{a_n}{A_{\max}}) \times \left[q\Delta f_n \cdot (f_n)^{-p} - r\right]$$

where a_n, f_n correspond to the amplitude and frequency of the predicted partial number *n*, A_{max} is the maximum amplitude, and Δf_n is the difference between the frequency of the predicted partial and its closest measured partial.

- Measured-to-predicted mismatch error

$$Err_{m\to p} = \sum_{k=1}^{K} E_w(\Delta f_k,\ f_k, a_k, A_{\max}) \quad (2)$$

$$= \sum_{k=1}^{K} \Delta f_k \cdot (f_k)^{-p} + (\frac{a_k}{A_{\max}}) \times \left[q\Delta f_k \cdot (f_k)^{-p} - r\right]$$

where a_k, f_k correspond to the amplitude and frequency of the measured partial number k, A_{max} is the maximum amplitude, and Δf_k is the difference between the frequency of the measured partial and its closest predicted partial.

The total error for the predicted fundamental frequency is then given by a combination of both errors:

$$Err_{total} = Err_{p\to m} / N + r \cdot Err_{m\to p} / K \quad (3)$$

The different parameters of the algorithm are set empirically.

This is the method used in the context of SMS (see [1]) including some improvements, as having pitch dependent analysis window, a selection of spectral peaks to be used, and an optimisation in the search for fundamental frequency candidates. In this algorithm the whole spectrum is processed at the same time, having as an input of the algorithm the collection of detected peaks from the magnitude spectrum.

3 Bandwise processing algorithm

Klapuri [3] proposed an algorithm for periodicity analysis that calculates independent fundamental frequencies estimates at separate frequency bands. Then, these values are combined to yield a global estimate. This solves several problems, one of which is inharmonicity. In inharmonic sounds, as stretched strings, the higher harmonics may deviate from their expected spectral positions, and even the intervals between them are not constant. However, according to the equation (4), we can assume the spectral intervals to be piece-wise constant at narrow enough bands, and increasing function of the center of the considered band.

$$f_n = nf_0\sqrt{1+bn^2} \quad (4)$$

where β is the inharmonicity factor, which value $\beta \in [0,0.0008]$.

Thus we utilize spectral intervals to calculate pitch likelihoods at separate frequency bands, and then combine the results in a manner that takes the inharmonicity into account. Another advantage of bandwise processing is that it provides robustness in the case of badly corrupted signals, where only a fragment of the whole frequency range is good enough to be used.

A single fast Fourier transform is needed, after which local regions of the spectrum are separately processed. Before the bandwise processing, the spectrum is equalized in order to remove both additive and convolutive noise simultaneously as explained at [3] and seen at equation (5). This method is based on the RASTA spectral processing [2].

First, a transformation is applied to the magnitude spectrum. This transformation makes additive noise go through a linear-like transformation while the harmonic spectrum go through a log-like transform.

Then, a moving average is subtracted in order to eliminate convolutive noise.

$$X_e(k) = \log\left[1 + J * X(k)\right] - X_{av}(k) \quad (5)$$

The equalized spectrum is processed in 18 logarithmically distributed bands that extend from 50Hz to 6000Hz. Each band comprises a 2/3-octave wide region of the spectrum that is subject to weighting with a triangular window. Overlap between adjacent bands is 50%, which makes them sum unity when the windowing gets into account. Fundamental frequency prominence vectors are calculated at each band as explained at [4] according to the following equation:

$$\begin{aligned} L_B(n) &= \max_{m \in M}\left\{ W(H) \sum_{h=0}^{H-1} X_e(k_B + m + hn) \right\} \\ m &\in M = \{0,1,\ldots,n-1\} \\ H &= \left[(K_B - m)/n\right] \\ W(H) &= 0.75/H + 0.25 \end{aligned} \quad (6)$$

Finally, the likelihood values are combined getting into account that fundamental frequency can increase as a function of the band center frequency for string instruments. Some improvements were made to provide robustness in interference, where pitch is observable only at a limited band, and to adapt the algorithm to signals containing a mixture of several harmonic sounds.

This filter bench is much alike to the human hearing system's filtering properties. Ear acts like an analyzer composed of a set of "continuous" band pass filters. The bandwidth of a noise affects the loudness of the sound, allowing the definition of a critical bandwidth, function of the band center. Critical bandwidth is usually between 1/6 and 1/3 octave.

We use here a discrete bench of 18 triangular filters whose efficient bandwidth is 1/3 octave, covering the 50Hz – 6400 Hz range (see figure 2). Furthermore, we normalize the filters in energy, so that the filter bench remains coherent when applied to a power spectrum.

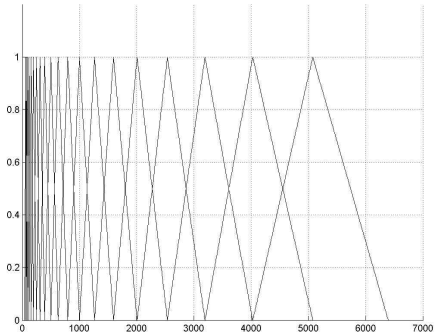


Figure 2: Bark bands triangular filter bench

4 Algorithm comparison

For isolated notes, we look at the output of both algorithms in order to decide if the correct pitch has been detected. As the fundamental frequency estimation is performed frame by frame, the algorithm performance at transitions tracking is not relevant.

For polyphonic sounds, we can in the same way judge if the predominant pitch has been correctly found.

We dealt with sounds of different natures:

- Quasi-harmonic sounds as wind instruments (saxophone, trumpet, clarinet, etc).
- Sounds whose harmonics are not equally spaced, presenting a small inharmonicity factor as string instruments (piano, guitar).
- Sounds with a strong inharmonicity as bells notes.
- Noisy sounds, in order to analyze the behavior against noise.
- Spectrums where a frequency band is filtered in order to measure the bandwise robustness.
- Polyphonic sounds: in this case, we try to extract the predominant fundamental frequency, i.e., the frequency that presents the clearest harmonics. The advantages of using a band-wise processing are also evaluated when dealing with polyphonic sounds, because the predominant frequency may be clear only in a small frequency band.

In order to study the behavior for a wide frequency range, sounds of low and high pitch have been used.

5 Results

For harmonic sounds, both algorithms performances are similar, as can be seen at the following figure.

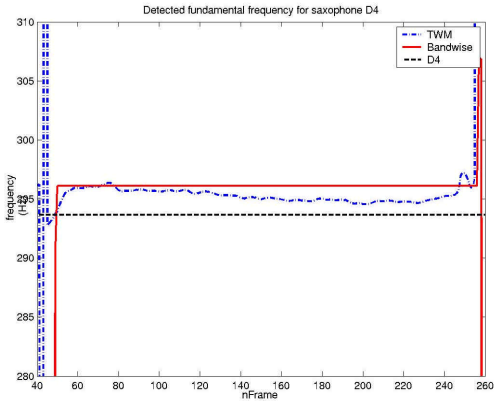


Figure 3: Detected fundamental frequency for saxophone D4 Note

The main difference between both methods is the frequency resolution. The TWM algorithm gets a better spectral resolution by interpolation of the magnitude spectrum when detecting spectral peaks. The spectral resolution could be also decreased using zero-padding.

For sounds whose harmonics are not equally spaced, some differences are found. First, the bandwise algorithm gets into account inharmonicity, which avoids a number of false pitch estimations, particularly if a part of the spectrum is damaged (overnoised or erased). And the algorithm estimates the inharmonicity factor, which is useful for multipitch detection.

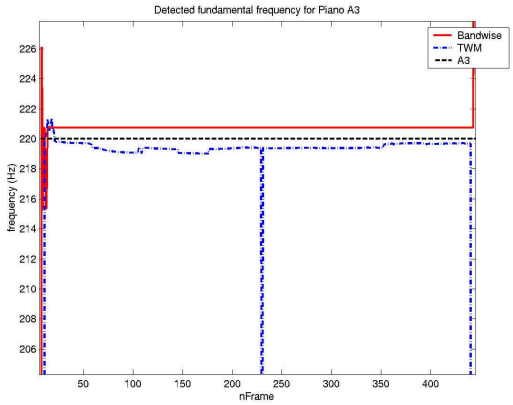


Figure 4: Detected fundamental frequency for piano A3

For noisy sounds, it appears that the bandwise algorithm is much more efficient, but this is mainly thanks to RASTA preprocessing which efficiently cleans the spectrums. Fundamental frequency is thus tracked nearly as long as the isolated note is clearly hearable in the original signal.

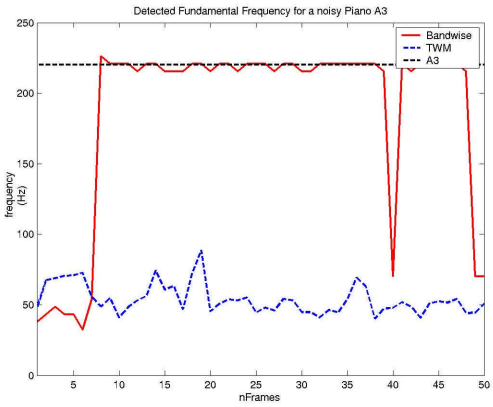


Figure 5: Detected fundamental frequency for noisy piano A3 note

Polyphonic sounds allow a comparison of the algorithms performance in another kind of noisy environment (mix containing many instruments). In this kind of environment, the TWM algorithm proved weaker than the bandwise processing.

Furthermore, bandwise processing aims towards multipitch estimation (MPE) explained at [4]. For each frame, the general model we use extracts one fundamental frequency and its associated inharmonicity coefficient at a time. This coefficient contains information about the locations of harmonics, making the building of a one-note spectrum more reliable. Before subtracting it to the equalized spectrum, we apply to the harmonics heights a smoothing (“smooth+min”) in order to leave a part of the partials coinciding with other notes’.

