Tools for music information retrieval and playing.

Antonello D'Aguanno, Goffredo Haus, Alberto Pinto, Giancarlo Vercellesi
Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano, 20135 Milano, Italy
{daguanno, haus, pinto, vercellesi}@dico.unimi.it

October 8, 2007

1 Introduction

This chapter deals with the state of the art of tools for music information retrieval (MIR) and playing and with state of the art musical feature extraction. We make use of a bottom-up strategy. First we focus on three different blind tasks: beat and tempo tracking, pitch tracking and automatic recognition of musical instruments; the attribute blind refers to the fact that these tasks deal with audio signals without paying attention to the symbolic information layer (score). Second we present the most useful algorithms which have proven to be most effective in solving these problems in general purpose situations, providing also an overview into specific task applications. These algorithms work both on compressed and uncompressed data, anyway particular attention will be given to MPEG audio formats like AAC and MP3. We then introduce second level tasks, such as automatic genre extraction and score extraction, that make use of proprietary algorithms too, which will be described in the chapter. We analyze the relationships between MIR and feature extraction presenting examples of possible applications. Finally we focus on automatic music synchronization, a non-blind task on score and the corresponding audio performance, pointing out both solving algorithms and their applications in MIR, music playing, music education and musicology. We introduce a new audio player that supports the MX logic layer and allows to play both the symbolic score and the related audio file coherently, offering a new experience in music listening.

2 The architecture of a MIR system

In this section we introduce the architecture of a MIR system which exploits the MX language in order to handle different formats of music data. The most innovative features of this MIR architecture is that it provides tools for efficient storage of structural musical data and for performing content-based queries on such data. The overall architecture of the Musical Data Management module is illustrated in Figure 1. The module consists of two main environments: the Musical Storage Environment and the Musical Query Environment. The musical storage environment has the purpose of representing musical information in the database, to make query by content efficient. The musical query environment provides methods to perform query by content on music scores, starting from a score or an audio fragment given as input.

The matching between the input and the scores stored into DB is performed in several steps, graphically illustrated in Figure 1. The input can be either an audio file or a score fragment, played by the user on a keyboard or sung or whistled into a microphone connected to the computer [13].

- **Musical Storage Environment** From the audio files, note-like attributes are extracted by converting the input into a sequence of note-numbers, i.e. the concatenation of pitch and duration of each input note. Such step is performed by the Symbolic Music Code Extractor module (Figure 1). The conversion uses different pitch-tracking algorithms. If the input is entered from a keyboard or it is a score fragment, conversion is not necessary and the sequence can be directly built.

- **Musical Query Environment** The Feature Extractor converts acoustic input first into an audio feature sequence and then into its related symbolic representation. Then the similarity function is computed.

The system we described allows to organize, manage and utilize information of a heterogeneous set of music source material. This work is a development of the information system described in the context of the "Teatro Alla Scala" project. The improvements regard the use of the graph model of musical data at different levels, the XML format for the representation of musical work and the new user interfaces for the navigation of musical source material. Such a system takes advantage of organizing, managing and utilizing information of a heterogeneous set of music source material through the XML multilayered structure.

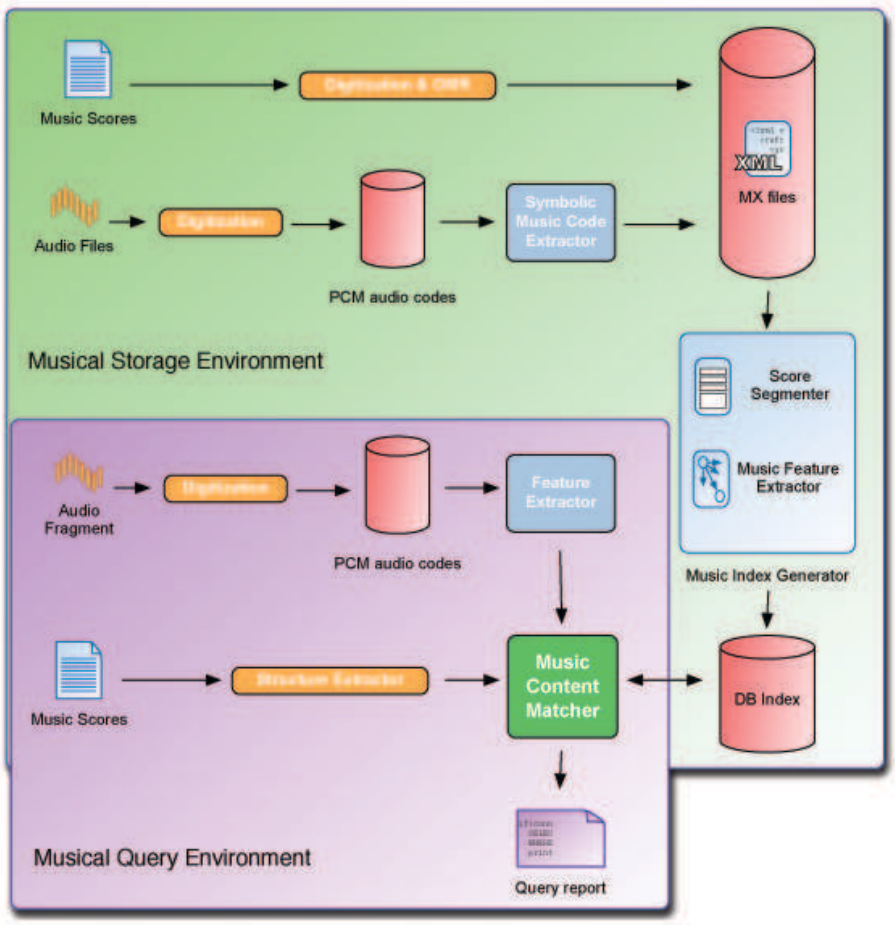

Figure 1: General architecture

The processing phase is characterized also by other feature extraction techniques and parameters typical of audio processing. All those techniques will be analyzed in the next sections.

