Chroma-based Predominant Melody and Bass Line Extraction from Music Audio Signals

Justin Jonathan Salamon

Master Thesis submitted in partial fulfillment of the requirements for the degree:
Master in Cognitive Systems and Interactive Media

Supervisor: Dr. Emilia Gómez

Department of Information and Communication Technologies
Universitat Pompeu Fabra
Barcelona, Spain

September 2008

Abstract

In this dissertation we present the research work we have carried out on melody and bass line extraction from music audio signals using chroma features. First an introduction to the task at hand is given and important relevant concepts are defined. Next, the scientific background to our work is provided, including results obtained by state of the art melody and bass line extraction systems. We then present a new approach to melody and bass line extraction based on chroma features, making use of the Harmonic Pitch Class Profile (HPCP) [Gómez 06a]. Based on our proposed approach, several peak tracking algorithms for selecting the melody (or bass line) pitch classes are presented. Next, the evaluation methodology and music collections and metrics used for evaluation are discussed, followed by the evaluation results.

The results show that as a salience function our proposed HPCP based approach has comparable performance to that of other state of the art systems, in some cases outperforming them. The tracking procedures suggested are shown to require further work in order to achieve a significant improvement in the results. We present some initial experiments on similarity computation, the results of which are very encouraging, suggesting that the extracted representations could be useful in the context of similarity based applications.

The dissertation is concluded with an overview of the work done and goals achieved, issues which require further work and some proposals for future investigation.

Contents

List of Figures

Acknowledgements

First and foremost, I would like to thank my tutor, Emilia Gómez, for her guidance and support throughout the year. My deep thanks and gratitude are owed to many more. I have attempted to list them here: Xavier Serra for his help and support, Perfecto Herrera and the members of Aula 316 for the feedback and brain-storming, Joan Serrà for his advice and suggestions, Anssi Klapuri, Matti Ryynänen and Masataka Goto for their wilful correspondence. Thanks to Narcís Parés and Paul Verschure for the hard work in creating the CSIM program of which I was part this year, and to all my colleagues from both the CSIM and TICMA Masters programs. A special thanks goes to my fellow office co-workers Marcelo, Gerard, Elena and Rena for their ideas, advice, and for tolerating the odd victory dance. Thanks to the PhD students and all other people who have taken apart in shaping my year here at the MTG.

I would also like to thank Martin Rohrmeier, Lawrence Paulson and the members of the Computer Laboratory and Centre for Music and Science (CMS) at Cambridge for setting me in the right path quite some time ago now. A warm thank you to Vassilis for the jams, Yotam for the moral support, Urinaca for the super-sake-therapy and all the people who have become my close friends this year. Thanks mum. Thanks aba. Thanks Guz.

Without a shadow of a doubt, there will be many people I have left out of the above list unintentionally, who deserve my gratitude. Sorry, and thank you!

Justin.

1 Introduction

1.1 Motivation

With the prevalence of digital media, we have seen exponential growth in the distribution and consumption of digital audio. With musical collections reaching vast numbers of songs, we now require novel ways of describing, indexing, searching and interacting with music. Over the past years the Music Information Retrieval (MIR) research community has made significant advances in our ability to describe music through direct analysis of the audio signal. Being able to extract various features of the music from the audio signal, or *descriptors*, is key to the creation of automatic content description tools which would be highly useful for music analysis, retrieval and recommendation.

Such descriptors are often classified as either low, mid or high-level descriptors, depending on the degree of abstraction of the descriptor [Lesaffre 05]. Features denoted as low-level are usually those which are closely related to the raw audio signal, computed from the signal in either a direct or derived way. Such descriptors will usually not be very musically meaningful for end-users, but are of great value for computational systems. Examples of low-level descriptors are acoustic energy, Mel Frequency Cepstral Coefficients (MFCCs) [Rabiner 93] and the Harmonic Pitch Class Profile (HPCP) [Gómez 06a] which we present in detail in the next chapters.

Mid to high-level descriptors are those we would consider more musically meaningful to a human listener rather than a machine. Examples of such descriptors are the beat, chords, melody, and even more abstract concepts such as mood, emotions, or the expectations induced in a human listener by a piece of music. When discussing music description, Serra proposes the "Music Information Plane" [Serra 05], a plane where the relevant information about music is placed in the context of two dimensions; one is the abstraction level of the descriptors (from physical to knowledge levels) and the other includes the different

media (audio, text, image). A visualisation of this plane is provided in figure 1.1, taken from [Serra 05] with the permission of the author.

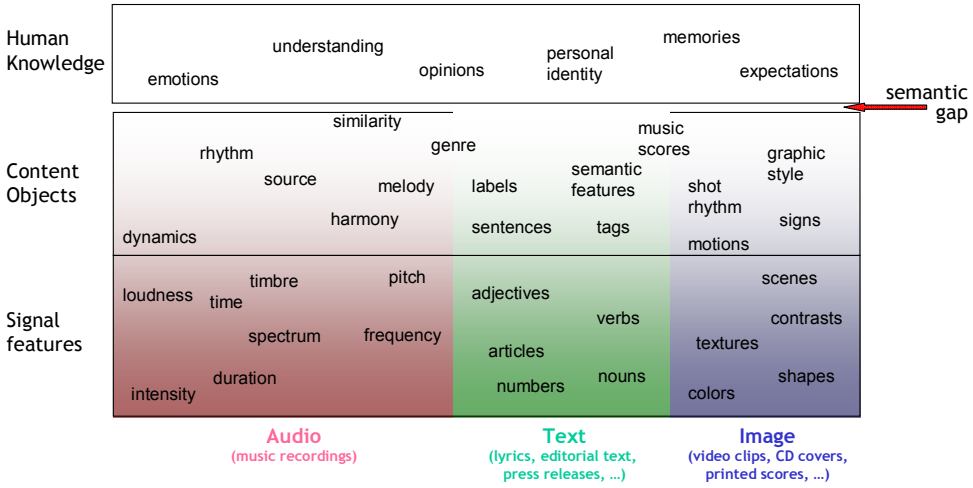

Figure 1.1: The "Music Information Plane".

One of the important musical facets presented in figure 1.1 is the melody. Melodic description has many potential applications and deserves proper attention. One such application is Query by Humming (QBH) - a content based search system allowing the user to search for music by singing (or humming) the tune. Currently, the large majority of existing QBH systems use databases of MIDI files (or similar symbolic representations) which need to be manually prepared. For such search systems to be truly functional on a large scale basis, an automatic method of extracting the melody is essential.

In addition to music searching, melody and bass line extraction would facilitate many other applications. Clustering variations of the same piece based on the melody or the chord progression (related to the bass) could assist song cover identification. Musicological research would benefit from the ability to group and analyse common melodic and harmonic primitives. An extracted melody could be used as a reduced representation (thumbnail) of a song in music applications, or on limited devices such as mobile phones. A melody and bass line extraction system could be used as a core component in other music computation tasks such as score following, computer participation in live human performances, or a music transcription system. For many years music transcription has been performed manually by musically trained people, a very time consuming task and practically infeasible for very large music collections. Extracting a mid-level symbolic representation of the melody and bass line would assist in automating this process.

In section 1.4 we examine more closely some of the potential application areas of melody and bass line extraction.

Going back to figure 1.1, we see that Serra identifies a "semantic gap" – the discrepancy between what can be recognised in music signals by current state-of-the-art methods and what human listeners associate with music [Serra 07c]. As noted by the authors, this "gap" is the main obstacle on the way towards truly intelligent and useful musical companions. Whilst not by any means a solution to the problem of the semantic gap, in developing tools for melody and bass line extraction (themselves perceptual concepts rooted in music cognition) we believe our work forms part of the effort in bridging this gap. Firstly, it provides descriptors currently missing from the landscape of the Music Information Plane, and one that could be used in attempt to reach higher-levels of abstraction still. In addition, and not less important, research into melody and bass line extraction also help us identify the inherent limitations of the bottom up approach and the places where a more interdisciplinary and wider notion of musical understanding is required.

1.2 Definitions

Before elaborating further on the task at hand, it is important that we clearly define what it is exactly that we aim to achieve. Namely, what do we mean by a "mid-level symbolic representation of the melody and bass line". In order to clarify this, we must have clear definitions of the terms "melody" and "bass line", of the source from which they are to be "extracted", and of the form the "extracted" result will take.

1.2.1 Musical Texture

The characteristics of the musical source heavily affect the nature of the intended task and the relevant approaches. Of these characteristics, one of the most elementary ones is the *musical texture* of the source.

Musical Texture is traditionally divided into three[1] classes [Copeland 57]:

- Monophonic – music with a single, unaccompanied line.
- Homophonic – a principal melodic line and a chordal accompaniment.
- Polyphonic – music consisting of two or more melodic lines.

[1] One might argue that it is four rather than three, if we include *hetrophony*, more common to Native American, Middle Eastern, and South African music.

Within the MIR community this classification is commonly narrowed down to two classes - monophonic and polyphonic (with homophonic included in polyphonic), as they require significantly differently approaches for information retrieval, and quite often tasks related to polyphonic music are considerably harder[2] than their polyphonic counterpart.

In this research we will focus on western popular music, which is usually either homophonic or polyphonic, and shall henceforth be referred to simply as polyphonic, in contrast to monophonic music.

1.2.2 Melody

Though seemingly intuitive, it is not a straight forward task to define what we perceive as the melody of a musical piece. The term "melody" is a musicological concept based on the judgement of human listeners [Poliner 07], and we can expect to find different definitions for the melody in different contexts. In [Ryynänen 08], Ryynänen and Klapuri note:

"The melody of a piece is an organized sequence of consecutive notes and rests, usually performed by a lead singer or by a solo instrument. More informally, the melody is the part one often hums along when listening to a music piece."

Typke notes that there are certain characteristics which are important to the perception of a melody [Typke 07], namely "melodic motion" (characterized by successive pitch intervals) and contour. The authors of [Poliner 07] define a melody in the following way:

"...the melody is the single (monophonic) pitch sequence that a listener might reproduce if asked to whistle or hum a piece of polyphonic music, and that a listener would recognise as being the 'essence' of that music when heard in comparison".

The above definition is adequate for the purpose of our work, and we adopt it as our definition of a melody for future reference.

[2] the challenges posed by polyphonic as opposed to monophonic music are explained section 2.2 of chapter 2.

1.2.3 Bass Line

The above discussion can be similarly applied to the definition of the bass line (perhaps more so in counterpoint music), however in the context of our work on popular music the bass line can be relatively clearly defined. Adopting the definition given in [Ryynänen 08], "the bass line consists of notes in a lower pitch register [than the melody] and is usually played with a bass guitar, a double bass, or a bass synthesizer".

The bass line (played by such instruments) will most often have the following characteristics:

- Low pitch – the bass instrument will usually be playing in the lowest register out of the ensemble, with fundamental frequencies reaching all the way down to the limit of perceptually audible sound (which is around 20Hz)[3].
- Limited range – the bass line will usually be played within a limited pitch range, a result of both the physical characteristics of the instrument, and musicological reasoning (emphasising the harmony and avoiding clashes with higher-pitched instruments).
- Slower note rate – quite often the bass line in western music will be played at a slower rate relative to the other parts, setting the rhythmic "feel" of the piece and following the harmony which changes at a slower rate than the notes of the melody (by-and-large).

1.2.4 Salience

Throughout this work, we shall be discussing the extraction of the *predominant* melody and bass line. As before, this requires that we define what we consider to be predominant, or *salient*. When discussing salience in music, we are referring to a musical part which "sticks out" more than the others, one which attracts our attention. Indeed, one might describe the melody in terms of salience – the melody is the most salient part of a piece of music, the part we listen to the most in a piece of music and hence most likely to identify it the most with the piece. More formally, we can define salience as the significance in perceptual terms of an element of music [Byrd 02].

The notion of salience manifests itself in melody extraction systems in the form of a *Salience Function*. That is, a function which given a candidate fundamental

[3]The biggest pipe in the King's College pipe organ in Cambridge is said to sound at 18Hz, whilst some of the biggest organs in the world go down to 8Hz!

frequency (or harmonic pitch class profile, as shall be explained later) for a specific frame, returns a value indicating its salience with relation to all other possible candidates in that frame. Such salience functions stand at the core of most melody extraction systems, as shown in section 2.2.

1.2.5 Extraction Versus Transcription

Another important distinction is between extraction and transcription. Music transcription refers to the task of taking audible music and writing it down using a formal notation, most commonly musical notation or MIDI[4] as a digital counterpart. This requires the segmentation of the melody (or bass line) into notes, quantisation of the pitch into semitones and possibly the generation of additional notations such as dynamics for example. The problem of transcription is of course an important one, and has been addressed more recently in the work of Ryynänen and Klapuri [Ryynänen 06, Ryynänen 07, Ryynänen 08]. However, we must note that such a formal transcription is not always desirable. An actual musical performance is quite often (if not always) different from the formal musical notation – the performer may vary the timing, sings with vibrato (which causes a pitch modulation), or use other ornamentation not in the original score. Furthermore, we might not be interested in the final score of the piece, for example when wishing to compare a sung query to a song in a database – a full score is not necessary for the computation of a similarity score. For these reasons, this work will focus on the extraction of a "symbolic mid-level representation". Symbolic implies a textual/numeric representation other than the audio samples. Mid-level means we are not performing musical transcription, but rather are aiming for a representation which is sufficient for the purposes of computing musical similarity. The exact format of this mid-level representation is detailed in chapter 3.

1.3 Musical Similarity

In the previous section we argued that the extracted representation should suffice for computing musical similarity. It is thus important that we state how we define melodic and bass line similarity. Similarly to the problem of how do we define a melody, the task of determining what makes one melody more similar to another is grounded in music cognition and musicological research.

[4]Musical Instrument Digital Interface [MIDI].

1.3.1 Melodic and Bass Line Similarity

We have already noted Typke's mention of the importance of melodic motion and contour. From musicological studies [Selfridge-Field 98] we can assert that the contour of a melody is indeed significant in its identification – small changes to the timing or slight shifting of individual notes will not change the overall identity of the melody. A melody's contour is transposition and tempo invariant – the melody maintains its identity when played either slower or faster, higher or lower. The importance of contour in melodic similarity has been acknowledged in musicological research for many years now [Fujitani 71, Dowling 78]. Thus, we propose a simple approach to melodic similarity – the similarity of two melodies is determined by a distance metric computed on the frequency (or in our case, harmonic pitch class) contour of the compared items. Justification for the use of this contour representation discussed in section 2.3.1. The selection of an appropriate distance metric is also an important matter and can influence the similarity judgement [Typke 07]. In section 5.4 we discuss the distance metric used in our work for the purpose of similarity evaluation.

1.4 Application Contexts

1.4.1 Query By Humming

As mentioned in section 1.1, one application area of interest is Query by Example (QBE) and Query by Humming (QBH) systems. QBH systems allow the user to search for songs by singing or humming a query, which is then matched against a database of songs. In light of the definitions of melody presented in section 1.2.2, we can expect the vast majority of queries to be a segment of a song's melody.

Notable early work on QBE is David Huron's Themefinder [ThemeFinder] which allows searching through a symbolic database by specifying a pitch, interval or contour sequence as a query. Much research has gone into QBE and QBH since, however the majority still use databases which are built from Midi files (or some format of symbolic data otherwise). Using manual annotations (in the form of Midi files) does not scale well, and melody and bass line extraction can assist in constructing large databases. The general architecture of a query-by-humming system (using a symbolic database) is presented in figure 1.2.

Still, such existing systems are very much of interest to us as they have resulted in the development of indexing algorithms, efficient search algorithms, and research into symbolic melodic similarity. Of note is the Vocal Search system developed by [Pardo 04]. The system records a sung query and transcribes it to

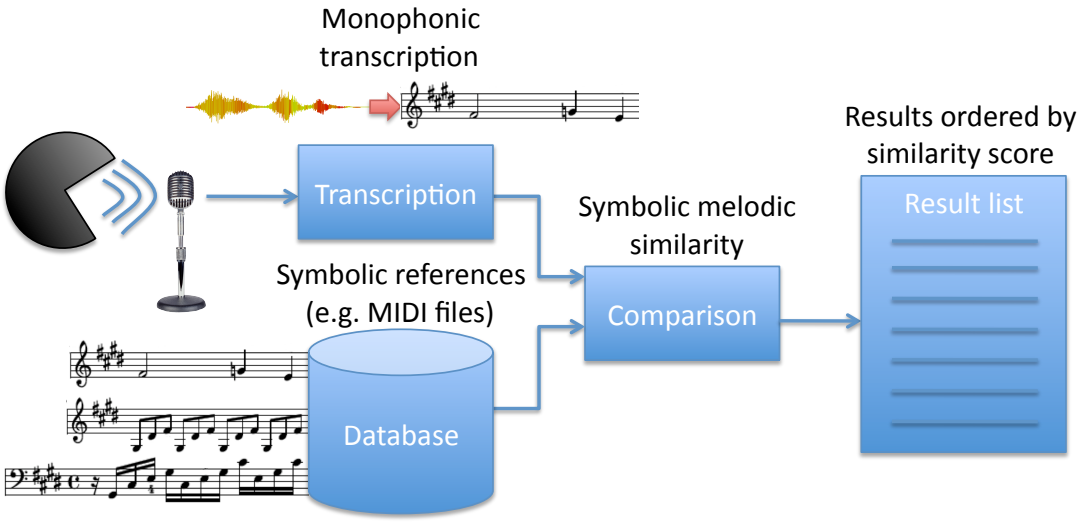

Figure 1.2: Query-by-humming general system architecture.

pitch and rhythm values (using a monophonic pitch tracker). Two approaches were compared for matching a query against themes in a database, one based on string matching techniques and the other on Hidden Markov Models (HMM). Their work presents important ideas on how a good representation of the query and targets can support robust matching (i.e. one which is key and tempo invariant) using local alignment algorithms. Further work on QBH evaluation and a comparative analysis of QBH systems are detailed in [Pardo 03, Dannenberg 07]. In this evaluation, string matching techniques which use a note interval symbolic representation were shown to outperform approaches based on N-grams and HMM, however Mean Reciprocal Rank (MRR) scores for the evaluation were still fairly low (the highest reported value was 0.282), indicating that there is much room for improvement.

1.4.2 Audio Cover Song Identification

Another important task is that of audio cover song identification. In addition to finding a predetermined target, users might be interested in discovering new music, for example through finding unknown cover versions of known songs. In [Yang 01], the author classifies five types of "similar" music pairs, with increasing levels of difficulty (in recognising that they are indeed a pair):

- Type I: Identical digital copy
- Type II: Same analog source, different digital copies, possibly with noise

- Type III: Same instrumental performance, different vocal components
- Type IV: Same score, different performances (possibly at a different tempo)
- Type V: Same underlying melody, different otherwise, with possible transposition.

More elaborate classifications of cover song types can be found in [Gómez 06a] and [Serrà 07b]. Examining types IV and V of the above classification, we note that in the case of type IV, we expect the harmony and melody to be the same for both versions of the song. It is a clear case of where both the bass line (which will normally be strongly related to the harmony) and melody can be useful for cover song identification. In type V, even the harmony of the song is altered, and thus finding similarity between the melodies of both versions might be the only way to recover such pairs[5].

Recently, good performance in audio cover song identification was demonstrated in [Serrà 07a] using tonal descriptors. As such, we could also utilise melody and bass line extraction as a powerful additional tool to enhance performance of existing systems, as opposed to using them as the base for a cover song identification system on their own.

1.4.3 Active Listening

In addition to the specific applications mentioned above, melody and bass line extraction would facilitate a range of applications which could enhance users' interaction with music through novel ways of exploring and browsing large collections, in which the user takes an *active* part in searching and finding new music. Melody and bass line extraction could stand at the core of novel interfaces for music interaction, for example allowing the user to skip or repeat parts of a song by browsing through the melody, or jump from one song to the next by targeting songs with similar bass riffs in selected locations. The enhancement of the listening experience through interaction, *Active Music Listening* [Goto 07], is an area of increasing importance, and with the growing capabilities of portable media devices, we only expect it to grow further.

1.5 Goals and Organisation of the Thesis

The main goals of the thesis are the following:

[5] A good example of a cover version in which all but the melody is changed is modern jazz group The Bad Plus' version of the ABBA song "Knowing Me, Knowing You".

- Provide scientific background and a summary of the literature in the field of melody and bass line extraction.
- Develop a new method for melody and bass line extraction based on chroma features.
- Compile and prepare music collections for evaluation and an evaluation methodology.
- Evaluate our method alongside state of the art systems.
- Discuss our results, conclude the work carried out and discuss future work.

In chapter 2 we present the scientific background underlying the work carried out in this research. In section 2.2 we start by explaining the challenges involved in melody and bass line extraction from polyphonic audio, and present a general architecture for a melody extraction system which outlines the a structure common to most melody extraction systems presented in recent years. In section 2.3 we examine in greater details two state of the art systems, and in section 2.4 we present chroma feature extraction and the descriptors used in our work.

In chapter 3 we explain our new approach to melody and bass line extraction using chroma features. We describe how we compute the chroma features and adapt them for our purposes (3.2) and the tracking algorithms we have implemented (3.3). Finally, we provide details about the implementation of our system as well as other algorithms implemented for the purpose of a comparative evaluation (3.4).

In chapter 4 we describe our evaluation methodology. This includes the selection of music collections for evaluation (4.2), the selection of appropriate evaluation metrics (4.3) and the preparation of the ground truth (4.5). In chapter 5 we present the results of the evaluation, including the results for salience function performance, tracking and similarity measurement. We comment on the results and draw some conclusions. We conclude the thesis in chapter 6, noting the goals we have accomplished, unresolved issues which require further work, and ideas for future investigation.

2
Scientific Background

2.1 Introduction

In the following sections we will provide the scientific background underlying the work carried out in this research. The chapter is divided into three themes – firstly, we start by giving an overview of the task of melody and bass line extraction in the literature, and outline a schematic architecture for melody and bass line extraction through which we can examine and compare different systems. Next, we examine more closely several relevant approaches and their implementation as state of the art systems. We then provide an overview of approaches to chroma feature extraction, including the approach used in our work, the Harmonic Pitch Class Profile (HPCP). We conclude this chapter with a discussion of the approaches introduced, and outline our selected approach to melody and bass line extraction.

2.2 General Architecture for Melody Extraction

Recognising the notes of different instruments in a piece of music is a relatively trivial task for the human listener. However, this seemingly simple task for a human has proven difficult and complex when we attempt to perform the same analysis automatically using computers.

Work on automated transcription traces back to the 1970's [Moorer 77], and there are good results for the transcription of monophonic signals – a detailed overview of techniques for monophonic pitch extraction can be found in [Brossier 06]. The problem of transcribing polyphonic audio however is more complex.

In monophonic music, the pitch of a single note is more easily ascertainable from its waveform, which has a relatively stable periodicity and will have a set of harmonics at integer multiples of the fundamental frequency under Fourier

analysis). Polyphonic music, as alluded to in chapter 1, will often have several overlapping notes. What is more, the fundamental frequencies of these notes might be in simple integer ratios, such that their harmonics coincide. Under Fourier analysis the spectral content of the different notes superimposes making the task of attributing specific bands and energy levels to specific notes a highly complex task, and an open research problem. Part of the problem is illustrated in figure 2.1. The top pane displays the spectrogram of a song, and the bottom pane displays the same spectrogram with the fundamental frequency of the melody overlayed.

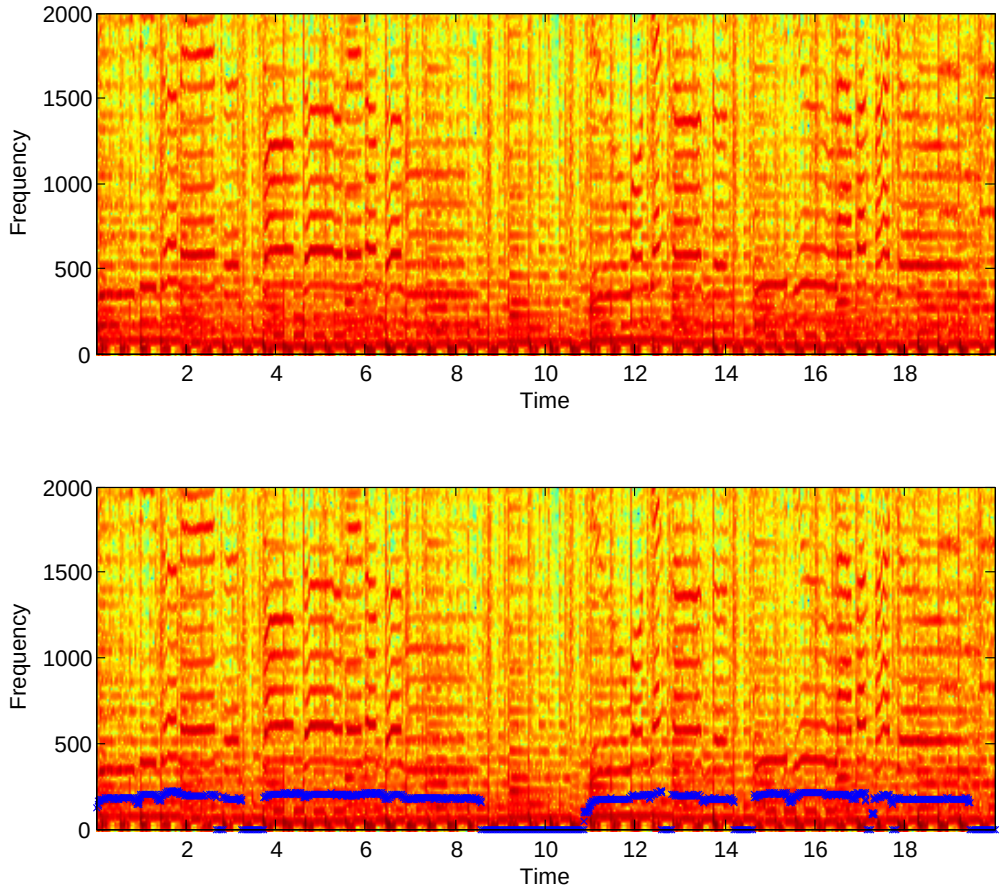

Figure 2.1: Spectrogram of "pop3.wav" from the MIREX2004 Audio Melody Extraction evaluation dataset.

In light of this problem, researchers started considering alternative formulations of the task other than full polyphonic transcription, and focused on extracting a single predominant line from polyphonic audio. Since the turn of the

millennium, many systems have been developed in what has grown into a very active research field. Evidence of this is the Audio Melody Extraction task which is part of the Music Information Retrieval Evaluation eXchange (MIREX) competitions [Downie 05], the first of which took place in 2004 and have continued annually since.

Following the MIREX competitions of 2004 and 2005 a review of the participating systems was made by [Poliner 07]. From this review some general conclusions were made about the common structure of most participating melody extraction systems, and the various differences and advantages of each system were brought to light. A common extraction architecture was identified, and is depicted in figure 2.2. It contains three main phases:

- Multi-pitch extraction: from an audio input, a set of fundamental frequency (F0) candidates for each time frame is obtained.
- Melody identification: selecting the trajectory of F0 candidates over time which forms the melody.
- Post processing: remove spurious notes or otherwise increase the smoothness of the extracted melody contour.

Figure 2.2: Common melody transcription architecture

Seven main algorithms out of the ones participating in the 2005 MIREX competition are reviewed in [Poliner 07]. In the following sections we elaborate on the general architecture presented above noting features common to most algorithms as well as comparing and contrasting the differences between them. The algorithms and their main characteristics are summarised at the end in table 2.1. In the sections that follow, we have further refined the architecture and divide it into five steps.

2.2.1 Step 1: Front End

The first step involves the initial signal processing applied to the audio signal in order to reveal the pitch content. Several different approaches are used, though

by-and-large they can be classified into two groups: those which perform the analysis in the *time domain* and those which perform the analysis in the *frequency domain*, i.e. spectral analysis.

The majority of the algorithms take the second approach, the most popular technique being to take the magnitude of the Short Time Fourier Transform (STFT). This involves taking small windows of the input audio (frames), calculating the Fourier Transform for each frame and taking the magnitude (denoted |STFT|). The result can be visualised as a spectrogram, as seen in figure 2.1. Pitched notes appear as a 'ladder' of more or less stable harmonics. The STFT can be summarised by the following formula:

$$X_l(k) = \sum_{n=0}^{N-1} w(n) \cdot x(n + lH) \cdot e^{-j\omega_k n} \qquad l = 0, 1, \ldots \tag{2.1}$$

where w denotes a real windowing function, l is the frame number and H is the time-advance value in samples (the "hop-size"). The process of taking the STFT is shown in figure 2.3, for one frame. The top pane contains the audio signal, the second pain the window function, the third pain the windowed signal (for one frame), and the bottom pane the magnitude spectrum of the Fourier Transform of the windowed signal.