After subtraction, the pitch detection algorithm may look for a new pitch in the same frame. The efficiency in MPE depends mainly on this one-note spectrum subtraction process. Therefore, it proved working well with mixtures of isolated notes whose harmonics are clear enough (violin, for example).

6 Conclusions and perspectives

The advantages of using bandwise processing for periodicity analysis have been tested. The main differences between both algorithms performances can be observed when the harmonicity is found in a particular frequency band, as for example filtered and polyphonic sounds.

It has also been proved that the equalization performed by the bandwise processing algorithm make this method more robust to the influence of noise.

As a perspective, we could try to test if the equalization is also valid as a general preprocessing method for the TWM algorithm, and we could think of applying some kind of post-processing to eliminate isolated errors and abrupt transitions between consecutive frames.

Another possibility is to apply a harmonic matching method to separated frequency bands, instead of computing frequency likelihoods or prominence vectors for each single frequency. This would imply an optimization of the computation charge of the algorithm.

Further developments may be done to improve multipitch estimation. In fact, we observed that, although it is a good modelisation, the use of the inharmonicity factor is not always precise enough to locate rightly the harmonics of a note. We actually tried to cross the detection methods. As RASTA preprocessing leads us to working on “denoised” spectrums, it is efficient to pick the peaks in the spectrum (like in TWM algorithm) and match them with the predicted sequence of harmonics to obtain an efficient reconstitution of a one-note spectrum before subtraction.

7 Acknowledgments

The authors would like to thank the members of the Music Technology Group for their help and support.

They would also like to thank Anssi Klapuri for his comments and explanations on bandwise processing for fundamental frequency detection.

The work reported in this paper has been partially funded by the IST European project CUIDADO [6] and by the TIC national project TABASCO.

References

[1] Cano, P, "Fundamental Frequency Estimation in the SMS Analysis", DAFX,1998.

[2] Hermansky, H, Morgan, N, and Hirsch, H. G, "Recognition of speech in additive and convolutional noise based on RASTA spectral processing.", ICASSP,1993.

[3] Klapuri, A, "Qualitative and quantitative aspects in the design of periodicity estimation algorithms", EUSIPCO,2000.

[4] Klapuri, A, Virtanen, T, and Holm, J. M, "Robust multipitch estimation for the analysis and manipulation of polyphonic musical signals", DAFX,2000.

[5] Maher, R. C and Beauchamp, J. W, "Fundamental frequency estimation of musical signals using a two-way mismatch procedure", Journal of the Acoustic Society of America, Vol. 95, page 2254-2263, 1993.

[6] CUIDADO IST project, http://www.cuidado.mu

Pitch-Based Solo Location

Gilles Peterschmitt
Emilia Gomez
Perfecto Herrera
Music Technology Group, Pompeu Fabra University
http://www.iua.upf.es/mtg
{gilles.peterschmitt, emilia.gomez, perfecto.herrera}@iua.upf.es

Abstract

The aim of this work is to study how a pitch detection algorithm can help in the task of locating solos in a musical excerpt. Output parameters of the pitch detection algorithm are studied, and enhancements for the task of solo location are proposed. A solo is defined as a section of a piece where an instrument is in foreground compared to the other instrument and to other section of the piece.

1 Introduction

Music browsing is a manifold activity that comprehends behavior as diverse as retrieving songs by different musical criteria, creating play lists that follow subjective criteria, selecting special excerpts, visualizing rhythmic or harmonic structures, etc. Recent advances and research projects on music content processing [1, 2, 3] give way to think that some of those activities will be performed soon in an automatic user-configurable way.

Instrumental solos are interesting and characteristic parts of a musical piece with a special status not only for a musicologist but even for a home-listener. A music browser should then provide with some functionalities in order to allow the user:

- to spot and fast browse solos inside music works;
- to visualize relevant music information of the solo (i.e. the score);
- to compile a list of solos by a given performer, or by a given instrument from the available music database; provided an adequate constraint satisfaction system, this mega-solo play list could follow some subjective and musical directions [4].

Besides the mentioned practical motivations, solo location has a research-related interest as a type of pre-processing intended to be useful for deriving instrumentation descriptions of complex music mixtures.

As far as we have been able to trace, there is no specific literature on automatic solo location. Therefore we have started our study with a conceptual analysis of the possible acoustic differences between what we may consider a "solo" and what we may consider an "ensemble" performance.

In this paper, a solo is defined as a section of a piece where an instrument is in foreground compared to the other instruments and to other sections of the piece. In physical terms, this means that spectra of solo sections should be dominated by one instrument. It is clear that the previous definition is highly debatable from the musicological point of view, but it should be accepted as a reasonable starting point from where some refinements can be done after careful study and testing. For a more formal definition of what can be considered a musical solo, the reader might consult the Grove Dictionary of Music [5].

One of the first approaches to getting some discriminative data for solo sections is looking at some spectral complexity measure. We assume that in audio segments where ensemble performance is predominant, the spectrum is more complex (i.e. with larger variability of spectral peaks location and amplitudes). Given that in the context of music browsing some pitch information is required, we thought that the above-mentioned measurements could be obtained as a "side-effect" of a pitch extraction process (hence without increasing the computational load of a system). We argued that in solo sections, pitch could be reasonably tracked by a common monophonic pitch detection algorithm, the *Two-Way Mismatch* [6], and therefore the pitch error indexes to be found in solo sections would be smaller than those to be found in "ensemble" sections. As we will see, the error indexes did not show enough discriminative power, and further enhancements were attempted.

2 TWM algorithm description

The used pitch estimation algorithm is described at [6]. This algorithm tries to extract a fundamental frequency from a set of spectral maximum of the magnitude spectrum of the signal. These peaks can be compared to the predicted harmonics for each of the possible candidate note frequencies. A particular fitness measure is described in [6] as a "Two-Way Mismatch" procedure. For each candidate, mismatches between the harmonics generated and the measured partials frequencies are averaged over a fixed subset of the available partials. The discrepancy between the measured and predicted sequences of harmonic partials is referred as the *mismatch error*. The solution presented on [6] is to employ two mismatch error calculations.

The first one is based on the frequency difference between each partial in the measured sequence and its nearest neighbor in the predicted sequence. The second is based on the mismatch between each harmonic in the predicted sequence and its nearest partial neighbor in the measured sequence.

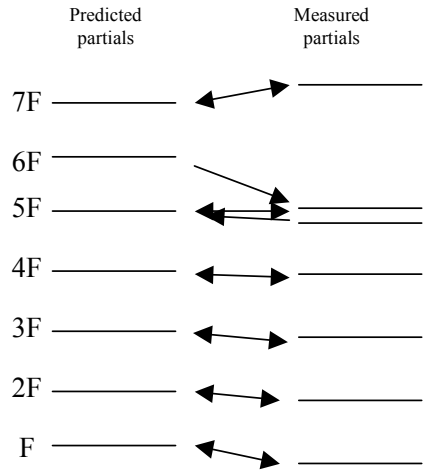


Illustration 0: TWM procedure

The two error measurements are computed as following:

- Predicted-to-measured mismatch error:

$$Err_{p\to m} = \sum_{n=1}^{N} E_\omega(\Delta f_n, f_n, a_n, A_{\max})$$

$$= \sum_{n=1}^{N} \Delta f_n \cdot (f_n)^{-p} + \left(\frac{a_n}{A_{\max}}\right) \times \left[q\Delta f_n \cdot (f_n)^{-p} - r\right] \quad (1)$$

where a_n, f_n correspond to the amplitude and frequency of the predicted partial number *n*, A_{max} is the maximum amplitude, and Δf_n is the difference between the frequency of the predicted partial and its closest measured partial.

- Measured-to-predicted mismatch error:

$$Err_{m\to p} = \sum_{k=1}^{K} E_\omega(\Delta f_k, f_k, a_k, A_{\max})$$

$$= \sum_{k=1}^{K} \Delta f_k \cdot (f_k)^{-p} + \left(\frac{a_k}{A_{\max}}\right) \times \left[q\Delta f_k \cdot (f_k)^{-p} - r\right] \quad (2)$$

where a_k, f_k correspond to the amplitude and frequency of the measured partial number k, A_{max} is the maximum amplitude, and Δf_k is the difference between the frequency of the measured partial and its closest predicted partial.

The total error for the predicted fundamental frequency is then given by:

$$Err_{total} = Err_{p\to m} / N + \rho \cdot Err_{m\to p} / K \quad (3)$$

The parameters *p, m, r* and ρ are set empirically and vary for each instrument.

3 TWM Output Errors Behavior

2.1 Errors

The three errors are crucial for pitch detection. However, from its definition, the PM error will be of first interest for our purpose. Indeed, the PM matches a set of predicted peaks with the set of measured spectral peaks. Its values are therefore usually lower and less erratic than that of the MP error, which tries to match a great number of measured peaks with the predicted peaks. The total error, which is a weighted combination of both errors, does not need precise description. We will therefore focus our attention to the study of the PM error.

2.2 PM Error Behavior

The optimal parameters input to the pitch detection algorithm are set carrying out tests on monophonic recordings of the instrument considered. If the parameters are optimally set for this instrument, the algorithm estimates the pitch correctly and the PM error is usually minimal. Using the algorithm on polyphonic sounds does not enable good pitch

estimation but leads to interesting output errors behavior.