3 Musical Feature Extraction

A Musical Feature Extraction system is a part contained in a MIR system that works requiring brute force and sophisticated signal-processing technology, which provides objective information about music content [14]. This section will describe the common tasks performed by this system.

3.1 Pitch Tracking

The conventional fashion of organization of music collection using singer's names, album's name, or any other text-based manner is becoming inadequate for effective and efficient usage of the music collection for average users. People sometimes prefer to access the music database by its musical content rather than textual keywords. Content-based music retrieval has thus become an active research area in recent years. Pitch extraction or estimation, more often called pitch tracking is a simple form of automatic music transcription which converts musical sound into a symbolic representation[15][16]. The basic idea of this approach is quite simple. Each note of music (including the query) is represented by its pitch. So a musical piece or segment is represented as a sequence or string of pitches. The retrieval decision is based on the similarity between the query and candidate strings. Pitch is normally defined as the fundamental frequency of a sound. To find the pitch for each note, the input music must first be segmented into individual notes. Segmentation of continuous music, especially humming and singing, is very difficult. Therefore, it is normally assumed that music is monophonic (produced using a single instrument) and stored as scores in the database. The pitch of each note is known. The common query input form is humming. To improve pitch tracking performance on the query input, a pause is normally required between consecutive notes. There are two pitch representations. In the first method, each pitch except the first one is represented as pitch direction (or change) relative to the previous note. The pitch direction is either U(up), D(down) or S(similar). Thus, each musical piece is represented as a string of three symbols or characters. The second pitch representation method represents each note as a value based on a chosen reference note. The value is assigned from a set of standard pitch values that is closest to the estimated pitch. If we represent each allowed value as a character, each musical piece or segment is represented as a string of characters. But in this case, the number of allowed symbols is much greater than the three that are used in the first pitch representation. After each musical piece is represented as a string of characters, the final stage is to find a match or similarity between the strings. Considering that humming is not exact and the user may be interested in find similar musical pieces instead of just the same one, approximate matching is used instead of exact matching. The approximate matching problem is that of string matching with k mismatches. The variable k can be determined by the user of the system. The problem consists of finding all instances of a query string Q = q1q2q3 . . . qm in a reference string R = r1r2r3 . . . rn such that there are

at most k mismatches (characters that are not the same). There are several algorithms that have developed to address the problem of approximate string matching [17]. Both the systems of Muscle Fish LLC [17] and the University Waikato produced good retrieval performance. But the performance depends on the accuracy of pitch tracking of hummed input signals. High performance is only achieved when a pause is inserted between consecutive notes. Since humming is the most natural way to formulate music queries for people who are not trained or educated with music theory. Therefore many researchers have proposed techniques for query-by-humming. Many techniques based on melody matching are proposed, and some of them can support query-by-humming [18] [19].

For a query-by-humming system to work well, reliable pitch detection in the humming is critical. There are 2 types of query-by-humming methods: (1) the method based on music notes segmentation and matching; (2) and the method based on continuous pitch contour matching. The first type of methods [20] requires the user to separate each music note with a short silence or hum with a particular syllable (such as Da). By such restrictions, it is assumed that each individual note can he accurately segmented by using signal energy. The pitch for the segmented note is then estimated. The restrictions, however, make the methods less practical for a real-world music retrieval system, since a user cannot always be aware of the note boundaries particularly when there are tied notes in the melody. 2illustrates the flow of such pitch detection method.

detection method.

The second type of query-by-bumming methods does not impose the above mentioned restrictions. The pitch value for each audio b e(a short time window) is estimated The melody is then represented using a pitch contour, or a time series of pitch values with no music note identities. Music retrieval is done by similarity matching of the pitch contours [21][19]. These approaches have show a better performance than the first type of method. Although note segmentation error does not exist in this method, the reliable pitch detection for each frame is difficult due to various dynamics in the sing voice, like pitch transitions and vocal registers. It can be a hard decision that whether an audio frame has a pitch and how reliable the detected pitch is. We have so far not seen a pitch detection method that is designed for this problem.

[22] presents a pitch tracking method for pitch contour extraction from humming voice.[22] proposes a harmonic analysis technique to efficiently group partials of a harmony in the power spectrum. A harmonic energy can be computed to estimate the reliability of a pitch value for a frame.

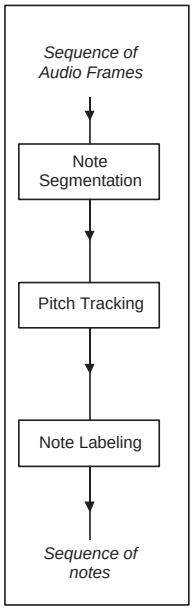

Figure 2: Illustrates the flow of such pitch

The pitch tracking method operates by initially finding the frames with reliable pitch and subsequently finalizing the pitch detection of other frames by region growing.

detection method.

Music and human voice are rich in harmonics. For a signal with a pitch there are quite a number of partials that are multiples of the fundamental frequency. The pitch is measured by the fundamental frequency; however the fundamental frequency may not be outstanding in the power spectrum. The partials can then be explored for robust pitch detection. The proposed techniques analyzes the harmonic structure for each audio frame (about 500 milliseconds). The analysis techniques can help determine whether there is pitch and what is the pitch in the signal. The analysis techniques a power spectrum analysis, peak grouping and harmonic energy estimation.

