Mach Learn (2006) 65:411–437
DOI 10.1007/s10994-006-9025-9

# A case based approach to expressivity-aware tempo transformation

**Maarten Grachten · Josep-Lluís Arcos · Ramon López de Mántaras**

Received: 23 September 2005 / Revised: 17 February 2006 / Accepted: 27 March 2006 / Published online: 15 June 2006
Springer Science + Business Media, LLC 2006

**Abstract** The research presented in this paper focuses on global tempo transformations of monophonic audio recordings of saxophone jazz performances. We are investigating the problem of how a performance played at a particular tempo can be rendered automatically at another tempo, while preserving naturally sounding expressivity. Or, differently stated, how does expressiveness change with global tempo. Changing the tempo of a given melody is a problem that cannot be reduced to just applying a uniform transformation to all the notes of a musical piece. The expressive resources for emphasizing the musical structure of the melody and the affective content differ depending on the performance tempo. We present a case-based reasoning system called *TempoExpress* for addressing this problem, and describe the experimental results obtained with our approach.

**Keywords** Music · Tempo transformation · Case based reasoning · Expressive performance

## 1. Introduction

It has been long established that when humans perform music from score, the result is never a literal, mechanical rendering of the score (the so-called nominal performance). As far as performance deviations are intentional (that is, they originate from cognitive and affective sources as opposed to e.g. motor sources), they are commonly thought of as conveying musical expressivity, which forms an important aspect of music. Two main functions of musical expressivity are generally recognized. Firstly, expressivity is used to clarify the musical structure (in the broad sense of the word: this includes for example metrical structure (Sloboda, 1983), but also the phrasing of a musical piece (Gabrielsson, 1987), and harmonic

**Editor:** Gerhard Widmer

M. Grachten (✉) · J. Arcos · R. de Mántaras
IIIA, Artificial Intelligence Research Institute,
CSIC, Spanish Council for Scientific Research,
Campus UAB, 08193 Bellaterra, Catalonia, Spain.
email: maarten@iiia.csic.es

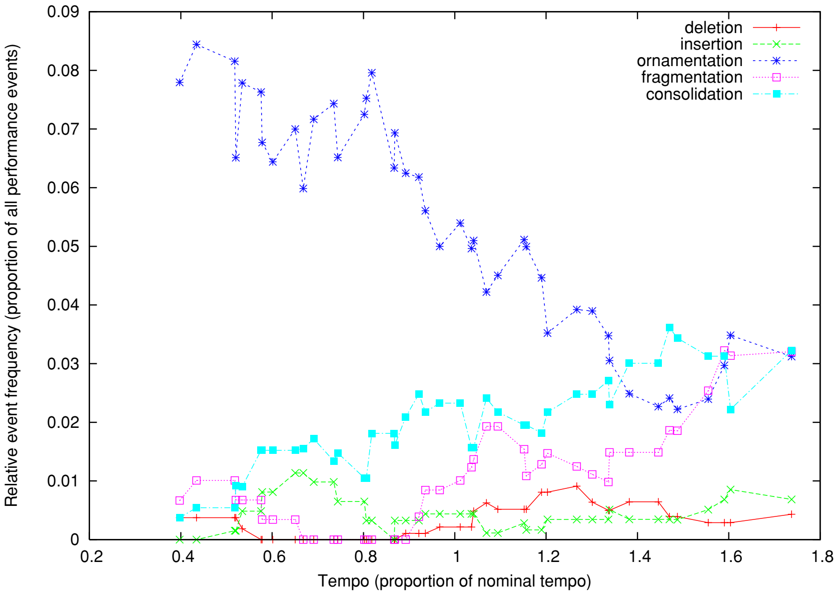


**Fig. 1** The frequency of occurrence of several kinds of performance events as a function of global performance tempo

structure (Palmer, 1996)). Secondly, expressivity is used as a way of communicating, or accentuating affective content (Juslin, 2001; Lindström, 1992; Gabrielsson, 1995).

Given that expressivity is a vital part of performed music, an important issue is the effect of tempo on expressivity. It has been argued that temporal aspects of performance scale uniformly when tempo changes (Repp, 1994). That is, the durations of all performed notes maintain their relative proportions. This hypothesis is called *relational invariance* (of timing under tempo changes). Counter-evidence for this hypothesis has also been provided however (Desain and Honing, 1994; Friberg and Sundström, 2002; Timmers et al., 2002), and a recent study shows that listeners are able to determine above chance-level whether audio-recordings of jazz and classical performances are uniformly time stretched or original recordings, based solely on expressive aspects of the performances (Honing, 2006).