For example, if the parameters are set optimally for a saxophone, good pitch estimation occurs if the saxophone plays on its own. The PM error is at its lowest as there is an evident match between the predicted peaks and the peaks present in the spectrum. In the presence of other instruments, the error is high (due to the addition of spectral peaks that belong to different harmonic series and instruments) and pitch estimation is usually corrupted. But if the saxophone "dominates" enough, some of the pitch can still be estimated. In the spectrum, some of the harmonic peaks of the saxophone are detectable and sensible matching is possible. The PM error in this case will be higher than in the monophonic case, but lower than when no instruments are in foreground. Moreover, the parameters being optimized for a given instrument -for example the saxophone-, it should give a lower error if the saxophone is in foreground than if an instrument with very different spectral characteristics is in foreground.

Figure 1 below shows the PM error output to the analysis of an extract of a piece by a Miles Davis ensemble. In this extract, the background (piano, drum and bass) is very quiet, and the saxophone plays clear and relatively loud notes, making the example visually explicit (note that it is not always the case).

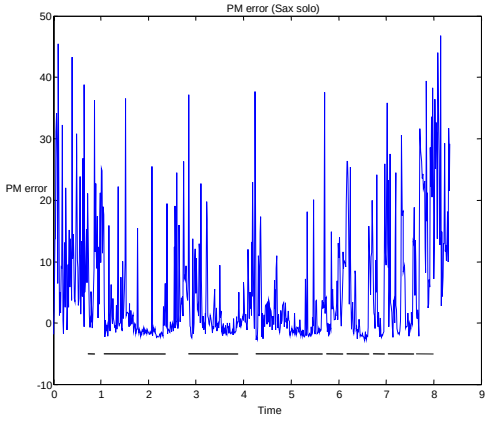


Figure 1: PM error against time for an extract of a piece by a jazz ensemble. The thick lines under the curve show the saxophone notes. Clear decreases can be observed when the saxophone plays.

The PM error behavior described above can be observed. The PM error, high for the ensemble mode, decreases a lot when the saxophone presence is dominant.

4 Solo Location

This behavior of the PM error, which in turn gets reflected in the Total error, can be used as a sign of spectral complexity, and therefore help in the task of solo location. Theoretically, the discrimination between "ensemble" and "solo" mode should be possible by detecting long-term changes in the mean of, say the PM error –high values representing "ensemble" mode and low values, solos.

From the statistical analysis of frame by frame data, it is clear that frames corresponding to solos have different average error values than frames corresponding to ensemble sections, and that both data distributions might have different origins (t-student -for means- and Kolmogorov-Smirnov -for distribution- tests results are omitted for convenience). Notwithstanding, an attempt to use a linear classifier (PDA) with the three features as predictors of category (solo/ensemble) yielded a mere 56% of success, after cross validation with a Jackknifed procedure. This was a disappointing result that could not be improved even using a quadratic classifier.

It seems clear, then, that solo location using the error parameters for the discrimination cannot be made on a frame by frame basis, as the variability of the error is large compared to the change in the mean we want to detect. Also, the change in the mean is neither very large nor neat at the solo boundaries. Using the TWM output PM and Total error alone, does not enable proper solo location.

A proposed way to tackle this problem was to use segmentation techniques such as Foote's similarity matrix [7] prior to the discrimination. Using such a technique with appropriate descriptors should enable to locate the boundaries of the different "parts" of the piece. The ensemble/solo discrimination using the long-term PM error level changes can then be done on these pre-located segments.

Studies were carried out in order to find relevant descriptors for locating these specific boundaries. Three ones were found to be particularly useful for this purpose: the spectral centroid, the skewness and the kurtosis. Adding the PM and Total error in the feature vector enabled better boundary location in a few cases where the spectral parameters were not sufficient for a good segmentation. However it sometimes misleads the segmentation result more than it helps. On 15 tests, adding these parameters helped 5 times (3 of which enabling segmentation when it was not previously possible), and corrupted the results importantly on only one occasion.

A summary of the procedure is given below:

1. The TWM input parameters, that are empirically set, are adapted to the solo instrument in order to improve the fundamental frequency estimation. In this step, monophonic sounds have been used.

2. The algorithm is applied to polyphonic sounds and the spectral features are calculated (Spectral Centroid, Skewness, Kurtosis). The analysis is done by frames of 0.0161s (512 samples at 44.1 kHz). This short frame length is necessary for a good performance of the pitch detection algorithm. The PM error, Total error and the three spectral parameters are extracted.

3. The features, averaged over segments of 50 frames, are input to the Foote's segmentation algorithm, and the candidates for Ensembles-Solos boundaries are automatically located according to the value of a "novelty score" (see figure 2) (the parameters for this step have to empirically set, although in the future the algorithm could adapt to the data).

4. The PM error is averaged over these pre-located segments and a decision taken for Solo or Ensemble mode.

5 A Case Study

As our initial database for testing is still rather small (15 songs), no significant and robust numerical data can be provided. Anyway, a case study will illustrate some specificities, pros, and cons of the procedure.

The following example illustrates the analysis for a piece by a John Coltrane ensemble. The ensemble is formed of alto saxophone, trumpet, piano, drum and bass. The piece starts with the ensemble until a saxophone solo starts around 37 seconds into the piece. The following parameters were input to the algorithm:

TWM input parameters (Optimal saxophone parameters):

- Window length of 0.0161 s (512 samples at sampling rate of 44100 Hz);
- Pitch range: the pitch detection analysis was carried out between 1000 and 3000 Hz;
- TWM parameters: p=0.5, q=1.4, r=0.5 in Equations (1) and (2), ρ=0.33 in Equation (3).

Segmentation parameters:

- Features were averaged over segments of 50 samples;
- Mahalanobis distances were calculated between feature vectors;
- Size of kernel: 30 averaged segments;
- Threshold for peak picking in Novelty Score curve: 3000 (empirically set).

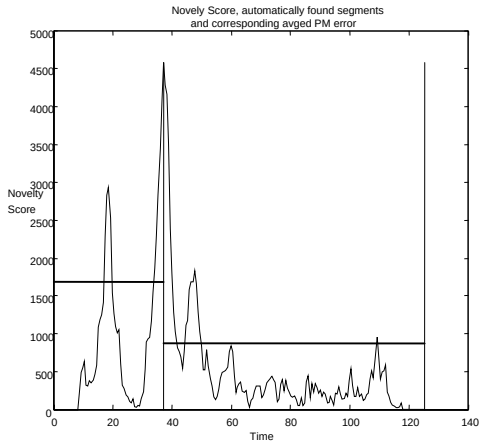


Figure 2: Novelty Score against time, with automatically found segments and corresponding PM error levels for an extract of "Blue Train" by John Coltrane. The transition form the theme to the solo was accurately found, and the error levels enable good discrimination.

The analysis of this piece showed to be particularly successful. The theme/solo boundary, located at 37 seconds into the piece was automatically found at 37.2 s, and the change in the mean of the PM error goes from 8.8 during the theme to 11.2 during the solo, enabling good discrimination. It can be noted that the good performance of the algorithm is very dependent on the piece, and that this piece is particularly suited for this purpose. There is a clear spectral change from the theme to the solo enabling a good boundary location. The solo is clear and continuous against a quite background giving easy ensemble/solo discrimination. This is not the case for all pieces.

6 Discussion and further work

First of all, problems with the TWM algorithm performance were encountered when dealing with instruments whose spectra do not show clear harmonic behavior. For example, very good results were obtained with quite harmonic solo instruments as saxophone, trumpet and violin, but with guitar and piano, the TWM could not give reliable results. A solution proposed to this problem is to use pre-processing techniques, such as the "noise suppression" technique (to remove additive and convolutive noise) proposed by A. Klapuri [8]. This technique enables to boost the spectral peaks, possibly enhancing the TWM performance. Including

an inharmonicity factor to correct for the stretched harmonics of the piano could also be beneficial.

Another problem resides in the automatic segmentation, whose input parameters vary from pieces to pieces (i.e. threshold for peak detection in novelty score curve, size of kernel, etc.). Studies have to be carried out in order to find ways to adapt the algorithm to the data. Also, more features should be added to the segmentation algorithm in order to get more robust boundaries location. Measures of spectral peak variability and amplitude modulation patterns (in order to detect beatings) are under current scrutiny.

Finally, the main problem resides in the concept we want to extract. This algorithm is basic and uses low-level descriptors to extract a rather abstract concept. This causes a number of inconsistencies and limits the robustness of the algorithm. First of all, the location is done from low-level features, which enables to locate solos with respect to these features only. That is, in order for a solo to be detected as such, it has to be:

- clear: relatively loud solo instrument compared to the background;
- continuous: if the solo consists of solo instrument lines with short ensemble intervention in between each solo lines, the level of the PM error will be raised considerably and the discrimination might be corrupted;

The ensemble mode parts have to be very characteristic as well: for example, either if the theme is played by one single instrument or by two instruments at unison, it might be considered as a solo.

This raises the problem of extracting a musical concept with low-level descriptors. It shows to work well in a lot of cases, but in reality, what is extracted is a “physical” concept of a solo (an instrument dominating the spectra) rather than a solo in a musical sense of the term. The variability of the abstract concept is too high for low-level physical features to describe it in its entirety. Higher-level descriptors and more powerful classification techniques could be used to take into account musical knowledge on solo location. For example, we know that it is statically more robust to detect ensembles than solos. Post-processing techniques could be used to correct the uncertain chunks of data with respect to these observations. Finally, recent studies were made on spectral flatness and the associated coefficient of tonality [9]. First tests showed this feature to be potentially useful in the task of locating solos [10], especially in the discrimination step. It could be added to the PM error feature in order to increase the discrimination robustness.

7 Acknowledgments

The work reported in this article has been partially funded by the IST European project CUIDADO [3].

