From MIDI to Traditional Musical Notation

Emilios Cambouropoulos

Austrian Research Institute for Artificial Intelligence

Schottengasse 3, A-1010 Vienna, Austria

emilios@ai.univie.ac.at

Abstract

In this paper a system that is designed to extract the musical score from a MIDI performance is described. The proposed system comprises of a number of modules that perform the following tasks: identification of elementary musical objects, calculation of accent (salience) of musical events, beat induction, beat tracking, onset quantisation, streaming, duration quantisation and pitch spelling. The system has been applied on 13 complete Mozart sonata performances giving very encouraging results.

Introduction

A system that attempts to extract the musical surface (i.e. a symbolic representation of notes in terms of quantised onsets and durations, and correctly spelled pitches) from a polyphonic MIDI performance is herein described. This system was developed as a means for obtaining the scores (in a symbolic machine-readable format) of a large number of performed piano works in the context of the project: 'Artificial Intelligence Models of Musical Performance'. Working on a score-extraction project would not only provide a useful tool for obtaining symbolic scores (optical recognition techniques are probably a more obvious candidate for this task) but would additionally give rise to invaluable insights into the relation of a musical performance to its corresponding musical score. In general, however, score-extraction techniques are indispensable for a plethora of applications that process performed MIDI input (e.g. music notation packages, interactive musical performance systems etc.).

The proposed system comprises of a number of modules that perform the following tasks: identification of elementary musical objects, calculation of accent (salience) of objects, beat induction, beat tracking, onset quantisation, streaming, duration quantisation and pitch spelling. The aim of this paper is to highlight a number of issues relating to score extraction and to give a quite broad understanding of problems relating to the development of and interaction among components that are necessary for score extraction tasks. Figure 1 gives an overall outline of the score extraction system.

The above system has been applied - to this date - to the midifiles of 13 complete sonatas by W.A.Mozart performed by a professional Viennese pianist. The system performs well in the above tasks. However, it does make mistakes. For instance, the beat tracker may add a few extra beats if a fermata is encountered, or it may shift temporarily off-beat if very weak notes appear on the beat (although this may be corrected by the quantisation preference process); the duration quantisation module strongly avoids rests and also goes wrong when a streaming mistake has been made; occasional pitch spelling mistakes may occur even though the pitch spelling module is in general very successful. Such mistakes can be corrected either interactively during the score-extraction process or manually after the symbolic representation has been generated.

The various components of proposed score extraction system are described in more detail in the sections below.

Salience of Musical Objects

As a first task elementary musical 'objects' are identified in the MIDI file and corresponding accent values are calculated for each of them.

Identification of Elementary Musical Objects

Musical 'objects' such as chords, arpeggiated chords, trills, mordents etc. are initially identified in the raw midifile. It is hypothesised that such collections of notes are perceived as 'wholes' before being broken down to their constituent parts. For instance, in Figure 2, if the arpeggiated chord is not identified in advance, then a quantisation procedure is likely to give rise to a wrong result such as the one depicted in the last example in Figure 2.

It is necessary, as a first step, to determine *when* a number of independent notes are close enough to be considered constituents of a chord. Empirical research has shown that two tones are heard as being synchronous if their onset differences are less than roughly 40 ms - for more than two tones this threshold is higher but usually not more than 70 ms (Handel 1989:214). We have used the threshold of 70 ms to convert the MIDI data into chords.