A brief look at the corpus of recorded performances we will use in this study (details about the corpus are given in subsection 3.1) reveals indeed that the expressive content of the performances varies with tempo. Figure 1 shows the frequency of occurrence of various types of expressivity, such as *ornamentation* and *consolidation*, as a function of the nominal tempo of the performances (the tempo that is notated in the score). In subsection 3.2.1 we will introduce the various types of performance events as manifestations of musical expressivity in detail. Note that this figure shows the occurrence of discrete events, rather than continuous numerical aspects of expressivity such as timing, or dynamics deviations. The figure clearly shows that the occurrence of certain types of expressivity (such as ornamentation) decreases with increasing tempo, whereas the occurrence of others (consolidation most notably) increases with increasing tempo. These observations amount to the belief that although in some circumstances relational invariance may hold for some aspects of expressivity, in general it cannot be assumed that all aspects of expressivity remain constant (or scale proportionally) when the tempo of the performance is changed. In other words, tempo transformation of musical performances involves more than *uniform time stretching* (*UTS*).

Throughout this paper, we will use the term UTS to refer to the scaling of the *temporal* aspects of a performance by a constant factor. For example, dynamics, and pitch will be left unchanged, and also no notes will be inserted or removed. Only the duration, and onsets of notes will be affected. Furthermore, we will use the term UTS in an abstract sense. Depending on the data under consideration it involves different methods to realize it. For example, it requires non-trivial signal-processing techniques to apply UTS to the audio recording of the performance. In symbolic descriptions of the performance on the other hand, UTS consists in a multiplication of all temporal values by a constant. Note that this holds if the descriptions measure time in absolute units (e.g. seconds). When time is measured in score units (e.g. beats) UTS makes no sense, since changing the tempo of the performance only changes the translation of score time units to absolute units of time.

Nowadays high-quality audio time stretching algorithms exist (e.g. Röbel, 2003; Bonada, 2000), making temporal expansion and compression of audio possible without significant loss in sound quality. The main aim of those algorithms is maintaining *sound* quality, rather than the *musical* quality of the audio (in the case of recorded musical performances). But as such, they can be used as tools to build higher level (i.e. content-based) audio transformation applications. A recent example of this is an application that allows the user to change the swing-ratio of recorded musical performances (Gouyon et al., 2003). Such audio applications can be valuable especially in the context of audio and video post-production, where recorded performances must commonly be tailored to fit specific requirements. For instance, for a recorded musical performance to accompany video, it must usually meet tight constraints imposed by the video with respect to timing or the duration of the recording.

In this paper we present a system for musical tempo transformations, called *TempoExpress*, that aims at maintaining the musical quality of recorded performances when their tempo is changed. That is, ideally listeners should not be able to notice from the expressivity of a performance that has been tempo transformed by *TempoExpress* that its tempo has been scaled up or down from another tempo. The system deals with monophonic audio recordings of expressive saxophone performances of jazz standards. For the audio analysis and synthesis, *TempoExpress* relies on an external system for melodic content extraction from audio, developed by Gómez et al. (2003c,b). This system performs pitch and onset detection to generate a *melodic description* of the recorded audio performance, in a format that complies with an extension of the MPEG7 standard for multimedia content description (Gómez et al., 2003a). To realize a tempo transformation of an audio recording of a performance, *TempoExpress* needs an XML file containing the melodic description of the performance, a MIDI file specifying the score, and the target tempo to which the performance should be transformed (the tempo is specified in terms of *beats per minute*, or *BPM*). The result of the tempo transformation is an XML file containing the modified melodic description, that is used as the basis for performing a resynthesis of the input audio.

The evaluation of *TempoExpress* presented in this paper consists in a comparison of tempo transformed performances to performances performed by a musician at the target tempo, using a distance measure that is optimized to be in accordance with human similarity judgments of performances. The evaluation is based on the modified melodic descriptions, rather than on the resynthesized audio, for two reasons. Firstly, we are primarily interested in testing the *musical* quality of the tempo transformed performance, whereas any kind of evaluation of the resynthesized audio would probably be strongly influenced by the *sound* quality. Secondly, the audio resynthesis is currently done in a semi-automatic way (that is, timing and dynamics changes are translated to audio transformations automatically, but for note insertions and similar extensive changes, manual intervention is still necessary). This limitation would prevent a large-scale evaluation, if the evaluation was to be done using resynthesized audio rather than the transformed melodic descriptions.