Approaches based on the STFT are used in [Dressler 05, Marolt 04, Goto 04b, Ryynänen 06, Poliner 05]. Two of the algorithms do not use the STFT in this manner, those of [Paiva 04] and [Vincent 05]. Though different, both are based on the same popular time-domain method for fundamental frequency estimation, the Autocorrelation Function (ACF)[1]. The maximum of the ACF corresponds to the fundamental frequency of periodic signals. Given a sequence $x(n)$ of length K, the ACF is defined as:

$$r(n) = \sum_{k=0}^{K-n-1} x(k) \cdot x(k + n) \tag{2.2}$$

2.2.2 Step 2: Multiple F0 Estimation

In the next step, the system must estimate which pitches are present in the audio signal, given the output from the front end. The '# pitches' in table 2.1 states how many simultaneous pitches may be reported by the system at any given time.

For the systems based on the |STFT|, the problem is to identify sets of harmonics and to properly credit the energy (or salience) of each harmonic to the

[1]The ACF can actually be also computed in the frequency domain, but we have left out the details for the sake of clarity.

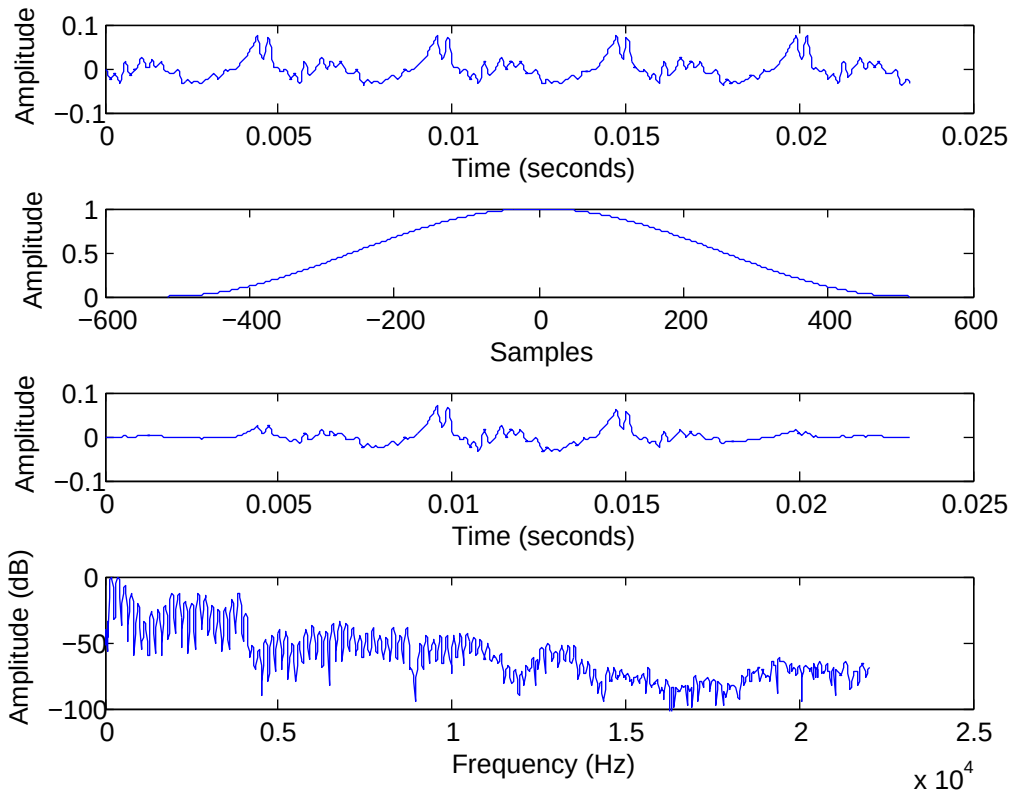

Figure 2.3: The steps involved in computing the STFT, for one frame.

appropriate fundamental, even though there need not be any energy at that fundamental for humans to perceive the corresponding pitch. As a result, one of the weaknesses of the —STFT— based approach is the possibility of reporting a fundamental frequency one octave too high, since if all the harmonics of a fundamental f_0 are present, so will the harmonics for an alleged $2f_0$. For estimating the pitches present in a given frame, the basic approach is to implement a "harmonic sieve", considering each possible F0 candidate and gathering evidence from the energy of its predicted harmonics. [Ryynänen 05] identifies lower fundamentals first and subtracts their spectrum from the overall spectrum before detecting further candidates, thus reducing evidence for fundamentals with octave errors. [Goto 04b] proposes a technique for estimating weights over all possible fundamentals to jointly explain the observed spectrum, which effectively lets different fundamentals compete for harmonics, based on Expectation-Maximization (EM) re-estimation of the set of unknown harmonic-model weights; this is largely suc-

cessful in resolving octave ambiguities. Further details of these systems will be presented in the following sections. [Marolt 04] and [Dressler 05] use a weighting system based on fundamentals which are equal to, or one octave below actual observed frequencies. A radically different approach is taken by [Poliner 05] which prefers to feed the entire Fourier Transform magnitude at each frame into a Support Vector Machine (SVM) classifier. This approach willfully ignores prior knowledge about the nature of pitched sounds, on the principle that it is better to let the machine learning algorithm figure this out for itself, where possible. The classifier is trained to report only one pitch – the appropriate melody. [Paiva 04] chooses the largest peaks of his ACF based front end, whilst Vincent uses a generative model for the time-domain wave form and selects the candidate fundamental with the largest posterior probability for its parameters under the model.

2.2.3 Step 3: Onset events

This step refers to the segmentation of frames (each with candidate F0s) into sets of distinct objects - individual notes or short strings of notes - each with a distinct start and end time. As can be seen in table 2.1, only some of the systems perform this step. [Goto 04b],[Poliner 05], and [Vincent 05] choose the best F0 candidate in each frame and return it at the final answer. [Dressler 05, Marolt 04] form distinct fragments of more-or-less continuous pitch and energy that are then the basic elements used in later processing. [Ryynänen 05] creates higher-level constructs, using a hidden Markov model (HMM) providing distributions over features including an onset strength which is related to the local temporal derivative of the total energy associated with a pitch. The result is a per-note HMM which groups F0 candidates over continuous frames.

2.2.4 Step 4: Post-processing

In this step the raw multi-pitch tracks are further cleaned up in order to get the final melody estimate. [Dressler 05], [Marolt 04] and [Paiva 04] use a set of rules that attempt to capture the continuity of good melodies in terms of pitch and energy in order to select a subset of the note fragments and form a single melody line (including gaps where no melody is selected). [Goto 04b] uses what he calls "tracking agents", agents which use a set of heuristics in order to compete for the F0 candidates. Each agent has an accumulated strength depending on past selected candidates and penalties, with agents killed and created depending on strength thresholds. The path of the strongest agent is reported as the extracted melody. Both [Ryynänen 05] and [Vincent 05] use HMMs, where Ryynänen feeds

his per-note HMM into a higher level note transition HMM, whilst Vincent only uses an HMM for smoothing the F0 contour.

2.2.5 Step 5: Voicing

The final step is that of voicing detection. This involves determining when the melody is present and when it is not. Once again, not all algorithms perform this step, and Goto and Vincent report their best pitch estimate at each frame. Poliner uses a global energy threshold to gate his initially continuous output, whilst Dressler, Marolt and Paiva's selection of note fragments naturally leads to gaps where there is no suitable element. Dressler further augments this with a local threshold to discount low energy notes. The algorithms and their main characteristics are summarised in table 2.1 below.

System	Front end	Multi-pitch	# pitches	Onset events	Post-processing	Voicing
[Dressler 05]	STFT+sines	Harmonic model fit	5	Fragments	Streaming rules	Melody + local thresh.
[Marolt 04]	STFT+sines	EM fit of tone models	>2	Fragments	Proximity rules	Melody grouping
[Goto 04b]	Hier. STFT+sines	EM fit of tone models	>2	–	Tracking agents	(continuous)
[Ryynänen 05]	Auditory + STFT	Harmonic Sieve	2	Note onsets	HMM	Background model
[Poliner 05]	STFT	SVM Classier	1	–	–	Global threshold
[Paiva 04]	Auditory correlogram	Summary autocorrelation	>2	Pitches	Pruning rules	Melody grouping
[Vincent 05]	YIN / Time windows	Gen. model inference	5 / 1	–	HMM	(continuous)

Table 2.1: Melody extraction algorithms, main characteristics.

2.2.6 Evaluation and Conclusion

As we have seen, the task of melody extraction can be approached in various ways. Until recently, a number of obstacles have impeded an objective comparison of these systems, such as the lack of a standardised test set or consensus regarding evaluation metrics. In 2004, the Music Technology Group (MTG) at the Pompeu Fabra University proposed and hosted a number of audio description contests in conjunction with the International Conference on Music Information Retrieval (ISMIR). These evaluations which included contests for melody extraction, genre classification/artist identification, tempo induction, and rhythm classification, evolved into the Music Information Retrieval Evaluation Exchange (MIREX) [Downie 05] which took place during the summer of 2005, organised and run by Columbia University and the University of Illinois at Urbana-Champaign.

In Chapter 4 we provide an overview of the MIREX competitions, examining the test sets and metrics used, as well as the results obtained by the aforementioned algorithms. Following the reading of recent papers on melody and/or bass line extraction, we note that the task of melody and bass line extraction still lacks a uniform evaluation methodology incorporating evaluation metrics and music collections for testing, however we believe that through the joint effort of the research community and initiatives such as the MIREX competitions and the *Real World Computing* music database (RWC) [Goto 04a] (discussed in detail in chapter 4) that a uniform methodology can be established. In this work we have made an effort to perform the evaluation in a way which supports a uniform and comparable evaluation methodology.

2.3 State of the Art Systems

In this section we take a closer look at two state of the art melody extraction systems. Firstly, we examine the PreFEst system by [Goto 04b], which has been briefly introduced in the previous section. Next, we examine a system developed by Klapuri and Ryynänen in [Klapuri 06]. Unlike the one by Goto, this is a system for multiple F0 estimation. It serves as the core for a full melody and bass line transcription system as well as a multiple F0 estimator, however we will not examine the system beyond what is presented in [Klapuri 06]. Our interest will be in evaluating this system as a *Salience Function*, as was described in section 1.2.4.

2.3.1 Probabilistic Modeling and Expectation Maximisation

Masataka Goto was the first to demonstrate successful melody and bass line extraction from *real world audio signals* such as the ones recorded on commercially distributed CDs [Goto 99, Goto 04b], in what is his now well-known PreFEest (Predominant F0 Estimation Method) system.

In [Goto 04b], the author starts by relating to the work carried out in a related field, which he refers to as Sound Source Segregation[2]. We can define it as the task of extracting from an audio signal without additional knowledge a set of audio signals whose mix is perceived similarly to the original signal, and where every extracted signal on its own is meaningful to a human listener. A full review of Audio Stream Separation is beyond the scope of this work, and we refer the reader to [Vinyes 05] for a comprehensive overview of the task and the different techniques used in attempt to solve it. What is important for us is what Goto explains with relation to Audio Stream Separation, namely that segregation is *not* necessary for understanding. That is, as human listeners we can make sense of two auditory streams, without necessarily separating them beforehand. This motivates the development of a method for musical understanding (in this case melody and bass line extraction) which does not depend on Audio Source Separation.

Goto then explains what he defines as the Music-Scene-Description problem. Music-Scene-Description is a process by which we obtains a description representing the input musical audio signal. When considering the form of this description, Goto notes that a transcription (in the form of a musical score) requires musical training and expertise, and what is more, does not capture non-symbolic properties such as the expressive performance of music. Instead, he identifies the requirements for a description as the following:

- An intuitive description that can be easily obtained by untrained listeners.
- A basic description that trained musicians can use as a basis for higher-level music understanding.
- A useful description facilitating the development of various practical applications.

Following these requirements, Goto proposes a description consisting of five sub-symbolic representations:

[2]Also referred to as Audio Stream Separation, Blind Source Separation (BSS), or simply Source Separation.

1. Hierarchical beat structure
2. Chord change possibility
3. Drum pattern
4. Melody line
5. Bass line

For melody and bass line, a continuous F0 contour is proposed as a fitting sub-symbolic representation. The PreFEst system performs melody and bass line extraction, and we now provide further details on how this is performed. Before explaining the actual method however, we must first identify the challenges of the task at hand, that is, of extracting a continuous F0 contour representing the melody or bass line from a polyphonic signal. Goto identifies three main problems, which for clarity we have named in the following way:

- The **Range** problem – Which F0 belongs to the melody and which to the bass line in polyphonic music.
- The **Estimation** problem – How to estimate the F0 in complex sound mixtures where the number of sound sources is unknown.
- The **Selection** problem – How to select the appropriate F0 when several ambiguous F0 candidates are found.

In order to address these problems, we have to make the following assumptions:

- The **Range** assumption – The melody will have most of its harmonic content in the middle to high range frequencies, whilst the bass will be more present in the low frequencies.
- The **Estimation** assumption – The melody and bass line have a harmonic structure, and we can use this fact to attempt and infer the appropriate F0s.
- The **Selection** assumption – The melody and bass line will tend to have temporally continuous trajectories.

Finally, based on these assumptions, we can suggest potential solutions which form the basis for the implemented system:

- The **Range** solution – we can limit the frequency regions examined for melody and bass line.
- The **Estimation** solution – we will regard frequency components as a weighted mixture of all possible harmonic structure tone models.
- The **Selection** solution – in the selection of the F0s, we will consider temporal continuity and select the most stable trajectory.

The PreFEst system can be divided into three parts. We will show how each of these parts incorporates one or more of the steps mentioned in section 2.2 as being part of a general melody extraction architecture. The three parts are:

- Front-end: performs spectral analysis on a limited frequency range of the input signal to produce frequency components for further analysis. Equivalent to step 1 (Front-end), as in section 2.2.1.
- Core: regards the observed frequency components as a weighted mixture of all possible harmonic-structure tone models. It estimates weights for the frequency components using Expectation Maximisation (EM), and the maximum weight model is considered the most predominant harmonic structure and its F0 is obtained. By taking the top weighted models we get a set of candidate F0s at each frame. This is step 2 (Multi-pitch), as in section 2.2.2.
- Back-end: given the F0 candidates, the most dominant and stable trajectory is chosen using a tracking-agent architecture, and returned as the resulting melody or bass line (depending on the frequency range used in the Front-end). This is step 4 (Post-processing), as in section 2.2.4. Note that steps 3 (Onset events) and 5 (Voicing) are not part of the system.

The PreFEst architecture is summarised in figure 2.4, taken from [Goto 04b] with permission of the author. In the following sections we elaborate on each of the three parts of the system.

2.3.1.1 Font-end

The Front-end divides further into three steps:

1) STFT

The first step involves taking the STFT of the signal. Differently from some of the other approaches presented earlier which use the STFT directly, Goto uses

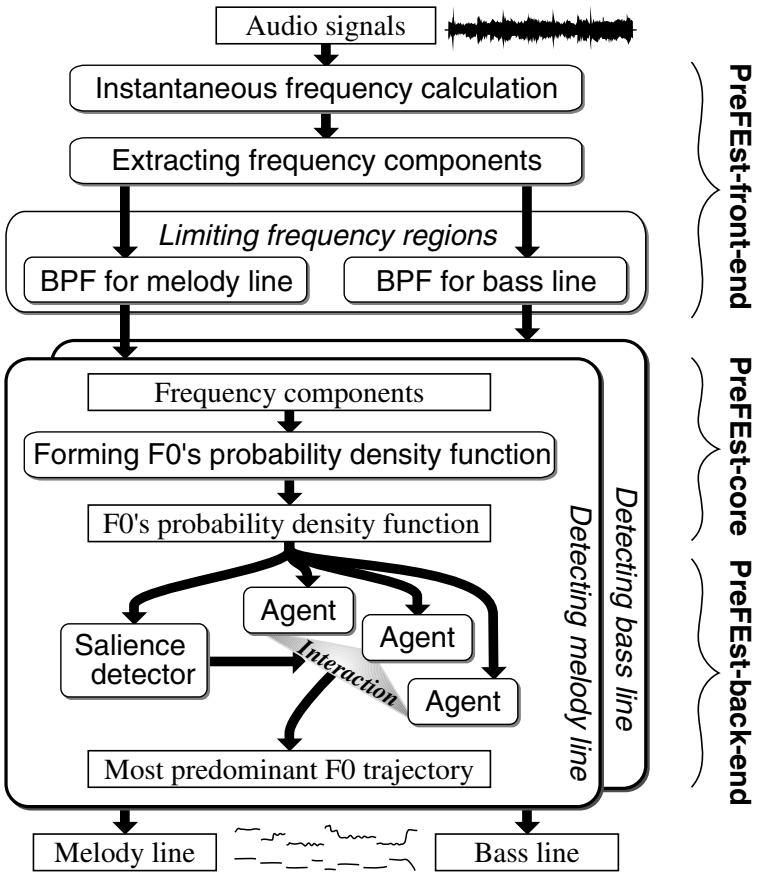

Figure 2.4: The PreFEst architecture.

an STFT-based multirate filterbank (figure 2.5). This involves taking the FFT for the highest frequency range, and then low-pass filtering and downsampling the signal before calculating the FFT for the next frequency range. This way optimal analysis parameters can be chosen at each step depending on the frequency range analysed, in order to get the best analysis possible given the time-frequency resolution trade-off inherent to Fourier Analysis [Cohen 89].

2) Instantaneous Frequency (IF) Components

From the output of the filter bank, the *instantaneous frequency* (IF) is calculated. The IF is defined as the rate of change of the signal phase. This involves the mapping of the center frequency ω of an STFT filter to the instantaneous frequency $\lambda(\omega t)$, where:

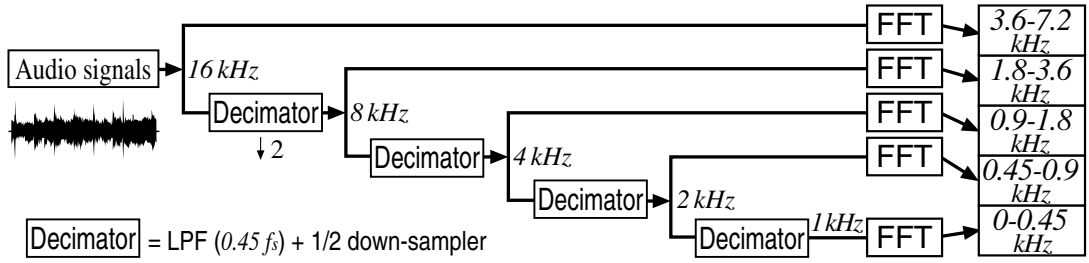

Figure 2.5: STFT-based multirate filterbank.

$$\lambda(\omega t) = \omega + \frac{a\frac{\partial b}{\partial t} - b\frac{\partial a}{\partial t}}{a^2 + b^2} \tag{2.3}$$

To get a and b we use a reformulation of the STFT for a signal $x(t)$ and a window function $h(t)$:

$$\begin{aligned} X(\omega, t) &= \int_{-\infty}^{\infty} x(\tau) h(\tau - t) e^{-j\omega\tau} \delta\tau \\ &= a + jb, \end{aligned} \tag{2.4}$$

Using this, we can obtain a set of instantaneous frequencies:

$$\Psi_f^{(t)} = \left\{ \psi \mid \lambda(\psi, t) - \psi = 0,\ \frac{\partial}{\partial \psi}(\lambda(\psi, t) - \psi) < 0 \right\} \tag{2.5}$$

By calculating the power of these frequencies, given by the STFT spectrum at $\Psi_f^{(t)}$, we can define the power distribution function $\Psi_p^{(t)}(\omega)$ as

$$\Psi_p^{(t)}(\omega) = \begin{cases} |X(\omega, t)| & \text{if } \omega \in \Psi_f^{(t)}, \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

In figure 2.6 we show first the amplitude spectrum of a signal and then the IF amplitude spectrum, reproduced from [Abe 96].

3) Limit Frequency Regions The final step is to limit the frequency regions used for melody and bass line using two band-pass filters (BPF). The frequency responses of the band-pass filters used for melody and bass line are given in figure 2.7, taken from [Goto 04b].

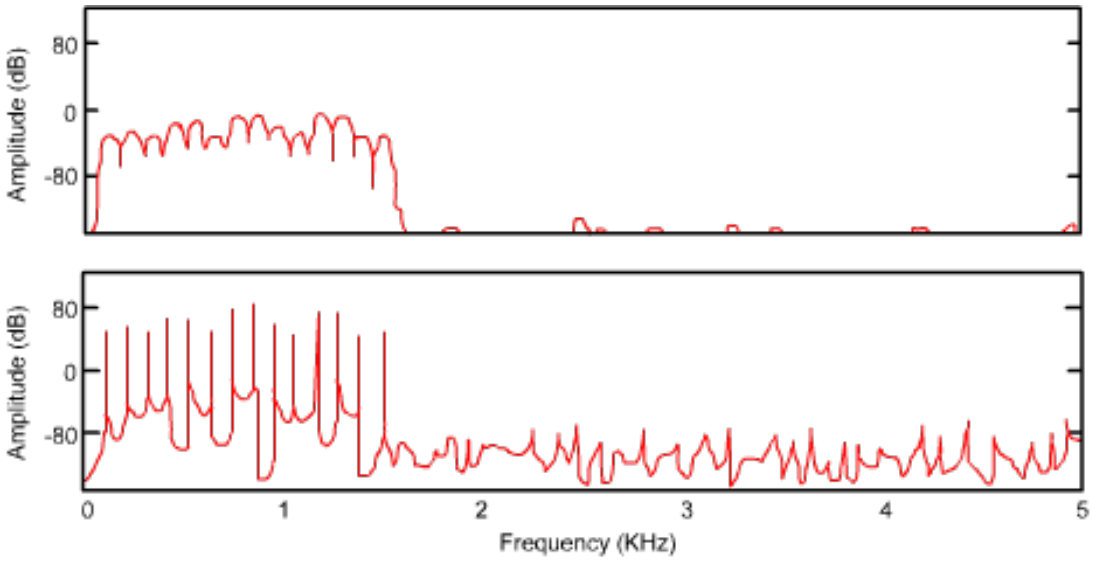

Figure 2.6: The amplitude spectrum and the IF amplitude spectrum.

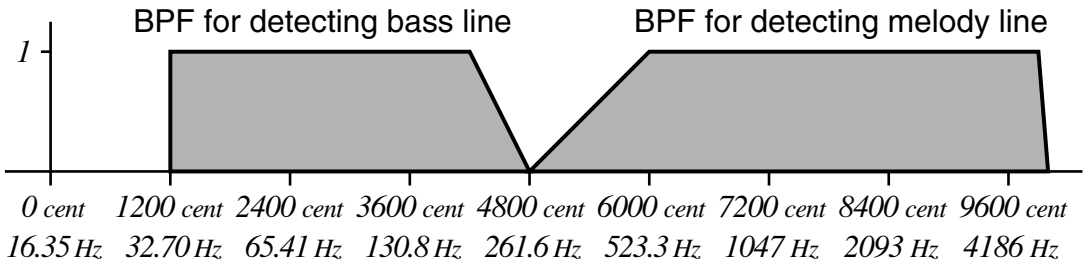

Figure 2.7: Frequency responses of BPFs used for melody and bass line in PreFEst

This allows us to express the observed frequency components as a probability density function (PDF) – the *observed* PDF, which will facilitate the use of statistical techniques in further steps:

$$p_{\Psi}^{(t)}(x) = \frac{BPF_i(x)\Psi_p'^{(t)}(x)}{\int_{-\infty}^{\infty} BPF_i(x)\Psi_p'^{(t)}(x)\delta x} \tag{2.7}$$

where $\Psi_p'^{(t)}(x)$ is $\Psi_p^{(t)}(\omega)$ expressed in *cents* (a musical interval measurement based on a logarithmic scale):

$$f_{cent} = 1200 \log_2 \frac{f_{Hz}}{440 \times 2^{\frac{3}{12}-5}} \tag{2.8}$$

2.3.1.2 Core

The PreFEst core is responsible for taking the observed PDF produced by the front end and outputting candidate F0s. We consider each observed PDF to have been generated from a weighted-mixture model of the *tone models* of all the possible F0s. A tone model is the PDF corresponding to a typical harmonic structure and indicates where the harmonics of the F0 tend to occur. Figure 2.8 shows examples of four tone models – (a) and (b) are examples of tone models for melody F0s, whilst (c) and (d) are for bass F0s. Because the weights of the tone models represent the relative dominance of every possible harmonic structure in the mixture, we can regard these weights as the F0's PDF: the more dominant a tone model is in the mixture, the higher the probability of the F0 corresponding to the model.

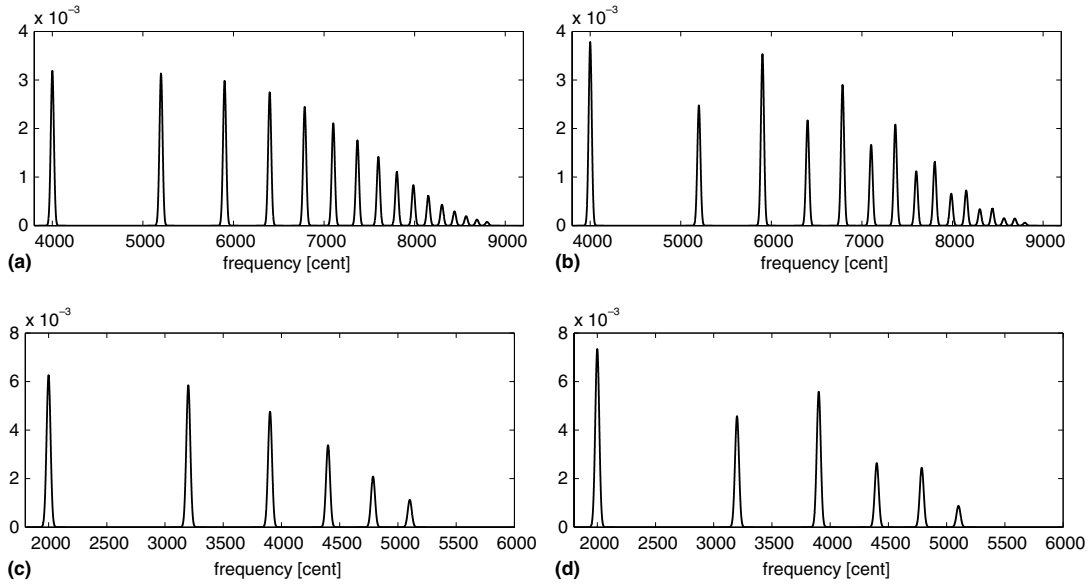

Figure 2.8: Tone models for melody and bass line fundamental frequencies.

Each tone model can be modeled mathematically as a Gaussian Mixture Model (GMM). This way, we can model the *observed* PDF as a weighted sum of all tone models. The goal is then to find the model parameters (tone model weights) which give the Maximum A Posteriori (MAP) probability that the *observed* PDF was generated by the model. We can then use the resulting F0 weighting function as a PDF for all F0s for a single frame, which is effectively what we defined in section 1.2.4 as a Salience Function. This maximisation can not be performed analytically, and so Goto makes use of the Expectation Maximisation (EM) algorithm in order to obtain the F0 PDF. A detailed account

of the mathematics involved in the application of the EM algorithm for finding the weights which give the MAP for the tone model mixture is beyond the scope of this dissertation, and we refer the reader to [Goto 04b] for further details. A good explanation of GMMs and the application of the EM algorithm can also be found in [Master 00].

2.3.1.3 Back-end

Given an F0 PDF for every frame of the input signal, the task is then to select the correct melody or bass line F0 at each frame – corresponding to one of the peaks in the F0 PDF. As explained earlier, the goal is to select the most dominant and stable F0 trajectory over the analysis frames, which is to be returned as the extracted melody (or bass line).

Goto performs this selection using an architecture of "tracking agents" – alternate hypotheses of the current and past pitch which compete to acquire the new pitch estimates from the current frame, and live or die based on a continuously-updated penalty that reflects the total strength of the past pitches they represent. The steps involved in this process are presented below, and illustrated in figure 2.9 taken from [Goto 04b].