[23]describes a singing transcription system, which could he divided into two modules. One is for the front-end voicing processing, including voice acquisition, end-point detection and pitch tracking, which deal with the raw singing signal and convert it to a pitch contour. The other is for the melody tracking, which maps the relatively variation pitch level of human singing into accurate music notes, represented as MIDI note number. The overall

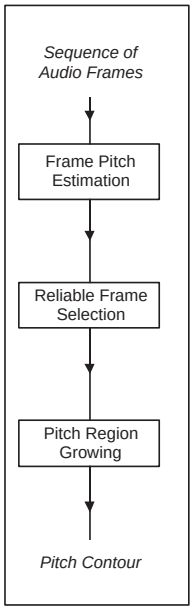

Figure 3: Illustrates the flow of such pitch

system block diagram can be shown as 4. detection method.

The melody tracker is based on Adaptive Round Semitones (ARS) algorithm, which converts a pitch contour of singing voice to a sequence of music notes. The pitch of singing voice is usually much more unstable than that of musical instruments. Furthermore, by adding on the transcription process a heuristic music grammar constraints based on music theory, the error rate can be reduced to the lowest.

3.2 Beat/Tempo Tracking

In this section we will describe beat and tempo tracking contest; to define these contests we want to cite Simon Dixon: *"The task of beat tracking or tempo following is perhaps best described by analogy to the human activities of foot-tapping or hand-clapping in time with music, tasks of which average human listeners are capable. Despite its apparent intuitiveness and simplicity compared to the rest of music perception, beat tracking has remained a difficult task to define, and still more difficult to implement in an algorithm or computer program."*[24] These algorithms should be able to estimate the tempo and the times of musical beats in expressively performed music. The input data may be either digital audio or a symbolic representation of music

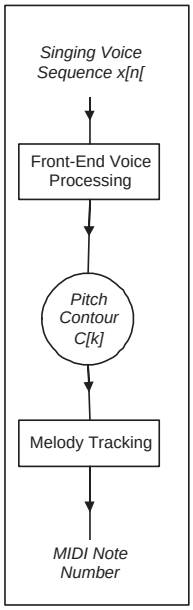

Figure 4: Illustrates the flow of such pitch

such as MIDI.[25] This kind of programs find application in tasks like beat-driven real-time computer graphics, computer accompaniment of a human performer, lighting control and many others. Tempo and beats tracking are directly applicated in MIR systems in fact every songs has a distinctive beats and metronome bpm. It should be noted that these tasks are not restricted to music with drums; in fact the human ear can identify beats and tempo even if the song has not a strong rhythmical accentuation. Obviously these tasks are more difficult in music without drums. The algorithms that implement tempo and beat tracking actually have less accuracy in music without drums. In this contest it is difficult to point out the accuracy rate of the various algorithms because some widespread data set and common methodological evaluation routine are not yet accepted. This situation is partly due to the choice of data set, which often depends on the goals of the system[26]. Various models have been proposed in order to extract the beat from performance data. The primary distinction we want to point out is between real-time and batch algorithms. For example automatic accompaniment systems have to use real-time algorithms. Transcription and analysis software tends to process data off-line, because rhythmically ambiguous sections can be frequently determined analyzing all the beat information found

in the song. Thus the choice between real-time and off-line systems it is directly related to algorithm aim. Actually the beat tracking system works on a two stage models. The first stage is an onset detector, the second one is an interpretative system, which gets the onset detector output and tries to understand the tempo of the song and the correct beat position. For music with drums to develop an onset detector system the simplest way is to high pass the signal and then clustering thae filtered signal and introduce a threshold to the cluster energies. Obviously this trivial algorithm works not very well but it permits to understand that a drums beat has a spectral frequency with rich high components. About the interpretative system is possible to find many different solutions like agents model [27][24], or probabilistic systems [28]and so on. We will present two different solutions to the interpretative problem that use agents model. In the next sections, four different tempo-beat tracking algorithms (one solely dedicated to Tempo tracking) will be described. The algorithms [24][27] work in PCM format, the algorithms presented in [29][30] are dedicated to MP3 standard.

3.2.1 The Goto Algorithm

The first algorithm was developed by Masataka Goto[27]. This algorithm is based on the previous Goto and Marauroka works, one for music with drums[31] and the other for music without drums[32]. In figure 5 is presented the algorithm scheme.

This algorithm describes a real-time beat-tracking system that can deal with the audio signals of popular-music compact discs in real time regardless of whether or not those signals contain drum sounds. The system can recognize the hierarchical beat structure comprising the quarter-note level (almost regularly spaced beat times), the half-note level, and the measure level.[27] The algorithm consists of two components: the first one extracts the musical elements from audio signals; the second component tries to understand the beat structure. The first step detects three kinds of musical elements as the beat tracking cues:

1. Onset times
2. Chord changes
3. Drum patterns

These elements are extracted from the frequency spectrum calculated with the FFT (1024 samples) of the input (16 bit / 22.05 kHz) using the Hanning window. The frequency spectrum is subdivided into 7 critical bands.