*TempoExpress* solves tempo transformation problems by case-based reasoning. Problem solving in case-based reasoning is achieved by identifying and retrieving the problem (or set of problems) most similar to the problem that is to be solved from a case base of previously solved problems (called cases), and adapting the corresponding solution to construct the solution for the current problem. In the context of a music performance generation system, an intuitive manner of applying case-based reasoning would be to view unperformed music (e.g. a score) as a problem description (possibly together with requirements about how the music should be performed) and to regard a performance of the music as a solution to that problem. As has been shown by Widmer (1996), relating expressivity to the musical score is easier when higher level structural aspects of the score are represented (e.g. using concepts such as 'step-wise ascending sequence'), than when only the surface of the score is represented (i.e. a sequence of individual notes). Therefore, a structural musical analysis is also included in the problem description. Moreover, because we are interested in changing the tempo of a specific performance (we deal with the task of performance *transformation*, rather than performance *generation*), the expressive resources used in that performance also have to be modeled as part of the problem requirements.

The corpus of musical data we use contains fourteen phrases from four jazz standards, each phrase being performed at about twelve different tempos, amounting to 4256 performed notes. Jazz standards, as notated in The Real Book (2004) typically consist of two to five phrases (monophonic melodies annotated with chord symbols). Phrases usually have a length of five to eight bars, typically containing 15 to 25 notes.

A preliminary version of *TempoExpress* was described in Grachten et al. (2004b). In this paper we present the completed system, and report the experimental results of our system over more than six thousand tempo transformation problems. The tempo transformation problems were defined by segmenting the fourteen phrases into segments (having a typical length of about five notes), and using the performances at different tempos as problem descriptions and solutions respectively. Although the system is designed to deal with complete phrases, we decided to evaluate on phrase segments rather than entire phrases for the sake of statistical reliability, since this increases both the number of possible tempo transformation problems to solve, and the amount of training data available, given the musical corpus.

The paper is organized as follows: Section 2 situates the work presented here in the context of related work. In section 3 we will present the overall architecture of *TempoExpress*. In section 4 we report the experimentation in which we evaluated the performance of *TempoExpress*. Conclusions and future work are presented in section 5.

## 2. Related work

The field of expressive music research comprises a rich and heterogeneous number of studies. Some studies are aimed at verbalizing knowledge of musical experts on expressive music performance. For example, Friberg et al. have developed *Director Musices* (DM), a system that allows for automatic expressive rendering of MIDI scores (Friberg et al., 2000). DM uses a set of expressive performance rules that have been formulated with the help of a musical expert using an analysis-by-synthesis approach (Sundberg et al., 1991a; Friberg, 1991; Sundberg et al., 1991b).

Widmer (2000) has used machine learning techniques like Bayesian classifiers, decision trees, and nearest neighbor methods, to induce expressive performance rules from a large set of classical piano recordings. In another study by Widmer (2002), the focus was on discovery of simple and robust performance principles rather than obtaining a model for performance generation.

Hazan et al. (2006) have proposed an evolutionary generative regression tree model for expressive rendering of melodies. The model is learned by an evolutionary process over a population of candidate models.

In the work of Desain and Honing and co-workers, the focus is on the cognitive validation of computational models for music perception and musical expressivity. They have pointed out that expressivity has an intrinsically perceptual aspect, in the sense that one can only talk about expressivity when the performance itself defines the standard (e.g. a rhythm) from which the listener is able to perceive the expressive deviations (Honing, 2002). In more recent work, Honing showed that listeners were able to identify the original version from a performance and a uniformly time stretched version of the performance, based on timing aspects of the music (Honing, 2006).

Timmers et al. have proposed a model for the timing of grace notes, that predicts how the duration of certain types of grace notes behaves under tempo change, and how their durations relate to the duration of the surrounding notes (Timmers et al., 2002).

A precedent of the use of a case-based reasoning system for generating expressive music performances is the *SaxEx* system (Arcos et al., 1998; López de Mántaras and Arcos, 2002). The goal of the *SaxEx* system is to generate expressive melody performances from an inexpressive performance, allowing user control over the nature of the expressivity, in terms of expressive labels like ‘tender’, ‘aggressive’, ‘sad’, and ‘joyful’.