- A salience detector is used to select peaks higher than a set threshold from the current frame's PDF.
- The peaks are then assigned to existing agents according to the peak's closeness to the previous peak selected by the agent, and the agent reliability.
- If any peaks are left unassigned, a new agent is created for them.
- Agents to which no peak is assigned receive a penalty.
- If the agent penalty reaches a set threshold, the agent dies.
- The reliability of an agent is determined by its reliability on the previous frame and the current peak's salience.
- The output F0 sequence is the peak trajectory of the agent with the highest reliability and greatest total power along the trajectory, where the total power is the power of all harmonics of the selected F0 at each frame.

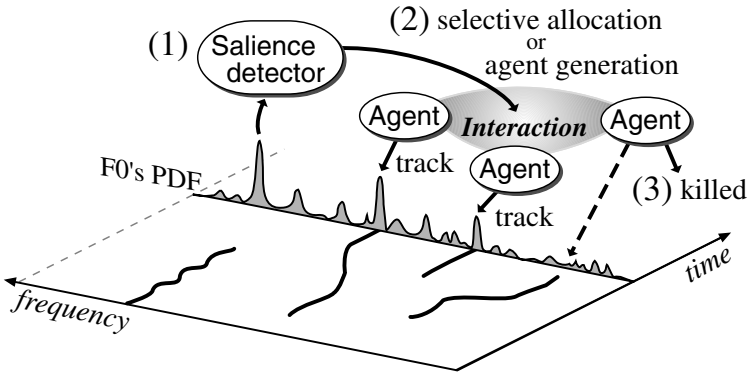

Figure 2.9: The PreFEst tracking agents architecture.

2.3.1.4 PreFEst Evaluation and Conclusion

The PreFEst system was tested on ten musical excerpts taken from compact disc recordings, in the genres of popular, jazz and orchestral music. Each excerpt was 20 seconds long and contained a single-tone melody and several instruments. The evaluation was performed by comparing the extracted F0 for every frame with a hand labelled F0 determined using an F0 editor developed by the author. The extracted F0 was judged correct if its distance from the labelled F0 was under 50 cents (i.e. if the extracted F0 was within a range of one semitone centred on the labelled F0). Only frames in which a melody was present were taken into consideration, i.e. performance was evaluated for voiced frames only.

The system obtained an average correct extraction rate of 88.4% for melody and 79.9% for bass line. These were amongst the first significant results to be obtained by a melody and bass line extraction system. It is important to note that the test set was very limited at the time (only 10 songs and only 20 seconds for each song), which is an example of the lack of evaluation data which was one of the problems in the early days of this field. PreFEst was later evaluated again using more test data and a larger set of metrics as part of the MIREX competitions, as discussed in chapter 4.

2.3.2 Multiple F0 Estimation by Summing Harmonic Amplitudes

We now present a second approach to melody extraction. Whilst the previous system is fairly complex in its mathematical detail and implementation, the following takes a simple approach, which is also computationally efficient. We shall

examine it through the work of Klapuri, presented in [Klapuri 06]. It is important to note that unlike the previous system which is presented as a complete melody and bass line extraction solution, the following algorithms are presented as multiple F0 estimators. As such, they are closer to a Salience Function, without the final post-processing step standard to melody extraction systems.

2.3.2.1 Introduction

In [Klapuri 06], the author presents three algorithms for multiple F0 estimation. At their core, they are all based on the same concept, which we present below. We then explain each of the three algorithms, labelled the *Direct* method, the *Iterative* method and the *Joint* method. The underlying approach common to all three algorithms has two primary steps:

1. Take the STFT of the input signal, and perform spectral whitening
2. Compute a salience value for candidate F0s, using a salience function calculated as the weighted sum of the amplitudes of the harmonic partials of the candidate F0.

For the three algorithms here, we work with candidate *periods* rather than candidate F0s, where the relation between a period τ and its corresponding frequency f is

$$\tau = \frac{f_s}{f} \tag{2.9}$$

where f_s is the sampling frequency of the input signal which for all data used in our work has the value of 44,100Hz. Klapuri defines the salience $s(\tau)$ of a period candidate τ as follows:

$$s(\tau) = \sum_{m=1}^{M} g(\tau, m)|Y(f_{\tau,m})| \tag{2.10}$$

where $Y(f)$ is the STFT of the *whitened* time-domain signal, $f_{\tau,m} = m \cdot f_s/\tau$ is the frequency of the m:th harmonic partial of a F0 candidate f_s/τ, M is the total number of harmonics considered and the function $g(\tau, m)$ defines the weight of partial m of period τ in the summation. Klapuri notes however that there is no efficient method for computing the salience function as given in equation 2.10, and proposes to replace it with a discrete version:

$$\hat{s}(\tau) = \sum_{m=1}^{M} g(\tau, m) \max_{k \in \kappa_{\tau,m}} |Y(k)| \tag{2.11}$$

where the set $\kappa_{\tau,m}$ defines a range of frequency bins in the vicinity of the m:th overtone partial of the F0 candidate f_s/τ

$$\kappa_{\tau,m} = [\langle mK/(\tau + \Delta\tau/2)\rangle, \ldots, \langle mK/(\tau - \Delta\tau/2)\rangle] \tag{2.12}$$

where $\Delta\tau = 0.5$, the spacing between fundamental period candidates τ is half the sampling interval.

2.3.2.2 Spectral Whitening

One of the challenges in a melody extraction system is to make it robust to different sound sources, or rather to different timbres. This can be achieved by the flattening of the spectral envelope of the signal, which largely defines the timbre of the sound. This process is referred to as *Spectral Whitening*. Klapuri achieves this by performing the following steps:

- Given a signal $x(t)$, we take the discrete Fourier Transform to get $X(k)$ (using a Hann window and zero padding to twice the window size).
- Next, a band-bass filterbank is simulated in the frequency domain, with the centre frequencies of the subbands uniformly distributed in a critical-band scale. Each subband with centre frequency c_b has a triangular power response starting at c_{b-1} and ending at c_{b+1}. The centre frequencies are given by equation 2.13, and the power response of the subbands is shown in figure 2.10.

$$c_b = 229 \times (10^{(b+1)/21.4} - 1) \qquad b = 1, \ldots, 30 \tag{2.13}$$

- Next, we calculate the standard deviations σ_b within the subbands b, where K denotes the size of the Fourier Transform

$$\sigma_b = \left(\frac{1}{K} \sum_k H_b(k) |X(k)|^2 \right)^{1/2} \tag{2.14}$$

- From these σ_b we can then calculate bandwise compression coefficients $\gamma_b = \sigma_b^{\nu-1}$, where ν is a parameter determining the degree of spectral whitening to be applied and was determined experimentally and set to 0.33.
- Finally we can obtain an expression for the whitened spectrum $Y(k)$ as

$$Y(k) = \gamma(k) X(k) \tag{2.15}$$

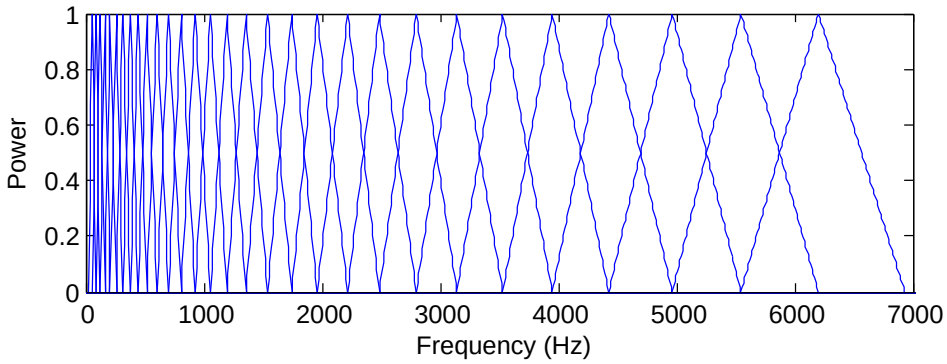

Figure 2.10: Power response for subbands $H_b(k)$ applied in spectral whitening

In figure 2.11 we provide an example of the effect of spectral whitening. The blue curve shows the original amplitude spectrum of the signal. The red curve shows the amplitude spectrum after spectral whitening. As expected, the resulting spectrum maintains the peak locations of the original spectrum, but it flattens the spectral shape such that all peaks have roughly the same amplitude.

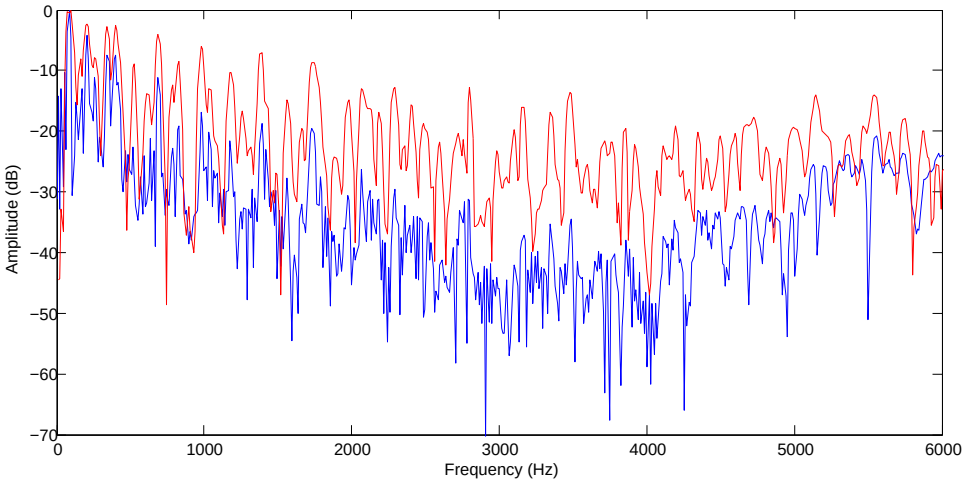

Figure 2.11: Spectral amplitude of a signal, before and after spectral whitening.

2.3.2.3 Direct Method

Based on the background provided above, Klapuri suggests three algorithms for multiple F0 estimation. In this section we present the first of the three and the

simplest, appropriately labelled by the author the "Direct" method.

The idea is to evaluate the salience function $\hat{s}(\tau)$ for a range of values of τ, and pick the desired number of local maxima as the estimated F0s. For predominant F0 estimation we would choose the greatest maximum at each frame, and the simplest extension to melody extraction would be to output this value as the melody F0.

In order to do this calculation the only thing missing is to define the weighting function $g(\tau, m)$ in a way which minimises the estimation error. Rather than attempting to find an analytical solution, the author solves this using optimisation with a large amount of training material:

- Training material is generated, consisting of random mixtures of musical instrument sounds with varying degrees of polyphony, starting with only one line (monophony), through 2, 4 and up to 6. 4,000 training instances were used in total.

- For every instance Klapuri performs both multiple F0 estimation and predominant F0 estimation, where the output for the predominant F0 estimation is deemed correct if it matches *any* of the reference F0s in the mixture[3]. The optimisation is aimed at minimising the average of the error rates of both multiple and predominant F0 estimations. For further details about the optimisation procedure we refer the reader to [Klapuri 06]. Finally, the author was able to obtain a functional representation for $g(\tau, m)$ of the following form

$$g(\tau, m) = \frac{f_s/\tau + \alpha}{m f_s/\tau + \beta} \tag{2.16}$$

  The values for α and β used by the author in later experiments are 27Hz and 320Hz respectively for a 46ms frame (analysis window of 2048 samples when $f_s = 44100Hz$), and 52Hz and 320Hz respectively for a 93ms frame (analysis window of 4096 samples).

Now that we have an expression for $g(\tau.m)$, $\hat{s}(\tau)$ can be computed using the Direct method, and all that is needed is to select a frequency range to examine. Needles to say, having to go through every candidate period τ in the range will be computationally expensive, and as we shall see the next method, in addition to introducing a more elaborate approach, is also amenable to efficient computation.

[3]This form of evaluation can not be applied to melody extraction where there is always only one correct F0 in the mixture.

2.3.2.4 Iterative Method

One of the shortcomings of the Direct method is that while the highest peak is a good indicator of one of the true periods τ, other peaks in $\hat{s}(\tau)$ might be a result of the same period τ, appearing at integer multiples of the corresponding F0. Klapuri suggests solving this through iterative estimation and cancellation – the spectrum of each detected sound is cancelled from the mixture and $s(\tau)$ is updated before estimating the next F0. We outline the algorithm in five steps:

1. Initialise a residual spectrum $Y_R(k)$ to equal $Y(k)$, and a spectrum of detected sounds $Y_D(k)$ to zero.

2. Estimate a fundamental period $\hat{\tau}$ using $Y_R(k)$ and Algorithm 1 (presented shortly). $\hat{\tau}$ is chosen as the maximum of $\hat{s}(\tau)$.

3. The harmonic partials of $\hat{\tau}$ are located in $Y_R(k)$ at bins $\langle mK/\tau \rangle$. We estimate each partial's frequency and amplitude and use it to calculate the magnitude spectrum at the few surrounding frequency bins. The magnitude spectrum of the m:th partial is weighted by $g(\hat{\tau}, m)$ and added to the corresponding position of the spectrum of detected sounds $Y_D(k)$.

4. Recalculate the residual spectrum as $Y_R(k) \leftarrow \max(0, Y(k) - dY_D(k))$, where d controls the amount of subtraction.

5. If there are any sounds remaining in $Y_R(k)$, return to step 2.

Unlike the Direct method which requires scanning through all candidate periods in order to find the maximum of $\hat{s}(\tau)$, the Iterative method can be computed using an efficient divide-and-conquer algorithm (Algorithm 1) which avoids calculating $\hat{s}(\tau)$ for every possible period τ. For further details on Algorithm 1 the reader is referred to [Klapuri 06].

These five steps are repeated until the desired number of sounds has been detected. When the number of sounds is not given, it has to be estimated. The task of *polyphony estimation* is performed by repeating the iteration until the newly-detected period τ_j at iteration j no longer increases the quantity

$$S(j) = \frac{\sum_{i=1}^{j} \hat{s}(\hat{\tau}_i)}{j^{\gamma}} \tag{2.17}$$

where $\gamma = 0.70$ was determined empirically.

Algorithm 1: Fast search for the maximum of $\hat{s}(\tau)$

$Q \leftarrow 1$; $\tau_{low}(1) \leftarrow \tau_{min}$; $\tau_{up}(1) \leftarrow \tau_{max}$; $q_{best} \leftarrow 1$;
while $\tau_{up}(q_{best}) - \tau_{low}(q_{best}) > \tau_{prec}$ **do**
    #Split the best block and compute the new limits
    $Q \leftarrow Q + 1$
    $\tau_{low}(Q) \leftarrow (\tau_{low}(Q_{best}) + \tau_{up}(q_{best}))/2$
    $\tau_{up}(Q) \leftarrow \tau_{up}(q_{best})$
    $\tau_{up}(q_{best}) \leftarrow \tau_{low}(Q))$
    #Compute new saliences for the two block-halves
    for $q \in \{q_{best}, Q\}$ **do**
        Calculate $s_{max}(q)$ using equations 2.11-2.12 with $g(\tau, m) = \frac{f_s/\tau_{low}(q)+\alpha}{m f_s/\tau_{up}(q)+\beta}$ where $\tau = (\tau_{low}(q) + \tau_{up}(q))/2$ and $\Delta\tau = \tau_{up}(q) - \tau_{low}(q)$
    end
    #Search the best block again
    $q_{best} \leftarrow \arg\max_{q\in[1,Q]} s_{max}(q)$
end
Return $\hat{\tau} = (\tau_{low}(q_{best}) + \tau_{up}(q_{best}))/2$
$\hat{s}(\hat{\tau}) = s_{max}(q_{best})$

2.3.2.5 Joint Method

As we have seen, the iterative method is both faster to compute and takes into consideration the issue of falsely detecting partials of a present F0 as other F0s. One issue still remains however, and that is the possibility that the iterative process of estimation and cancellation has some undesirable effect on the results. To examine this, Klapuri suggests to factor the cancellation into the salience function and compute a joint estimation for all F0s simultaneously. This procedure is described in five steps as follows:

1. Calculate the salience function $\hat{s}(\tau)$ according to equation 2.11.
2. Choose the I highest local maxima of $\hat{s}(\tau)$ as candidate fundamental period values τ_i with $i = 1, \ldots, I$.
3. For each candidate i, compute the following quantities:
    (a) The frequency bins of the harmonic partials $k_{i,m}$
    (b) The candidate spectrum $Z_i(k)$
4. Let us denote the number of simultaneous F0s to estimate by P, and a set of P different candidate indices i by $\mathcal{I}$.

5. Then, find such an index set $\mathcal{I}$ that maximises

$$G(\mathcal{I}) = \sum_{i \in \mathcal{I}} \sum_{m} g(\tau_i, m) |Y(k_{i,m})| \prod_{j \in \mathcal{I} \setminus i} (1 - Z_j(k_{i,m})) \qquad (2.18)$$

Equation 2.18 can be broken down relatively simply – the summation at the centre of the expression is the salience function $\hat{s}(\tau)$ as we have seen in before. The product to its right is the cancellation factor from all other candidates i in the examined set $\mathcal{I}$, and finally the summation on the extreme left sums the resulting salience value for all candidates i in $\mathcal{I}$, giving us an overall salience value for the set $\mathcal{I}$. A problem with equation 2.18 is that the computational complexity of evaluating $G(\mathcal{I})$ for all $\binom{I}{P}$ different index combinations $\mathcal{I}$ is too great for it to be feasible. A reasonably efficient implementation is possible by making use of the lower bound $\tilde{G}(\mathcal{I})$ of $G(\mathcal{I})$. The complete details are beyond the scope of this section, but the reader is referred to [Klapuri 06] for the full mathematical detail and a relatively efficient algorithm for performing the computation. What should be noted, is that since an initial set of I maxima needs to be found, Algorithm 1 can not be used in this case, and so the Joint method can not be computed as efficiently as the Iterative method.

2.3.2.6 Evaluation and Conclusion

The three algorithms were evaluated using a significantly different approach to that used by Goto or the MIREX competitions (as detailed in chapter 4).

The test data for the evaluation was generated by the authors, by creating random mixtures of musical instrument samples with F0s between 40 and 2100 Hz. First an instrument was allotted randomly, and then a sound from the prescribed range was randomly selected. The process was repeated until the desired number of sounds was obtained, which were then mixed with equal mean-square levels. The authors used a total of 2842 samples from 32 musical instruments. As polyphony estimation is a difficult task in its own right, polyphony estimation and multiple F0 estimation were evaluated separately, and we only present the results for the latter.

The three algorithms, Direct, Iterative and Joint (labelled in the diagram as *d*,*i* and *j* respectively) were compared against three reference algorithms, presented in [Tolonen 00], [Klapuri 03] and [Klapuri 05] (labelled in the diagram as [3], [4] and [5] respectively). An extracted F0 was judged correct if it deviated less than 3% from the reference F0. The authors also evaluated predominant F0 estimation, by judging an F0 to be correct if it matches *any* of the true F0s in the mixture. We note again how this evaluation methodology is highly different

from the one we saw earlier for the evaluation of Goto's PreFEst. As such, it is hard to directly compare the results with those we have seen earlier. Klapuri and Ryynänen did however present a complete melody extraction system [Ryynänen 05] (though it is based on a different approach to the one presented here), which took part in the MIREX evaluations and can be more easily compared to other systems. More recently, a full system for melody, bass line and chord estimation based on the salience function we have presented above was developed by Ryynänen and Klapuri [Ryynänen 08], and evaluated using the RWC, more on which in chapter 4. The evaluation results for the Direct, Iterative and Joint methods are presented in figure 2.12, taken from [Klapuri 06] with the permission of the authors.

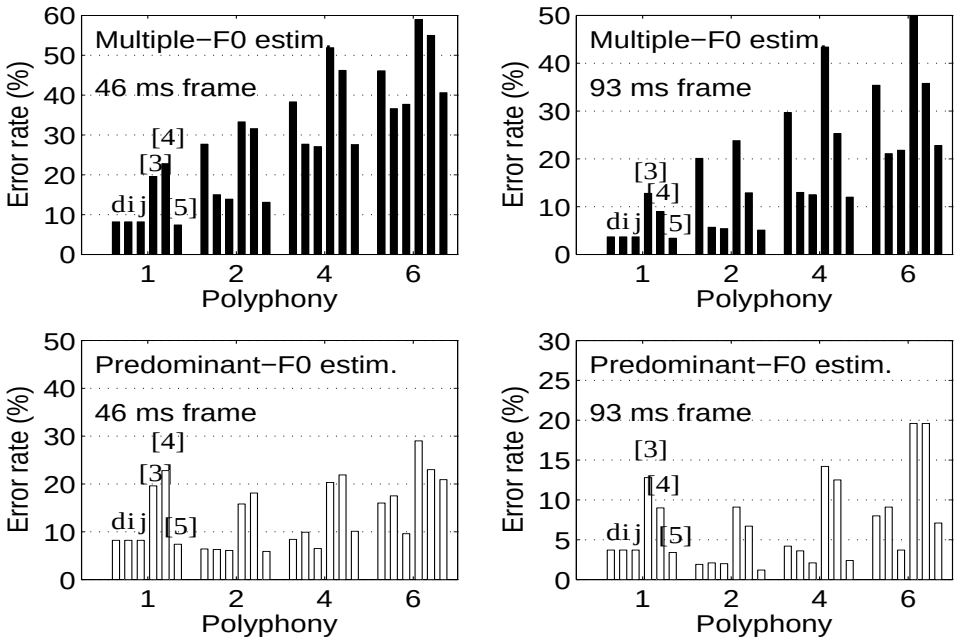

Figure 2.12: Results for multiple and predominant F0 estimation taken from [Klapuri 06].

As seen in figure 2.12, the performance of the Iterative and Joint approaches for multiple F0 estimation is almost the same and outperforms the rest. For predominant F0 estimation, the error rates for the Direct and Iterative methods are similar to that of [5], whilst the Joint method outperforms the rest for high polyphonies. Though not comparable to other results we have presented so far for melody and bass line extraction (or those presented in chapter 4), we can make the overall observation that these approaches seem to perform well and are com-

parable (if not better in some cases) to the more elaborate (and computationally expensive) approaches presented in [Tolonen 00, Klapuri 03, Klapuri 05]. This further motivates us to investigate the use of a simple salience function based on the summation of harmonic amplitudes for the purpose of melody and bass line extraction, as explained in the following chapters.

2.4 Chroma Feature Extraction

In the previous sections we reviewed the general architecture for melody and bass line extraction, as well as two relevant state of the art systems and the details of the "Salience Functions" used at their core. In this section we provide the scientific background for what forms the Salience Function at the core of our system, namely Chroma Feature Extraction.

2.4.1 Pitch Class Distribution - An Overview

Chroma features refer to the induction of tonality information from the audio signal. The nomenclature for this feature is varied and also includes pitch-class distribution (PCD), pitch histograms and pitch-class profile (PCP). Most of these refer to the same concept, though their method of computation can vary significantly. Generally speaking, the pitch-class distribution of music is a vector of features describing the different tones (or pitches) in the audio signal (the granularity of the analysis can be as coarse as a complete audio signal or as fine as a single analysis frame), and it is directly related to the tonality of a piece. [Fujishima 99] proposed a chord recognition system based on the *pitch-class profile* (henceforth PCP), defined by Fujishima as a twelve dimensional vector representing the intensities of the twelve semitone pitch classes. An example of such a PCP (otherwise referred to as a 12-bin chroma histogram) is given in figure 2.13. The twelve bins correspond to the pitch classes A, $A\sharp$,...,G,$G\sharp$.

In [Gómez 06a], Gómez defines the requirements that should be fulfilled by reliable a pitch class distributions:

1. Represent the pitch class distribution of both monophonic and polyphonic signals.

2. Consider the presence of harmonic frequencies – the first harmonics of a complex tone belong to the major key defined by the pitch class of the fundamental frequency, and all but the 7th harmonic belong to its tonic triad.

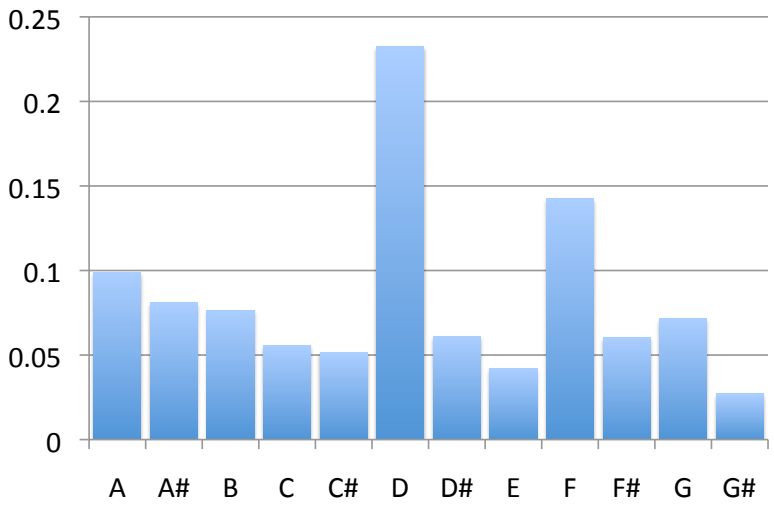

Figure 2.13: Pitch-class profile example

3. Be robust to noise: ambient noise (e.g. live recordings), percussive sounds, etc.

4. Be independent of timbre and instrument type.

5. Be independent of loudness and dynamics.

6. Be independent of tuning, so that the reference frequency can be different form the standard A 440Hz.

Gómez points out that all approaches for computing the instantaneous evolution of the pitch class distribution follow the same schema, shown in figure 2.14. In the following sections we briefly review the different approaches taken towards computing each step of this schema, and in the final section we present the *Harmonic Pitch Class Profile*, an extension of the PCP presented in [Gómez 06a] and the tonal descriptor used in our work on melody and bass line extraction.

2.4.1.1 Pre-processing

The main task of this step is to prepare the signal for pitch class distribution description, enhancing features that are relevant for the analysis. As such it should help fulfil the third requirement mentioned above, i.e. provide robustness against noise.

All approaches found in the literature are based on spectral analysis in the frequency domain. Fujishima [Fujishima 99, Fujishima 00] uses the Discrete Fourier

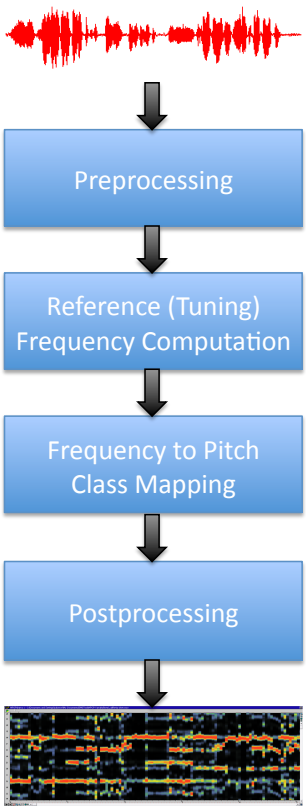

Figure 2.14: General schema for pitch class distribution computation from audio.

Transform (DFT) with a frame size of 2048 samples and a sampling rate of 5.5KHz (i.e. a 400ms frame). The DFT is also used for computing the HPCP as we shall see in section 2.4.2. It is also common to restrict the frequency range for the analysis – different approaches use various ranges (63.5Hz-2032Hz in [Fujishima 99], 25Hz-5000Hz in [Pauws 04], 100Hz-5000Hz in [Gómez 06a], and there are several other variations). As we shall see in chapter 3, limiting the frequency range for the HPCP computation plays an important role in our system.

An alternative to the DFT is the constant-Q transform [Brown 91, Brown 92], used for the *constant-Q profile* [Purwins 00] and *pitch profile* [Zhu 05]. It is beyond the scope of our work to give further details about this approach, and the reader is referred to [Gómez 06a] and the above cited papers for further information.

Finally, there are several other pre-processing steps in addition to frequency analysis utilised by some of the authors. [Fujishima 99] uses non-linear scaling and silence and attack detection to avoid noisy features. [Gómez 06b] uses

transient detection and peak selection for considering only local maxima of the spectrum.

2.4.1.2 Reference Frequency Computation

Whilst A 440Hz is considered as the standard reference frequency for pitch class definition, we cannot assume that bands (or orchestras) will always be tuned to this reference frequency. Though a series of approaches uses a fixed reference frequency ([Fujishima 99, Purwins 00, Pauws 04] and others), there are several approaches which try to take this issue into consideration.