References

[1] Agrain, P. (Guest Editor) "Musical Content Feature Extraction". Journal of New Music Research, 28 (4), 1999.

[2] Music Content Analysis web pages: http://www.cs.tut.fi/sgn/arg/music/. Tampere University of Technology, Finland.

[3] CUIDADO project. European Comission IST Program. http://www.cuidado.mu/

[4] Pachet, F. Roy, P. "Automatic Generation of Music Programs", Proceedings of the CP (Constraint Programming) Conference, Washington (USA), October 1999.

[5] Grove Dictionary of Music, http://www.grovemusic.com.

[6] Maher, R. C and Beauchamp, J. W, "Fundamental frequency estimation of musical signals using a two-way mismatch procedure", Journal of the Acoustic Society of America, Vol. 95, page 2254-2263, 1993.

[7] Foote, J. “Automatic audio segmentation using a measure of audio novelty”, Proceedings of IEEE International Conference on Multimedia and Expo, vol. I, pp. 452-455, 2000.

[8] Klapuri, A. Virtanen, T. Eronen, A. Seppänen, J. “Automatic Transcription of Musical Recordings”, http://www.cs.tut.fi/sgn/arg/klap/crac2001/crac2001.pdf, 2001.

[9] Johnsotn, J.D. 1988. “Transform Coding of Audio Signals Using Perceptual NoiseCriteria”, IEEE Journal of Selected Areas in Communication, Vol. 6, pp. 314-323, 1988.

[10] Izmirili, O. “Using a Spectral Flatness Based Feature for Audio Segmentation and Retrieval”, http://ciir.cs.umass.edu/music2000/posters/izmirli.pdf, 2000.

Music Content Description Schemes and the MPEG-7 Standard

EMILIA GÓMEZ, FABIEN GOUYON, PERFECTO HERRERA, XAVIER AMATRIAIN
Music Technology Group, Institut Universitari de l'Audiovisual
Universitat Pompeu Fabra
Passeig de Circumval·lació, 8, 08003 Barcelona
SPAIN
{emilia.gomez, fabien.gouyon, perfecto.herrera, xavier.amatriain}@iua.upf.es
http://www.iua.upf.es/mtg/

Abstract: The aim of this paper is to review and discuss possible ways of describing the content of musical files in the context of a specific software application (a tool for content-based management and edition of samples and short audio phrases). Available or feasible descriptors and description schemes for different musical layers (melodic, rhythmic and instrumental) are examined, and its relationship with the existing standard for multimedia content description MPEG-7 is discussed.

Keywords: - music description, MPEG-7, standard, melody, rhythm, instrument.

1. Introduction

Describing the musical content of audio files has been a pervasive goal in the computer music and music processing research communities. Though it has been frequently equated to the problem of "transcription", describing music content usually implies an applied context that has a "home-" or "non-scholar" user in the final end of the chain. Therefore, it is usually the case that "conventional" music data types are not the perfect ones nor the final structures for storing content descriptions that are going to be managed by people with different backgrounds and interests (probably quite different from the purely musicological). This approach to music content description has also been that of the standardizing initiative carried out since 1998 by the ISO workforce that has been known as MPEG-7. MPEG-7 is a standard for multimedia content description that was officially approved in 2001 and is currently being further expanded. It provides descriptors (hence D's) and description schemes (hence DS's) for different audio-related needs such as speech transcription, sound effects classification, and melodic or timbral-based retrieval.

The CUIDADO project (Content-based Unified Interfaces and Descriptors for Audio/music Databases available Online) is also committed with "applied" music description in the context of two different software prototypes, the so-called *Music Browser* and *Sound Palette*. The former is intended to be a tool for navigation in a collection of popular music files, whereas the latter is intended to be a tool for music creation based on short excerpts of audio (samples, music phrases, rhythm loops…). More details on these prototypes can be found in [25]. The development of the Sound Palette calls for a structured set of description schemes covering from signal-related or low-level descriptors up to user-centered or high-level descriptors.

2. Melody description

In this section, we will review how melody has been described and represented in the literature. There has been an important contribution in the context of *Query by Humming* systems, where it is necessary to represent data in a efficient manner so that it can be matched against a database of melodies. In this context, [16] identifies the properties that a melody representation scheme should have: *compactness* (so that the representation can be easily stored), *expressiveness* (or the ability to retain the expressive qualities of the singing voice) and *portability* (the representation must be adaptable to many types of inputs and objects it seeks to match).

Melody has been mainly described using pitch information. As a forward step, pitch contour has also been used in several applications such as query by humming, similarity matching or melodic classification, because it has been found to be more significant to listeners in determining melodic similarity. The earliest approaches disregarded timing information completely, but more recent studies showed that durational values facilitates melody recognition (see [10] for review).

Other information besides pitch and duration can also be taken into account. One example appears in the MPEG-7 standard, where key and scale information are also coded. This description scheme can be found at [1] and [2], and is explained in [17].

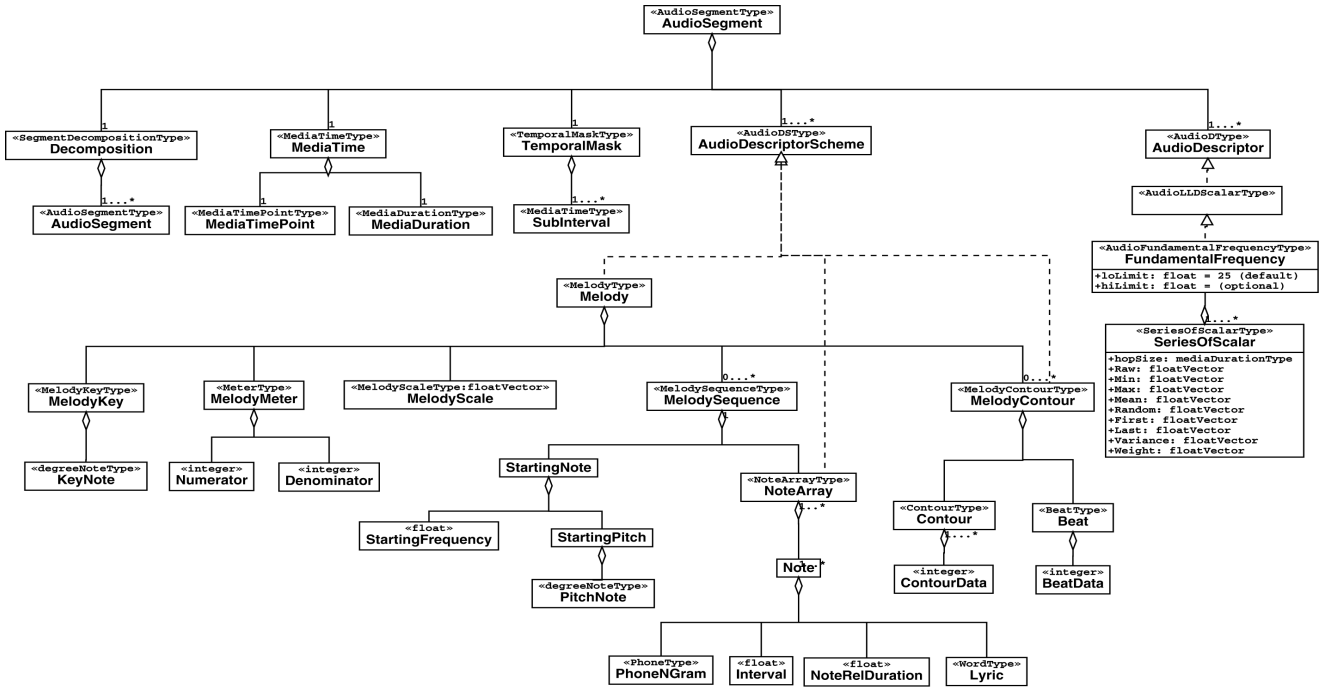


Figure 1: MPEG-7 Melody DS

MPEG-7 proposes two levels of melodic description by the *mpeg7:MelodySequence* and the *mpeg7:MelodyContour* descriptors, plus some information about scale, meter, beat and key (see figure 1). The melodic contour uses a 5-step contour (from –2 to +2) in which intervals are quantized, and also represents basic rhythm information by storing the number of the nearest whole beat of each note, which can drastically increase the accuracy of matches to a query. However, this contour has been found to be inadequate for some applications, as melodies of very different nature can be represented by identical contours. One example is the case of having a descendant chromatic melody and a descendant diatonic one. Both of them have the same contour although their melodic features are very unlike.

For applications requiring greater descriptive precision or reconstruction of a given melody, the *Melody* DS supports an expanded descriptor set and higher precision of interval encoding (*mpeg7:MelodySequence*). Rather than quantizing to one of five levels, the precise pitch interval (with cent or greater precision) between notes is kept. Timing information is stored in a more precise manner by encoding the relative duration of notes defined as the logarithm of the ratio between the differential onsets. In addition to these core descriptors, MPEG-7 define a series of *optional support* descriptors such as *lyrics*, *key*, *meter*, and *starting note*, to be used as desired for an application.

Other features based on pitch information have been used for melody description. They can be classified in four different categories when considering three different levels of analysis:

1. **Distributional and frequential features:** features derived from a numerical or statistical analysis of pitch information.

- Tessitura: or pitch range, that is the amplitude of the melodic contour [23].
- Interval distribution: types of intervals used [22].
- Pitch variety: measure of diversity of the pitch class set used in the melody [23].
- Repetitions: measure the number of repeated pitches [23], that represents a measure of the melodic movement.
- Melodic profile: depending on the pitch contour we could define several types of pitch profiles: ascending, descending, constant, etc. [8,23].
- Melodic density: or degree of melodic activity, which can be defined in relation with the rhythmic distribution of notes.
- Contour features: some features derived from an analysis of the contour information are proposed in [23]:
  - Contour direction: overall tendency of the melody to rise or fall.
  - Contour stability: the proportion of intervals for which the following interval is in the

same direction gives an idea of stability in melodic direction.