Another case-based reasoning system is *Kagurame* (Suzuki, 2003). This system renders expressive performances of MIDI scores, given performance conditions that specify the desired characteristics of the performance. Although the task of *Kagurame* is performance *generation*, rather than performance *transformation* (as in the work presented here), it has some sub-tasks in common with our approach, such as performance to score matching, segmentation of the score, and melody comparison for retrieval. *Kagurame* also employs the edit-distance for performance-score alignment, but it discards deletions/insertions and retains just the matched elements, in order to build a list of timing/dynamics deviations that represent the performance. Furthermore, its score segmentation approach is a hierarchical binary division of the piece into equal parts. The obtained segments thus do not reflect melodic structure. Another difference is that *Kagurame* operates on polyphonic MIDI, whereas *TempoExpress* deals with monophonic audio recordings. *Kagurame* manipulates local tempo, durations, dynamics, and chord-spread as expressive parameters.

Recently, Tobudic and Widmer (2004) have proposed a case-based approach to expressive phrasing, that predicts local tempo and dynamics and showed it outperformed a straightforward k-NN approach.

To our knowledge, all of the performance rendering systems mentioned above deal with predicting expressive values like timing and dynamics for the notes in the score. Contrastingly *TempoExpress* not only predicts values for timing and dynamics, but also deals with note insertions, deletions, consolidations, fragmentations, and ornamentations.

## 3. System architecture

In this section we will explain the structure of the *TempoExpress* system. We will first give a short description of the tempo transformation process as a whole, and then devote subsections to each of the steps involved, and to the formation of the case-base. A schematic view of the

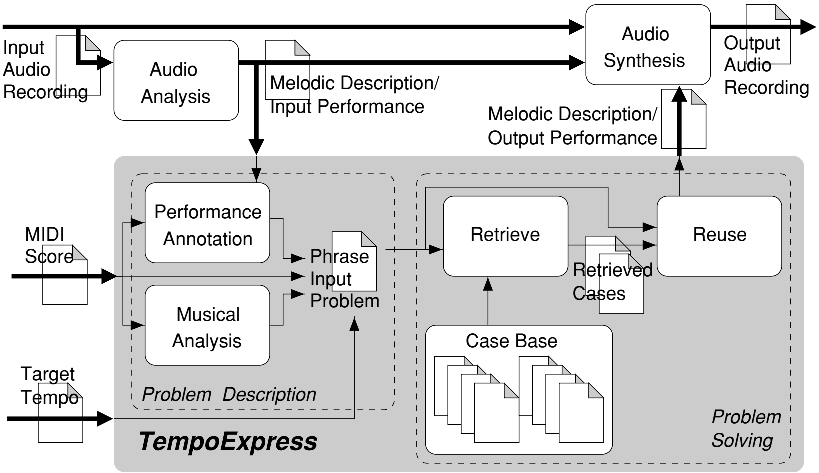


**Fig. 2** Schematic view of the *TempoExpress* system

system as a whole is shown in figure 2. We will focus on the part inside the gray box, that is, the steps involved in modifying the expressive parameters of the performance at the musical level. For a more detailed account of the audio analysis/synthesis components, we point the reader to Gómez et al. (2003b); Maestre and Gómez (2005).

Given a MIDI score of a phrase from a jazz standard, and given a monophonic audio recording of a saxophone performance of that phrase at a particular tempo (the *source* tempo), and given a number specifying the *target* tempo, the task of the system is to render the audio recording at the target tempo, adjusting the expressive parameters of the performance to be in accordance with that tempo. In the rest of this paper, we will use the term performance to specifically refer to a symbolic description of the musician's interpretation of the score, as a sequence of performed notes.

In order to apply the CBR problem solving process, the first task is to build a *phrase problem specification* from the given input data. This is a data structure that contains all information necessary to define a tempo transformation task for a musical phrase, and possibly additional information that may improve or facilitate the problem solving. A phrase problem specification contains the following information:

1. a *MIDI score*, the score as a sequence of notes;
2. a *musical analysis*, an abstract description of the melody;
3. a *source tempo*, the tempo (in BPM) of the input performance;
4. a *target tempo*, the tempo (in BPM) at which the output performance should be rendered;
5. an *input performance*, the performance as a sequence of performed notes;
6. a *performance annotation*, a description of the expressivity in the performance;
7. a list of *segment boundaries*, that indicates how the score of the phrase is divided into segments.