[Fujishima 00] adjusts the PCP values according to the reference frequency *after* the PCP is computed. The technique is based on ring shifting the PCP with a resolution of 1 cent and computing the mean and variance for 12 semitone width segments, where the minimum variance indicates the unique peak position. [Zhu 05] determines the tuning frequency before computing the PCD, and then uses this frequency for the frequency to pitch mapping. The approach is based on statistical analysis of the frequency positions of prominent peaks of the constant-Q transform, and a similar approach is proposed in [Gómez 06a].

2.4.1.3 Frequency Determination and Mapping to Pitch Class

Following the transformation of the signal to the frequency domain and determination of the reference frequency, the next step is to determine the pitch class values.

[Leman 00] and [Tzanetakis 02] take a multipitch estimation oriented approach, applying periodicity analysis to the output of a filter-bank using autocorrelation. They extract a set of K predominant frequencies f_{pk}, where $k = 1 \dots K$, which are used for tonal description. Leman matches these to pitch classes using

$$PCD(n) = \sum_{f_{pk}\ s.t.\ M(f_{pk})=n} 1 \tag{2.19}$$

where $n = 1 \dots 12$ (12 pitch classes), and the function $M(f_{pk})$ maps a frequency value to the PCD index

$$M(f_{pk}) = round(12 \cdot log_2 \left(\frac{f_{pk}}{f_{ref}} \right) mod\ 12) \tag{2.20}$$

where f_{ref} is the reference frequency and goes into PCD(0). Although pitch class C is often assigned to this bin, in our work we will assign pitch A, such that $f_{ref} = 440Hz$ for a piece tuned to this frequency.

[Fujishima 99] considers all frequencies of the DFT rather than just predominant ones. The weight given to each frequency is determined by the square of its spectral amplitude:

$$PCD(n) = \sum_{i\ s.t.\ M(i)=n} |X_N(i)|^2 \tag{2.21}$$

where $n = 1 \dots 12$ and $i = 0 \dots N/2$ and N is the size of the DFT. $M(i)$ maps a spectrum bin index to the PCP index:

$$M(i) = \begin{cases} -1 & \text{if } i = 0 \\ round(12 \cdot log_2 \left(\frac{f_s \cdot i/N}{f_{ref}} \right) mod\ 12) & \text{if } i = 1, 2, \dots, N/2 \end{cases} \tag{2.22}$$

where f_s is the sampling rate, f_{ref} is the reference frequency that falls into $PCP(0)$, and $f_s \cdot i/N$ is the frequency of the spectrum at bin i. Other approaches (e.g. [Purwins 00]) use the magnitude $|X_N(i)|$ in place of the squared magnitude $|X_N(i)|^2$.

[Gómez 06a] introduces a weighting scheme based on a cosine function as detailed in section 2.4.2. Another important issue is the consideration of harmonics. Several approaches such as [Pauws 04] and [Zhu 05] take harmonics into account in different ways, and we explain the one used for the computation of the HPCP shortly.

2.4.1.4 Interval Resolution

An important aspect of the PCD is the frequency resolution used to describe the pitch classes, the traditional value being a resolution of one semi-tone (i.e. 12 PCD bins each of 100 cents), as used in [Pauws 04]. However, increasing the resolution can help improve robustness against tuning and other frequency oscillations. [Fujishima 99] and [Zhu 05] use 12 bin PCDs, but use greater resolutions (1 and 10 cents respectively) during the first analysis steps. [Purwins 00] and [Gómez 06b] use 36 PCD bins, i.e. a resolution of one third of a semitone. When it comes to melody and bass line extraction, we care about the resolution even more, since the greater the resolution, the more accurate the contour we extract to describe the melody or bass line. As we shall see in chapter 3, we employ a PCD (the HPCP) with 120 pitch class bins – a resolution of one tenth of a semitone.

2.4.1.5 Post-Processing Methods

Similarly to the melody extraction systems we reviewed earlier, PCD computation is also often followed by some post-processing. As one of the requirements for PCDs is robustness to variations in dynamics, [Gómez 06b] normalises each PCD frame by its maximum value. [Leman 00] adds to each feature vector a certain amount of the previous one, whilst [Fujishima 00] proposes a more complex peak enhancement procedure based on summing the correlations between ring-shifted versions of the PCP and the original version. Others propositions include summing over larger time segments and smoothing by averaging.

2.4.2 Harmonic Pitch Class Profile

Following our review of different approaches for the computation of a pitch class distribution, we now provide further details of the approach used in this work, the *Harmonic Pitch Class Profile* (HPCP) introduced by Gómez in [Gómez 06a]. In this section we explain how the HPCP is computed once the signal has already been processed and the tuning frequency determined. For full details of these steps please see the above reference.

The HPCP is based on the Pitch Class Profile (PCP) presented earlier which was proposed by [Fujishima 99]. To reiterate, this vector measures the intensity of each of the *twelve* semitones of the diatonic scale. The HPCP introduces three main modifications to the PCP:

1. Weighting – a weight is introduced into the feature computation.
2. Harmonics – the presence of harmonics is taken into consideration (hence the 'H' in HPCP).
3. Higher resolution – a higher PCD bin resolution is used.

[Gómez 06a] uses a frequency range of 100Hz to 5000Hz, i.e. only considering spectral peaks whose frequency is within this interval. In chapter 3 we shall see how the frequency range under consideration is further adjusted for the task of melody and bass line extraction. The HPCP vector is defined as:

$$HPCP(n) = \sum_{i=1}^{nPeaks} w(n, f_i) \cdot a_i^2 \qquad n = 1 \ldots size \tag{2.23}$$

where a_i and f_i are the linear magnitude and frequency of peak i, $nPeaks$ is the number of spectral peaks under consideration, n is the HPCP bin, $size$ is the

size of the HPCP vector (the number of PCD bins) and $w(n, f_i)$ is the weight of frequency f_i for bin n.

2.4.2.1 Weighting Function

Instead of having each frequency f_i contribute to a single HPCP bin, we define a weighting function $w(n, f_i)$ such that f_i contributes to the HPCP bins contained in a certain window around this frequency. The contribution of peak i is weighted using a $cos()^2$ function centred around the frequency of the corresponding bin. For a given bin n, the weight is adjusted according to the distance between f_i and the centre frequency of the bin f_n:

$$f_n = f_{ref} \cdot 2^{\frac{n}{size}} \qquad n = 1 \ldots size \tag{2.24}$$

The distance d is measured in semitones and given by:

$$d = 12 \cdot \log_2 \left(\frac{f_i}{f_n} \right) + 12 \cdot m \tag{2.25}$$

where m is an integer chosen to minimise $|d|$. Thus, the weight is computed by:

$$w(n, f_i) = \begin{cases} \cos^2 \left(\frac{\pi}{2} \cdot \frac{d}{0.5 \cdot l} \right) & \text{if } |d| \leq 0.5 \cdot l \\ 0 & \text{if } |d| > 0.5 \cdot l \end{cases} \tag{2.26}$$

where l is the length of the weighting window. l is a parameter of the algorithm and [Gómez 06a] empirically sets it to $\frac{4}{3}$ of a semitone. In figure 2.15 we show the weighting function when we use a resolution of $\frac{1}{3}$ of a semitone (36 bins) and $l = \frac{4}{3}$ of a semitone. The red bar indicates one bin in the HPCP, and we see how each spectral peak contributes to four HPCP bins.

2.4.2.2 Consideration of Harmonic Frequencies

The frequency spectrum of a note will contain peaks at several of its harmonics, that is frequencies which are integer multiples of the fundamental frequency $(f, 2f, 3f, 4f, \ldots)$. These harmonics affect the HPCP, and we must assure that harmonics contribute to the pitch class of the fundamental frequency. To do so, [Gómez 06a] proposes a weighting procedure – each spectral peak at frequency f_i contributes to all frequencies for which it is a harmonic frequency $\left(f_i, \frac{f_i}{2}, \frac{f_i}{3}, \frac{f_i}{4}, \ldots, \frac{f_i}{nHarmonics}\right)$, where the contribution decreases according to the curve:

$$w_{harm}(n) = s^{n-1} \tag{2.27}$$

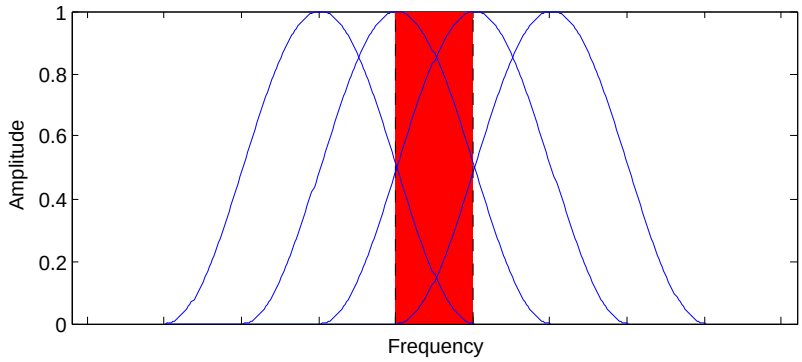

Figure 2.15: Weighting function used in HPCP computation.

where n is the harmonic number and $s < 1$, and set by the author to 0.6[4]. This curve is shown in figure 2.16.

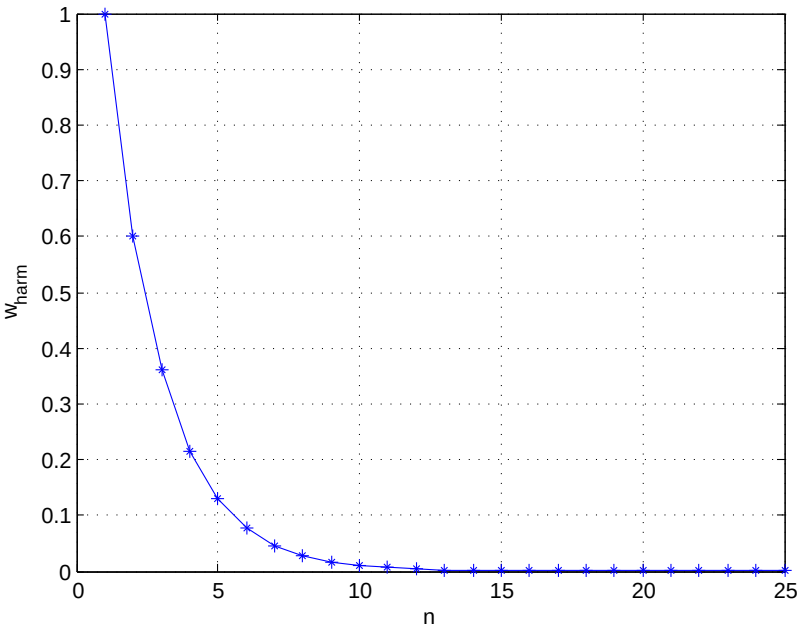

Figure 2.16: Weighting function for harmonic frequencies, $s = 0.6$.

[4]Ideally the value of s should be set according to the timbre of the instrument.

2.4.2.3 Spectral Whitening for HPCP

As previously explained, spectral whitening increases robustness against timbre variations. The process of spectral whitening was explained in detail in section 2.3.2.2. Using timbre normalisation, notes in high octaves will contribute equally to the HPCP vector as notes in a lower pitch range.

2.4.2.4 HPCP Normalisation

The HPCP is computed for each analysis frame of the signal, an its values are normalised with respect to its maximum in the specific analysis frame:

$$HPCP_{normalised}(n) = \frac{HPCP(n)}{\max_n(HPCP(n))} \qquad n = 1 \ldots size \tag{2.28}$$

Together with peak detection, this process makes the HPCP independent of dynamics, overall volume and the present of soft noise. Only spectral peaks beyond a threshold are selected, so that very low energy frames will return a flat HPCP. Normalisation means the peak of every HPCP is set to one, such that amplitude variations in the signal do not affect the HPCP.

2.5 Discussion

In this chapter we provided an extensive scientific background to the work undertaken in the following chapters. We started by reviewing the general architecture used in most melody extraction systems. This architecture will also form a rough schema for our melody and bass line extraction method presented in chapter 3. We then examined two state of the art systems in greater detail. When reviewing the work carried out by [Goto 04b], we noted why a mid-level contour based representation may be desirable, as opposed to full transcription. We adopt the same approach in our system, extracting a mid-level representation. We also examined the use of peak tracking for the purpose of melody and bass line selection out of a set of candidate F0s, and this too will form part of our work. Next we reviewed three different algorithms (or salience functions) for multiple and predominant F0 estimation, presented in [Klapuri 06]. We observed how a simple approach based on the summation of harmonic amplitudes can produce good results for predominant F0 estimation, indicating its potential for melody and bass line extraction. Furthermore, we introduced some important spectral analysis steps such as spectral whitening. Following this review, we can identify several important principles which we shall follow in our selected approach:

- We will extract a mid-level contour based representation.
- Pre-processing steps such as spectral whitening to increase robustness against timbre variation and frequency range splitting for melody and bass line will be employed.
- A simple approach based on summation of harmonic amplitudes has the potential of providing good results for melody and bass line extraction.
- Tracking rules can be employed for the selection of the melody or bass line.

Based on these observations, we identified the HPCP as a technology with the potential of being beneficial for melody and bass line extraction:

- It includes spectral processing steps such as peak selection and spectral whitening which have been shown to be beneficial for this kind of task.
- The HPCP is in essence a pitch-class based salience function. As detailed in chapter 3, we will make the assumption that the tonality of a musical signal in the low frequency region is strongly affected by the bass line, whilst the tonality in the mid to high frequency region is affected by the melody. The frequency range examined by the HPCP can be easily modified, hence allowing us to use different frequency range for melody and bass line.
- The HPCP is based on the summation of harmonic amplitudes, an approach that has been shown to be promising. One important difference from the salience functions presented in [Klapuri 06] is that by definition, the HPCP does not contain any octave information. Whilst in accordance with the notion of a mid-level representation, one of our goals will be to examine how this lack of octave information affects performance.

An overview of pitch class distribution computation was given, and the specific details of the HPCP were explained. In chapter 3, we further elaborate on our selected approach. In chapter 4 we explain our evaluation methodology and the process of preparing music collections for evaluation. This includes the implementation of the three algorithms proposed in [Klapuri 06] and reviewed in section 2.3.2 for the purpose of a comparative evaluation. In chapter 5 we present the specific experiment we have carried out and the results we have obtained. Finally in chapter 6 we conclude our work with a discussion of the results and future directions for the work carried out in this research.

3

Melody and Bass Line Extraction

3.1 Introduction

In this chapter we explain our selected approach in detail. Following our overview of the HPCP, we start by explaining how we adapt the HPCP as presented in [Gómez 06a] for the purpose extracting a mid-level representation of the melody and bass line. As we shall see, this corresponds to steps 1 and 2 from the general melody extraction architecture as explained in section 2.2. Similarly to the approach taken by Goto in [Goto 04b], we do not perform onset detection or voicing detection, and we extract a continuous contour as our representation. Finally, we explain how our method was implemented.

3.2 Chroma Features for Salience Estimation

Following the reasoning given in section 2.5, we propose the use of the *Harmonic Pitch Class Profile* (HPCP) as a salience function for melody and bass line extraction. As previously explained, the HPCP returns a relative (or absolute, depending on whether normalisation is performed) salience value for each pitch class in the analysed segment, which depends on the presence of its harmonic frequencies in the frequency spectrum of the signal. In the following sections we explain how this "salience function" is fine tuned for the purposes of our specific task.

3.2.1 Frequency Filtering

The HPCP as formulated in [Gómez 06a] examines a relatively wide range of the audible spectrum, taking into consideration frequencies between 100Hz and 5000Hz. Following the rationale of Goto, we argue that bass line frequencies will be more predominant in the low frequency range, whilst melody frequencies will

be more predominant in the mid to high frequency rage. Our proposition is thus to limit the frequency band analysed during the HPCP computation, depending on whether we are focusing on the melody or bass line. We adopt the ranges proposed in [Goto 04b] – 32.7Hz (1200 cent) to 261.6Hz (4800 cent) for bass line, and 261.6Hz (4800 cent) up to 5KHz (9907.6 cent) for melody. In chapter 2 a PCP was be visualised by means of a histogram. In order to visualise the evolution of an HPCP over time, we plot it on its side, in what we call a *chromagram*. The x-axis represents time, whilst on the y-axis we indicate the salience of the pitch classes (going full cycle from A through A♯, B, C... back to A) by colour, going from blue (low) to red (high). In figure 3.1 we show the chromagram for a 5 second segment from the song RM-P047 from the RWC popular music collection, using a frequency range of 32.7Hz to 5KHz, and an analysis window of 8192 samples with a sampling rate of 44100Hz.

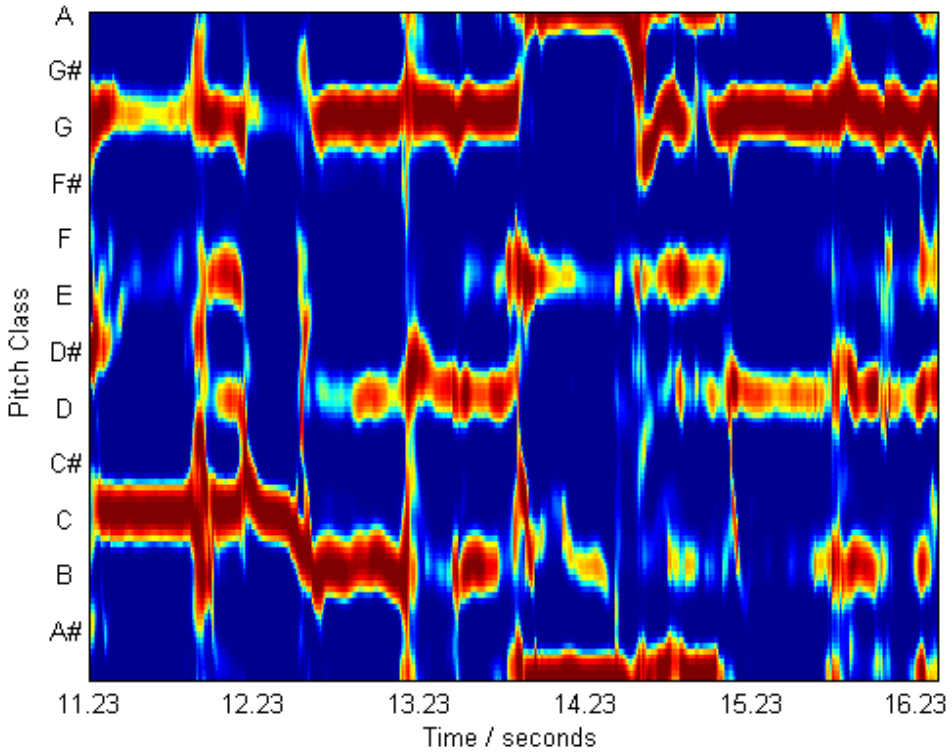

Figure 3.1: Chromagram for 5 second segment from RM-P047.

Now, we perform the same analysis again, only we first limit the frequency range to 261.6Hz-5000Hz for melody and then to 32.7Hz-261.6Hz for bass. In

figure 3.2 we provide three chromagrams – the top chromagram is the same as the one in figure 3.1. The middle one is the chromagram with frequency filtering for melody, and the bottom one is the chromagram with frequency filtering for bass. The middle and bottom chromagrams also have the reference melody/bass line overlayed as white lines (the reference has been mapped from frequency values to HPCP bins).

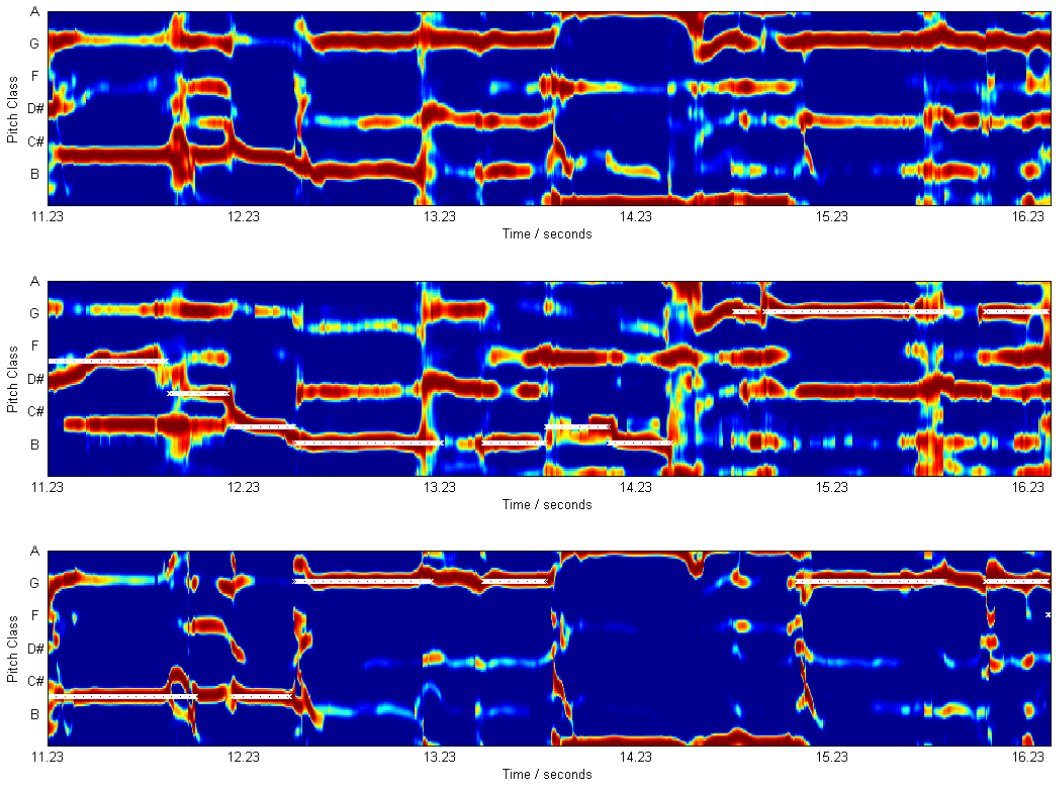

Figure 3.2: Original, melody and bass line chromagrams for RM-P047

As can be seen from figure 3.2, limiting the frequency range for the HPCP computation has a significant effect. We comment that the top pane (containing the HPCP using the entire frequency range) is closely related to the bass line and harmony. Once we apply the "filtering" for melody, we observe that pitch classes previously not at all salient appear, and they are in fact (some but not all) the ones of the melody. We also note however that limiting the frequency region does not entirely "solve" the problem, as there are often several salient pitch classes at each frame, only one of which is the melody pitch class. For bass line the results are even more encouraging, and we can see that every bass line note in

this segment seems to be detected with good accuracy[1] (experimental results are provided in chapter 5).

3.2.2 HPCP Resolution

As explained in section 2.4.1.4, one of the important factors in the computation of the HPCP (or any PCD for that matter) is the resolution, i.e. the number of pitch class bins into which we divide the diatonic scale. Whilst a 12 or 36 bin resolution may suffice for tasks such as key or chord estimation, if we want to properly capture subtleties such as vibrato and glissando, as well as the fine tuning of the singer or instrument, a higher resolution is needed. It is possible that for certain applications such a high resolution will not be needed, but we find it better to start off with a high resolution, which can easily be reduced at a later stage should the need be. In figure 3.3 we show the HPCP for the same 5 second segment of "train05.wav" from the MIREX 2005 collection, taken at a resolution of 12, 36, and 120 bins. We see that as we increase the resolution, elements such as glissando (seconds 1-2) and vibrato (seconds 3-4) become better defined.

[1]The references for the RWC are discretised to the nearest semitone, such that real characteristics of the audio such as glissando or vibrato may look like errors when compared to the reference. More on this in the following chapters.

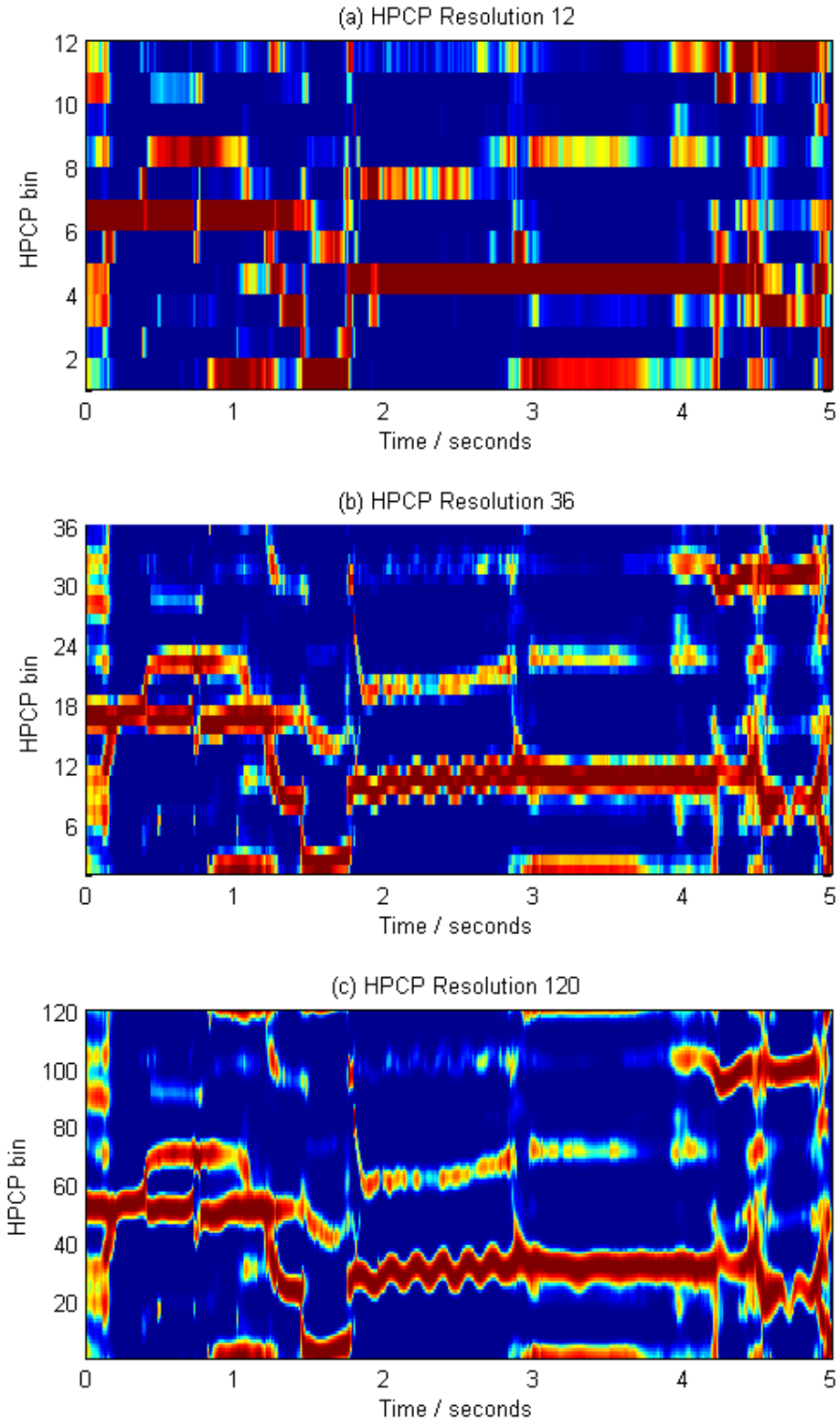

Figure 3.3: HPCP taken at different resolutions.

3.2.3 Window Size

Another important parameter is the window size used in the Fourier Analysis step of the HPCP computation. As mentioned in section 2.3.1.1, there is a trade-off between the time and frequency resolution of the analysis, depending on the window size. In our case however, there is another, related trade-off – using a small window gives us good time resolution, which means we can more accurately track the subtle changes in the melody or bass line. However, we are also more likely to have single frames where the melody or bass line is momentarily not the most (or one of the most) salient lines, resulting in spurious peaks and what we can generally refer to as noise. Following experiments using different window sizes we empirically set the window size to 8192 samples (186ms). In figure 3.4 we present the chromagrams of HPCPs computed for the song train05.wav from the MIREX05 collection with increasing window size. Evaluation results for different window sizes using the RWC Music Database are given in chapter 5.