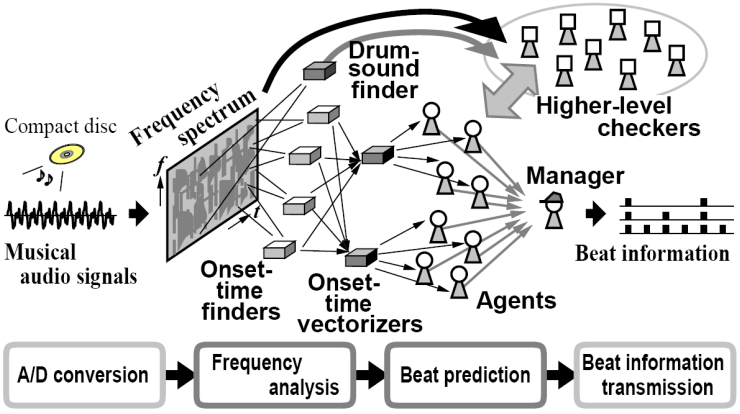

Figure 5: Overview of the Beat tracking system proposed in [27]

The onset times can be detected by a frequency analysis process that takes into account the rapidity of an increase in power and the power present in nearby time frequency bands. The results of this algorithm are stored in an onset-time vectors. By using autocorrelation and cross-correlation of the onset-time vectors, the model determines the inter-beat interval and predicts the next beat time. The output of this stage is the provisional beat times vector. The provisional beat times obtained is just a single hypothesis of the quarter-note level. To calculate the Chord-changes the frequency spectrum is sliced into the point indicated by the provisional beat times component. In this point the dominant frequencies of the spectrum are estimated by using a histogram of frequency components. Chord-change possibilities are then obtained by comparing dominant frequencies between adjacent point indicated by the provisional beat times element. The drums patterns are restricted to bass drum and snare. A drum-sound finder detects the onset time of the bass drum; the onset time of the snare is found using a noise detector. The second step handled ambiguous situations when the beat-tracking cues are interpreted; a multiple-agent model in which multiple agents examine various hypotheses of the beat structure in parallel was developed. Each agent uses its own strategy and makes various hypotheses. The agent manager gathers all hypotheses and then determines the final output on the basis of the most reliable one. It should be noted in 5 that the drums line is not mandatory, but it helps the algorithm furnishing more information.

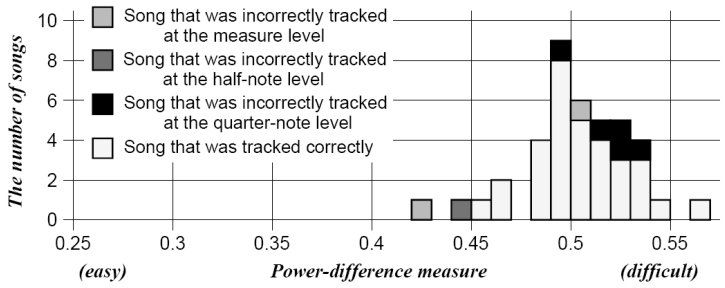

Figure 6: Histogram for 40 songs without drum-sounds[27].

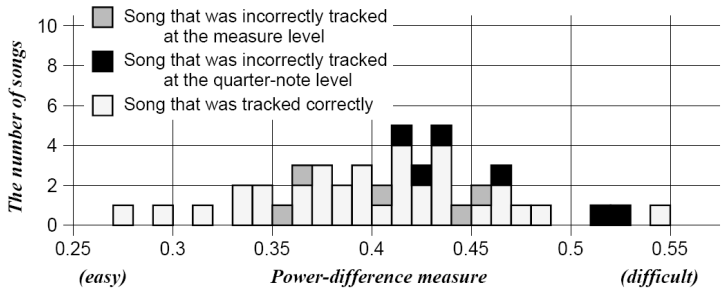

Figure 7: Histogram for 45 songs with drum-sounds[27].

In figure 6 are presented the algorithm results on music without drums and in figure 7 the results on music with drums. In [27] is proposed a quantitative measure of the rhythmic difficulty, called the *power-difference measure* (see also [32] for further information) that considers differences between the power on beats and the power on other positions. This measure is defined as the mean of all the normalized power difference $diff_{pow}(n)$ in the song:

$$diff_{pow}(n) = 0.5\frac{pow_{other}(n) - pow_{beat}(n)}{max(pow_{other}(n), pow_{beat}(n))} + 0.5$$

where $pow_{beat}(n)$ represents the local maximum power on the n-th beat and $pow_{other}(n)$ represents the local maximum power on positions between the n-th beat and (n + 1)-th beat. The power-difference measure takes a value between 0 (easiest) and 1 (most difficult). For a regular pulse sequence with a constant interval, for example, this measure takes a value of 0[27].

3.2.2 The Dixon Algorithm

Dixon in [24] describes an audio beat tracking system using multiple identical agents, each of which represents a hypothesis of the current tempo and synchronization (phase) of the beat. The system works well for pop music, where tempo variations are minimal, but does not perform well with larger tempo changes. [33] extends this work to provide for significant tempo variations as found in expressive performances of classical music. They use the duration, amplitude and pitch information available in MIDI data to estimate the relative rhythmic salience (importance) of notes, and prefer that beats coincide with the onsets of strong notes. In this paper, the salience calculation is modified to ignore note durations because they are not correctly recorded in the data. Processing is performed in two stages: tempo induction is performed by clustering of the time intervals between near note onsets, to generate the initial tempo hypotheses, which are fed into the second stage, beat tracking, which searches for sequences of events which support the given tempo hypothesis. Agents perform the search. Any agent represents a hypothesized tempo and beat phase, and try to match their predictions to the incoming data. The closeness of the match is used to evaluate the quality of the agents' beat tracking, and the discrepancies are used to update the agents' hypotheses. Multiple reasonable paths of action result in new agents being created, and agents are destroyed when they duplicate each other's work or are continuously unable to match their

predictions to the data. The agent with the highest final score is selected, and its sequence of beat times becomes the solution.[26]