As figure 2 shows, only items (1) and (4) of this list are directly provided by the user. The musical analysis (2) is derived from the MIDI score and contains information about various kinds of structural aspects of the score, like metrical structure, an analysis of the melodic surface, and note grouping. The phrase segmentation (7) is also derived from the MIDI score, and is intended to capture the musical groupings inherent in the phrase.

The performance annotation module takes the MIDI score and the melodic description of the input audio recording as input, and provides the source tempo (3), the input performance (5), and the performance annotation (6). The melodic description is an XML formatted file containing both a frame-by-frame description of the audio (with descriptors like fundamental frequency candidates, and energy), and a segment-by-segment description of the audio. The audio segment descriptions (not to be confused with phrase segments) correspond to the individual notes detected in the audio, and apart from their begin and end time (i.e. note onset and offset) they include mean energy (dynamics), and estimated fundamental frequency (pitch) as descriptors. The source tempo (3) is estimated by comparing the total duration of the audio (time in the melodic description is specified in seconds) and the duration of the MIDI score (which specifies time in musical beats). Although this way of estimating the global tempo is simple, it works well for the data we used.[1] The input performance (5) is a symbolic representation of the performed notes, with MIDI pitch numbers (estimated from the fundamental frequency), duration, onset, and dynamics information. This information is readily available from the melodic description. To facilitate comparison between the performed notes and the MIDI score notes, the duration and onset values of the performed notes are converted from seconds to beats, using the computed source tempo. Finally, the performance annotation (6) is computed by comparing the MIDI score and the input performance.

We will refer to the phrase problem specification that was built from the input data as the *phrase input problem*; this is the problem specification for which a solution should be found. The solution of a tempo transformation will consist in a performance annotation. The performance annotation can be interpreted as a sequence of changes that must be applied to the MIDI-score in order to render the score expressively. The result of applying these transformations is a sequence of performed notes, the *output performance*, which can be directly translated to a melodic description at the target tempo, suitable to be used as a directive to synthesize audio for the transformed performance.

In a typical CBR setup, the input problem is used to query the case base, where the cases contain problem specifications similar in form to the input problem, together with a solution. The solution of the most similar case is then used to generate a solution for the input problem as a whole. In the current setting of music performance transformation however, this approach does not seem the most suitable. Firstly, the solution is not a single numeric or nominal value, as in e.g. classification, or numeric prediction tasks, but it rather takes the form of a performance annotation, which is a composite structure. Secondly, melodies are usually composed of parts that form wholes in themselves (a phrase is typically composed of various motifs). The first observation implies that solving a problem as a whole would require a huge case base, since the space of possible solutions is so vast. The second observation on the other hand suggests that a solution may be regarded as a concatenation of separate (not necessarily independent)[2] partial solutions, which somewhat alleviates the need for a very large case base, since the partial solutions are less complex than complete solutions.

This has led us to the design of the problem solving process that is illustrated in figure 3. The phrase input problem is broken down into phrase segment problems (called *segment input problems*, or simply *input problems* henceforward), which are then solved individually. The solutions found for the individual segments are concatenated to obtain the solution for

[1] Tempo estimates computed for 170 performances have a mean error of 0.2 BPM and a standard deviation of 1.1 BPM.

[2] For example, the way of performing one motif in a phrase may affect the (in)appropriateness of particular ways of playing other (adjacent, or repeated) motifs. Although such constraints are currently not defined in *TempoExpress*, we will explain in section 3.4 how the reuse infrastructure can easily accommodate this.