- Movement by step: proportion of intervals that are diatonic steps. A high score indicates a smooth melodic curve with few large leaps.
- Leap returns: proportion of large (leap) intervals NOT followed by a return interval. A large leap is greater than or equal to 8 semitones (minor 6th). The returning interval must be at least 1 semitone but less than the leap interval preceding it.
- Climax Strength: measured as the inverse of the number of times the climatic note is repeated in the melody. The highest value of 1 for this feature occurs when the climatic note is used only once. More frequent use lessens the climatic impact.

2. **Tonality related features:** these features require a musical analysis of the pitch data:

- Key information [17].
- Scale information: scale and type of scale (diatonic, chromatic, pentatonic, etc) [17].
- Key centered: defined in [23] as the proportion of *quanta* (being *quantum* equal to the shortest duration of a note) where the pitch is primary, that is either tonic or dominant. This feature is an indication of how strongly the melody has a sense of the key.
- Non-scale notes: indication of how strongly tonal the melody is by measuring the number of quanta that does not belong to the scale [23].
- Dissonant intervals: measure of the fraction of dissonant intervals [23].
- Cadence information: which types of cadences are used.

3. **Features derived from a structural analysis:** it is necessary to analyze the structure of the pitch sequence.

- Motives and patterns: analyze which are the melodic patterns or motives that are used. Some references can be found in [10] section 5.
- Phrase description: some features can be derived from phrase segmentation (also including musical knowledge).

The introduced features corresponding to these three categories have been successfully used for different applications in different contexts (e.g. algorithmic composition [23], comparative analysis [22] or music information retrieval [8]). However, if we want to share descriptions between different systems, these existing features have to be assembled into a common multi-level description scheme, which is a very hard task to carry on.

4. **Emotional descriptors:** Some meaningful and interesting higher-level features (as for example emotional descriptors) may be derived from the ones introduced above. For example: an ascending profile and a medium-large density could be related with a certain subjective category. Perceptual or subjective features will be treated as secondary in this article, but it should be interesting to consider them in a future work. Some emotional/textual melodic descriptors could be defined, as for example sad, happy, or charming. One interesting approach more related to the concept of **expressivity** is the one of the Department of Speech, Music and Hearing of the Royal Institute of Technology at Stockholm[1]. They define some context dependent rules characterizing a music performance. These rules affect duration, pitch, vibrato and intensity of notes, and can create crescendos, diminuendos, change the tempo or insert pauses between tones. They can be classified into three groups: differentiation rules, grouping rules and ensemble rules, and combinations of these rules define different emotional qualities (fear, anger, happiness, sadness, tenderness, and solemnity). We could also proceed in the opposite direction if we analyze the performance to get duration, pitch, vibrato and intensity features and we try to apply these rules to obtain some emotional descriptors of this performance.

3. Rhythm description

Considering the word 'rhythm' in a broad sense, one can wish to characterize the rhythm of a single note, of a pattern or of an entire musical movement, either monophonic or polyphonic. Moreover, rhythmic descriptors may be relevant in different types of applications, ranging e.g. from performance investigations to song comparisons. The literature concerning automatic extraction of rhythmic features from audio or symbolic data is indeed wide (see e.g. [12] for a review), let us give a few pointers here.

In the context of audio signal classifiers, [20] and [24] propose low-level rhythmic features characterizing in some way the rhythmic strength of the signal. The rate at which one taps his feet to the music (i.e. the Beat) is a more common rhythmic

[1] http://www.speech.kth.se/music/performance

concept that has been addressed widely from both computational and perceptual points of view (see e.g. [19]). Other periodicities that the perceptually most important one have also been proposed. For instance, among others, [3], [21] and [11] focus on the concept of "tatum" (smaller rhythmic pulse). Some propose to consider jointly the beat tracking and rhythm parsing issues, i.e. the variation of a musical piece's pace and the quantization of note timings (see [7] and [18]). Going to a higher level of abstraction, some (although much less numerous than those focusing on the Beat) seek Meter and rhythmic pattern recognition (see e.g. [4], [9]). Finally, many address the notion of expressive timing, and focus on features like tempo changes, deviations, or event-shifts (see e.g. [3]). An insightful discussion regarding the intertwining of expressive timing, metrical structure and pace of a musical signal is given in [14].

Current elements of the MPEG-7 standard that convey a rhythmic meaning are the following:

- The *Beat (mpeg7:BeatType)*
- The *Meter (mpeg7:MeterType)*
- The *note relative duration*

The *Beat* and *note relative duration* are embedded in the melody description. The *Meter*, also illustrated in [1] in the description of a melody, might be used as a descriptor for any audio segment.

Here, the *Beat* refers to the pulse indicated in the feature *Meter* (which doesn't necessarily corresponds to the notion of perceptually most prominent pulse). The B*eatType* is a series of numbers representing the quantized positions of the notes, with respect to the first note of the excerpt (the positions are expressed as integers, multiples of the measure divisor, the value of which is given in the denominator of the meter). The *note relative duration* is the "logarithmic ratio of the differential onsets for the notes in the series" [1]. The *MeterType* carries in its denominator a reference value for the expression of the beat series. The numerator serves, in conjunction to the denominator, to refer somehow to pre-determined templates of weighting of the events. (It is assumed that to a given meter corresponds a defined "strong-weak" structure for the events. For instance, in a 4/4 meter, the first and third beats are assumed to be strong, the second and the fourth weak. In a 3/4 meter, the first beat is assumed to be strong, and the two others weak.)

As for melodic attributes, the organization of rhythmic attributes in a coherent description scheme seems a necessary step for sharing descriptions between different systems and addressing different applications.

4. Instrument description

When dealing with multi-timbral polyphonies, or when working with sound samples, a need for instrument labeling of segments arises. In the first case, we assume the labeling to be done manually, as the current state of the art does not provided satisfactory tools for doing source separation in an automatic and reliable way; in the second case, a situation that arises in the context of the above mentioned Sound Palette, automatic modeling and labeling of instrument classes can be done with high success rates by means of building statistical models of the classes that exploit the information conveyed by the so-called low-level descriptors (e.g. spectral centroid, log-attack time, etc.). This model-based approach is also useful when the complexity of audio mixtures is low, as it is the case in "drum loops", another type of audio material that is managed by the Sound Palette.

A convenient way for achieving automatic labeling of monophonic samples or phrases in terms of instrumentation (i.e. piano, kick+hihat, etc.) consists of deriving, by means of using a training procedure with a large set of sound files, a model that characterizes the spectro-temporal distinctive features for each one of the classes to be learned. Models can be computed and stored using quite different techniques (see [13] for a comprehensive review). The appropriate association between models and classification schemes yields automatic labeling to sound files of the above-mentioned types (sound samples and rhythmic loops), and the existing labels can be organized into taxonomies. Taxonomies can be easily handled with MPEG-7 Description Schemes (specifically with the *ClassificationScheme*), but even though it provides a large set of Description Schemes for representing models, most of them have been included without considering audio modeling. The most audio-specific one, the *AudioModel*, relies specifically on Subspace Components Analysis [6], which is only one of the available techniques. Models describable with the Predictive Model Markup Language (PMML)[2], such as binary trees or neural networks would be an important addition to the standard.

An interesting differentiation to be commented here is that of instrument description versus timbre

[2] http://www.dmg.org

description. MPEG-7 provides descriptors and Description Schemes for timbre as a perceptual phenomenon. This set of D's and DS's are useful in the context of search by similarity in sound-samples databases, where perceptual distances are computed by means of weighted combinations of those descriptors (as for example, log attack time, harmonic spectral variation, spectral and temporal centroids, etc. Complementary to them, one could conceive the need for having D's and DS's suitable for performing categorical queries (in the same sound-samples databases), or for describing instrumentation if only in terms of culturally-biased instrument labels and taxonomies, as has been previously mentioned.

5. Conclusions

As we said in the introduction, we address the issue of musical description in a specific framework, that of the development of an application, a tool for content-based management, edition and transformation of sound samples, phrases and loops: the *Sound Palette*. We intended to cope with the description needs of this application, and therefore we have still left out issues of harmony, expressivity or emotional load descriptions, as they do not seem to be priorities for such a system.

6. Acknowledgments

The work reported in this article has been partially funded by the IST European project CUIDADO and the TIC project TABASCO.

References

[1] MPEG Working Documents, MPEG, 2001. <http://www.cselt.it/mpeg/working_documents.htm>

[2] MPEG-7 Schema and description examples, Final Draft International Standard (FDIS), 2002. <http://pmedia.i2.ibm.com:8000/mpeg7/schema/>

[3] Bilmes J., Timing is of the Essence: Perceptual and Computational Techniques for Representing, Learning, and Reproducing Expressive Timing in Percussive Rhythm, Master of Science Thesis, MIT, 1993.

[4] Brown J., Determination of the meter of musical scores by autocorrelation, Journal of the Acoustical Society of America, vol. 94, 1993.

[5] Casey, M.A., General sound classification and similarity in MPEG-7, Organized Sound, vol. 6, pp. 153-164, 2001.

[6] Casey, M.A. and Westner, A. Separation of mixed audio sources by independent subspace analysis, In Proceedings of the International Computer Music Conference, Berlin, Germany 2000.

[7] Cemgil A., Desain P. and Kappen B., Rhythm quantization for transcription, Computer Music Journal, vol. 24, 2000.