3.2.3 Beat Tracking in Compressed Domain

In order to better understand the rest of the section, a brief overview of the basic concepts about MP3 audio standard is provided. We focus on the Window-Switching pattern and its onset detector behavior. Further information about MPEG standards can be found in [34],[35],[36] and [37]. MP3 uses 4 different MDCT window types: long, long-to-short, short, short-to-long indexed with 0,1,2,3 respectively. The long window, allows greater frequency resolution for audio signals with stationary characteristics, while the short one provides better time resolution for transients [37]. In short blocks there are 3 sets of window values for a given frequency, in a window there are 32 frequency sub bands, further subdivided into 6 finer sub bands by MDCT. 3 short windows are then grouped in one granule. The values are ordered by frequency, then by window. The switch between long and short blocks is not instantaneous. The two-window type long-to-short and short-to-long serves to transition between long and short window types. Because MDCT processing of a sub band signal provides better frequency resolution, it consequently has poorer time resolution. The quantization of MDCT values will cause errors that are spread over the long time window so it is more likely that this quantization will produce audible distortions. Such distortions usually manifest themselves as pre-echo because the temporal masking of noise occurring before a given signal is weaker than the masking of noise after [37][38]. This situation appears frequently in music, with strong drums line, when the drummer plays snare or bass drum and than the Window-Switching Pattern may be used as simple onset detector with an high threshold. Wang in [29] proposes the Window-Switching Pattern (WSP) as information to refine the output of an MDCT[1] coefficients analysis in a beat tracking contest in order to perform better error concealment for music transmission in noisy channel. The algorithm extracts the sub-band MDCT coefficients and than it calculates any sub-band energy. A search window is defined. The basic principle of onset selection is setting a proper threshold for the extracted sub band energy values. The local maxima within a search window, which fulfils certain conditions, are selected to be beat candidates. This process is performed in each band separately. Than the WSP identified by the encoder is used to refine the MDCT analysis. The algorithm extracts the sub-band MDCT coefficients and than it

[1] Modified Discrete Cosine Transform

calculates any sub-band energy. A search window is defined. The basic principle of onset selection is setting a proper threshold for the extracted sub band energy values. The local maxima within a search window, which fulfils certain conditions, are selected to be beat candidates. This process is performed in each band separately. Than the WSP identified by the encoder is compared with the beat candidates. A Statistical model selects the correct beat from the beat candidate set. [30] presents a template matching technique (based on WSP only) to reach a general purpose tempo tracker on music with drums. Because the WSP is structured coherently with the drums line it is possible to compare this pattern with a simple template made up by a vector filled with 0 and with a 1 value where is a metronome beat. Any elements of this array represent an MP3 granule. This array has to be matched with the WSP found by the MP3 encoder. An estimation function is required. This function has to yield the distance between the metronome template examined and the real MP3 window-switching pattern. In figure 8 the metronome template is represented by the darkest line with -1 peak. Any peak is a metronome beat. In this figure is clear the WSP structure is coherent with song time even if the song has a very complex drums line. A first implementation reached a correct bpm recognition in the 50% songs and in another 30% it is possible to estimate the correct bpm by the program results. The algorithm fails in the 20% of the songs. These values come from an experimentation finalized to demonstrate the WSP capabilities. Obviously the WSP alone is not sufficient to act like a beat tracker but it is adequate to solve tempo tracking contest.

3.3 Score Extraction

Score extraction can be defined as the act of listening to a piece of music and to write down the score for the musical events that constitute the piece. This implies the extraction of specific features out of a musical acoustic signal, resulting in a symbolic representation that comprises notes, pitches, timings, dynamics, timbre and so on. The score extraction task is not simple and intuitive like beat tracking. People without musical education find this task very difficult and they could not be able to perform it. The automatic transcription of music is a well-understood problem just for monophonic music. To transcribe monophonic music many solution algorithm have been proposed, including time-domain techniques based on zero-crossing and autocorrelation, as well as frequency frequency-domain based on the discrete Fourier transform and the cepstrum[39][40][41]. These algorithms proved to be reliable and commercially applicable. In polyphonic music transcription

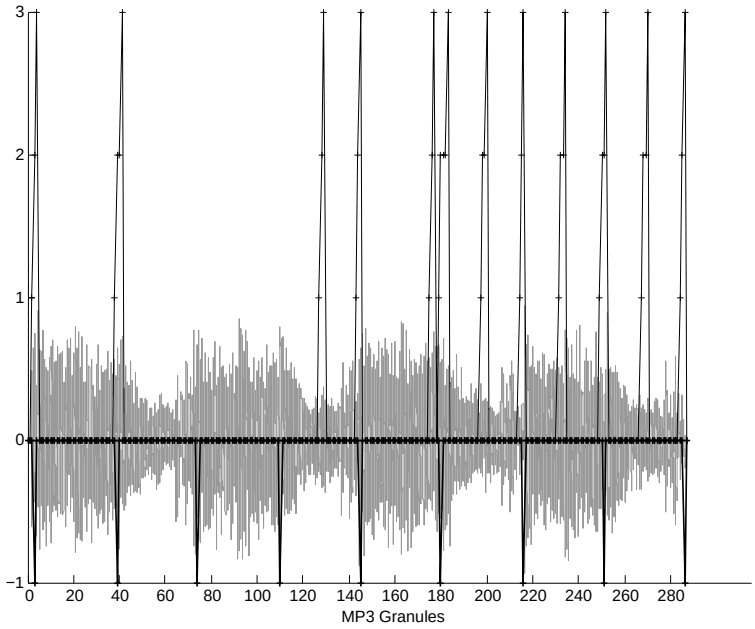

Figure 8: The comparison between song waveform (gray) and its corresponding MP3 window-switching pattern (black line). Horizontal axis is the MP3 granule index. The four window types (long, long-to-short, short and short-to-long) are indexed with 0, 1, 2 and 3 respectively. The metronome template is represented by black line with -1 peaks.[30]