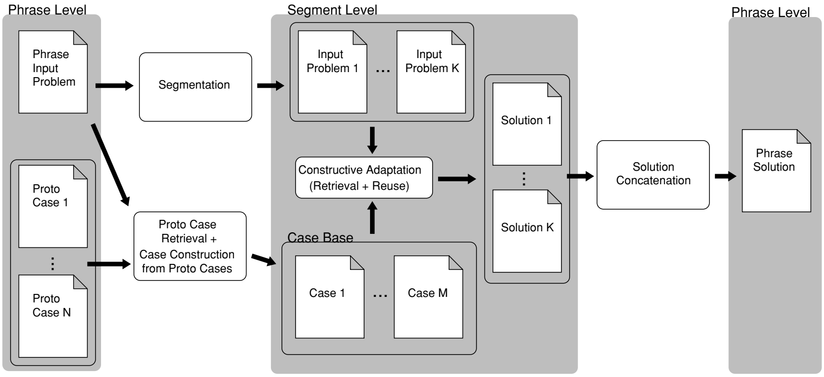


**Fig. 3** The problem solving process from phrase input problem to phrase solution

the phrase input problem. Furthermore, a preliminary retrieval action is performed using the problem at the phrase level. The goal of this preliminary retrieval is to set up the case base for the segment-level problem solving, from what we will call *proto cases*. Proto cases are information units that contain phrase related information like the MIDI score, the musical analysis, the segmentation boundaries, and all performances (with varying global tempos) available for that phrase. The case base is formed by pooling the segments of the selected proto cases, hence the number of cases $M$ it will contain depends on the selectivity of the preliminary retrieval, and the number of segments per phrase: If $C$ is the subset of proto cases that were selected during preliminary retrieval, and $s_i$ the number of segments in the $i^{th}$ proto case from $C$, then the case base size is:

$$M = \sum_{i=1}^{|C|} s_i$$

The case base obtained in this way contains *cases*, consisting of a segment problem specification and a solution at the segment level. The cases contain the same type of information as the input problem specifications and solutions at the phrase level, but they span a smaller number of notes. Solving the phrase input problem is achieved by searching the space of partially solved phrase input problems. A partially solved phrase input problem corresponds to a state where zero or more segment input problems have a solution. A complete solution is a state where all segment input problems have a solution. Solutions for the segment input problems are generated by adapting retrieved (segment) cases. This technique for case reuse is called *constructive adaptation* (Plaza and Arcos, 2002).

The expansion of a state is realized by generating a solution for a segment input problem. To achieve this, the retrieve step ranks the cases according to similarity between the MIDI scores of the segment input problem and the cases. The reuse step consists of mapping the score notes of the retrieved case to the score notes of the input problem, and using this mapping to ‘transfer’ the performance annotation of the case solution to the input problem.

In the following subsections, we will address the issues involved in more detail. Subsection 3.1 specifies the musical data that makes up the corpus we have used in this study. Subsection 3.2 elaborates on the steps involved in automatic case acquisition, and explains the construction of cases from proto cases. The retrieval step is explained in subsection 3.3. Finally, subsection 3.4 deals with the reuse step.

**Table 1** Songs used to populate the case base

| Title | Composer | Song Structure | Tempo Range |
|---|---|---|---|
| Body and Soul | J. Green | $A_1$ $A_2$ $B_1$ $B_2$ $A_1$ | 35–100 BPM |
| Like Someone in Love | Van Heusen/Burke | $A$ $B_1$ $B_2$ | 65–260 BPM |
| Once I Loved | A.C. Jobim | $A$ $B$ $C$ $D$ | 55–220 BPM |
| Up Jumped Spring | F. Hubbard | $A_1$ $A_2$ $B$ | 90–270 BPM |

### 3.1. Musical corpus

Four different songs were recorded using professional recording equipment. The performing artist was a professional jazz saxophone player. Every song was performed and recorded at various tempos. One of these tempos was the nominal tempo. That is, the tempo at which the song is intended to be played. This is usually notated in the score. If the nominal tempo was not notated, the musician would determine the nominal tempo as the one that appeared most natural to him. The other tempos were chosen to be around the nominal tempo, increasing and decreasing in steps of 5 (in slow tempo ranges) or 10 BPM (in faster tempo ranges). About 12 tempos per song were recorded. The musician performed on his own, accompanied by a metronome indicating the global tempo of the piece. In total 170 interpretations of phrases were recorded, amounting to 4256 performed notes. Table 1 shows the top level phrase structure of the songs (determined manually from the score) and the tempo range per song.

The musician was instructed to perform the music in a way that seemed natural to him, and appropriate for the tempo at which he was performing. Note that the word natural does not imply the instruction to play in-expressively, or to achieve ‘dead-pan’ interpretations of the score (that is, to imitate machine renderings of the score). Rather, the musician was asked not to strongly color his interpretation by a particular mood of playing.

### 3.2. Automated case acquisition/problem description

An important issue for a successful problem solving system is the availability of example data. We have therefore put effort in automatizing the process of constructing cases from non-annotated data (that is, the audio files and MIDI scores). Note that since cases contain problem specifications, the problem description step (see figure 2) is a part that case acquisition from available data has in common with the normal system execution cycle when applying a tempo transformation as outlined in figure 2. Case acquisition differs from the normal execution cycle however, in the sense that the melodic description that describes the performance at the target tempo (the output of the reuse-step) is not inferred through the problem solving process, but is rather given as the correct solution for the problem.