[8] Chi Lap, Y. and Kao, B., A study of musical features for melody databases, In Proceedings of the 10th International Conference and Workshop on Database and Expert Systems Applications, DEXA, 1999.

[9] Gasser M., Eck D. and Port R., Meter as mechanism: a neural network that learns metrical patterns, Connection Science, vol. 1, 1999.

[10] Gómez, E., Klapuri, A. and Meudic, B., Melody description and extraction in the context of music content processing, Journal of New Music Research, to appear, 2002.

[11] Gouyon F., Herrera P. and Cano P., Pulse-dependent analyses of percussive music, In Proceedings of 22nd AES International Conference on Virtual, Synthetic and Entertainment Audio, 2002.

[12] Gouyon F. and Meudic B., Towards rhythmic content processing of musical signals-Complementary approaches, Journal of New Music Research, to appear, 2002.

[13] Herrera, P., Amatriain, X., Batlle, E. and Serra, X., Towards instrument segmentation for music content description: A critical review of instrument classification techniques, In Proceedings of International Symposium on Music Information Retrieval, 2000.

[14] Honing H., From time to time: The representation of timing and tempo, Computer Music Journal, vol. 35, 2001.

[15] Kartomi, M., On Concepts and Classification of Musical Instruments, The University of Chicago Press, Chicago, 1990.

[16] Lindsay, A.T., Using Contour As a Mid-Level Representation of Melody, Master of Science in Media Arts and Sciences Thesis, Massachusetts Institute of Technology, 1996.

[17] Lindsay, A. T. and Herre, J., MPEG-7 and MPEG-7 Audio -An Overview, Journal of the Audio Engineering Society, vol. 49, pp. 589-594, 2001.

[18] Raphael C., Automated rhythm transcription, In Proceedings of the International Symposium on Music Information Retrieval, 2001.

[19] Scheirer E., Tempo and beat analysis of acoustic musical signals, Journal of the Acoustical Society of America, vol. 103, 1998.

[20] Scheirer E. and Slaney M., Construction and evaluation of a robust multifeature Speech/Music discriminator, In Proceedings of IEEE-ICASSP Conference, pp. 1331-1334, 1997.

[21] Seppänen J., Tatum grid analysis of musical signals, In Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2001.

[22] Toiviainen, P. and Eerola, T., A method for comparative analysis of folk music based on musical feature extraction and neural networks, In Proceedings of VII International Symposium on Systematic and Comparative Musicology and III International Conference on Cognitive Musicology, 2001.
<http://www.cc.jyu.fi/~ptee/publications.html>

[23] Towsey, M., Brown, A., Wright, S. and Diederich, J., Towards Melodic Extension Using Genetic Algorithms, Educational Technology & Society, vol. 4-2, 2001.
<http://ifets.ieee.org/periodical/vol_2_2001/towsey.html>

[24] Tzanetakis G., Essl G. and Cook P., Automatic Musical Genre Classification of Audio Signals, In Proceedings of International Symposium for Audio Information Retrieval, 2001.

[25] Vinet, H., Herrera, P. and Pachet, F., The CUIDADO project, In Proceedings of ISMIR Conference, Paris, October 2002.

Using the MPEG-7 Standard for the Description of Musical Content

EMILIA GÓMEZ, FABIEN GOUYON, PERFECTO HERRERA, XAVIER AMATRIAIN
Music Technology Group, Institut Universitari de l'Audiovisual
Universitat Pompeu Fabra
Passeig de Circumval·lació, 8, 08003 Barcelona
SPAIN
{emilia.gomez, fabien.gouyon, perfecto.herrera, xavier.amatriain}@iua.upf.es,
http://www.iua.upf.es/mtg/

Abstract : The aim of this paper is discussing possible ways of describing some music constructs in a dual context: that of a specific software application (a tool for content-based management and edition of samples and short audio phrases), and that of the current standard for multimedia content description (MPEG-7). Different musical layers, melodic, rhythmic and instrumental, are examined in terms of usable descriptors and description schemes. After discussing some MPEG-7 limitations regarding those specific layers (and given the needs of a specific application context), some proposals for overcoming them are presented.

Keywords: - music description, MPEG-7, music content analysis, melody, rhythm, instrument.

1. Introduction

Describing the musical content of audio files has been a pervasive goal in the computer music and music processing research communities. Though it has been frequently equated to the problem of "transcription", describing music content usually implies an applied context that has a "home" or "non-scholar" user in the final end of the chain. Therefore, it is usually the case that "conventional" music data types are not the perfect ones nor the final structures for storing content descriptions that are going to be managed by people with different backgrounds and interests (probably quite different from the purely musicological). This approach to music content description has also been that of the standardizing initiative carried out since 1998 by the ISO workforce that has been known as MPEG-7. MPEG-7 is a standard for multimedia content description that was officially approved in 2001 and is currently being further expanded. It provides descriptors and description schemes for different audio-related needs such as speech transcription, sound effects classification, and melodic or timbral-based retrieval.

The CUIDADO project (Content-based Unified Interfaces and Descriptors for Audio/music Databases available Online) is also committed with "applied" music description in the context of two different software prototypes, the so-called *Music Browser* and *Sound Palette*. The former is intended to be a tool for navigation in a collection of popular music files, whereas the latter is intended to be a tool for music creation based on short excerpts of audio (samples, music phrases, rhythm loops…). More details on these prototypes can be found in [10]. The development of the Sound Palette calls for a structured set of description schemes covering from signal-related or low-level descriptors up to user-centered or high-level descriptors. Given our previous experience and involvement in the MPEG-7 definition process ([6], [9]), we have developed a set of music description schemes according to the MPEG-7 Description Definition Language (hence DDL). Our goals have been manifold: first, coping with the description needs posed by a specific application (the Sound Palette); second, keeping compatibility with the standard; and third, evaluating the feasibility of these new Description Schemes (hence DSs) for being considered as possible enhancements to the current standard. We have then addressed very basic issues, some of them are yet present but underdeveloped in MPEG-7 (melody), some of them are practically absent (rhythm), and some of them seem to be present though using an exclusive procedure (instrument). Complex music description layers, as it is the case of harmony or expressivity descriptions, have been purposively left out from our discussion.

2. MPEG-7 musical description

2.1 Melody description

In this section, we will briefly review the work that have been done inside the MPEG-7 standard to represent melodic features of an audio signal. The MPEG-7 DSs are explained in [1,2,8].

MPEG-7 proposes two levels of melodic description: *MelodySequence* and *MelodyContour* values, plus some information about scale, meter, beat and key (see Figure 1). The melodic contour uses a 5-step contour (from –2 to +2) in which intervals are

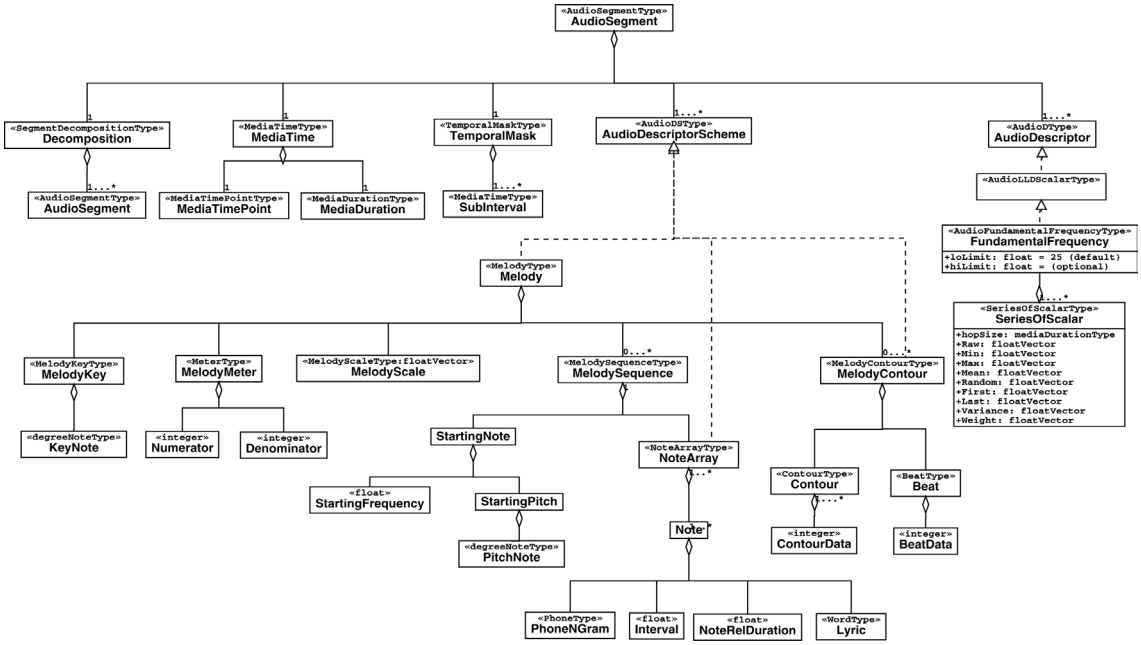


Figure 1: MPEG-7 Melody DS

quantized, and also represents basic rhythm information by storing the number of the nearest whole beat of each note, which can drastically increase the accuracy of matches to a query. However, this contour has been found to be inadequate for some applications, as melodies of very different nature can be represented by identical contours. One example is the case of having a descendant chromatic melody and a descendant diatonic one. Both of them have the same contour although their melodic features are very unlike.