the situation is not so positive. These results are not so encouraging because of the increased complexity of the signals in question. It should be noted that score extraction is a composed task. In fact we can subdivide this problem in a set of different tasks: pitch tracking to get information about the notes, beat tracking to understand the correct rhythmical figures, source separation to separate a single instrument part from the other, timbre extraction to understand which instruments have to be insert in the score and so on. Many algorithms have been proposed to solve the problem constrained to mono-timbrical music (i.e. a piano score with many voices simultaneously). These algorithms are very similar to the algorithm presented in 3.1 using the same low-level feature extractor but with a second stage dedicated to interpret the low-level results. The low-level features are often identified with the term mid-level representation. A good mid-level representation for audio should be able to separate individual sources, be invertible in a perceptual sense, reduce the number of components and reveal the most important attributes of the sound. Current methods for automatic music transcription are often based on modeling the music spectrum as a sum of harmonic sources and estimating the fundamental frequencies of these sources. This information constitutes an ad hoc mid-level representation. In order to successfully create a system for automatic music transcription; the information contained in the analyzed audio signal must be combined with knowledge of the structure of music[42].

3.4 Genre Extraction

Musical genres are categorical descriptions that are used to characterize music in music stores, radio stations and now on the Internet. Although the division of music into genres is somewhat subjective and arbitrary there are perceptual criteria related to the texture, instrumentation and rhythmic structure of music that can be used to characterize a particular genre. Humans are remarkably good at genre classification as investigated in [43] where it is shown that humans can accurately predict a musical genre based on 250 milliseconds of audio. This finding suggests that humans can judge genre using only the musical surface without constructing any higher level theoretic descriptions as has been argued in [44]. Up to now genre classification for digitally available music has been performed manually. Therefore techniques for automatic genre classification would be a valuable addition to the development of audio information retrieval systems for music.

[45] address the problem of automatically classifying audio signals into an hierarchy of musical genres. More specifically, three sets of features for rep-

resenting timbral texture, rhythmic content and pitch content are proposed. Although there has been significant work in the development of features for speech recognition and musicspeech discrimination there has been relatively little work in the development of features specifically designed for music signals. Although the timbral texture feature set is based on features used for speech and general sound classification, the other two feature sets (rhythmic and pitch content) are new and specifically designed to represent aspects of musical content (rhythm and harmony). The performance and relative importance of the proposed feature sets is evaluated by training statistical pattern recognition classifiers using audio collections collected from compact disks, radio, and the Web. Audio signals can be classified into an hierarchy of music genres, augmented with speech categories. The speech categories are useful for radio and television broadcasts. Both whole-file classification and real-time frame classification schemes are proposed.

[45] identifies and review two different approaches to Automatic Musical Genre Classification. The first approach is prescriptive, as it tries to classify songs in an arbitrary taxonomy, given a priori. The second approach adopts a reversed point-of-view, in which the classification emerges from the songs.

- **Prescriptive approach:** it makes the same assumption that a genre taxonomy is given and should be superimposed on the database of songs). They all proceed in two steps:

  1. Frame-based Feature extraction: the music signal is cut into frames, and a feature vector of low-level descriptors of timbre, rhythm, etc. is computed for each frame.
  2. Machine Learning/Classification: a classification algorithm is then applied on the set of feature vectors to label each frame with its most probable class: its genre. The class models used in this phase are trained beforehand, in a supervised way.

  The features used in the first step of automatic, prescriptive genre classification systems can be classified in 3 sets: timbre related, rhythm related and pitch related. There have been numerous attempts at extracting genre information automatically from the audio signal, using signal processing techniques and machine learning schemes.

- **Similarity relations approach:** The second approach to automatic genre classification is exactly opposite to the prescriptive approach just reviewed. Instead of assuming that a genre taxonomy is given a priori, it tries to emerge a classification from the database, by clustering songs

according to a given measure of similarity. While the prescriptive approach adopts the framework of supervised learning, this second point-of-view is unsupervised. Another important difference is that in the first approach, genre classifications are considered as natural and objective, whereas in this approach it is similarity relations which are considered as objective.

4 Audio - Score automatic synchronization

Music language is made up of many different and complementary aspects. Music is the composition itself as well as the sound a listener hears, and is the score that a performer reads as well as the execution provided by a computer system. The encoding formats commonly accepted and employed are often characterized by a partial perspective of the whole matter: they describe data or metadata for score, audio tracks, computer performances of music pieces, but they seldom encode all these aspects together. Nowadays we have at our disposal many encoding formats aimed at a precise characterization of only one (or few) music aspect(s). For example, MP3, AAC and PCM formats provide ways to encode audio recordings; MIDI represents among other things a well known standard for computerdriven performance; TIFF and JPEG files can result from a scanning process of scores; NIFF, Enigma, Finale formats are aimed at score typing and publishing. The first problem to face is finding a comprehensive way to encode all these different aspects in a common framework, without repudiating the accepted formats. An important advantage of such effort is keeping together all the information related to a single music piece, in order to appreciate the richness of heterogeneous representations of music (aural, visual, logical, structural descriptions). But this key advantage has an interesting consequence: the possibility to create a strongly interconnected and synchronized environment to enjoy music. The purpose of our MXDemo is illustrating the full potentialities of an integrated approach to music description. This goal can be achieved thanks to three cooperating elements:

1. a comprehensive XML-based format to encode music in all its aspects.
2. a software environment aimed to the integrated representation. The software application will provide a graphic interface to read, watch, and listen to music, keeping the different levels synchronized.
3. an automatic system to synchronize music score and the related audio signal.