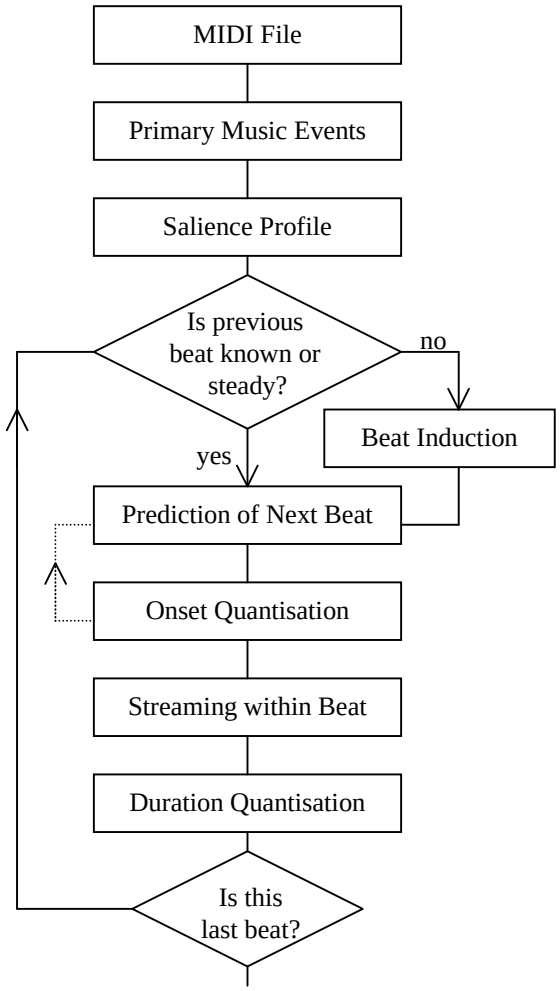

Figure 1: Overall the score extraction system (pitch spelling is a separate module).

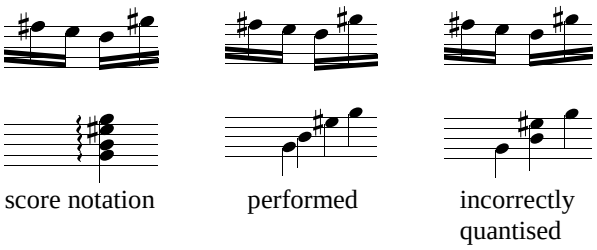

Figure 2: Arpeggiated chord: as notated in the score, as actually performed and as incorrectly quantised (should be identified early on as a musical 'object')

Musical knowledge is essential for identifying other kinds of objects such as trills, grace-notes, mordents, arpeggiated chords and so on. The exact definition of such objects and the development of routines that can compute them is not trivial. For instance, a rapid alternation of two pitches (at a 2nd interval) may belong to a trill or to a very fast melodic passage; note duration is not sufficient for detecting grace-notes as staccato notes have very short durations as well. As this is a large topic in its own right it will not be discussed further in this paper.

Musical Accents

For each musical object an 'accent' strength is computed that indicates the relative salience of the object. Musical salience is determined by a number of factors such as note duration, dynamics, pitch, harmony, cadences and so on (for instance, longer notes or harmonically more stable notes are more salient). In the current version of this system the following three salience factors have been taken into account: a. duration in ms of notes in each event, b. dynamic value (MIDI velocity) and c. pitch value. Notes with longer durations, higher dynamic values or lower pitch are considered to be more salient. The rationale behind the last factor is that notes in lower voices (especially the bass line) tend to be less ornamented and more accurate in terms of keeping time whereas higher voices (e.g. melody) have more expressive flexibility and variation. The salience strength of events (e.g. chords) is taken to be the sum of the strengths of their constituent notes.

A possible salience strength function for each note is proposed:

$f(d,p,v) = (v/p) \cdot d$ where:

d is duration in ms (*d* = Offset - Onset),
v is dynamic value (MIDI-velocity), $30 < v < 90$
p is pitch (MIDI-pitch), $30 < p < 60$

Values for *v* and *p* smaller or greater than the minimum or maximum of each range are taken to be equal to the minimum or maximum respectively. The most significant factor into the above function is the duration of notes as this can take a wide range of values (from roughly 100 ms to a few seconds). The dynamic and pitch factors become more influential when notes have relatively similar durations.

In the next section, it will be shown that knowledge of such accent strengths can enhance beat-processing applications.

Finding the Beat

Beat is taken to refer to a regular pulse perceived when listening to music - this is usually the temporal level at which listeners tap their feet or clap their hands. The action of finding an appropriate beat rate (tempo) will be referred to as *beat-induction* whereas the task of following the beat will be referred to as *beat-tracking*. Ordinary human

listeners are competent in performing these tasks; equivalent performance on the computer has proved however remarkably hard to achieve.

Perhaps computer systems are quite weak in simulating this behaviour (except for simple strict tempo cases) because they often do not have access to musical knowledge that can adjust the tracking process when significant local variations in the beat occur. Most beat-tracking and beat-induction systems rely solely on note onsets or inter-onset intervals (Desain and Honing, 1992; Rosenthal 1992; Rowe 1992). It has often, however, been proposed in theoretical work that finding the beat (the lowest meaningful level of metrical structure) involves matching a regular grid to the *accentuation* structure of a music work (e.g. Lerdahl & Jackendoff, 1983; Povel & Essens, 1985).