For applications requiring greater descriptive precision or reconstruction of a given melody, the *mpeg7:Melody* DS supports an expanded descriptor set and higher precision of interval encoding, the *mpeg7:MelodySequence*. Rather than quantizing to one of five levels, the precise pitch interval (with cent or greater precision) between notes is kept. Timing information is stored in a more precise manner by encoding the relative duration of notes defined as the logarithm of the ratio between the differential onsets. In addition to these core descriptors, MPEG-7 define a series of optional support descriptors such as *lyrics*, *key*, *meter*, and *starting note*, to be used as desired for an application.

2.2 Rhythm description

Current elements of the MPEG-7 standard that convey a rhythmic meaning are the following:

- The *Beat (BeatType)*
- The *Meter (MeterType)*
- The *note relative duration*

The *Beat* and *note relative duration* are embedded in the melody description. The *Meter*, also illustrated in [1] in the description of a melody, might be used as a descriptor for any audio segment.

Here, the *Beat* refers to the pulse indicated in the feature *Meter* (which doesn't necessarily corresponds to the notion of perceptually most prominent pulse). The B*eatType* is a series of numbers representing the quantized positions of the notes, with respect to the first note of the excerpt (the positions are expressed as integers, multiples of the measure divisor, the value of which is given in the denominator of the meter). The *note relative duration* is the "logarithmic ratio of the differential onsets for the notes in the series" [1]. The *MeterType* carries in its denominator a reference value for the expression of the beat series. The numerator serves, in conjunction to the denominator, to refer somehow to pre-determined templates of weighting of the events. (It is assumed that to a given meter corresponds a defined "strong-weak" structure for the events. For instance, in a 4/4 meter, the first and third beats are assumed to be strong, the second and the fourth weak. In a 3/4 meter, the first beat is assumed to be strong, and the two others weak.)

2.3 Instrument description

The MPEG-7 *ClassificationScheme* defines a scheme for classifying a subject area with a set of terms organized into a hierarchy. This feature can be used, for example, for defining taxonomies of instruments. A term in a classification scheme is referenced in a description with the *TermUse* datatype. A term

represents one well-defined concept in the domain covered by the classification scheme. A term has an *identifier* that uniquely identifies it, a *name* that may be displayed or used as a search term in a target database, and a *definition* that describes the meaning of the term. Terms can be put in relationship with a *TermRelation* descriptor. It represents a relation between two terms in a classification scheme, such as synonymy, preferred term, broader-narrower term, and related term. When terms are organized this way, they form a classification hierarchy. This way, not only content providers but also individual users can develop their own classification hierarchies.

An interesting differentiation to be commented here is that of instrument description versus timbre description. The current standard provides descriptors and Description Schemes for timbre as a perceptual phenomenon. This set of Ds and DSs are useful in the context of search by similarity in sound-samples databases. Complementary to them, one could conceive the need for having Ds and DSs suitable for performing categorical queries (in the same sound-samples databases), or for describing instrumentation if only in terms of culturally-biased instrument labels and taxonomies.

2.3.1 Classification Schemes for instruments

A generic classification scheme for instruments along the popular Hornbostel-Sachs-Galpin taxonomy (cited by [7]), could have the schematic expression depicted below. More examples using the ClassificationSchemed DS can be found in [3].

```
<ClassificationScheme term="0" scheme="Horbonstel-Sachs Instrument Taxonomy">
<Label>"HSIT"</Label>
<ClassificationSchemeRef scheme="Cordohpones" />
<ClassificationSchemeRef scheme="Idiophones" />
<ClassificationSchemeRef scheme="Membranonphones" />
<ClassificationSchemeRef scheme="Aerophones" />
<ClassificationSchemeRef scheme="Electrophones" />
</ClassificationScheme>

<ClassificationScheme term="1" scheme="Cordophones">
<Label>"Cordophones"</Label>
<ClassificationSchemeRef scheme="Bowed" />
<ClassificationSchemeRef scheme="Plucked" />
<ClassificationSchemeRef scheme="Struck" />
</ClassificationScheme>

<ClassificationScheme term="2" scheme="Idiophones">
<Label>"Idiophones"</Label>
<ClassificationSchemeRef scheme="Struck" />
<ClassificationSchemeRef scheme="Plucked" />
<ClassificationSchemeRef scheme="Frictioned" />
<ClassificationSchemeRef scheme="Shakened" />
</ClassificationScheme>

<ClassificationScheme term="3" scheme="Membranophones">
…
</ClassificationScheme>
```

3. Use of the standard

We have reviewed in last section the description schemes that the MPEG-7 provides for music description. In this section, we will see how we have used and adapted this description scheme in our specific application context.

3.1 On MPEG-7 descriptions

Regarding the *mpeg7:Note* representation, some important signal-related features like e.g. intensity, intra-note segments, articulation or vibrato are needed by the application. It should be noted that some of these features are already coded by the MIDI representation. This *Note* type, in the *Melody* DS, includes only note relative duration information, silences are not taken into account. Nevertheless, it would sometimes be necessary to know the exact note boundaries. Also, the note is always defined as a part of a descriptor scheme (the *noteArray*) in a context of a *Melody*. One could object that it could be defined as a segment, which, in turn, would have its own descriptors.

Regarding melody description, MPEG-7 also includes some optional descriptors related to key, scale and meter. We need to include in the melodic representation some descriptors that are computed using the pitch and duration sequences. These descriptors will be used for retrieval and transformation purposes.

Regarding rhythmic representation, some comments could be made regarding MPEG-7. First, there is no direct information regarding the tempo, nor to the speed at which pass pulses. Second, in the *BeatType*, when quantizing an event time occurrence, there is a rounding towards $-\infty$, thus in the case where an event is slightly before the beat (as it can happen in expressive performance) it is attributed to the preceding beat. Third, this representation cannot serve for exploring fine deviations from the structure; furthermore as events are characterized by beat values, it is not accurate enough to represent already-quantized music where sub-multiples are commonly found. Finally, it is extremely sensitive to the determination of the meter, which is still a difficult task for the state-of-the-art in rhythm computational models.

Regarding instrument description capabilities, there is no problem for a content provider to offer exhaustive taxonomies of sounds. It could also be possible that a user would define her/his own devised taxonomies. But for getting some type of automatic labelling of samples or simple mixtures, there is a need for DSs capable of storing data defining class models. Fortunately, MPEG-7 provides description

schemes for storing very different types of models: discrete or continuous probabilistic models, cluster models, or finite state models, to name a few. The problem arises in the connection between these generic-purpose tools and the audio part: it is assumed that the only way of modeling sound classes is through a very specific technique that computes a low-dimensional representation of the spectrum, the so-called spectrum basis [4] which de-correlates the information that is present in the spectrum.

3.2 Extensions

3.2.1 Audio segment derivation

The first idea would be to derive two different types from *mpeg7:AudioSegmentType***.** Each of the segments would cover a different scope of description and would logically account for different DSs.

- **NoteSegment**: Segment representing a note. The note has an associated DS, accounting for melodic, rhythmic and instrument descriptors, as well as the low-level descriptors (LLDs) inherited from *mpeg7:AudioSegmentType*.
- **MusicSegment:** Segment representing an audio excerpt, either monophonic or polyphonic. This segment will have its associated Ds and DSs and could be decomposed in other *MusicSegments* (for example, a polyphonic segment could be decomposed in a collection of monophonic segments, as illustrated in Figure 2) and in *NoteSegments*, by means of two fields whose types derive from *mpeg7:AudioSegmentTemporalDecompositionType* (see Figure 3). The *MusicSegment* has an associated DS differing from that of the note.

The note has different melodic, rhythmic and instrumental features than a musical phrase or general audio excerpt, and there are some attributes that do not have any sense associated to a note (for example, *mpeg7:Melody*). But a Note *is* an *AudioSegment* with some associated descriptors.

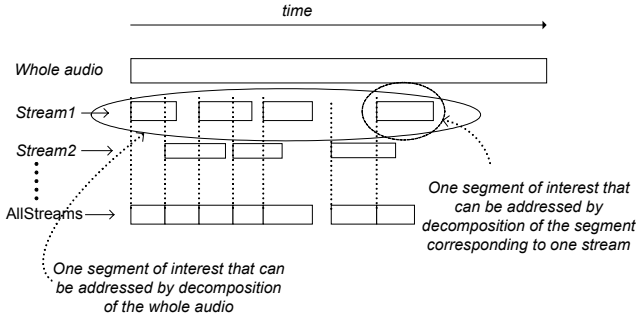


Figure 2: Audio segment decomposition

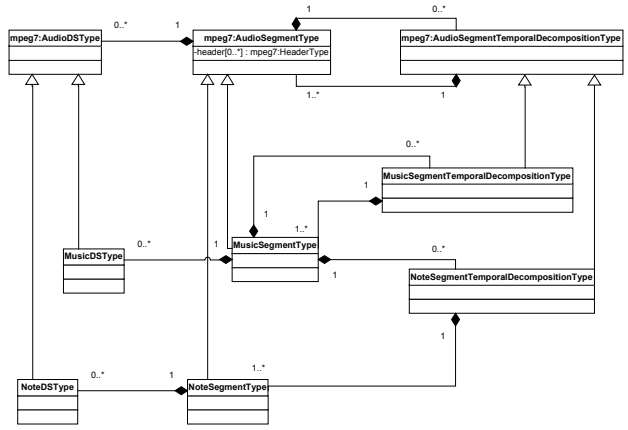


Figure 3: Class diagram of MPEG-7 *AudioSegment* and *AudioSegmentTemporalDecomposition* derivations

3.2.2 Definition of Description Schemes:

Description Scheme associated to *NoteSegmentType*:

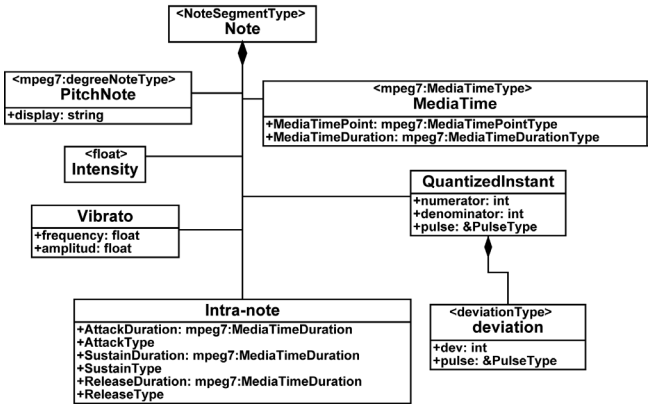


Figure 4: Note DS

- The exact temporal location of a note is described by the ***mpeg7:MediaTime*** attribute inherited from the *mpeg7:AudioSegment*.