Problem description involves two main steps: annotation of the performance, and a musical analysis of the score. Performance annotation consists in matching the notes of the performance to the notes in the score. This matching leads to the *annotation* of the performance: a sequence of *performance events*. The annotation can be regarded as a description of the musical behavior of the player while he interpreted the score, and as such conveys the musical expressivity of the performance. The second step in the case acquisition is an analysis of the musical score that was interpreted by the player. The principal goal of this analysis is to provide conceptualizations of the score at an intermediate level. That is, below the phrase level (the musical unit which the system handles as input and output), but above the note level.

One aspect is the segmentation of the score into motif level structures, and another one is the categorization of groups of notes that serves as a melodic context description for the notes.

### *3.2.1. Performance annotation*

It is common to define musical expressivity as the discrepancy between the musical piece as it is performed and as it is notated. This implies that a precise description of the notes that were performed is not very useful in itself. Rather, the *relation* between score and performance is crucial. The majority of research concerning musical expressivity is focused on the temporal, or dynamic variations of the notes of the musical score as they are performed, e.g. (Canazza et al., 1997; Desain and Honing, 1994; Repp, 1995; Widmer, 2000). In this context, the spontaneous insertions or deletions of notes by the performer are often discarded as artifacts, or performance errors. This may be due to the fact that most of this research is focused on the performance practice of classical music, where the interpretation of notated music is usually strict. Contrastingly, in jazz music performers often favor a more liberal interpretation of the score, in which expressive variation is not limited to variations in timing of score notes, but also comes in the form of e.g. deliberately inserted and deleted notes. We believe that research concerning expressivity in jazz music should pay heed to these phenomena.

A consequence of this broader interpretation of expressivity is that the expressivity of a performance cannot be represented as a straight-forward list of expressive attributes for each note in the score. A more suitable representation of expressivity describes the musical behavior of the performer as *performance events*. The performance events form a sequence that maps the performance to the score. For example, the occurrence of a note that is present in the score, but has no counterpart in the performance, will be represented by a *deletion event* (since this note was effectively deleted in the process of performing the score). Obviously, deletion events are exceptions, and the majority of score notes are actually performed, be it with alterations in timing/dynamics. This gives rise to *correspondence events*, which establish a correspondence relation between the score note and its performed counterpart. Once a correspondence is established between a score and a performance note, other expressive deviations like onset, duration, and dynamics changes, can be derived by calculating the differences of these attributes on a note-to-note basis.

Analyzing the corpus of monophonic saxophone recordings of jazz standards described in subsection 3.1, we encountered the following types of performance events:

**Insertion** The occurrence of a performed note that is not in the score
**Deletion** The non-occurrence of a score note in the performance
**Consolidation** The agglomeration of multiple score notes into a single performed note
**Fragmentation** The performance of a single score note as multiple notes
**Transformation** The change of nominal note features like onset time, duration, pitch, and dynamics
**Ornamentation** The insertion of one or several short notes to anticipate another performed note

These performance events tend to occur persistently throughout different performances of the same phrase. Moreover, performances including such events sound perfectly natural, so much that it is sometimes hard to recognize them as deviating from the notated score. This supports our claim that even the more extensive deviations that the performance events describe, are actually a common aspect of (jazz) performance.

A key aspect of performance events is that they refer to particular notes in either the notated score, the performance, or both. Based on this characteristic a taxonomy can be formulated,

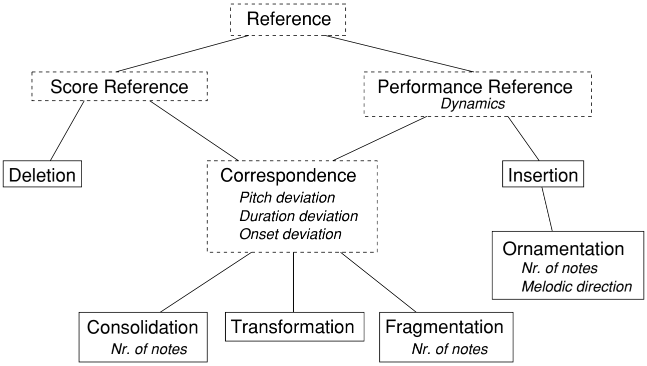


**Fig. 4** A taxonomy of performance events

as shown in figure 4. The concrete events types, listed above, are depicted as solid boxes. The dotted boxes represent abstract types of events, that clarify the relationships between different types of events. In actual performance annotations, the abstract events will always be instantiated by a concrete subtype. The most prominent attributes of the various event types are added in italic. For example, the "number of notes" attribute specifies the value of $n$ in a $n$-to-1 (consolidation), 1-to-$n$ (fragmentation), or 0-to-$n$ (ornamentation) mapping between score and performance.