It is herein hypothesised that more salient (accented) musical events tend to have a stronger influence on beat-tracking processes than less important ones, i.e. beats tend to fall on more accented/salient events.

Beat Induction

Extracting an adequate beat is achieved by 'looking' into a pre-specified time window for the most common inter-onset interval. All the possible IOIs formed by all the onsets within the window (e.g. 3 seconds) are calculated. For each IOI, a salience value is attached that takes into account the accents of its two delimiting notes (in this implementation the IOI salience is simply the sum of the two accents). The IOIs are sorted by size and 'clustered' into small overlapping time bins (70ms). For each time bin the sum of all the IOI saliences is calculated. The cluster with the highest salience value determines the local tempo (i.e. inter-beat interval).

The IOI clustering algorithm is described in detail in (Dixon et al. 2000). The current version differs from that algorithm in that it takes into account the salience of each IOI.

The beat induction module is activated at the beginning of the piece and whenever the beat variance of preceding beats exceeds a certain value (meaning that the beat-tracker is 'lost').

Beat Tracking

For a given moment in a piece a prediction is made for the next beat based on the previous few beats (e.g. average of the last three beats). The next beat position is calculated using a Gaussian function that is centered at the predicted time point and that has an overall width equal to the beat value. The 'steepness' of the function depends on the observed variance of the previous beats. The accent strengths of musical objects within the prediction range are adjusted according to the function and the greatest value is selected. If there are no events in the predicted window, then a beat position is placed at the predicted point and beat-tracking proceeds from there on.

This method takes into account the relative salience of musical events (see figure 3). For instance, a weaker event that falls closer to the predicted beat position may be disregarded as the algorithm may select a stronger event that has a greater distance to it. This way, the beat tracker can track correctly sections with greater temporal deviations (more rubato). In a recent study (Dixon et al, 2000) it is shown that the performance of a multi-agent beat tracking system improved from 75% to over 90% when musical event accents were taken into account (the system was applied to the same Mozart data).

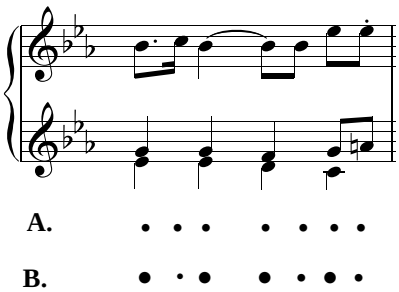

Figure 3: Sequence A illustrates the event onsets, and sequence B the accents of musical events. The proposed beat-tracking algorithm is applied to sequence B.

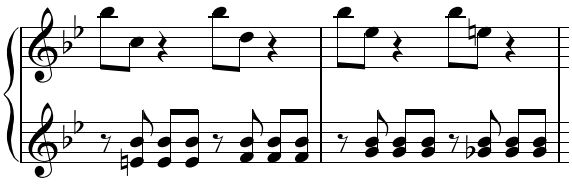

Figure 4: In this example the beat may erroneously shift forward by one eighth note as the notes on every other actual beat are less accented than their succeeding three note chords.

This beat-tracking algorithm can make mistakes especially in cases where weak events appear on beats and strong events appear very close to them. In Figure 4 the beat-tracker may shift forward by one eighth note; if, however, the previous tempo has been very steady it is likely that this section will be tracked correctly as well as the prediction function will be very sharp. The beat-tracker usually recovers from mistakes as soon as strong musical events indicate the correct beat (there are cases however where the beat-tracker goes astray). Points where the beat tracker has gone wrong can be corrected interactively by the user; after a particular beat is corrected the system is allowed to track the beat again from that point onwards.

Time Quantisation

Once a beat is detected, an attempt is made to quantise all the onsets and durations of notes within the range of this beat and its preceding beat.

Onset Quantisation

Onsets within the range of two successive beats are quantised by selecting the time grid (multiples of 2 and 3) that fits best to the observed onset values. Rather than 'moving' the observed event onsets to the closest points of the quantisation grids and then selecting the grid for which the minimum deviation error is computed (as many commercial quantisers do), quantisation is applied to each inter-event interval. The last inter-event interval in the beat equals to 1 minus the sum of the previous quantised beat fractions (this way a round-off error is avoided). A goodness value that is inversely related to the deviation error is attached to each subdivision level. The quantisation subdivision level that gives the highest goodness value is selected (further discussion on the quantisation problem can be found in Desain and Honing, 1992).