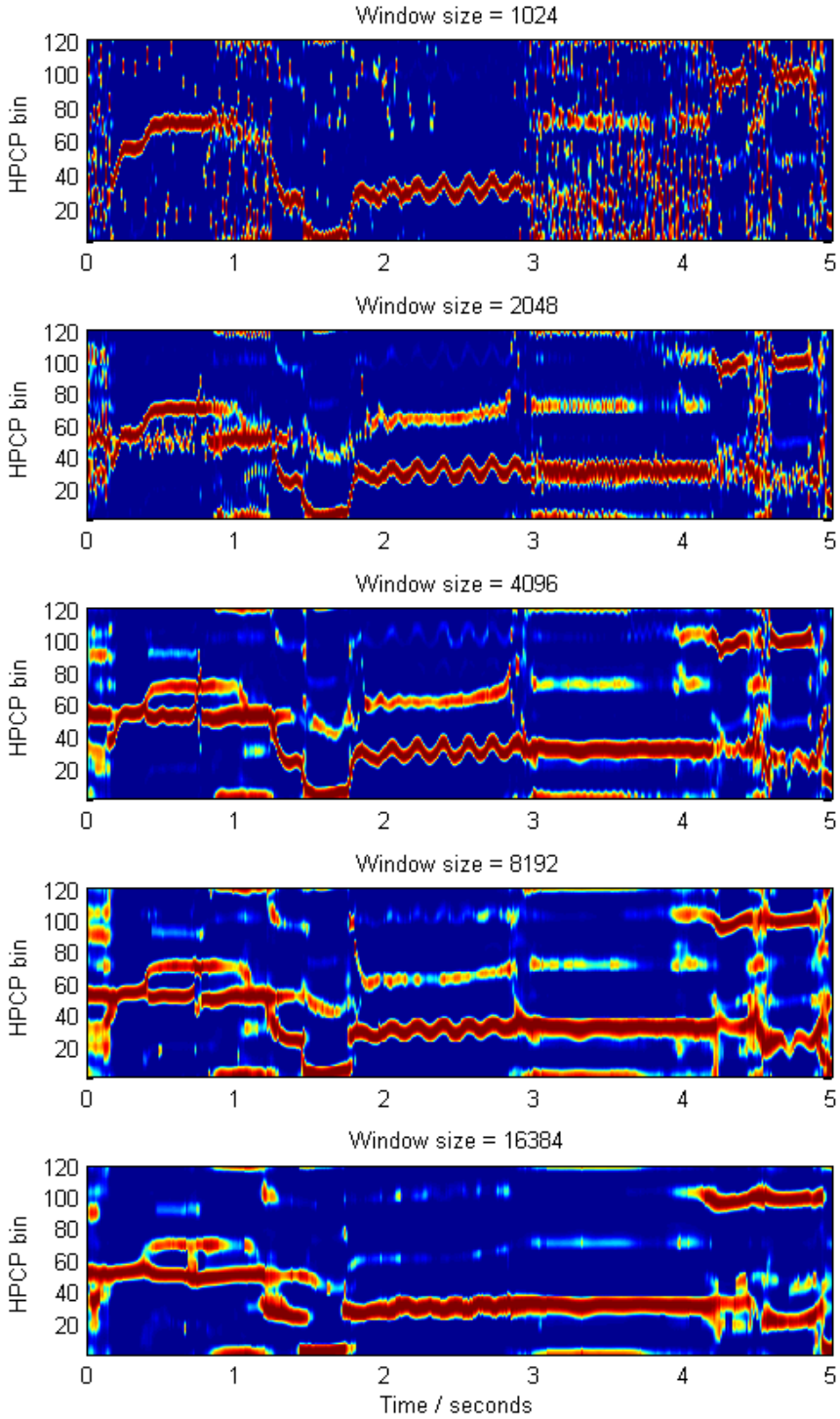

Figure 3.4: HPCP computed with different window sizes.

3.2.4 Normalisation

In the chromagrams shown so far, we have been using the normalisation procedure as explained in [Gómez 06a]. Another option would be to use the non-normalised HPCP, if it was shown that using the absolute HPCP peak values is beneficial for the F0 tracking stage of the system. However, preliminary experiments showed that this was not in fact the case, and so we have not explored this possibility further and for the rest of the work we use normalised HPCPs.

3.3 Peak Tracking

Given the HPCP for every frame of the analysed piece, the final task is to select the correct peak out of the potential candidates in each frame (corresponding the the Post-processing step in the general melody extraction architecture). One important question when considering this task is – how many peaks must we consider in the analysis? Once again we are presented with a trade-off – the more peaks we consider, the greater the likelihood that the true F0 (translated into an HPCP bin) is amongst one of the peaks. However, the more peaks we consider, the more complicated our tracking algorithm must be in order to cope with the increased number of potential candidates. For this reason, we have adopted the following approach – we start working with two peaks. The first thing we do is evaluate the "glass ceiling" for two peaks, that is, what is the best performance possible if we were always to select the correct peak out of the two, in the case that the correct peak is present (presented in chapter 5. We then propose a set of tracking algorithms, and evaluate them with relation to this glass ceiling. Concurrently, we evaluate the glass ceiling for an increasing number of peaks, in order to examine what overall results are obtainable using our approach, and whether it has any inherent limitations.

In the following sections we present two main approaches to HPCP peak tracking. Based on each approach we have written several algorithms with slight variations, resulting in a total of six algorithms.

3.3.1 Proximity-Salience based Tracking

The first set of tracking algorithms is based on two simple assumptions:

1. The melody (or bass line) is more likely to be found in the most salient peak of the HPCP.

2. The melody (or bass line) will tend to have a continuous contour, such that peaks should be "rewarded" for proximity to the previous selected peak.

Based on these assumptions, we have devised a set of tracking algorithms which consider the peak salience and peak proximity as parameters in the selection of the next peak. Before we present the algorithms however, we must first discuss the concept of proximity in the context of the HPCP. When considering a standard sequence of candidate F0s, calculating proximity is fairly straight forward – the distance between two frequencies represented in cents f_1 and f_2 is simply $|f_1 - f_2|$. In the case of the HPCP however, we are dealing with bins (with values between 1 and 120) rather than frequencies. What is more, given two bins b_1 and b_2, we cannot simply compute the value $|b_1 - b_2|$, since the HPCP is *cyclic*. That is, when computing the HPCP we loose the octave information, and thus we need to think in terms of a pitch chroma circle rather that pitch height, as visualised in figure 3.5.

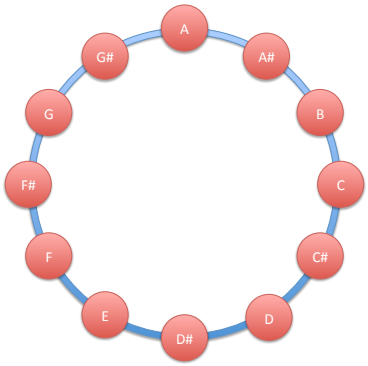

Figure 3.5: Chroma circle.

For this reason, we define the distance between two HPCP bins b_1 and b_2 as the shortest distance between the two bins along the chroma circle, as follows:

$$distHPCP(b_1, b_2) = \begin{cases} \min(b_1 - b_2, b_2 + 120 - b_1) & \text{if } b_1 > b_2 \\ \min(b_2 - b_1, b_1 + 120 - b_2) & \text{otherwise} \end{cases} \quad (3.1)$$

It is important to note that this is a rather different notion of pitch distance (it is more a pitch-class distance), which might make the tracking task more complicated. Another option would be to use a different distance function based on musicological knowledge (for example considering two notes to be closer if they are in the same mode with relation to the current chord), however we have not

Tracking Algorithm 1:

```
1.  for i = 1 to number of frames
2.      //Get the bins and saliences of the two highest peaks of the HPCP
3.      [bin_1 bin_2] = peaks(HPCP)
4.      [salience_1 salience_2] = [HPCP(bin_1) HPCP(bin_2)]
5.      if i == 1
6.          melody(i) = bin_1
7.      else
8.          //Compute the distances of the candidate peaks from the previous peak
9.          dist_1 = distHPCP(bin_1, melody(i − 1))
10.         dist_2 = distHPCP(bin_2, melody(i − 1))
11.         if dist_2 < dist_1 and salience_2\salience_1 > threshold
12.             melody(i) = bin_2
13.         else
14.             melody(i) = bin_1
15.         end
16.     end
17. end
```

experimented with this option. We now present the tracking algorithms based on salience and proximity:

As can be seen, Tracking Algorithm 1 implements a simple heuristic – always select the highest peak of the HPCP, unless the second highest is closer to the previously selected peak and has a salience value which is at least *threshold* of the salience value of the highest peak. We set *threshold* empirically to 0.8. By changing *threshold* we can get two more simple variations on Tracking Algorithm 1 – if we set *threshold* to 1, the algorithm will always select the highest peak of the HPCP. If we set it to 0, it will always take the peak closest to the previous one.

Tracking Algorithm 2 is identical to Tracking Algorithm 1, with the exception of line 11. The condition is now changed to:

11. **if** $(dist_2 < dist_1$ **and** $salience_2 \backslash salience_1 > threshold)$ **or** $(dist_2 < distThreshold)$

The appended **or** condition means we give priority to highly close peaks regardless of their salience, and we set *distThreshold* to 10 (so that the two bins are withing one semitone of each other). The third algorithm attempts to make further use of temporal knowledge. That is, rather than just considering the next peak, it looks further into the future. The function $peakSalience(b)$ evaluates the

salience of peak b by summing the salience of the next 10 closest peaks (starting at peak b). The rationale is that if the peak is closely followed by a continuous sequence of strong peaks, it is a good indication that it is the correct peak to select at the current frame.

Tracking Algorithm 3:

```
1.  for i = 1 to number of frames
2.      //Get the bins and saliences of the two highest peaks of the HPCP
3.      [bin_1 bin_2] = peaks(HPCP)
4.      [salience_1 salience_2] = [peakSalience(bin_1) peakSalience(bin_2)]
5.      if i == 1
6.          melody(i) = bin_1
7.      else
8.          if salience_1 < salience_2
9.              melody(i) = bin_2
10.         else
11.             melody(i) = bin_1
12.         end
13.     end
14. end
```

3.3.2 Note-Segmentation based Tracking

In the previous section we presented three variations on tracking where the two main parameters are the candidate peak's salience and its proximity to the previous peak. The first two make the decision based only on the current frame's peaks, and the third variations tries to look a little further into the future, however the selection is still made on a per-frame basis. In this section we present algorithms based on an approach which further develops the notion of a peak's salience depending on the sequence of peaks it is part of.

The significant step we perform here is what we call "note segmentation". It is important to make clear that we are not in any way attempting to segment the peaks into notes with a single pitch value and start and end time as they would appear in a musical score. Rather, here we define a "note" as a sequence of peaks where the distance between every peak and the previous peak in the "note" is smaller than a certain threshold. The rationale here is to group together peaks which are part of the same note or sequence of notes. In this way, we can compute the salience on a per-segment basis (by summing the salience of its constituent

peaks), and make our selection based on this segment salience. Moreover, once we select a segment, we continue selecting the peaks of the segment until we need to make another decision. This happens either when our segment is over, or when a new segment starts whilst the current segment is still ongoing. The point at which we make this decision may have a significant effect on the results, and thus we have created two algorithms, each using a different heuristic for deciding when to switch between segments.

3.3.2.1 Segment Creation

In this section we quickly explain how the segments are created. As we always consider two simultaneous peaks at every given frame, we use two containers, A and B to contain all note segments. Thus, every container contains for each frame one of the two available peaks, and marker to indicate whether it is the start of a new segment or the continuation of an existing one. Similar to the first three algorithms, we use peak proximity and individual peak salience as parameters in the grouping. Given the two containers A and B, and two candidate peaks p_1 and p_2, the peaks are allocated based on the following heuristics (note – the proximity of a peak to a container is determined by the distance between the peak and the last peak in the container:

- A container is allocated the peak closest to it.
- If one peak is closer to both containers than the other peak is, it is allocated to the container to which it is closest, and the remaining peak is allocated to the other container.
- A peak is considered to be part of the existing note segment in the container to which it is allocated if its distance from the container is less than a set threshold.
- If a container is allocated a peak whose distance is greater than the set threshold, the peak is marked as the start of a new segment.

In essence, the choice of heuristic is determined by four boolean conditions. Given one of the containers, we ask:

- Which peak is closest to this container?
- Is that peak closer to the other container?
- Is the peak we are allocated within the set threshold?

- Is the peak allocated to the other container within the set threshold?

These conditions lay out 16 possible allocation and segment continuation scenarios. Given this segmentation, we now present the algorithms we have created for peak tracking. As they are more elaborate, they are presented in a slightly more abstract pseudo-code fashion:

Tracking Algorithm 4:

```
1.  initiate the frame index to 1
2.  perform the note segment grouping: [A, B] = grouping(HPCPs for all frames)
3.  while index < frames
4.      compute the salience of the segments in A and B from the current index to the end
        of the segment, salience_A and salience_B
5.      compute the distances from the first peak of each segment to the last selected peak
7.      //Check if we have a proximity overruling
6.      if the distance from the closest segment to the previous peak is below distThreshold
7.          select all peaks belonging to that segment
8.          advance index to the frame following the last frame of the segment
9.      //Otherwise select the next segment based on its salience
10.     else
11.         if salience_A > salience_B
12.             select all peaks belonging to the segment in container A
13.             advance index to the frame following the last frame of the segment
14.         else
15.             select all peaks belonging to the segment in container B
16.             advance index to the frame following the last frame of the segment
17.         end
17.     end
18. end
```

Simply put, the algorithm determines the most salient segment at the current position, and selects all of its peaks. It then determines the next most salient segment (note that unless two segments happen to start at the same time, one of the segments will have its beginning cut off) at the current position and repeats. Similarly to Tracking Algorithm 2, it too has a clause such that if a segment is closer than a certain threshold, it is selected even if it is not the most salient one. The fact that we follow a segment all the way to its end may be a disadvantage, in the case where we select a wrong note, and ignore the beginning of a new segment which is actually correct. For this reason we created a variant, Tracking Algorithm 5, where if a new segment starts in the middle of the currently selected

segment and has a greater salience than the current segment, we immediately switch to the new segment.

Tracking Algorithm 5:

1. initiate the frame *index* to 1
2. perform the note segment grouping: $[A, B] = grouping(HPCPs\ for\ all\ frames)$
3. compute the salience of the segments in A and B from the current index to the end of the segment, $salience_A$ and $salience_B$
4. **while** $index < frames$
5. **if** $salience_A > salience_B$
6. **while** there is no *segmentinterrupt*
7. select the peak belonging to the segment in container A
8. advance *index*
9. **if** $A(index)$ is the start of a new segment, recompute $salience_A$
10. **if** $B(index)$ is the start of a new segment, recompute $salience_B$ **and if** $salience_B > salience_A$, produce a *segmentinterrupt*
11. **end**
12. **else**
13. **while** there is no *segmentinterrupt*
14. select the peak belonging to the segment in container B
15. advance *index*
16. **if** $B(index)$ is the start of a new segment, recompute $salience_B$
17. **if** $A(index)$ is the start of a new segment, recompute $salience_A$ **and if** $salience_A > salience_B$, produce a *segmentinterrupt*
18. **end**
19. **end**
20. **end**

The final variant of this approach, **Tracking Algorithm 6**, makes an attempt to discard note segments that are too short, under the assumption that these segments are too short to be part of the melody or bass line. It is the same as Tracking Algorithm 5, but we now add a further condition to the **if** statements on lines 5 and 12, requiring the segment to be longer than $frameThreshold$ to be considered. If neither of the segments fulfills this requirement at a given frame, the F0 for this frame is set to 0 (denoted by bin 0 for our HPCP representation). $frameThreshold$ was set empirically to 15 frames, i.e. 87ms.

3.3.3 Smoothing

A potential problem with the note segmentation approach we have proposed is the presence of "noise" – one or several frames which break the continuity

Tracking Algorithm 6:

1. initiate the frame *index* to 1
2. perform the note segment grouping: $[A, B] = grouping(HPCPs\ for\ all\ frames)$
3. compute the salience of the segments in A and B from the current index to the end of the segment, $salience_A$ and $salience_B$
4. **while** $index < frames$
5. **if** $salience_A > salience_B$ **and** $frames_A > frameThreshold$
6. ...
12. **elseif** $frames_B > frameThreshold$
13. ...
18. **else**
19. $melody(index) = 0$
20. advance *index*
21. **if** $A(index)$ is the start of a new segment, recompute $salience_A$
22. **if** $B(index)$ is the start of a new segment, recompute $salience_B$
23. **end**
24. **end**

of an otherwise continuous sequences of HPCP peaks. Clearly, this affects the segmentation algorithm, and as a result the peak tracking algorithm. The first thing we have done to minimise this "noise" is the selection of a large window size for the HPCP computation, as mentioned in section 3.2.3. This window effectively acts as a low pass filter on the pitch-class evolution. Nonetheless, in an attempt to reduce any remaining outliers, we propose a smoothing algorithm which is to be executed before the segmentation process.

Using a further low-pass filter would not serve our purpose in this case, as it would cause the "smudging" of note transitions, i.e. creating a continuous transition between different notes of the melody or the bass line where they should not exist. Rather, we define a selective filter which is only applied under certain conditions. Given a frame f_i with a peak at bin b_i, we alter the bin value if:

- at *lookAhead* frames into the future, the next *size* frames (starting at $f_{i+lookAhead}$) contain bins with a distance smaller than *smoothThreshold* between them.
- at *lookAhead* frames into the past, the last *size* frames (ending at $f_{i-lookAhead}$) contain bins with a distance smaller than *smoothThreshold* between them.

- The distances between $b_{i-lookAhead}$ and $b_{i+lookAhead}$ is less than $smoothThreshold$.
- the distance between b_i and $b_{i-lookAhead}$ and the distnace between b_i and $b_{i+lookAhead}$ are both greater than $smoothThreshold$.

If these conditions are met, b_i is changed to the average of $b_{i-lookAhead}$ and $b_{i+lookAhead}$. Note that we use a mean which takes the cyclic nature of the HPCP into account, such that for example the mean of bins 10 and 110 would be 120. The conditions required for smoothing to occur are visualised in figure 3.6.

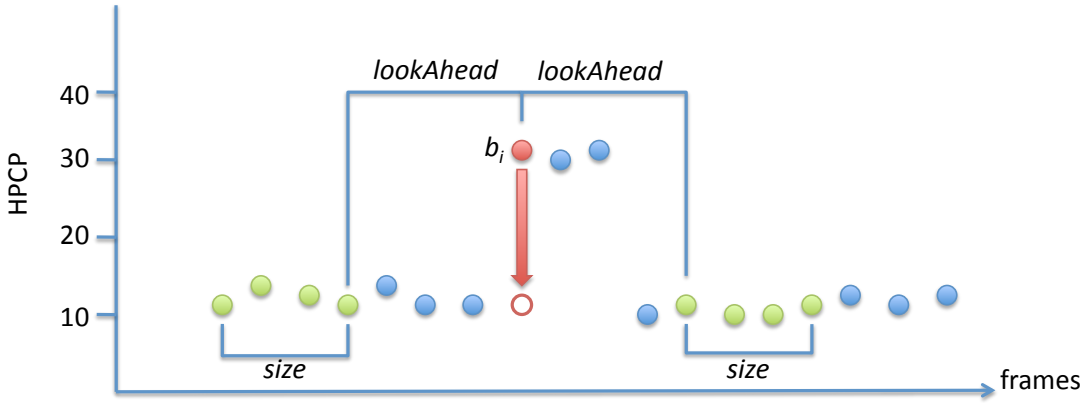

Figure 3.6: The parameters involved in the smoothing process.

As can be inferred from the diagram, the larger the *lookAhead*, the greater the sequence of outliers we can smooth (but we should avoid trying to smooth sequences which are long enough to be actual notes and not outliers). The *size* parameter allows us to set how strict the smoothing condition is – the larger *size* is, the longer the sequence in which the outlier exists must be for the outlier to be smoothed. Finally, we note that in order to perform this smoothing, the HPCP peaks for all frames must be divided into sequences. Clearly if the optimal division into sequences was known, we could divide them into melody and non-melody. As this is not the case, we approximate this by dividing the peaks into two sequences such that the first sequence always contains the highest peak, and the second sequence the other peak present in that frame.

3.3.4 Voicing

In previous sections we mentioned that we do not perform voicing detection in our system. Though we have not included it officially as part of our system, we

have performed some initial experiments with voicing detection. We use a simple approach based on the energy of the current frame – our assumption is that where the melody is present the overall energy will be greater:

$$melody\_present(i) = \begin{cases} \text{true} & \text{if } energy(i) > voicingThreshold \\ \text{false} & \text{otherwise} \end{cases} \quad (3.2)$$

Initial experiments showed that as expected, this threshold must be altered depending on the song. Next, we observed that whilst selecting the threshold manually for a song can produce good results even with this simple approach, selecting it automatically is not a straight forward task. The simplest heuristic one might suggest is the following:

$$voicingThreshold = \overline{energy} - \sigma(energy) * factor \quad (3.3)$$

As we shall see in chapter 5, it seems that successful voicing detection requires a somewhat more elaborate approach, and is suggested as one of the topics for future research in chapter 6.

3.3.5 Conclusion

The results obtained using the various tracking algorithms presented in this section are presented in chapter 5. These algorithms represented two fundamental approaches, each with several slight variations. As mentioned earlier, we use the output of these algorithms as the final output of our system, without any further processing.

3.4 Implementation Details

In the final section of this chapter, we provide a quick description of how we implemented our suggested approach, as well as how we implemented the three algorithms presented in [Klapuri 06] for the purpose of a comparative evaluation.

As detailed in previous sections, our approach is comprised of three main phases – the pre-processing, the computation of the HPCP (our salience function), and the melody or bass line tracking. We can consider the evaluation step as one more phase in the process.

3.4.1 Pre-processing and HPCP Computation

The first two phases of our approach are actually combined into one step. For the computation of the HPCP, we use the implementation available in *Essentia* [Essentia], an in-house library created at the Music Technology Group of the Pompeu Fabra University, which provides a collection of algorithms and descriptors used to extract features from audio files. It is written in C++, and supports a scripting language for setting specific descriptor parameters, such as the ones we have discussed for computing the HPCP. Thus, given an input file, Essentia will perform the pre-processing and the HPCP computation and produce an output file with the values of the 120 HPCP bins for each frame of the analysed signal.

This output is then passed into the second module, which we have written in Matlab. It includes all the tracking algorithms presented in this chapter, as well as the smoothing, voicing detection, note segment grouping and other auxiliary functions mentioned so far. This module takes the HPCPs for all frames of the analysed song as input and produces an output file with two columns – the first contains the time-stamp of each frame, and the second the selected melody or bass line F0 for the given frame. The final output is given in frequencies rather than HPCP bins for the purpose of the evaluation phase as explained in chapter 4. The conversion of HPCP bin b into frequency f is done as follows:

$$f = \begin{cases} 27.5 \cdot 2^{(b/120) \cdot factor} & \text{if } b > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

where $factor = 2, 4, 8, \ldots$ determines the octave into which we transpose the HPCP.

3.4.2 Evaluation

The final phase is the evaluation module. We base our evaluation on the evaluation metrics used in the MIREX 2004 and 2005 competitions, more on which in the following chapter. The evaluation metrics are implemented Matlab and freely available on [ISMIR 04]. We further modified the metrics to match those used in the MIREX2005 competition and so that they can be used to evaluate performance using the RWC database. For the preparation of the music collections for evaluation, we have written an auxiliary tool in Java for converting references provided in MIDI format into the two-columned format presented above.

3.4.3 Implementation of the Algorithms Presented in [Klapuri 06]

A considerable amount of effort was involved in the implementation of the three algorithms presented in [Klapuri 06]. The Direct, Iterative and Joint methods were all implemented from the bottom up, in Matlab. The algorithms take the audio file as input, and produce the values of the salience function for each frame as output, with the exception of the Iterative method which only produces peak values (as it avoids computing the entire salience function). We based our implementations on the information provided in [Klapuri 06]. It is important to note that though we have made every attempt to replicate the exact algorithms as described in the reference paper, some details are not fully specified, and we have had to make some assumptions in order to complete the implementation. Nonetheless, such occasions were seldom enough for us to believe that our implementation is reliable for the purpose of a comparative evaluation. In figures 3.7 and 3.8 we present visualisations of the salience function computed by the Direct method for RM-P047.wav from the RWC Popular Music Database and train05.wav from the MIREX05 collection respectively.

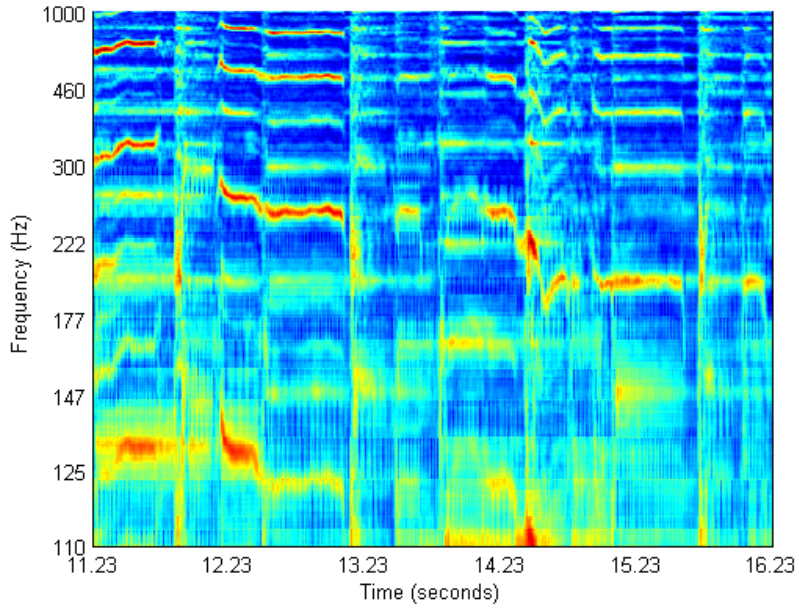

Figure 3.7: Salience for RM-P047.wav computed by the Direct method.

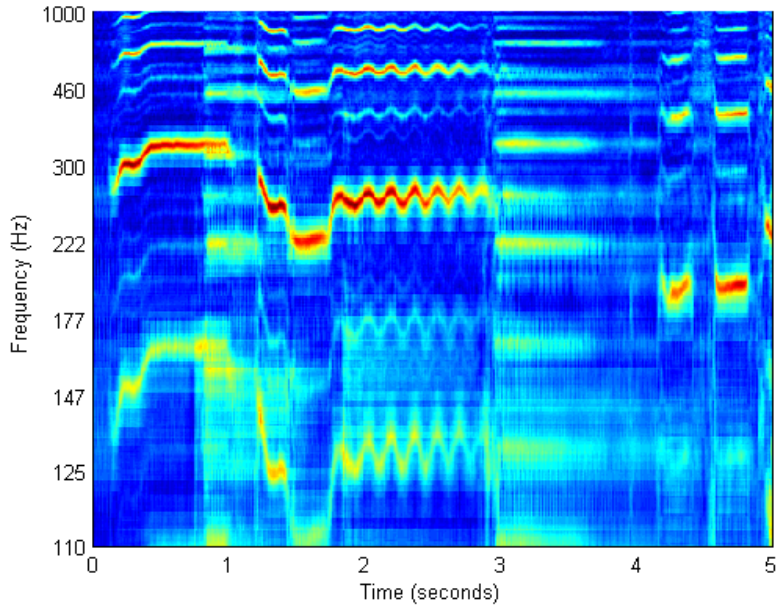

Figure 3.8: Salience for train05.wav computed by the Direct method.

4
Evaluation Methodology

In this chapter we discuss the matter of evaluation, which forms an important part of our work and of the field of melody and bass line extraction in general. We start by providing some background to the task of evaluating melody and bass line extraction, and review the efforts made by the research community in this area so far. Next, we describe the evaluation methodology used in our research. This includes the selection of music collections for evaluation, the preparation of the reference annotations (ground truth), and the evaluation metrics used. The results of the evaluation are presented in chapter 5.

4.1 Introduction

Until recently, a number of obstacles have impeded the objective comparison of melody extraction systems, such as the lack of a standardised test set or consensus regarding evaluation metrics. The problem is evident from the variety of music collections and metrics mentioned in different papers published in the field, including some of the papers we presented in chapter 2.

In 2004, the Music Technology Group (MTG) at the Pompeu Fabra University proposed and hosted a number of audio description contests in conjunction with the International Conference on Music Information Retrieval (ISMIR). These evaluations which included contests for melody extraction, genre classification/artist identification, tempo induction, and rhythm classification, evolved into the Music Information Retrieval Evaluation Exchange (MIREX) [Downie 05] which took place during the summer of 2005, organised and run by Columbia University and the University of Illinois at Urbana-Champaign. The MIREX competitions continue to have an important role in the field, and continue to take place annually.