About point number one, some example of XML-based format to encode music are presented in [46] [47]. They are not discussed in this section. About the second one, [48] proposes a generic service for realtime access to context-based music information such as lyrics or score data. In our web-based client-server scenario, a client application plays back a particular (waveform) audio recording. During playback, the client connects to a server which in turn identifies the particular piece of audio as well as the current playback position. Subsequently, the server delivers local, i.e., position specific, context-based information on the audio piece to the client. The client then synchronously displays the received information during acoustic playback.[48] demonstrates how such a service can be established using recent MIR (Music Information Retrieval) techniques such as audio identification and synchronization. [49] propose the MXDemo,a stand-alone software which illustrates the full potentialities of an integrated approach to music description.

In order to solve the third point, all present approaches to score-to audio synchronization proceed in two stages: In the first stage, suitable parameters are extracted from the score and audio data streams making them comparable. In the second stage, an optimal alignment is computed by means of dynamic programming (DP) based on a suitable local distance measure. Turetsky et al. [7] first convert the score data (given in MIDI format) into an audio data stream using a synthesizer. Then, the two audio data streams are analyzed by means of a short-time Fourier transform (STFT) which in turn yields a sequence of suitable feature vectors. Based on an adequate local distance measure permitting pairwise comparison of these feature vectors, the best alignment is derived by means of DTW. The approach of Soulez et al. [50] is similar to [7] with one fundamental difference: In [7], the score data is first converted into the much more complex audio formatin the actual synchronization step the explicit knowledge of note parameters is not used. Contrary, Soulez et al. [50] explicitly use note parameters such as onset times and pitches to generate a sequence of attack, sustain and silence models which are used in the synchronization process. This results in a more robust algorithm with respect to local time deviations and small spectral variations. Since the STFT is used for the analysis of the audio data stream, both approaches have the following drawbacks: Firstly, the STFT computes spectral coefficients which are linearly spread over the spectrum resulting in a bad low-frequency resolution. Therefore, one has to rely on the harmonics in the case of low notes. This is problematic in polyphonic music where harmonics and fundamental frequencies of different notes often coincide. Secondly, in order to obtain a sufficient time resolution one has

to work with a relatively large number of feature vectors on the audio side. (For example, even with a rough time resolution of 46 ms as suggested in [7] more than 20 feature vectors per second are required.) This leads to huge memory requirements as well as long running times in the DTW computation. In the approach of Arifi et al. [51], note parameters such as onset times and pitches are extracted from the audio data stream (piano music). The alignment process is then performed in the score-like domain by means of a suitably designed cost measure on the note level. Due to the expressiveness of such note parameters only a small number of features is sufficient to solve the synchronization task, allowing for a more efficient alignment. One major drawback of this approach is that the extraction of score-like note parameters from the audio data a kind of music transcriptionconstitutes a difficult and time-consuming problem, possibly leading to many faultily extracted audio features. This makes the subsequent alignment step a delicate task. [52] presents an algorithm, which solves the synchronization problem accurately and efficiently for complex, polyphonic piano music. In a first step, they extract from the audio data stream a set of highly expressive features encoding note onset candidates separately for all pitches. This makes computations efficient since only a small number of such features is sufficient to solve the synchronization task. Based on a suitable matching model, the best match between the score and the feature parameters is computed by dynamic programming (DP). To further cut down the computational cost in the synchronization process, they introduce the concept of anchor matches, matches which can be easily established. Then the DP-based technique is locally applied between adjacent anchor matches.

References

[1] Goffredo Haus and Emanuele Pollastri. A multimodal framework for music inputs (poster session). In *ACM Multimedia*, pages 382–384, 2000.

[2] François Pachet. Content management for electronic music distribution. *Commun. ACM*, 46(4):71–75, 2003.

[3] K.D. Martin. Automatic transcription of simple polyphonic music: Robust front end processing. Technical Report Technical Report No. 399, M.I.T. Media Laboratory Perceptual Computing Section, 1996.

[4] Eric D. Scheirer. *Using Musical Knowledge to Extract Expressive Performance Information from Audio Recordings.* In Readings in Computational Auditory Scene Analysis, 1997.

[5] E.Wold et al. Content-based classification, search, and retrieval of audio. *IEEE Multimedia*, Vol. 3, No. 3:pp. 2736, 1996.

[6] J. Logan A. Ghias and D. Chamberlin. Query by humming. In *Proceedings of ACM Multimedia 95*, pages 231–236, 1995.

[7] Q. Tian. Y. Zhu, M Kankanhalli. Similarity matching of continuous melody contours for humming querying of melody databases. In *International Workshop on Multimedia Signal Processing*, 2002.

[8] E. Pollastri. A pitch tracking system dedicated to process smging voice for music retrieval. In *IEEE International Conference on Multimedia and Expo*, 2002.

[9] H. Lee J. R. Jang and M. Kao. Content-based music retrieval using linear scaling and branch-and-bound tree search. In *Proceedings of IEEE International Conference on Multimedia and Expo.*, 2001.