- **PitchNote**: as defined in *mpeg7: degreeNoteType. MIDI-Note* could also be used as pitch descriptor, making a direct mapping between melodic description and MIDI representation.

- As well, some symbolic time representation (quarter of note, etc) would be needed if we want to work with MIDI files.

- **Intensity**: floating value indicating the intensity of the note. It is necessary when analyzing phrasing and expressivity (crescendo, diminuendo, etc) in a melodic phrase, although it could be represented by using the *mpeg7:AudioPower* low-level descriptor.

- **Vibrato**: also important when trying to characterize how the musical phrase has been performed, it is defined by the vibrato frequency and amplitude.

- **Intra-note** segments: as explained in last section, it is important for some applications to have

information about articulation, as attack and release duration. It can be represented by some descriptors indicating the duration and type of the intra-note segments. In addition to intra-note segment durations, some more descriptors could be defined to characterize articulation.

- **Quantized instant**: If one wishes to reach a high level of precision in a timing description, then the decomposition of the music segment into note segments is of interest. In addition to the handling of precise onsets and offsets of musical events, it permits to describe them in terms of position with respect to the metrical grids. In our quantized instant proposal, given a pulse reference that might be the Beat, the Tatum, etc., a note is attributed a rational number representing the number of pulses separating it from the previous one. This type can be seen as a generalization of the *mpeg7:BeatType*, improvements being the following:

- One can choose the level of quantization (the reference pulse does not have to be the time signature denominator as in the *BeatType*).
- Even when a reference pulse is set, one can account for (i.e. represent without rounding) durations that don't rely on this pulse (as in the case of e.g. triplets in a quarter-note-based pattern). This feature is provided by the fact that the quantized instants are rational numbers and not integers.
- The rounding (quantization) is done towards the closest beat (not towards -∞).

In addition, the **deviation** of a note from its closest pulse can be stored. The deviation is expressed in percentage of a reference pulse, from –50 to +50. (Here, the reference pulse can be different than that used for quantizing, one might want to e.g. quantize at the Beat level and express deviations with respect to the Tatum.) This may useful for analyzing phrasing and expressivity.

Description Scheme associated to *MusicSegmentType*:

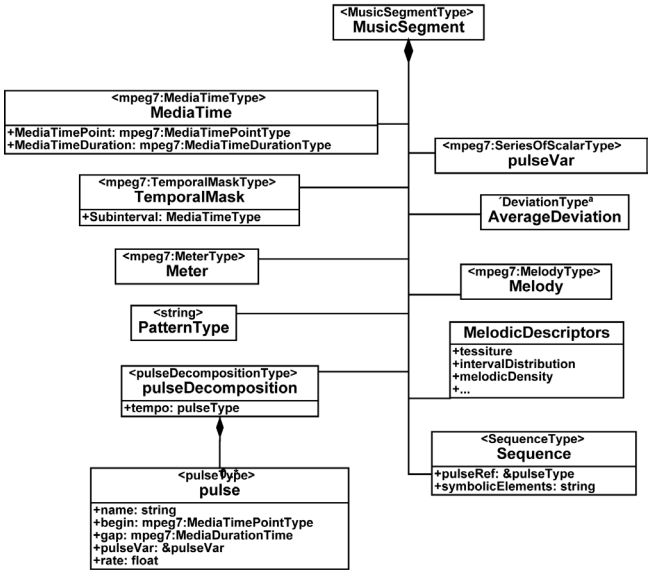


Figure 5: Music DS

- The exact temporal location of the music segment is also described by the ***mpeg7:MediaTime*** attribute derived from the *mpeg7:AudioSegment*.

- The ***mpeg7:Melody*** DS is used to describe contour and melody sequence attributes of the audio excerpt.

- **Melodic descriptors**: se need to incorporate some "*unary"* descriptors derived from the pitch sequence information, modeling some aspects as tessitura, melodic density or interval distribution. These features provide a way to characterize a melody without explicitly giving the pitch sequence and should be included in the *MusicSegment* DS.

- The **Meter** type is the same as MPEG-7's.

- Considering that several **pulses**, or metrical levels, coexist in a musical piece is ubiquitous in the literature. In this respect, our description of a music segment accounts for a decomposition in pulses, each pulse has a name, a beginning time index, a gap value and a rate (which is logically proportional to the inverse of the gap; some might prefer to apprehend a pulse in terms of occurrences per minute, some others in terms of e.g. milliseconds per occurrence). It is clear that in much music, pulses are not exactly regular, here resides some of the beauty of musical performance; therefore, the regular grid defined by the previous 'beginning' and 'gap' can be warped according to a time function representing tempo variations, the '**pulseVar**'. This function is stored in the music segment DS, a pulse can hold a reference to the *pulseVar*. Among the hierarchy of pulses, no pulse is by any mean as important as the tempo. In addition, the reference pulse for writing down the rhythm often coincides with the perceptual pulse. Therefore, it seemed important to provide a special handling of the **tempo**: the **pulse decomposition** type holds a mandatory pulse named tempo, in

addition to it, several other pulses can optionally be defined. Additional pulses can be e.g., the Tatum, the Downbeat, etc.

- **Sequence type**: A simple series of letters can be added to the description of a music segment. This permits to describe a signal in terms of recurrences of events, with respect to the melodic, rhythmic or instrumental structure that organizes musical signals. For instance, one may wish to categorize the succession of Tatums in terms of timbres –this would look e.g. like the string *'abccacccabcd'*–, and then look for patterns. Categorize segments of the audio "chopped up" with respect to the Beat grid might also reveal interesting properties of the signal. One might want to describe a signal in the context of several pulses; therefore several sequences can be instantiated.

- Rather than restricting one's time precision to that of a pulse grid, one might wish to categorize musical signals in terms of accurate time indexes of occurrences of particular instruments (e.g. the ubiquitous bass drums and snares). This, in order to post-process these series of occurrences so as to yield rhythmic descriptors. Here, the **decomposition** of a music segment in its constituent instrument streams is needed (see Figure 2). For instance, a music segment can be attributed to the occurrences of the snare, another one to those of the bass-drum; timing indexes lie in the *mpeg7:TemporalMask*, inherited from the *mpeg7:AudioSegment*, that permits to describe a single music segment as a collection of sub-regions disconnected and non-overlapping in time.

4. Conclusions

As mentioned above, we address the issue of musical description in a specific framework, that of the development of an application, a tool for content-based management, edition and transformation of sound samples, phrases and loops: the Sound Palette. We intended to cope with the description needs of this application, and therefore we have still left out issues of harmony, expressivity or emotional load descriptions, as they do not seem to be priorities for such a system. We believe that adding higher-level descriptors to the current Ds and DSs (e.g. presence of rubato, swing, groove, mood, etc.), needs a solid grounding and testing on the existing descriptors, defining interdependency rules that currently cannot be easily devised. New descriptors and description schemes have been proposed keeping also in mind the need for compatibility with the current MPEG-7 standard; they should be considered as the beginning of an open discussion regarding what we consider as the current shortcomings of the standard.

5. Acknowledgments

The work reported in this article has been partially funded by the IST European project CUIDADO and the TIC project TABASCO.

References

[1] MPEG Working Documents, MPEG, 2001. <http://www.cselt.it/mpeg/working_documents.htm>

[2] MPEG-7 Schema and description examples, Final Draft International Standard (FDIS), 2002. <http://pmedia.i2.ibm.com:8000/mpeg7/schema/>

[3] Casey, M.A., General sound classification and similarity in MPEG-7, Organized Sound, vol. 6, pp. 153-164, 2001.

[4] Casey, M.A., Sound Classification and Similarity, In Manjunath, BS., Salembier, P. and Sikora ,T. (Eds.), Introduction to MPEG-7: Multimedia Content Description Language, pp. 153-164, 2002.

[5] Herrera, P., Amatriain, X., Batlle, E. and Serra, X., Towards instrument segmentation for music content description: A critical review of instrument classification techniques, In Proceedings of International Symposium on Music Information Retrieval, 2000.

[6] Herrera, P., Serra, X. and Peeters, G., Audio descriptors and descriptor schemes in the context of MPEG-7, In Proceedings of International Computer Music Conference, 1999.

[7] Kartomi, M., On Concepts and Classification of Musical Instruments, The University of Chicago Press, Chicago, 1990.

[8] Lindsay, A.T. and Herre, J., MPEG-7 and MPEG-7 Audio - An Overview, Journal of the Audio Engineering Society, vol. 49, pp. 589-594, 2001.

[9] Peeters, G., McAdams, S. and Herrera, P., Instrument sound description in the context of MPEG-7, In Proceedings of the International Computer Music Conference, 2000.

[10] Vinet, H., Herrera, P. and Pachet, F., The CUIDADO project. In Proc. of ISMIR Conference, Paris, October 2002.