The quantisation of onsets within the beat can affect beat tracking as well. If the beat-tracker does not have any strong indication of which of two equally accented events is on the beat, it can examine both possibilities selecting the one that gives a 'better' quantisation (e.g. a beat event that results in a quantisation of five equally spaced onsets is less preferred to another event that results in a quantisation of four equally spaced ones).

Duration Quantisation

The actual offsets of notes are of hardly any use in quantisation, especially for a percussive instrument such as the piano (even more so if the pedal is used). In figure 6, the lower notes of the arpeggios are commonly held longer than the higher notes – if these performed durations were quantised to the closest nominal values then lower notes would be notated as eighth notes or even dotted eighth notes. Calculation of note durations has thus been based in the current system on an elementary 'streaming' algorithm whereby notes within a beat are split into independent streams (voice parts) and then durations are considered to be equivalent to the inter-onset intervals for the notes of each stream.

The streaming algorithm is based on the Gestalt principle of proximity and simply tries to find the shortest streams that connect all the onsets within a beat (figure 5). Crossing of streams is not allowed. The number of streams is always equal to the number of notes in the largest chord. The solution to this problem is not trivial and appropriate searching techniques are required for developing an efficient algorithm. The current elementary version of the algorithm makes mistakes (see figure 6) but can be

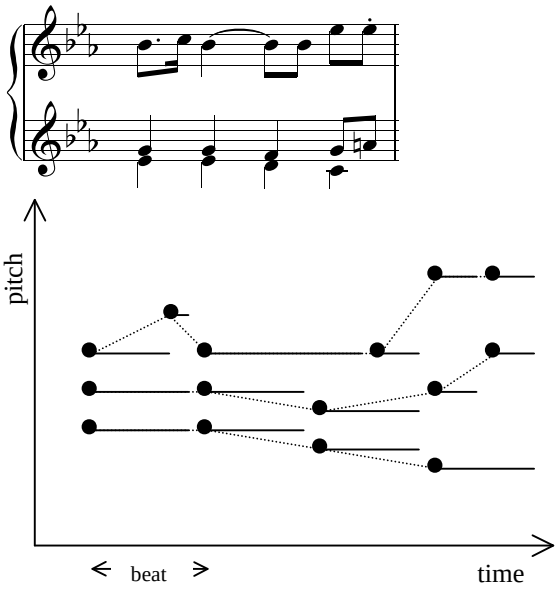

Figure 5: Duration quantisation relies on a streaming algorithm. Dots in the graph represent the onsets of the notes in the musical segment; dotted lines show the three streams detected by the streaming algorithm; horizontal lines indicate the inter-onset intervals for each stream. The algorithm in this case gives the correct score durations.

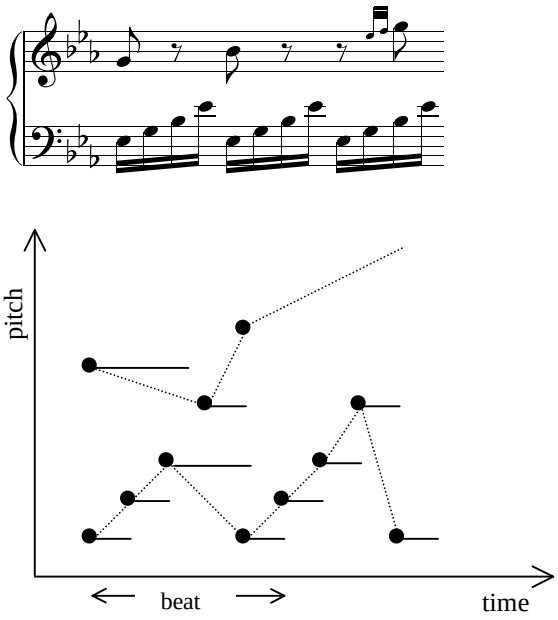

Figure 6: Duration quantisation fails locally as the streaming algorithm has given wrong results (see caption of fig.5 for explanation of graph).

improved if other principles like 'goodness of continuation' are taken into account. Streaming is a large research topic in its own right (see Bregman 1990) and the current algorithm is only a means for providing a crude

streaming of notes so that durations of notes may be calculated as interonset intervals of each stream. Obviously this methodology avoids rests (rests, however, may appear at the beginning of beats if no note was held from a previous beat and if the beginning of the beat has fewer streams than its continuation).