[10] Y. Zhu and MS Kankanhalli. Robust and efficient pitch tracking for query-by-humming. *Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on*, 3:1586–1590, 2003.

[11] C. Wang, R. Lyu, and Y. Chiang. A robust singing melody tracker using adaptive round semitones (ars). *Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis*, 1:549–554, 2003.

[12] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39 – 58, March 2001.

[13] Nicolas Scaringella and Giorgio Zoia. A real-time beat tracker for unrestricted audio signals. In *SMC'04 Conference Proceedings*, Paris, France, October 2004.

[14] S. Dixon. An empirical comparison of tempo trackers. *Proceedings of the 8th Brazilian Symposium on Computer Music*, 2001.

[15] Masataka Goto. An audio-based real-time beat tracking system for music with or withoutdrum-sounds. *Journal of New Music Research*, 30(2):159 – 171, May 2001.

[16] J.C.Sethares W.A. Sethares, R.D. Morris. Beat tracking of musical performances using low-level audio features. *IEEE Transactions On Speech and Audio Processing*, 13(2):275 – 285, March 2005.

[17] Ye Wang and Miikka Vilermo. A compressed domain beat detector using mp3 audio bitstreams. In *MULTIMEDIA '01: Proceedings of the ninth ACM international conference onMultimedia*, pages 194–202, Ottawa, Canada, 2001. ACM Press.

[18] A. D'Aguanno, G. Haus, and G. Vercellesi. Mp3 window-switching pattern preliminary analysis for general purposes beat tracking. In *Proceedings of the 120th AES Convention*, Paris, France, 20-23 May 2006.

[19] M. Goto and Y Muraoka. Music understanding at the beat level real-time beat tracking for audio signals. *D. F. Rosenthal & H. G. Okuno, (Eds.), Computational AuditoryScene Analysis*, page 157176, 1998.

[20] M Goto and Y. Muraoka. Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions. *Speech Communication*, 27(34):311335, 1999.

[21] S. Dixon and E. Cambouropoulos. Beat tracking with musical knowledge. *ECAI 2000: Proceedings of the 14th European Conference on Artificial Intelligence*, pages 626–630, 2000.

[22] ISO/IEC International Standard IS 11172-3. Information technology - coding of moving pictures and associated audiofor digital storage media at up to about 1.5 mbits/s - part 3: Audio.

[23] ISO/IEC International Standard IS 13818-3. Information technology - generic coding of moving pictures and associatedaudio, part 3: Audio.

[24] D. Noll. Mpeg digital audio coding. *IEEE Signal Processing Magazine*, 14(5):59–81, 1997.

[25] Davis Pan. A tutorial on mpeg/audio compression. *IEEE Multimedia*, 2(2):60 – 74, Summer 1995.

[26] Dou Weibei Hou Zhaorong and Dong Zaiwang. New window-switching criterion of audio compression. In *Multimedia Signal Processing, 2001 IEEE Fourth Workshop on*, pages 319–323, Cannes, France, October 2001.

[27] J. C. Brown. Musical fundamental frequency tracking using a pattern recognition method. *Journal of the acoustical Society of America*, 92(3):1394–1402, 1992.

[28] J.C. Brown and M.S. Puckette. A high resolution fundamental frequency determination based on phase changes of the fourier transform. *Journal of the acoustical Society of America*, 94:662–667, 1993.

[29] J. C. Brown and Bin Zhang. Musical frequency tracking using the methods of conventional and narrowed autocorrelation". *Journal of the acoustical Society of America*, 89(5):2346–2354, 1991.

[30] A.P. Klapuri. Automatic music transcription as we know it today. *Journal of New Music Research*, 33(3):269–282, 2004.

[31] F. Pachet and D. Cazaly. A classification of musical genre. In *RIAO Content-Based Multimedia Information Access Conference*, 2000.

[32] S. Davis and P. Mermelstein. Experiments in syllable-based recognition of continuous speech. *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 28:pp. 357366, 1980.

[33] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING*, VOL. 10, NO. 5:293–302, 2002.

[34] Goffredo HAUS and Maurizio LONGARI. Towards a symbolic/time-based music language based on xml. In *Proceedings of the First International IEEE Conference on Musical Applications Using XML (MAX2002)*, 2002.

[35] Perry ROLAND. The music encoding initiative (mei). In *Proceedings of the First International IEEE Conference on Musical Applications Using XML (MAX2002)*, 2002.

[36] F. Kurth, M. Muller, A. Ribbrock, T. Roder, D. Damm, and C. Fremerey. A prototypical service for real-time access to local context-based music information. *ISMIR, Barcelona, Spain*, 2004.

[37] HAUS Goffredo LUDOVICO-Luca Andrea e VERCELLESI Giancarlo BARATE', Adriano. Mxdemo: a case study about audio, video, and score synchronization. In *Proceedings of IEEE Conference on Automatic Production of Cross Media Content for Multi-channel Distribution (AXMEDIS)*, pages pages 45–52, 2005.

[38] F. Soulez, X. Rodet, and D. Schwarz. Improving polyphonic and poly-instrumental music to score alignment. *4th International Conference on Music Information Retrieval*, pages 143–148, 2003.

[39] Clausen M. Kurth-F. Muller M. Arifi, V. *Automatic Synchronization of Musical Data: A Mathematical Approach.* MIT Press, 2004.

[40] Meinard Muller Frank Kurth Tido Roder. Towards an efficient algorithm for automatic score-to-audio synchronization. In *5th International Conference on Music Information Retrieval, ISMIR 2004*, Barcellona, Spain, October 2004.