4.2 Music Collections

Although a great deal of music is available in digital format, the number of corresponding transcriptions time-aligned with the audio is rather limited. In this section we present three important music collections which have already been used for melody and bass line extraction evaluation, and are used for the evaluation of our system.

4.2.1 The Real World Computing Music Collection

In an attempt to address the lack of standard evaluation material, Goto et al. prepared the *Real world Computing* (RWC) Music Database [Goto 02]. The initial collection contained 215 songs in four databases: Popular Music (100 pieces), Royalty-Free Music (15 pieces), Classical Music (50 pieces) and Jazz Music (50 pieces). The current version contains an additional Music Genre Database (100 pieces) and a Musical Instrument Sound Database (50 instruments) [Goto 04a]. All 315 musical pieces in the database have been originally recorded, and the database is available for researchers around the world at a cost covering duplication and shipping expenses. For the purpose of our evaluation we have used the Popular Music Database. The database consists of 100 songs - 20 songs with English lyrics performed in the style of popular music typical of songs on the American hit charts in the 1980s, and 80 songs with Japanese lyrics performed in the style of modern Japanese popular music typical of songs on the Japanese hit charts in the 1990s. Important to us, it is the only collection out of the ones mentioned in this chapter which has transcriptions for the bass line as well as melody.

For every piece in the database the authors have prepared a transcription in the form of a Standard MIDI File (SMF) [MIDI], containing the parts of all instruments and voices in the piece. Most of the pieces were transcribed by ear given the audio signal. The files are stored in SMF format 1 (multiple tracks) and conform to the GS format. As such, a conversion process is needed for converting the relevant track in the MIDI file (originally containing the tracks of all instruments) into the two-columned time-stamp F0 format mentioned in section 3.4 and further explained in section 4.3. The conversion process must ensure that the reference F0 sequence is synchronised with the audio, which is not trivial. As seen in section 4.5, we were able to synchronise 73 files out of the existing 100 in the Popular Music database. Of these, 7 files did not have a bass line, leaving us with 73 files for melody 66 files for bass line evaluation.

Finally, it is important to note that as the transcriptions are provided in SMF, the pitch values in the transcription are discretised to the nearest semitone. Any metric used to evaluate performance on the RWC database must take this into account, as further explained in section 4.5.

4.2.2 The MIREX 2004 and 2005 Collections

Whilst the RWC database *can* be used for melody (and bass line) extraction evaluation, discretising audio to the nearest semitone results in the omission of a significant amount of expressive detail (such as vibrato and glissando). Thus, the organisers of the MIREX competitions opted to create novel sets of recording-transcription pairs. Twenty such pairs were created for the MIREX 2004 competition. By using songs for which the original tracks were available, they were able to use existing monophonic pitch tracking tools such as SMSTools [Cano 98] to estimate the fundamental frequency of the isolated, monophonic melody track. The transcriptions were created in the aforementioned two-column time-stamp F0 format, where the time-stamps increase in 5.8ms steps. As a convention, frames in which the melody is unvoiced are labeled 0Hz. As a final step the transcriptions were manually verified and corrected. For the 2005 competition, 25 new recording-transcription pairs were prepared, although only 13 of these are readily available (those which were released for system tuning prior to the evaluation), as the rest are still used for evaluation in competitions. The ground truth melody transcriptions for the 2005 set were generated at 10ms steps using the ESPS get_f0 method implemented in WaveSurfer [Sjölander 00], and manually verified and corrected. Tables 4.1 and 4.2 provide a summary of the test data used in each competition.

Category	**Style**	**Melody Instrument**
Daisy	Pop	Synthesised voice
Jazz	Jazz	Saxophone
Midi	Folk (2), Pop (2)	Midi instruments
Opera	Classical Opera	Male voice (2), Female voice (2)
Pop	Pop	Male Voice

Table 4.1: Summary of data used in the 2004 melody extraction evaluation.

We note that the 2005 test set is more biased towards pop-based corpora as opposed to the 2004 set which is fairly balanced. The shift was motivated by the relevance of the genre to commercial applications, and the availability of multi-

Melody Instrument	Style
Human voice (8 f, 8 m)	R&B (6), Rock (5), Dance/Pop (4), Jazz(1)
Saxophone (3)	Jazz
Guitar (3)	Rock guitar solo
Synthesised Piano (3)	Classical

Table 4.2: Summary of data used in the 2005 melody extraction evaluation.

track recording. The 2005 test set is more representative of real-world recordings, and as such it is also more complex than the 2004 collection.

4.3 Evaluation Metrics

In this section we review the metrics used to evaluate our work. Two sets of metrics are presented – the ones used for the MIREX 2004 evaluation, the ones used for the MIREX 2005 evaluation which we also use for evaluation with the RWC database. By using a the metric originally used to evaluate each of the MIREX collections, we are able to compare our results with those obtained in the two competitions, and guarantee that our results with the RWC database can be compared with future work.

One issue which is of great importance is that by using the HPCP, our extracted melodies and bass lines do not contain octave information. In each of the metric sets presented below, there are two versions, one which considers octave errors as a mistake, and one which ignores octave errors (which we label the "chroma" metric). Only results obtained using the chroma metric can be compared with our work, as the metric taking octave information into account is not applicable to our mid-level representation. For completeness, we present both versions in the following sections, however in chapter 5 we will focus on the octave agnostic version only.

4.3.1 MIREX 2004 Metrics

The 2004 evaluation included two metrics. The first computes the raw transcription concordance, whilst the second computes the chroma transcription concordance, that is, both the reference and the output of the algorithm are mapped onto one octave before the comparison is made. The final result in the 2004 competition was the average of these two metrics. The metrics were computed

for both voiced and unvoiced frames, such that voicing detection was implicitly taken into account in the final score.

The raw transcription concordance is a frame-based comparison of the estimated fundamental frequency to the reference fundamental frequency on a logarithmic scale. Both the estimated and reference fundamental frequency are converted to the cent scale using equation 2.8 shown in section 2.3.1.1. For a given frame n, the frame concordance error is measured by the absolute difference between the estimated and reference pitch value:

$$err_n = \begin{cases} 100 & \text{if } |f_{cent}^{est}[n] - f_{cent}^{ref}[n]| \geq 100 \\ |f_{cent}^{est}[n] - f_{cent}^{ref}[n]| & \text{otherwise} \end{cases} \quad (4.1)$$

The overall transcription concordance for a segment of N frames is given by the average concordance over all frames:

$$score = 100 - \frac{1}{N}\sum_{n=1}^{N} err_n \quad (4.2)$$

Since octave transpositions and other errors in which the frequency of the estimated pitch is an integer multiple of the reference frequency were frequent, a second metric, the chroma transcription concordance, ignores octave errors by folding both estimated and reference transcriptions into a single octave of 12 semitones (maintaining a resolution of one cent) before performing the calculation as presented in equations 4.1 and 4.2. The mapping onto one octave is performed as follows:

$$fchroma_{cent} = 100 + mod(f_{cent}, 1200) \quad (4.3)$$

It is fortunate that it was decided to compute this second metric as well as the first, as it is the only out of the two which is applicable to our approach, and allows the comparison of our results with those obtained in the competition. It is also important to note, that the metrics above penalise for every cent of error, a condition which was relaxed for the 2005 evaluation and must be relaxed for the evaluation of the RWC where the reference is discretised to the nearest semitone.

4.3.2 MIREX 2005 Metrics and RWC Metrics

For the 2005 competition, two main changes were made to the evaluation metrics. Firstly, the tasks of pitch estimation and voicing detection were explicitly calculated separately, unlike the 2004 results in which voicing detection evaluation is implicit in the calculation. Nonetheless, the final score is still a combination of

the voicing detection and pitch estimation performance. In the process of this evaluation, six metrics were computed:

- **Overall Accuracy** – the proportion of frames labelled correctly with both raw pitch accuracy and voicing detection.
- **Raw Pitch Accuracy** – the proportion of voiced frames in the estimated transcription which are correctly labelled, out of the total number of voiced frames in the reference transcription.
- **Raw Chroma Accuracy** – the same as raw pitch accuracy but mapping both estimated and reference frequencies onto a single octave.
- **Voicing Detection Rate** – the proportion of frames labelled as voiced in the reference transcription that are also labelled as voiced in the estimated algorithm out of the total number of frames labelled as voiced in the reference transcription.
- **Voicing False Alarm Rate** – the proportion of frames labelled as unvoiced in the reference which are labelled as voiced in the estimated transcription, out of the total number of unvoiced frames in the reference transcription.
- **Discriminability** – the voicing detection rate can be increased by biasing the algorithm towards labelling every frame as voiced. However, this in return increases the false alarm rate. The discriminability d' is a metric which evaluates the ability of the algorithm to obtain a good voicing detection rate whilst maintaining a low rate of false alarms.

The second major difference is the way in which the pitch and chroma accuracy are computed. Whilst the 2004 metrics penalised deviations as small as one cent from the reference frequency, the new metrics consider the estimated frequency to be correct if it is within $\pm\frac{1}{4}$ tone ($\pm$50 cents) of the reference frequency. In this way the algorithms are not penalised for small variations in the reference frequency.

This modification becomes absolutely necessary when using the RWC for evaluation. This is because, as previously mentioned, the RWC references are in Midi format and so the pitches are discretised to the nearest semitone. As such, it makes no sense to penalise for deviations from the reference which are smaller than $\frac{1}{4}$ tone, since even a perfect transcription would have such deviations whenever for example the melody is sung with vibrato, or is not perfectly in tune. The raw pitch accuracy can be computed using a modification of equation 4.1:

$$err_n = \begin{cases} 100 & \text{if } |f_{cent}^{est}[n] - f_{cent}^{ref}[n]| > 50 \\ 0 & \text{otherwise} \end{cases} \quad (4.4)$$

4.4 MIREX 2004 and 2005 Evaluation Results

Following the presentation of the test data and metrics used in the MIREX 2004 and 2005 melody extraction evaluations, in this section we briefly present the results obtained by the participant algorithms.

We present the results for the 2004 evaluation without providing further background on the participating algorithms. For the purpose of comparing the results with those obtained by our system, we show the results obtained for voiced frames only (as voicing detection is not included in our system). Details about the participating algorithms and the full set of evaluation results are available in [Gomez 06c]. The pitch estimation results for both raw pitch and chroma metrics are shown in table 4.3, and we have highlighted the chroma metric results as they are of greater interest to us. The participant IDs are taken from the reference paper.

Participant ID	**Paiva (1)**	**Tappert and Batke (2)**	**Poliner and Ellis (3)**	**Bello (4)**
Raw Pitch Accuracy (voiced only)	75.25%	39.73%	50.95%	48.99%
Chroma Accuracy (voice only)	**75.83%**	**56.11%**	**52.12%**	**56.89%**

Table 4.3: Results for the 2004 melody extraction evaluation, voiced frames only.

A thorough analysis of the results can be found in [Gomez 06c]. Next, we present the results obtained for the 2005 evaluation. The participant algorithms of the 2005 evaluation were reviewed in section 2.2, and further information is available in the referenced papers. The results for all six metrics detailed in section 4.3.2 are presented in table 4.4, with the chroma accuracy for voiced frames only highlighted in bold.

The algorithms marked by an asterisk return a continuous sequence of F0 estimates for every frame, and do not perform voicing detection. The overall winner of the competition was the algorithm by Dressler [Dressler 05], and as we can see from the table this is primarily due to its good performance on voicing detection (it has the highest d' value). We see that the results for raw chroma

Rank	Participant	Overall Accuracy	Raw Pitch	Raw Chroma	Voicing Detection	Voicing FA	Voicing d'
1	Dressler	71.4%	68.1%	**71.4%**	81.8%	17.3%	1.85
2	Ryynänen	64.3%	68.6%	**74.1%**	90.3%	39.5%	1.56
3	Poliner	61.1%	67.3%	**73.4%**	91.6%	42.7%	1.56
3	Paiva 2	61.1%	58.5%	**62.0%**	68.8%	23.2%	1.22
5	Marolt	59.5%	60.1%	**67.1%**	72.7%	32.4%	1.06
6	Paiva 1	57.8%	62.7%	**66.7%**	83.4%	55.8%	0.83
7	Goto *	49.9%	65.8%	**71.8%**	99.9%	99.9%	0.59
8	Vincent 1 *	47.9%	59.8%	**67.6%**	96.1%	93.7%	0.23
9	Vincent 2 *	46.4%	59.6%	**71.1%**	99.6%	96.4%	0.86

Table 4.4: Results for the 2005 melody extraction evaluation.

accuracy lie between 60% and 75% roughly, with the majority centered around 70%.

4.5 Data Preparation

In this section we describe how the music collections and corresponding reference files were prepared for the evaluation process. Little preparation was necessary for the MIREX 2004 and 2005 data sets and reference files, as the metrics evaluation code we use is based on that originally used in the MIREX04 competition. As such, we were able to use the reference files directly without any further conversion. It is important to note once more that only about half of the MIREX 2005 dataset used in the competition is available to researchers, thus our results are not directly comparable to those given in table 4.4, though we should still be able to make general observations.

Unlike the MIREX datasets, the RWC database on the other hand required much preparation and pre-processing before we could use it for evaluation. In the following sections we describe the steps we performed in the preparation of the RWC Popular Music Database for evaluation.

4.5.1 Alignment Verification and Offsetting

Unlike the MIREX reference files which were produced by analysing the audio directly using pitch tracking tools, the majority of the RWC files were manually

transcribed by ear into SMF (MIDI). This process presents several potential problems:

1. Initial offset – there may be an initial offset between the start time of the audio and that of the reference, caused by different lengths of silence at the beginning of the recording and the MIDI file.

2. Tempo alignment – the majority of reference MIDI files in the database use a fixed tempo value[1]. If the transcription tempo is slightly different from the audio tempo, or if the tempo of the audio varies in a way which can not be expressed in the MIDI transcription, there will be a misalignment between the audio and reference which can not be compensated for by simple shifting.

3. Note modelling – the task of segmenting a sung word into discrete notes does not necessarily have one solution, and may introduce artificial breaks between notes in the transcription.

4. Octave errors – when transcribing sung voice into MIDI, a certain MIDI instrument must be selected to represent the voice. The difference in timbre, combined with the fact that octave perception is subjective, may result in octave errors in the transcription.

We assume that problems (3) and (4) have been addressed as best as possible through the careful preparation of the database by its authors. In our case the matter of octave errors would not pose a problem even if it were present, as we use the chroma metric which is octave agnostic. We do however have to deal with problems (1) and (2).

In order to be able to easily compare the audio and references, we have synthesised the SMF reference files into audio signals sampled at 44.1KHz, using the freely available iTunes software by Apple. The problem of initial offset is solved by shifting all the notes of the reference transcription so that the starting time of the first note matches that of the first "note" in the audio. This times offset was found by searching for the first audio sample whose amplitude is greater than a threshold (set to 0.01) in both the recording and the synthesised MIDI, and measuring the difference.

Ensuring there is no tempo variation between the recording and reference files requires more thought. The most convenient way of doing this which does not involve manually checking every single note of every song is to use an alignment

[1]Effort was made by the authors to include tempo changes in the MIDI files where necessary, for example at the end of a song if it slows down for the finale.

algorithm to compare the recording and reference file. As before, we start by synthesising the MIDI file into audio. Next, we compute the HPCP descriptor for both recording and synthesised audio, using a resolution of 12 bins, a frequency range of 40Hz to 5000Hz, a window size of 4096 samples and a hop size of 256 samples. This provides us with a description of the tonality of both audio files reliable enough for alignment. We use a local alignment algorithm for HPCPs courtesy of Joan Serrà, as explained in [Serrà 07b]. A good overview of local alignment as well as other string based alignment algorithms can be found in [Navarro 02]. In order to perform the local alignment computation in reasonable time (and due to memory limitations), we average the HPCPs of every 16 frames. If we denote the HPCP for a single frame i as a 12 dimensional vector $\vec{v}_i$, then the average of $\vec{v}_1, \ldots, \vec{v}_{16}$ is given by:

$$\vec{v}_{avg} = \frac{\sum_{i=1}^{16} \vec{v}_i}{\max\left(\sum_{i=1}^{16} \vec{v}_i\right)} \tag{4.5}$$

which ensures the resulting averaged HPCP is still normalised to values between 0 and 1. Once we have the HPCP sequence for both files, we perform the alignment, as seen in figure 4.1.

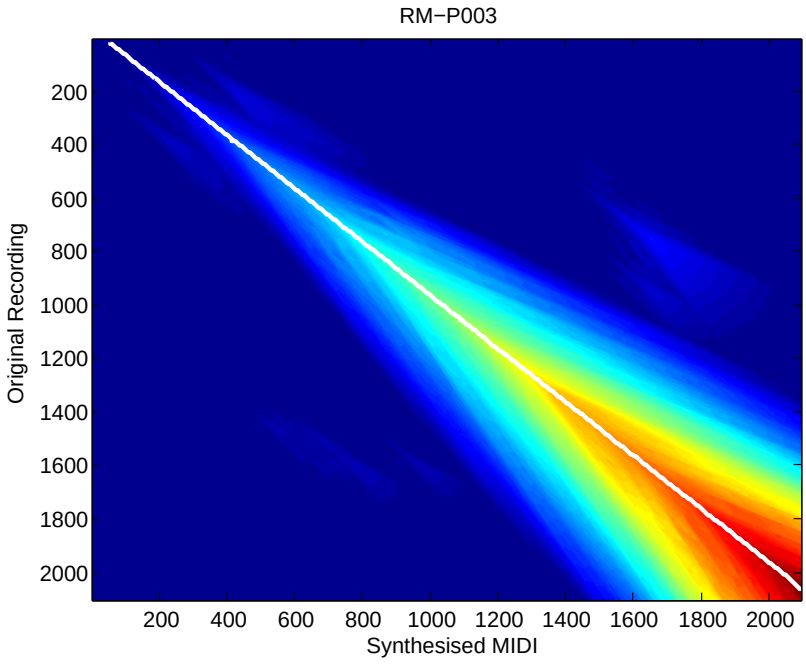

Figure 4.1: Alignment of RWC recording RM-P003 to the synthesised reference.

The plot represents the scores of the alignment matrix, going top-down left to right. We place the original recording on the Y-axis and the synthesised MIDI on the X-axis, and use colour to indicate the best alignment score for the given position, going from blue (for a score of 0) to red (the maximum score in the matrix). The best alignment path is indicated by the white line. For a perfect alignment, we would expect the the white line to have the following properties:

- It starts at the top left corner and ends at the bottom right, indicating that both tracks have equal initial and final silence length (ideally zero).
- It has a slope of -45 degrees, indicating that both files are at the exact same tempo.
- The line is perfectly straight, indicating every part of the reference can be matched against a corresponding part in the original recording, without any skipping or time bending.

In figure 4.1 we provide a clear example of an alignment which is overall successful, however an initial segment of the synthesised MIDI is skipped, indicating that the reference will have to be shifted in order to match the timing of original recording. In figure 4.2 we provide an example in which the timing of the notes does not match perfectly between the original and the reference, resulting in a "wiggly" curve.

The following procedure was followed for each of the 100 songs in the RWC Popular Music Database:

- The files were aligned using the procedure described above.
- The slope of the alignment curve was calculated.
- The curve was examined for irregularities.
- The initial shift required for the timing of both files to match was calculated.

For suspicious alignment curves, the following further procedure was carried out:

- The original recording was run through a monophonic pitch detection algorithm provided by Essentia. The output was plotted against the reference MIDI file, which was converted into the two-column time-stamp F0 format using the tool described in the section 4.5.2. Though the output of the pitch estimator is very noisy (as it is not intended for polyphonic signals), it normally contains several places throughout the song where melody note

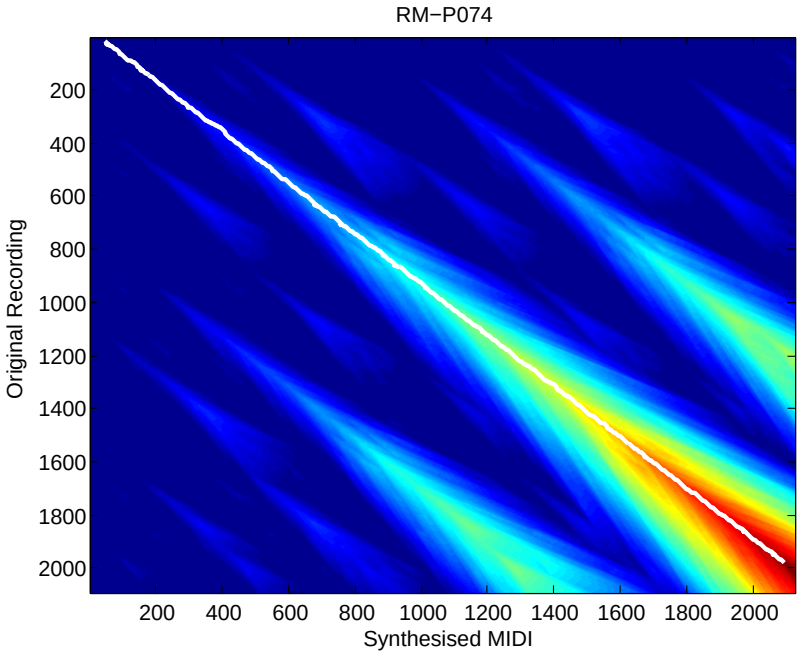

Figure 4.2: Alignment of RWC recording RM-P074 with the synthesised reference.

boundaries can be clearly identified. These places were used as reference points to match against the MIDI transcription.

- In some cases the alignment was confirmed to be successful, with the irregularities in the alignment curve resulting from a-tonal segments in the song.
- In other cases, the initial shift had to be manually adjusted, but there were otherwise no problems.
- In the final case, a slight tempo difference between the reference and the recording was identified, such that there was no initial shift value for which all notes were well aligned. These files had to be discarded.

In addition, we report one case in which we had problems synthesising the MIDI file, and one case in which the melody was divided between two tracks in the reference and discarded for this reason. All in all we were able to synchronise 73 files, which are listed in appendix A. Out of these 73 files, 7 songs did not have a proper bass line, leaving us with 66 files used for the evaluation of bass line extraction, also listed in appendix A.

4.5.2 Format Conversion

In this section we briefly describe the conversion of the SMF reference files available with the RWC Music Collection into the two-column time-stamp F0 format. As previously mentioned, for every audio piece in the database there is a reference transcription provided in SMF. This MIDI file contains the parts of all instruments (including voices and sound effects) in the piece, each in an individual track. From this MIDI file, we need to extract either the melody or bass line track, and convert it to the format we use for our evaluation. This process involves the following steps:

- Melody/bass line track identification
- Tempo calculation
- Reference generation

The first step was performed manually as described below. The second and third are both computed by an auxiliary tool we have written in Java. Given a MIDI file and the desired track number, the program reads the MIDI byte code and outputs the reference file in the desired format, as detailed in sections that follow.

4.5.2.1 Track Identification

Due to the variety of styles and arrangements in the RWC Popular Music Database, many of the reference files use a different set of instruments. This means we must examine every file to ensure that it indeed has a track labelled as the melody, and a bass line track. Unfortunately, it is also the case that the RWC database does not use a consistent track number for placing the melody and bass line, which means we must also check for each song in which tracks are the melody and bass line placed. This was performed manually by opening all the reference SMF files in Cubase and observing the relevant track numbers. Tracks which did not have a bass line were discarded.

4.5.2.2 Introduction to MIDI and the SMF

The *Musical Instrument Digital Interface* (MIDI) specification defines a message format (the "Midi Protocol") for transferring musical data between electronic devices. For example, to sound a note on a midi device you send a "Note On" message, which specifies a key (pitch) value and a velocity (intensity) value. The protocol includes messages for turning a note on or off, changing instrument

etc. The midi specification also defines a set of messages not directly related to the production of sound, such as Meta Messages for transmitting text strings (e.g. lyrics, copyright notice, tempo changes) and System Exclusive messages for transmitting manufacturer specific instructions.

The Standard Midi File Format is a storage format in which every message is combined with a time-stamp to form a "midi event", so that messages can be recalled and replayed in the correct order at a later date. A midi time-stamp is specified in "ticks" can be converted into an actual time value. Further details about the MIDI protocol and the SMF are available at [MIDI].

4.5.2.3 Tempo Calculation

The SMF supports several ways of defining the tempo of a song. In the case of the RWC, all files use the same method – they define a number of ticks per beat (where a beat normally corresponds to one quarter note), and the duration of a single beat is specified in microseconds in track 0 of the SMF. The initial tempo, and any tempo changes throughout the song are performed through a MetaMessage on track 0 which specifies a new beat duration. In order to properly transcribe the MIDI file, we must track all tempo changes. The first step of the process is thus scanning through all messages of track 0, making a list of all tempo changes and their tick time-stamp.

4.5.2.4 Reference Generation

The reference is generated a by process which simulates the "analysis" of the MIDI track as if it were an audio file sampled at 44.1KHz with a 256 sample hope size. The current frequency is set to 0Hz and frame counter to 1. The process reads all the messages on the track sequentially, and performs the following steps for each new event:

- Compute the time-stamp of the current event by summing the durations of all ticks up to the current event's tick, making sure to factor in tempo changes up to the current tick.
- Next, if the event is a Note On event, we must "fill" the frames up to the event's time-stamp – the time-stamp for the current frame is given by $time = frame * hop / f_s$. We store this time stamp together with the current frequency and increase the frame counter, and repeat this as long as the

computed time-stamp is smaller than the event's time-stamp. Finally, we update the current frequency to that specified by the Note On event[2].

- If the event is a Note Off event, we first check whether the event's frequency matches the last frequency specified by a Note On event. If the frequencies match, we fill the frames up to the current event's time-stamp with the current frequency as explained above, and finally set the current frequency to 0. If however the frequencies do no match, it means that we have detected a note overlap in the reference transcription. That is, a new note has started before the previous has ended. In this case, our policy is to cut the current note short and start the new one, meaning the Note Off event is no longer relevant (as it refers to a note we have already cut), and is discarded.

At the end of this process we have a two columned list, the left column specifying the frame time-stamps, and the right column the frequency value for the given frame. This list is saved to a text file which is then used for the evaluation.

4.6 Conclusion

In this chapter we reviewed the various aspects relevant to the evaluation of our work. We presented the music collections used in our evaluation, together with the metrics used in conjunction with each collection. We then presented the relevant results obtained by the competing systems in the MIREX 2004 and 2005 competitions. Finally, we detailed the steps involved in the preparation of the RWC reference files for evaluation. In the following chapter we present the evaluation results for our approach, using the various algorithms presented in chapter 3, evaluated on the three aforementioned music collections. In parallel, we compute the results for our implementation of the algorithms presented in [Klapuri 06] and compare them to those obtained by our system.

[2]The message will specify a MIDI *note* number, which we convert to a frequency value using $f = 440 * 2^{(note-69)/12}$

5 Results

5.1 Introduction

In this chapter we present the evaluation results for our melody and bass line extraction approach using chroma features, as presented in chapter 3. In parallel, we give the results for our implementation of the algorithms presented in [Klapuri 06]. The results presented in this chapter are those obtained for voiced frames only, as voicing detection is not part of our system and we wish to evaluate how well our algorithm succeeds in detecting the correct melody or bass line pitch class when the melody or bass line is present. The chapter is divided into three main sections, reflecting three different parts of the evaluation.

In section 5.2, we evaluate our method as a raw salience function. That is, we examine the performance of the HPCP for melody and bass line extraction when we do not attempt to make any intelligent peak selection, and always select the highest peak at every frame. We perform the same evaluation for the Direct, Iterative and Joint methods suggested by Klapuri, and comment on the results[1]. In section 5.3, we evaluate our method now with the various tracking algorithms presented in chapter 3. Finally, in section 5.4 we provide the results for some initial experiments we performed on similarity computation using the extracted mid-level representations provided by our system.

5.2 Salience Functions Performance

In the first part of our evaluation we evaluate the performance of our HPCP based approach as a raw salience function, always selecting the peak of the HPCP as the melody (or bass line) peak. The results are then compared to those obtained

[1]The Direct, Iterative and Joint methods are only evaluated for melody extraction since they would require further adaption in order to perform well as a bass line salience function.

by our implementation of the three algorithms proposed by Klapuri. For these (the Direct, Iterative and Joint methods) we used a window size of 2048 samples[2] and a frequency range between 110Hz and 1000Hz. This initial evaluation gives us an idea of what we can expect to achieve overall.