Pitch spelling

In (Cambouropoulos, 2000) an algorithm is presented that generates a correctly spelled diatonic score from a MIDI file. The algorithm can be applied to polyphonic music. This algorithm is based on two principles:

a. notational parsimony (i.e. double-sharps and double flats are avoided) and

b. diatonic interval optimisation (i.e. common diatonic intervals are preferred whereas rare intervals such as augmented, diminished, chromatic intervals are avoided).

The above optimisation procedure is applied locally on a MIDI-file using a shifting overlapping windowing technique. The spelling of the pitch classes within the window that gives an optimal solution for the above two rules is selected. This transcription algorithm is unique in that it requires no previous key- or tonality-finding mechanisms; on the contrary, it can be used as a precursor to key-finding algorithms.

It has been tested on 3 complete Mozart piano sonatas (kv279, kv280, kv281) for which it gives 97% correct results (total number of notes is 13002; number of notes with accidentals is 3546; notes mis-spelled by algorithm is 111) - further extended tests are currently underway. Minor mistakes occur occasionally, mainly because of local conflicts introduced by voice-leading concerns (see figure 7); an example of correct pitch spelling is given in figure 8.

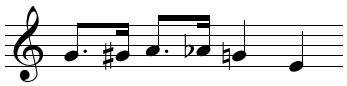

Figure 7: The pitch spelling algorithm would spell both notes with accidentals as G# as it has no knowledge of voice-leading.

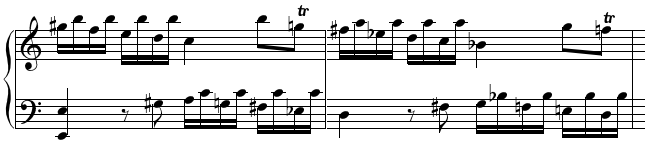

Figure 8: Musical section from Mozart's Sonata kv279 spelled correctly by the algorithm.

The proposed pitch-spelling component, on the one hand, highlights the importance of the traditional diatonic pitch-interval system when dealing with tonal music and, on the other hand, provides a very simple and robust algorithm that can be easily incorporated in a number of music applications (e.g. music notation software, score-extraction systems, music analytic systems and so on).

Conclusions

In this paper a system that attempts to extract the musical surface from a performed MIDI files has been presented. A wide range of independent but interacting modules was described. Special emphasis has been given to making use of musical accents for determining the beat level and then using this for quantising all the note onsets. Streaming was also presented as a means for deriving the duration of notes. Finally a module was described that converts MIDI numbers to the traditional pitch notation. The current score extraction system can be further improved by refining the existing modules and determining more accurately their interdependencies; additionally, new components may be introduced to cope with aspects of musical scores that depend on knowledge of musical structure.

Acknowledgements

This research is part of the project Y99-INF, sponsored by the Austrian Federal Ministry of Education, Science, and Culture in the form of a START Research Prize. The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Education, Science, and Culture.

References

Bregman, A. S. 1990. *Auditory Scene Analysis*. Cambridge, Mass: The MIT Press.

Cambouropoulos, E. 2000. Pitch Spelling: from Numbers to Sharps and Flats. Submitted to ICMC 2000, Berlin.

Cambouropoulos, E. 1996. A General Pitch Interval Representation: Theory and Applications. *Journal of New Music Research*, 25(3): 231-251.

Desain, P. and Honing H. 1992. *Music, Mind and Machine*. Amsterdam: Thesis Publishers

Dixon, S. and Cambouropoulos E. 2000. Beat Tracking with Musical Knowledge. Accepted for ECAI 2000, Berlin.

Handel, S. 1989. *Listening. An Introduction to the Perception of Auditory Events.* Cambridge, Mass: The MIT Press.

Lerdahl, F. and Jackendoff, R. 1983. *A generative Theory of Tonal Music.* Cambridge, Mass: The MIT Press.

Povel, D. J. and Essens, P. 1985. Perception of Temporal Patterns. *Music Perception* 2:411-440.

Rosenthal, D. 1992. Emulation of Human Rhythm Perception. *Computer Music Journal* 16(10):64-76.

Rowe, R. 1993. *Interactive Music Systems (Machine Listening and Composing).* Cambridge, Mass: The MIT Press.