5.2.1 Results for Melody Extraction

In table 5.1 we present the results obtained using the HPCP, Direct, Iterative and Joint methods. It is important to note once again that for each music collection the relevant metric was used, and that only half of the files used in the MIREX05 competition were available for our evaluation.

Music Collection	Metric	HPCP	Direct	Iterative	Joint
MIREX04	Chroma (cent)	62.66%	69.41%	70.26%	69.27%
MIREX05	Chroma (semitone)	61.12%	66.64%	66.76%	66.59%
RWC Pop	Chroma (semitone)	56.47%	52.66%	52.65%	–

Table 5.1: Salience function performance.

The first thing we observe is that for all approaches, the results are lower for the MIREX05 music collection compared to those for the MIREX04 collection, and lower still with the RWC. We can interpret this as confirmation that the latter collections, being more representative of real world music recordings, are more complex and make it harder to extract the melody from the analysed sound mixture.

For the MIREX04 collection, we note that our HPCP based approach (even without any tracking) performs well in comparison to the results listed in table 4.3, though it does not outperform the winning algorithm. Furthermore, we note that the algorithms proposed in [Klapuri 06] all perform considerably better. It is also interesting to note that we hardly observe any difference between the Direct, Iterative and Joint approaches, indicating that when used solely for extracting the maximum of the salience function at each frame (and not for multiple-F0 estimation), the three approaches are more or less equivalent. For the MIREX05 collection, we note the slight drop in performance due to increased complexity of the data. Once again the algorithms proposed by Klapuri outperform the HPCP when used as a raw salience function.

[2]Experiments with a window size of 4096 and the MIREX04 and MIREX05 collections resulted in reduced performance and hence are not included in the results.

When examining the results for the RWC database, we note that our HPCP based approach actually outperforms the other algorithms[3]. In [Ryynänen 08], the authors present a system for melody and bass line extraction which makes use of the salience function at the core of the Direct, Iterative and Joint methods which is then passed through elaborate post-processing using acoustic and musicological modelling. The system was also tested with a subset of the RWC Music Collection, however the results are not directly comparable, as the system attempts to perform full transcription (into musical notes) and is evaluated using a different set of metrics. Still, the authors achieve promising results (see [Ryynänen 08] for evaluation results), indicating that if our approach is comparable in its performance as a salience function to the ones presented by Ryynänen and Klapuri, its potential as the basis for a melody extraction system with more elaborate post-processing is worth further investigation. In figure 5.2 we show examples of successfully extracted melodies (plotted in red x's) against the reference melodies (in blue circles) for songs from each of the aforementioned collections.

5.2.1.1 Effect of Window Size

Finally for this section, we present the performance results for our proposed approach, using different analysis window sizes. As explained in section 3.2.3, there is a trade-off between using a bigger window for smoothing out "noisy" frames and using a smaller window for a more accurate description of the temporal evolution of the extracted melody or bass line. We evaluated our approach on the RWC collection, using different windows sizes. The results are presented in figure 5.1.

As expected, we see an increase in performance as we increase the window size (up to a certain limit), with the highest score achieved with a window size of 16384 samples (371ms). As previously mentioned, increasing the window size also reduces the ability to model more subtle elements of the melody. As we have argued, we opt for a window size of 8192 samples (186ms), which performs almost as well as the next window size up, at the benefit of gaining a more refined description of the melody.

5.2.2 Results for Bass Line Extraction

Here we present the result for bass line extraction, again without any tracking. Only the RWC database was used for the evaluation, as it is the only one which

[3]Due to time constraints we only computed the Direct and Iterative salience functions, though we can assume that the Joint method would have similar results.

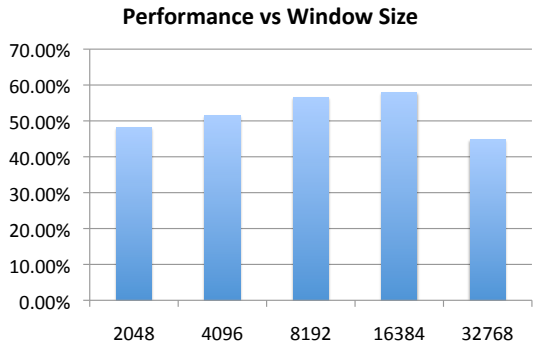

Figure 5.1: Results for our HPCP based approach with the RWC database using different window sizes.

provides references for the bass line as well as the melody. Similarly, we can only evaluate our approach, as the Direct, Iterative and Joint methods are not adapted in our implementation for bass line extraction. The result is given in table 5.2:

Music Collection	**Metric**	**HPCP**
RWC Pop	Chroma (semitone)	73.00%

Table 5.2: Salience function performance for bass line.

We see that the result for bass line extraction is considerably higher than its equivalent for melody extraction. We can explain this as being the result of two main factors: Firstly, the bass line, almost always, is simpler than the melody, and usually has fewer but longer notes. This means there is less chance of the analysis missing short notes or making mistakes at note transitions. Second and more important, we can argue that the bass line will almost certainly be the most dominant instrument in the low frequency range of the spectrum. Unlike the melody, it does not have to compete with other instruments for salience. Furthermore, the bass line is closely tied to the harmony of the piece, and so the bass frequency will often be supported not only by the harmonics of the bass note, but by the frequencies of other harmonic instruments in the mixture. In figure 5.2 we present (alongside extracted melodies) an example in which the bass line is successfully extracted. The performance for the melodies and bass line presented in figure 5.2 are 73%, 80%, 78% and 95% for daisy1.wav (MIREX04), train05.wav (MIREX05), RM-P014 (RWC, melody) and RM-P069 (RWC, bass) respectively.

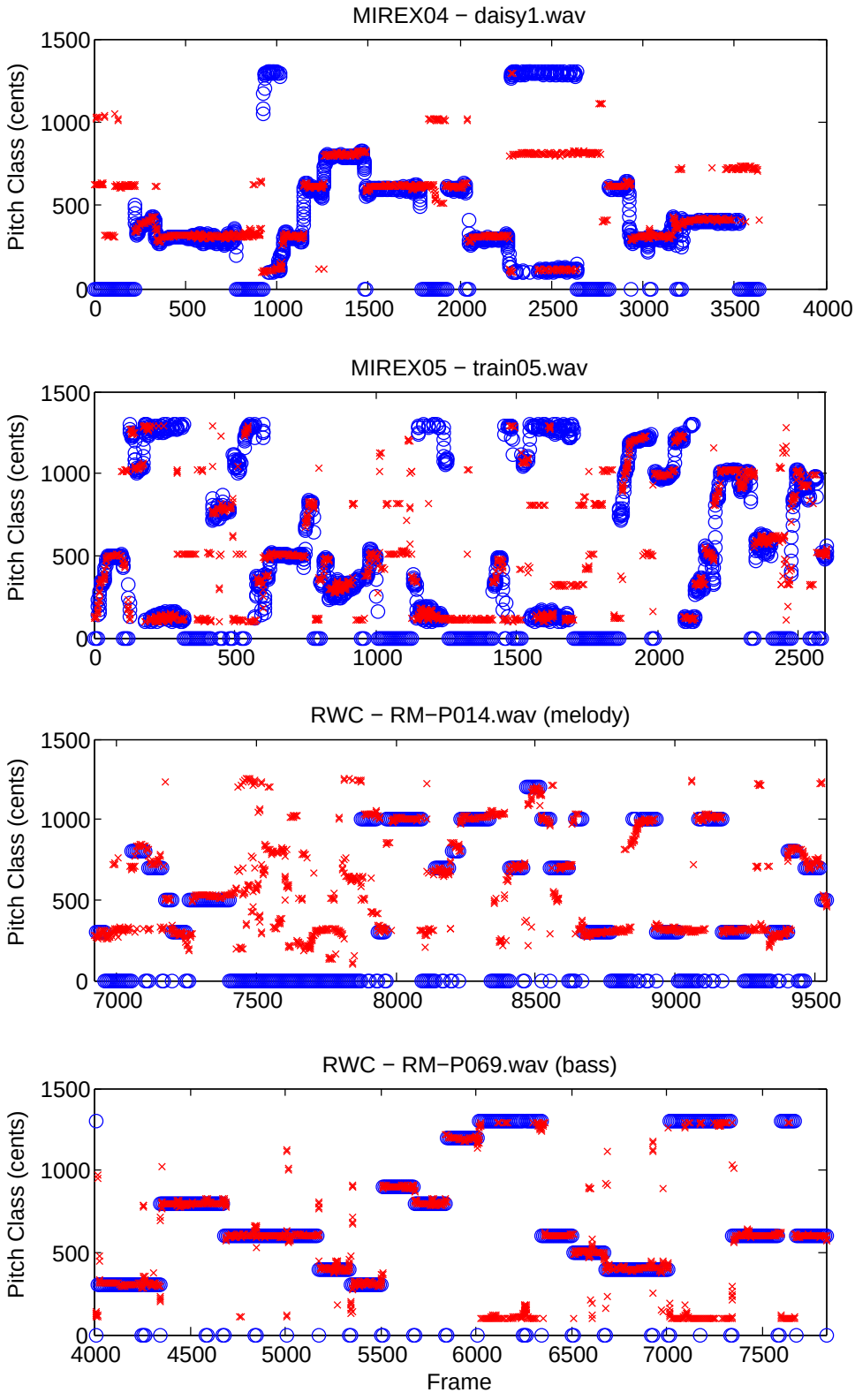

Figure 5.2: Extracted melodies and bass line against references for all collections.

5.2.3 Voicing Experiment

As previously mentioned, voicing detection does not form part of our final algorithm and we have not devoted much time in investigating this matter. Nonetheless, we have performed very initial experiments with voicing detection as mentioned in section 3.3.4, the results of which are presented here for completeness.

In figure 5.3 we present the results for the MIREX04 collection, using increasing values of the *factor* parameter (from 0.5 to 1.5) in equation 3.3. Three curves are presented – the blue curve shows the raw pitch detection performance, the red curve the unvoiced frame detection rate, and the green curve the overall performance taking into consideration both voicing and pitch detection.

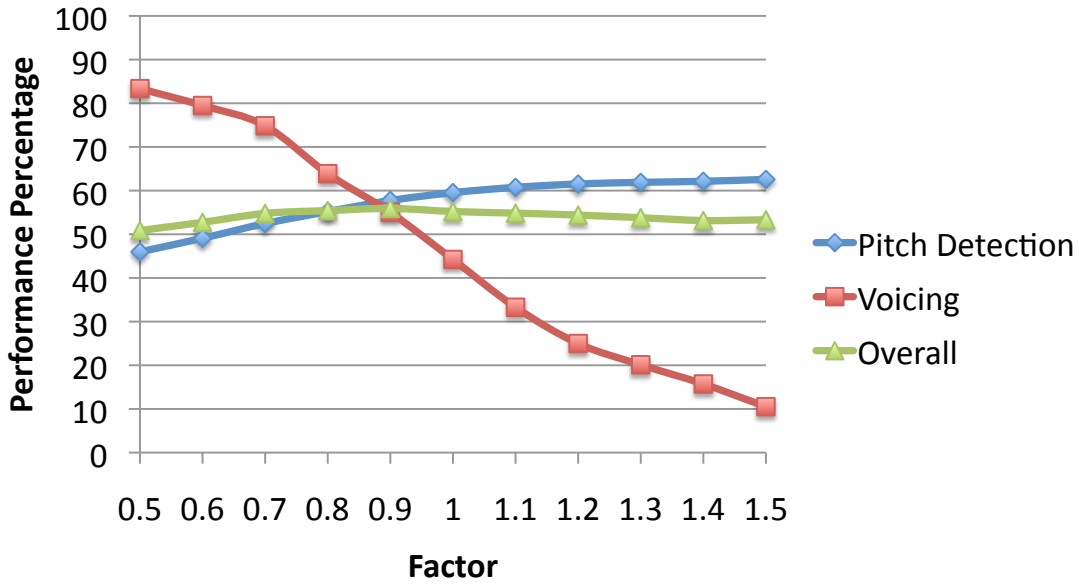

Figure 5.3: Pitch detection, voicing and overall performance for MIREX2004.

The results are fairly straight forward – the greater the *factor*, the higher the threshold, resulting in less frames labelled as unvoiced. As a result, the unvoiced frame detection rate drops whilst the raw pitch detection rate goes up. The overall performance is a combined score for all frames in the song. Recall that we are using the MIREX04 Chroma (cent) metric, meaning every deviation (down to one cent) from the reference is penalised. The optimal value for the *factor* in equation 3.3 is selected as the location of the maximum of the green curve, in this case 0.9. For this value, the unvoiced frame detection rate is 55.02%, the raw pitch detection is 57.74% and the overall performance 55.92%. In figure 5.4 we present the same analysis for the MIREX05 collection, where the optimal value

for $factor$ is 0.6. In this case the unvoiced frame detection rate is 73.11%, the raw pitch detection is 55.37% and the overall performance is 55.12%. Finally, in figure 5.5 we show the extracted melody for daisy1.wav (MIREX04), first without voicing detection and then with, for $factor = 0.9$.

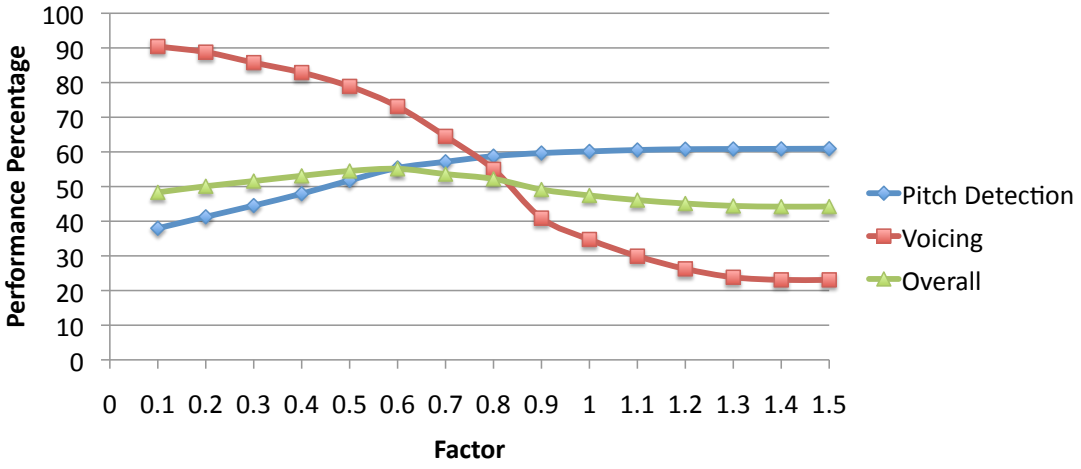

Figure 5.4: Pitch detection, voicing and overall performance for MIREX2005.

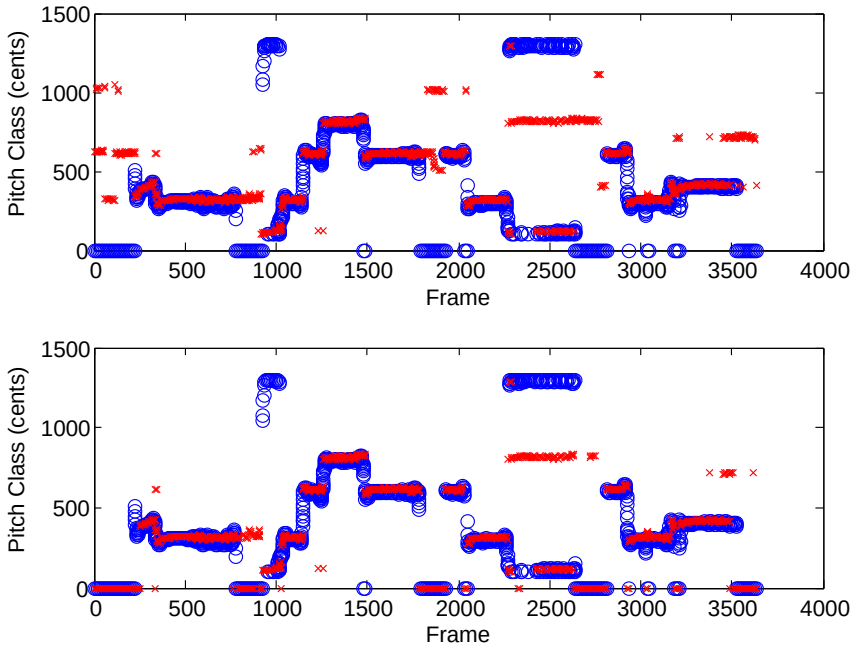

Figure 5.5: Extracted melody for daisy1.wav, with and without voicing detection.

5.3 Tracking Performance

Following the presentation of the raw salience function performance of our HPCP based approach in the previous section, in this section we present the results obtained using the various tracking algorithms proposed in chapter 3. So far, we have always chosen the highest peak of the HPCP as the output frequency (pitch class) for the given frame. The idea behind all tracking algorithms is that using a simple set of heuristics, it is perhaps possible to make a more calculated peak selection at each frame, given that highest peak is not always the correct one.

5.3.1 Glass Ceiling

Before presenting the results, we must first examine what might be called the "glass ceiling" of our approach. That is, given the number of extracted peaks at every frame, what is the highest possible score we could achieve, if our tracking algorithm always selected the right peak out of the available candidates. We recall that the proposed tracking algorithms in chapter 3 consider two candidate peaks at every frame (the highest two). Table 5.3 presented the glass ceiling for our HPCP based approach using two peaks at each frame, for the MIREX04, MIREX05 and RWC Popular Music (melody and bass) collections.

Music Collection	Metric	Raw Salience	Glass Ceiling
MIREX04	Chroma (cent)	62.66%	72.92%
MIREX05	Chroma (semitone)	61.12%	70.81%
RWC Pop (melody)	Chroma (semitone)	56.47%	69.91%
RWC Pop (bass)	Chroma (semitone)	73.00%	79.49%

Table 5.3: Glass ceiling for RWC using 2 peaks.

There are several things to note from this table. Firstly, that the glass ceiling is relatively low (the maximum possible being 100%). This indicates that there are relatively many cases in which the correct melody (or bass line) frequency is not present in neither of the top two peaks in the HPCP. The second thing to note is that the raw results obtained by always selecting the highest peak are fairly close to the glass ceiling, indicating that it might be hard to improve on current results without taking more peaks into consideration. This is most clear for the bass line results, where we are only 6.5% below the maximum achievable using 2 peaks. Following these observations, we calculated the glass ceiling for our approach using an increasing number of peaks. The results are presented in

table 5.4. For completeness, we have also included the evaluation results for the MIREX04 collection with the evaluation metric used for the MIREX05 and RWC collections (labelled "Chroma (semitone)").

Music Collection	Metric	Raw Salience	2 Peaks	3 Peaks	4 Peaks	5 Peaks	6 Peaks	7 Peaks	8 Peaks	9 Peaks
MIREX04	Chroma (cent)	62.66%	72.92%	76.13%	77.75%	78.61%	79.14%	79.34%	79.38%	79.38%
MIREX04	Chroma (semitone)	71.23%	82.805%	86.36%	88.15%	89.09%	89.66%	89.88%	89.93%	89.93%
MIREX05	Chroma (semitone)	61.12%	70.81%	74.94%	77.35%	78.88%	80.05%	81.09%	81.40%	81.40%
RWC Pop (melody)	Chroma (semitone)	56.47%	69.91%	76.51%	80.61%	83.49%	85.41%	86.30%	86.54%	86.55%
RWC Pop (bass)	Chroma (semitone)	73.00%	79.49%	83.22%	85.66%	87.44%	88.76%	89.46%	89.66%	89.68%

Table 5.4: Glass ceiling for increasing peak number.

The first thing we note, for all collections, is that the glass ceiling does not reach 100%, as we might have hoped for. The highest glass ceiling is as expected that of the simplest collection, the MIREX04 collection, when evaluated with the Chroma (semitone) metric. The glass ceiling for the Chroma (cent) metric is of course lower, and we can expect it to never reach 100%, as any deviation (as small as once cent) from the reference is penalised, even if the compared frequencies would be considered within the same semitone pitch class.

These results reveal what is perhaps an inherent limitation of our approach in its current form, that is – there are some frames in which the melody (or bass line) is not present in any of the peaks of the HPCP, regardless of their height ranking. The glass ceiling could potentially be "pushed up" by using further preprocessing in the HPCP computation, though we have not explored this in our work. Nonetheless, the results are by no means discouraging – averaging at 86.9% (for the chroma (semitone) metric) including 86.5% for melody and 89.7% for bass line with the RWC database (which is the collection closest to the real-world recordings that would be used in actual application contexts), getting close to this ceiling would already result in very high performance when compared to current state of the art systems.

5.3.2 Tracking Results

In table 5.5 we present the results obtained using tracking algorithms 1 through 6, proposed in chapter 3. We ran experiments on all three aforementioned music collections, with and without smoothing as a preprocessing step. For reference, we also include the glass ceiling (indicating the maximum result possible) for 2 peaks.

Music Collection	Metric	Raw Salience	smoothing	Proximity-Salience			Note-Segmentation			Glass Ceiling
				Alg 1	Alg 2	Alg 3	Alg 4	Alg 5	Alg 6	
MIREX04	Chroma (cent)	62.66%	no	**63.84%**	58.66%	63.057%	50.00%	57.97%	57.73%	72.92%
MIREX05	Chroma (semitone)	61.12%	no	60.40%	57.09%	**61.19%**	46.51%	53.77%	52.51%	70.81%
RWC Pop (melody)	Chroma (semitone)	56.47%	no	**57.92%**	55.40%	57.17%	49.32%	53.95%	53.47%	69.91%
RWC Pop (bass)	Chroma (semitone)	73.00%	no	71.80%	66.17%	**73.92%**	59.59%	68.76%	68.10%	79.49%
MIREX04	Chroma (cent)	62.66%	yes	62.08%	57.00%	**62.95%**	51.39%	56.97%	56.62%	72.15%
MIREX05	Chroma (semitone)	61.12%	yes	58.94%	56.02%	**59.92%**	47.74%	53.31%	51.38%	69.13%
RWC Pop (melody)	Chroma (semitone)	56.47%	yes	**58.46%**	55.70%	57.91%	50.26%	53.75%	53.09%	69.98%
RWC Pop (bass)	Chroma (semitone)	73.00%	yes	73.81%	68.39%	**75.60%**	62.26%	69.49%	68.57%	80.72%

Table 5.5: Results for tracking algorithms.

There are several observations we can make from the results presented in table 5.5. We start by noting that of all algorithms, **Tracking Algorithm 3** performs best on average, with **Tracking Algorithm 1** closely after it. In general, the Proximity-Salience based algorithms dramatically outperform the Note-Segmentation based ones. This indicates that whilst some improvement can be achieved using simple heuristics and a per-frame selection process, performing successful selection based on a larger temporal scope is a much more challenging task. When examining the results for the best note-segmentation based algorithm of the three proposed, **Tracking Algorithm 5**, we were able to observe several potential causes for the low performance:

- Segmentation process – the first potential problem is with the actual note segmentation process. As the HPCP peak data is fairly noisey (even with a large window and smoothing), it is often the case that a continuous note is broken into several notes. As a result, the note salience is divided between these notes, making it harder to recognise salient melody notes.

- Selection process – once the segmentation is done, the next challenge is in selecting the right note. This introduces a new problem – if the wrong note is selected, we continue following this wrong note either until it ends or until it is interrupted by a newer more salient note. Thus, whilst for the raw salience we simply select the maximum at each frame and are not penalised in future frames if we made the wrong selection, here making the wrong selection results in a greater penalty. Clearly, the opposite occurs when we select the correct note – we are rewarded in future frames for our current selection. However, we can tell from the results that on average we get penalised more than rewarded, resulting in a decrease in the results. A possible reason for this could be that melody notes tend to be shorter than notes by other instruments, and thus committing to the correct melody note results in a reward smaller than the penalty for committing to a wrong note.

Next, we observe that the smoothing process is beneficial for the tracking process when evaluated with the RWC melody and bass line collections, whilst detrimental for the MIREX04 and MIREX05. One could argue that as the RWC collection is more complex, the smoothing by-and-large helps the tracking process more than it changes correct notes into incorrect notes, whilst for the MIREX collections, which are simpler, the opposite occurs.

Finally, we note that though we were able to achieve some improvement over the raw salience performance for each of the collections, the improvement was not significant (roughly 2% for the RWC collections). We believe there are two main reasons for this:

- Glass ceiling – the average difference between the raw salience performance and the glass ceiling is roughly 10% on average. This means, that given the peak data we obtain from the HPCP, we are already doing almost as best as possible in selecting the correct peak in every frame. That is, out of the frames for which one of the two HPCP peaks is the true melody peak, we select the correct peak over 85% of the time. This has two further implications – firstly, that it could be very complex to come up with a heuristic which covers these extra 15% of the peaks without reducing the performance overall. Second and more important, is that since our glass ceiling for two peaks is lower than 100%, it means that there are frames in which neither of the two peaks is the true melody peak. Thus, these peaks throw the tracking algorithm "off track", leading it to make tracking decisions based on frames in which both peaks are erroneous.

- Octave information – the other possible problem is the fact that when we use the HPCP we fold all notes onto one octave, effectively forfeiting all octave information. Whilst not a problem in itself (as we are not aiming at full transcription), this could potentially make the tracking process harder. For example, consider two consecutive notes at an interval of a major seventh apart. With octave information, these notes would be considered fairly distant. Due to the cyclic nature of distance when using HPCP, these notes are considered equally close to each other as two notes which are only a semitone apart in reality. As such, using proximity as one of the major factors in the peak selection (alongside salience) becomes a more complex task. A potential improvement would be to consider distance in a way which is not solely based on the peak's location on the chroma circle, but which also takes musicological knowledge into account (e.g. likely and unlikely intervals, whether the note is within the mode of the current chord, etc.).

Following this analysis, our main conclusion is that for effective tracking we would have to consider more than two peaks. From table 5.3 we we can assert that no more than 8 peaks per frame need be considered to assure that we can potentially reach the highest possible results given our current approach, and in fact using as few as 5 peaks would still allow us to obtain highly satisfactory results. Still, as we increase the number of candidate peaks, so we increase the complexity of the tracking procedure. in chapter 6 we propose the adaptation of the tracking algorithms presented in chapter 3 for use with more than two peaks as a possible direction for future work.

5.4 Similarity Performance

As previously mentioned, our goal is to extract a mid-level representation of the melody and bass line which can be used for similarity computation based applications (such as QBH and cover song identification). Thus, in addition to the extraction preformance evaluation detailed in the sections above, we have also performed some initial similarity based experiments.

In these experiments, we used our extracted melodies and bass lines from the RWC databas. We compared every extracted melody to every melody reference and every extrated bass line to every bass line reference, using the distance metric mentioned in section 5.4.1. For the comparison, the references were converted from frequency values to HPCP bin values. The results are presented in the form of a confusion matrixm indicating the degree to which every extracted melody matches every melody reference, and every extracted bass line matches every bass line reference. The goal of this initial experiment is to show that the extracted mid-level representation adequatly represents the original melodies and bass lines (indicated by the references) such that when compared against the entire database it will match the correct song. As noted, it is only an initial experiment, and further work would be required to fully asses the usefulness of the extracted mid-level representations, proposed as future work.

5.4.1 Distance Metric

As previously mentioned, it is importnat to select the appropriate distance metric for the given task. In the case of this experiment, we wish to compare the entire extracted melody or bass line to the entire reference. We can then ask – what is the smallest number of operations needed to transform the extracted representation into the reference? The answer to this is given by the Levenshtein Distance [Levenshtein 66]. Below we present a commonly-used bottom-up dynamic programming algorithm for computing the Levenshtein distance. For further details and a detailed overview of string matching techniques the reader is referred to [Navarro 02]. For the purpose of matching our HPCP-based mid level representation, we define the *cost* function as:

$$cost(b_1, b_2) = \begin{cases} 0 & \text{if } distHPCP(b_1, b_2) <= 5 \\ 1 & \text{otherwise} \end{cases} \quad (5.1)$$

The final output of the algorithm thus is the cost of converting one sequence into the other, or *distance*. In figure 5.6 we show the distance matrix d for match-

Global alignment: Levenshtein Distance:

1. Given two sequences s and t of lengths M and N respectively:
2. **for** $i = 0$ to M
3. $\quad d(i, 0) = i$
4. **for** $j = 0$ to N
5. $\quad d(0, j) = j$
6. **for** $i = 1$ to M
7. $\quad$ **for** $j = 1$ to N
8. $$d(i,j) = \min \begin{cases} d(i-1,j)+1 & //\text{deletion} \\ d(i,j-1)+1 & //\text{insertion} \\ d(i-1,j-1)+cost(s(i),t(j)) & //\text{substitution} \end{cases}$$
15. $\quad$ **end**
16. **end**
17. **return** $d(M, N)$

ing the extracted melody representation of RM-P003.wav against its reference from the RWC database.

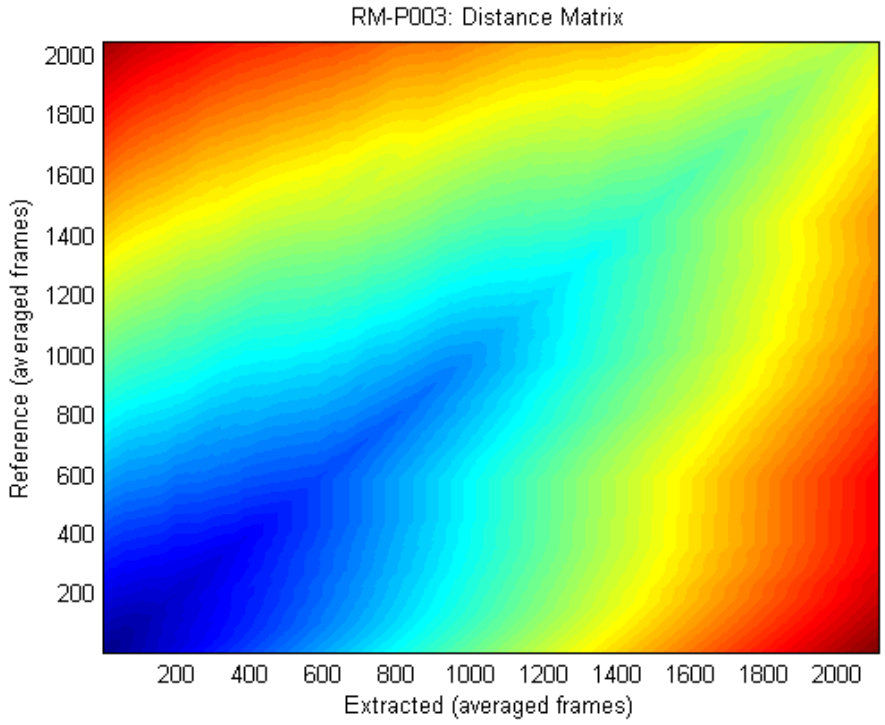

Figure 5.6: Distance Matrix for RM-P003.wav (melody).

As can be inferred from the algorithm, the maximum possible distance value depends on the length of the sequences compared, more specifically on the length of the longer sequence of the two. Thus, in order to be able to compare between distances measured for songs of different lengths, we normalise the final output of the algorithm by the length sequence of the longer sequence.

5.4.2 Results

We present the results in the form of a confusion matrix. In figure 5.7 we show the confusion matrix for the extracted melodies of the RWC database. In figure 5.8 we show the same for the extracted bass lines.

We see that in both cases, the diagonal along the matrix clearly stands out. This indicates that for the large majority of songs in the database, the extracted representation is always closest to its corresponding reference. We note that the diagonal for the bass lines is even clearer than the one for melodies, which is explained by the higher overall extraction rate achieved for bass lines. Though further experiments would be required in order to assert the usefulness of our extracted mid-level representation, the results presented here are clearly encouraging, suggesting the extracted representations could be useful in the context of similarity based applications.

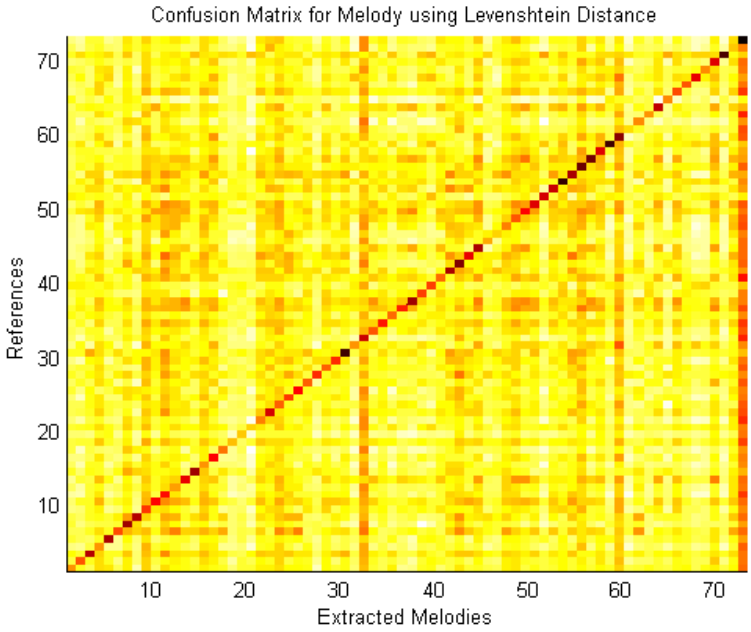

Figure 5.7: Confusion matrix for extracted melodies.

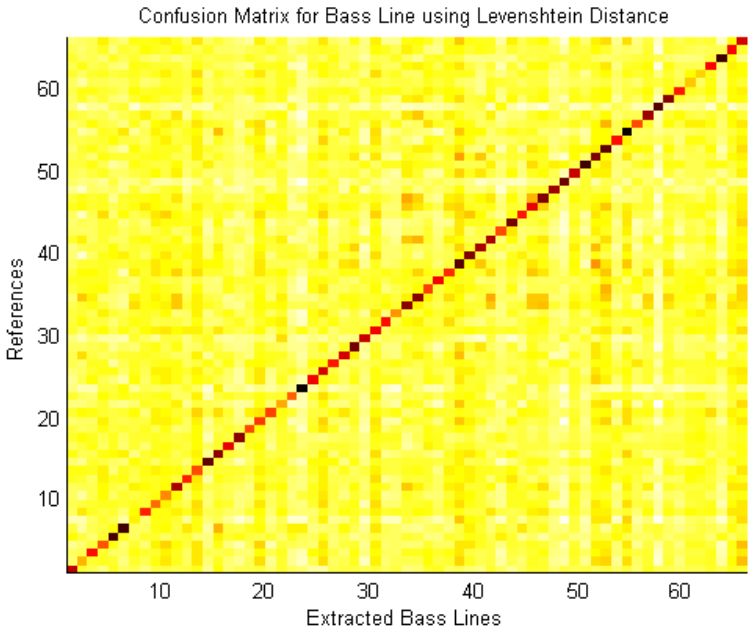

Figure 5.8: Confusion matrix for extracted bass lines.

5.5 Conclusion

In this chapter we presented the results for the evaluation of our proposed melody and bass line extraction system. We started be evlauating the performance of our approach as a *salience function*. For melody extraction, we saw that the results for the MIREX datasets were within the same performance region as the other state of the art algorithms (62.66% for MIREX04) though for the MIREX05 dataset our results (61.12%) were significantly lower than those of the other systems. However, we then noted that for the RWC database our results (56.47%) were comparable and even better than the compared approaches.

We then examined the effect of the window size on the analysis, showing that the two highest results are for windows of 8192 and 16384 samples, and we opted for the former for the reasons detailed in section 3.2.3. Next we presetned the performance results of our approach a salience function for bassline extraction, achieving 73.00% on average. Initial voicing experiments showed that the task of selecting a threshold for voiced frames is not a straight forward one, and that a more complex heuristic might be necessary.

Next, we presented the results for our approach now using tracking. We started by computing the glass ceiling for our approach given the number of peaks under consideration at every frame. We noted that for two peaks the glass ceiling is relatively low (72.92%, 70.81%, 69.91% and 79.49% for MIREX04, MIREX05, RWC (melody) and RWC (bass) respectively), and that our results are only about 10% below this glass ceiling. We noted that this means it might be hard to improve on the results obtained so far without taking more peaks into consideration. We then computed the glass ceiling for an increasing number of peaks, we observed that the glas ceilings also have a glass ceiling (79.38%, 89.93%, 81.40%, 86.55% and 89.68% for MIREX04 (cent), MIREX04 (chroma), MIREX05, RWC (melody) and RWC (bass) respectively).

We then presented the results obtained with the various tracking algorithms. We saw that smoothing was beneficial for the RWC collection but for the MIREX datasets, and that Tracking Algorithm 3 performed best overall. However, we noted that the improvement obtained by tracking was by-and-large insignificant (about 2%), indicating that we do in fact need to consider more peaks in every frame in order to achieve a significant improvement over the salience function results.

Finally we presented the results for some initial experiments carried out in similarity computation. We saw that for the large majority of extracted melodies and bass lines, the extracted mid-level representation was closest to its correponding reference. We noted that though we can not make any assertions based

on these initial experiments, the results are encouraging and suggest that the extracted representations could be useful in the context of similarity based applications.

6 Conclusion

In the last chapter of this dissertation, we provide a final overview of the work carried out. We present the goals achieved and contributions made, as well as discuss issues which remain unresolved or that require further investigation. We then make suggestions for future work, and briefly present a proposal for research which is to be carried out as part of a PhD. The chapter ends with some final conclusions.

6.1 Contributions

We start by refering back to the goals we set out for the work during the introduction (section 1.5). We note that all the goals mentioned have been met:

- Scientific background and a summary of the literature in the field of melody bass line extraction was provided.
- A new method for melody and bass line extraction based on chroma features was developed and presented in chapter 3.
- Our proposed method was evaluated and compared to exisiting state of the art systems, including ones we implemented and evaluated personally.
- The evaluation results were presented and discussed.

Focusing on the evaluation results, we can make some final conclusions. We note that as a salience function, our approach performs comparably to other state of the art systems. The relatively compact data representation of the HPCP and its efficient computation thus make it a good candidate for real world applications. Improving the results using tracking was shown to be difficult when only considering two peaks, and the tracking algorithms proposed must be extended to cope with a larger number of peaks if we wish to significantly improve the results.

It is importnat to note that as our proposed approach is based on chroma features (which do not contain octave information), it can not be used for the purpose of full transcription. Nonethless, we were able to show some initial results which suggest that our approach could be useful for a range of applications based on melody or bass line similarity.

6.2 Future Work

6.2.1 Improving Current Results

Throughout this dissertation we have metioned several issues which remain unresolved, and require further work. For some, a potential solution was already proposed when the issue was discussed. In this section we list those issues which have remained problematic, as well as those we find interesting and would like to pursue further.

- Voicing detection – as previously mentioned, we have only performed some very prelimiary work on voicing detection, which is not included as part of our final system in its current state. Though we were able to show some promising results for similarity computation without any voicing detection, it is clear that these results could be improved if combined with a succesful voicing detection algorithm. The algorithm by [Dressler 05] won the MIREX05 competition largely due to its high performance in voicing detection.

- Tracking – already discussed several times before, there is much room for improvement on the peak tracking phase of our system. This primarily requires the consideration of more than 2 peaks per frame, and the extension of the tracking algorithms to cope with the increased number of peaks (and hence noise).

- More experiments on similairity – the initial results given in chapter 5 seem promising, and without doubt merit further investigation. It is only in the context of a real world application such as a QBH search engine that we can fully evaluate the usefulness of our extracted mid-level representations. We also note that improving on the two aforementioned issues would automatically improve any results we obtain for similarity based applications.

6.2.2 Proposal for PhD Work

Following the work carried out for this Master's thesis, in this section we provide an introduction to our proposed PhD research. Generally speaking, our work was focused on the extraction of the melody and bass line. In the following two sections we explain how this work can be placed in a much broader context, extending the range of potential applications.

6.2.2.1 Musical Stream Estimation From Polyphonic Audio Based on Perceptual Characteristics

As listeners, we identify different types of "lines" when listening to polyphonic music. Often these lines are defined by their source such as instrument type, or predominance, such as the melody or bass line as we have done in this work. However, this is only a subset of perceptual characteristics we may use to define what we call *musical streams*, what we view as an extension of the concept of "auditory streams" first introduced by Bregman in 1990 [Bregman 90]. Other characteristics might be the timbre, pitch evolution, monotonousness, harmonicity, rhythmic feel, mood, etc. Moreover, though these streams are likely to follow a single sound source, they should not be constrained to one – we can imagine a musical segment strongly characterised by a certain rhythmical sequence, which is played by different instruments during different parts of the segment. In this sense, our perception of music goes far beyond that which is currently modeled by existing systems. This brings us back to the problem of the semantic gap, as discussed in our introduction to this dissertation.

In secion 1.1 we noted how the prevalence of digital media has resulted in an exponential growth in the distribution and consumption of digital audio. With this growth, our needs have evolved, as consumers, artists and researchers. We presented application areas that would greatly benefit from melody and bass line extraction, such as Query by Humming systems, which in order to be truly functional on a large scale basis, necessitate an automatic method of melody extraction.

But we can think about this problem beyond the goal of finding a known song – to the task of an exploratory search. Producers might be interested in musical lines with a certain sound or feel, regardless of the specific source. Artists might wish to incorporate a sample from another piece, or create a new piece altogether by the mixing of musical streams with certain perceptual characteristics (musical stream mosaicing). Music researchers can benefit from the comparison of certain perceptual themes (as an extension of the analysis of melodic themes for example). Utilising musical streams can enhance users' interaction with music

through novel ways of exploring and browsing large collections, in which the user takes an *active* part in searching and finding new music. The enhancement of the listening experience through interaction, *Active Music Listening* [Goto 07], was discussed in our introduction to this dissertation. In addition, we believe the research on developing this technology will help us in better understanding the human perception of music and will help towards bridging the semantic gap. The primary goal of the research will be to develop the technology for musical stream estimation. This means developing a system that given a set of perceptual characteristics (with potential parameterisation), can access a large database of polyphonic music and return meaningful musical streams. The development entails both theoretical work on models for musical stream estimation and relations between human perception and audio features, and applied work on the development and implementation of algorithms and their incorporation in functional systems. Once developed, this technology will be incorporated in systems which demonstrate the potential aplication of musical stream estimation.

6.3 Final Words

I found the he work on this project was both interesting and challenging. Through it I have aquired a wide background in the area of melody and bass line extraction as well as related disciplines in the field of Music Information Retrieval. I have had the opportunity to learn from many people, and look forward to future collaborations. I believe the results of the work are interesting and intend to continue investigating them as part of my intended PhD work. Finally, I would like to thank once more all the people who helped me (in any way) write this dissertation. Thanks.

Justin J. Salamon.

Bibliography

[Abe 96] T. Abe, T. Kobayashi & S. Imai. *Robust Pitch Estimation with Harmonics Enhancement in Noisy Environments based on Instantaneous Frequency.* In Proc. of ICSLP 96, pages 1277–1280, 1996.

[Bregman 90] A. Bregman. Auditory scene analysis. MIT Press, Cambridge, Massachussetts, 1990.

[Brossier 06] P. Brossier. *Automatic Annotation of Musical Audio for Interactive Applications.* PhD thesis, Queen Mary, University of London, 2006.

[Brown 91] J. C. Brown. *Calculation of a Constant-Q Spectral Transform.* Journal of the Acoustic Society of America, vol. 81, no. 1, pages 425–434, 1991.

[Brown 92] J. C. Brown & M. S. Puckette. *An Efficient Algorithm for the Calculation of a Constant-Q Transform.* Journal of the Acoustic Society of America, vol. 92, page 26982701, 1992.

[Byrd 02] D. Byrd & T. Crawford. *Problems of Music Information Retrieval in the Real World.* Information Processing & Management, vol. 38, pages 249–272, 2002.

[Cano 98] P. Cano. *Fundamental Frequency Estimation in the SMS Analysis.* In Proc. COST G6 Conference on Digital Audio Effects DAFx-98, Barcelona, 1998.

[Cohen 89] L. Cohen. *Time-Frequency Distributions– a Review.* In Proceedings of the IEEE, volume 77, pages 941–981, July 1989.

[Copeland 57] A. Copeland. What To Listen For In Music. McGraw Hill, 1957.

[Dannenberg 07] R. B. Dannenberg, W. P. Birmingham, B. Pardo, N. Hu, C. Meek & G. Tzanetakis. *A Comparative Evaluation of Search Techniques for Query-by-Humming Using the* Musart *Testbed.* Journal of the American Society for Information Science and Technology, February 2007.

[Dowling 78] W. J. Dowling. *Scale and Contour: Two Components of a Theory of Memory for Melodies.* Psychological Review, vol. 85, no. 4, pages 341–354, July 1978.

[Downie 05] S. Downie, K. West, A. Ehmann & E. Vincent. *The 2005 Music Information Retrieval Evaluation Exchange (MIREX 2005): Preliminary Overview*. In ISMIR, pages 320–323, 2005.

[Dressler 05] K. Dressler. *Extraction of the Melody Pitch Contour from Polyphonic Audio*. In Proc. 6th International Conference on Music Information Retrieval, Sept. 2005.

[Essentia] Essentia. *http://mtg.upf.edu/technologies/essentia*.

[Fujishima 99] T. Fujishima. *Realtime Chord Recognition of Musical Sound: a System using Common Lisp Music*. In Computer Music Conference (ICMC), pages 464–467, 1999.

[Fujishima 00] T. Fujishima. *Apparatus and Method for Recognizing Musical Chords*. U.S. Patent 6,057,502 Yamaha Corporation, Hamamatsu, Japan, 2000.

[Fujitani 71] D. S. Fujitani & W. J. Dowling. *Contour, Interval, and Pitch Recognition in Memory for Melodies*. The Journal of the Acoustical Society of America, vol. 49, no. 2B, pages 524–531, February 1971.

[Gómez 06a] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2006.

[Gómez 06b] E. Gómez. *Tonal Description of Polyphonic Audio for Music Content Processing*. INFORMS Journal on Computing, Special Cluster on Computation in Music, vol. 18, no. 3, 2006.

[Gomez 06c] E. Gomez, S. Streich, B. Ong, R. Paiva, S. Tappert, J. Batke, G. Poliner, D. Ellis & J. Bello. *A Quantitative Comparison of Different Approaches for Melody Extraction from Polyphonic Audio Recordings*. Technical Report MTG-TR-2006-01, Univ. Pompeu Fabra Music Tech. Group, 2006.

[Goto 99] M. Goto & S. Hayamizu. *A Real-Time Music Scene Description System: Detecting Melody and Bass Lines in Audio Signals*. Working Notes of the IJCAI-99 Workshop on Computational Auditory Scene Analysis, pages 31–40, 1999.

[Goto 02] M. Goto, H. Hashiguchi, T. Nishinura & R. Oka. *RWC Music Database: Popular, Classical, and Jazz Music Databases*. In Proc.

Third International Conference on Music Information Retrieval ISMIR-02, Paris, 2002. IRCAM.

[Goto 04a] M. Goto. *Development of the RWC music database*. In Proc. of ICA, pages I–553–556, 2004.

[Goto 04b] M. Goto. *A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals*. Speech Communication, vol. 43, pages 311–329, 2004.

[Goto 07] M. Goto. *Active Music Listening Interfaces Based on Signal Processing*. In IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007), volume IV, pages 1441–1444, 2007.

[ISMIR 04] ISMIR. *Melody Extraction Contest 2004 (MIREX 2004). http://ismir2004.ismir.net/melody_contest/results.html*, 2004.

[Klapuri 03] A. P. Klapuri. *Multiple Fundamental Frequency Estimation based on Harmonicity and Spectral Smoothness*. In IEEE Trans. Speech and Audio Processing, volume 11, 2003.

[Klapuri 05] A. P. Klapuri. *A Perceptually Motivated Multiple-F0 Estimation Method for Polyphonic Music Signals*. In IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, NY, 2005.

[Klapuri 06] A. Klapuri. *Multiple fundamental frequency estimation by summing harmonic amplitudes*. In Proc. 7th International Conference on Music Information Retrieval, Victoria, Canada, October 2006.

[Leman 00] M. Leman. *An Auditory Model of the Role of Short Term Memory in Probe-Tone Ratings*. Music Perception, Special Issue on Tonality Induction, vol. 17, no. 4, pages 481–509, 2000.

[Lesaffre 05] M. Lesaffre. *Music Information Retrieval: Conceptual Framework, Annotation and User Behavior*. PhD thesis, Ghent University, Belgium, December 2005.

[Levenshtein 66] V. Levenshtein. *Binary Codes Capable of Correcting Deletions, Insertions and Reversals*. Sov. Phys. Dokl, vol. 10, no. 8, pages

707–710, 1966. Original in Russian in Dokl. Akad. Nauk SSSR 163, 4, 845-848, 1965.

[Marolt 04] M. Marolt. *On Finding Melodic Lines in Audio Recordings.* In DAFX, Naples, 2004.

[Master 00] A. S. Master. Speech spectrum modelling from multiple sources. Master's thesis, University of Cambridge, 2000.

[Midi] Midi. *Musical Instrument Digital Interface. http://www.midi.org/.*

[Moorer 77] J. Moorer. *On the Transcription of Musical Sound by Computer.* Computer Music Journal, vol. 1, no. 4, pages 32–38, 1977.

[Navarro 02] G. Navarro & M. Raffinot. Flexible Pattern Matching in Strings. Cambridge University Press, Cambridge, UK, 2002.

[Paiva 04] R. P. Paiva, T. Mendes & A. Cardoso. *A Methodology for Detection of Melody in Polyphonic Music Signals.* In 116th AES Convention, 2004.

[Pardo 03] B. Pardo & W.P. Birmingham. *Query by Humming: How Good Can it Get?* In Workshop on Music Information Retrieval, pages 107–109, Toronto, Canada, 2003.

[Pardo 04] B. Pardo, J. Shifrin & W. Birmingham. *Name That Tune: A Pilot Study in Finding a Melody From a Sung Query.* Journal of the American Society for Information Science and Technology, 2004.

[Pauws 04] S. Pauws. *Musical Key Extraction from Audio.* In International Conference on Music Information Retrieval, Barcelona, Spain, 2004.

[Poliner 05] G. Poliner & D. Ellis. *A Classification Approach to Melody Transcription.* In Int. Conf. on Music Info. Retrieval ISMIR-05, London, 2005.

[Poliner 07] G. E. Poliner, D. P. W. Ellis, F. Ehmann, E. Gómez, S. Steich & O. Beesuan. *Melody Transcription from Music Audio: Approaches and Evaluation.* IEEE Transactions on Audio, Speech and Language Processing, vol. 15, no. 4, pages 1247–1256, 2007.

[Purwins 00] H. Purwins, B. Blankertz & K Obermayer. *A New Method for Tracking Modulations in Tonal Music in Audio Data Format.* Neural Networks – IJCNN, IEEE Computer Society, vol. 6, pages 270–275, 2000.

[Rabiner 93] L. R. Rabiner & B. H. Juang. Fundamentals of Speech Recognition. Prentice, Englewood Cliffs, NJ, 1993.

[Ryynänen 05] M. P. Ryynänen & A. Klapuri. *Polyphonic Music Transcription using Note Event Modeling.* In IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, Mohonk, NY, 2005.

[Ryynänen 06] M. Ryynänen & A. Klapuri. *Transcription of the Singing Melody in Polyphonic Music.* In Proc. 7th International Conference on Music Information Retrieval, Victoria, Canada, Oct. 2006.

[Ryynänen 07] M. Ryynänen & A. Klapuri. *Automatic Bass Line Transcription from Streaming Polyphonic Audio.* In ICASSP, 2007.

[Ryynänen 08] M. Ryynänen & A. Klapuri. *Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music.* Computer Music Journal, 2008.

[Selfridge-Field 98] E. Selfridge-Field. *Conceptual and Representational Issues in Melodic Comparison.* Computing in Musicology, vol. 11, pages 3–64, 1998.

[Serra 05] X. Serra. *Bridging the Music Semantic Gap.* Talk given at the Information Day on "Audio-Visual Search Technologies" organized by the Commission services of DG Information Society of the EC in Brussels, December 2005.

[Serrà 07a] J. Serrà. *A Cover Song Identication System Based on Sequences of Tonal Descriptors. Music Information Retrieval EXchange (MIREX) extended abstract.* In the 8th International Symposium on Music Information Retrieval (ISMIR), Vienna, Austria, 2007.

[Serrà 07b] J. Serrà. Music Similarity Based on Sequences of Descriptors: Tonal Features Applied to Audio Cover Song Identification. Master's thesis, Universitat Pompeu Fabra, Barcelona, 2007.

[Serra 07c] X. Serra, R. Bresin & A. Camurri. *Sound and Music Computing: Challenges and Strategies.* Journal of New Music Research, vol. 36, no. 3, pages 185–190, 2007.

[Sjölander 00] K. Sjölander & J. Beskow. *WaveSurfer - an Open Source Speech Tool*. In Proc. Int. Conf. on Spoken Language Processing, 2000.

[ThemeFinder] ThemeFinder. *http://themefinder.org/*.

[Tolonen 00] T. Tolonen & M. Karjalainen. *A Computationally Efficient Multipitch Analysis Model*. In IEEE Trans. Speech and Audio Processing, volume 8, pages 708–716, 2000.

[Typke 07] R. Typke. *Music Retrieval based on Melodic Similarity*. PhD thesis, Utrecht University, Netherlands, 2007.

[Tzanetakis 02] G. Tzanetakis. *Pitch Histograms in Audio and Symbolic Music Information Retrieval*. In 3rd International Symposium on Music Information Retrieval, Paris, 2002.

[Vincent 05] E. Vincent & M. Plumbley. *Predominant-f0 Estimation using Bayesian Harmonic Waveform Models*. In MIREX Audio Melody Extraction Contest Abstracts, London, 2005.

[Vinyes 05] M. Vinyes. Auditory stream separation in commercial music productions: implementation of a real-time method based on pan location. Master's thesis, UPC, Barcelona, 2005.

[Yang 01] C. Yang. *Music Database Retrieval based on Spectral Similarity*. In Int. Symposium on Music Information Retrieval (Poster) (ISMIR), Bloomington, Indiana, 2001.

[Zhu 05] M. S. Zhu Y.and Kankanhalli & S. Gao. *Music Key Detection for Musical Audio*. In 11th International Multimedia Modelling Conference (MMM'05), 2005.

A

RWC Music Database File List

The following list contains the song IDs of the songs in the RWC Popular Music Database used in our evaluation of the melody extraction task:

RM-P001, RM-P002, RM-P003, RM-P004, RM-P005, RM-P006, RM-P007, RM-P008, RM-P011, RM-P012, RM-P014, RM-P016, RM-P017, RM-P018, RM-P019, RM-P020, RM-P021, RM-P022, RM-P023, RM-P024, RM-P025, RM-P026, RM-P027, RM-P028, RM-P032, RM-P034, RM-P035, RM-P036, RM-P037, RM-P039, RM-P040, RM-P041, RM-P042, RM-P043, RM-P044, RM-P046, RM-P047, RM-P048, RM-P049, RM-P050, RM-P051, RM-P052, RM-P054, RM-P055, RM-P058, RM-P059, RM-P061, RM-P063, RM-P064, RM-P065, RM-P067, RM-P068, RM-P069, RM-P070, RM-P075, RM-P077, RM-P079, RM-P080, RM-P081, RM-P083, RM-P084, RM-P085, RM-P086, RM-P087, RM-P088, RM-P089, RM-P091, RM-P092, RM-P093, RM-P094, RM-P096, RM-P097, RM-P100.

The following list contains the song IDs of the songs in the RWC Popular Music Database used in our evaluation of the bass line extraction task:

RM-P001, RM-P002, RM-P004, RM-P006, RM-P007, RM-P008, RM-P011, RM-P012, RM-P014, RM-P016, RM-P017, RM-P018, RM-P019, RM-P020, RM-P021, RM-P022, RM-P023, RM-P024, RM-P025, RM-P026, RM-P027, RM-P028, RM-P032, RM-P034, RM-P035, RM-P036, RM-P037, RM-P039, RM-P040, RM-P041, RM-P042, RM-P044, RM-P046, RM-P047, RM-P048, RM-P049, RM-P050, RM-P051, RM-P052, RM-P054, RM-P055, RM-P058, RM-P059, RM-P061, RM-P063, RM-P064, RM-P065, RM-P067, RM-P068, RM-P069, RM-P070, RM-P081, RM-P083, RM-P084, RM-P085, RM-P086, RM-P087, RM-P088, RM-P089, RM-P091, RM-P092, RM-P093, RM-P094, RM-P096, RM-P097, RM-P100.