In order to obtain a sequence of performance events that represent the expressive behavior of the performer, the notes in the performance (the sequence of performed notes extracted from the melodic description of the audio) are matched to the notes in the score using the *edit-distance*. The edit-distance is defined as the minimal cost of a sequence of editions needed to transform a source sequence into a target sequence, given a predefined set of edit-operations (classically deletion, insertion, and replacement of notes). The cost of a particular edit-operation is defined through a *cost function* $w$ for that operation, that computes the cost of applying that operation to the notes of the source and target sequences that were given as parameters to $w$. We write $w^{K,L}$ to denote that $w$ operates on a subsequence of length $K$ of the source sequence, and a subsequence of length $L$ of the target sequence. For example, a deletion operation would have a cost function $w^{1,0}$. For a given set of edit-operations, let $W$ be the set of corresponding cost functions, and let $V_{i,j} = \{w^{K,L} \mid K \leq i \wedge L \leq j\} \subseteq W$ be the subset of cost functions that operates on source and target subsequences with maximal lengths of $i$ and $j$, respectively. Furthermore, let $\mathbf{s}_{1:i} = \langle s_1, \cdots, s_i \rangle$ and $\mathbf{t}_{1:j} = \langle t_1, \cdots, t_j \rangle$ be the source and target sequences respectively. Then the edit-distance $d_{i,j}$ between $\mathbf{s}_{1:i}$ and $\mathbf{t}_{1:j}$ is defined recursively as[3]:

$$d_{i,j} = \min_{w^{K,L} \in V_{i,j}} (d_{i-K,j-L} + w^{K,L}(\mathbf{s}_{i-K+1:i}, \mathbf{t}_{j-L+1:j})\,) \qquad (1)$$

where the initial condition is: $d_{0,0} = 0$. The minimal cost $d_{i,j}$ has a corresponding *optimal alignment*, the sequence of edit-operations that constitutes the minimal-cost transformation of $\mathbf{s}$ into $\mathbf{t}$. When we equate the (concrete) performance events shown above to edit-operations, we can interpret the optimal alignment between the score and a performance as an annotation

[3] We adopt the convention that $\mathbf{s}_{i:j}$ denotes the sequence $\langle s_i \rangle$ whenever $i = j$, and denotes the empty sequence $\emptyset$ whenever $i > j$.

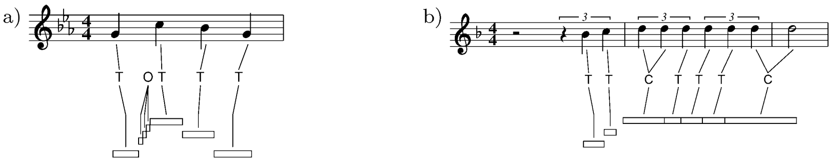


**Fig. 5** Graphical representations of performance annotations of excerpts from a) *Like Someone in Love, A*, and b) *Once I Loved, A1*

of the performance. To do this, we have defined cost functions for every type of performance event, that can subsequently be used to solve equation (1). Details of the cost functions are described in Arcos et al. (2003). Two example performance annotations are shown in figure 5. The bars below the staff represent performed notes (the vertical positions indicate the pitches of the notes). The letters in between the staff and the performed notes represent the performance events that were identified (‘T’ for transformation, ‘O’ for ornamentation, and ‘C’ for consolidation).

This method allows for automatically deriving a description of the expressivity in terms of performance events. With non-optimized edit-operation costs, the average amount of annotation errors is about 13% compared to manually corrected annotations (evaluating on the complete set of performances in the corpus). Typical errors are mistaking consolidations for a duration transformation event followed by note deletions, or recognizing a single ornamentation event, containing two or three short notes, as a sequence of note insertions (which they are, formally, but it is more informative to represent these as an ornamentation). In previous work (Grachten et al., 2004a), we have shown that by evolutionary optimization of edit-operation costs using manually corrected annotations as training data, the amount of errors could be reduced to about 3%, using a cross-validation setup on the performances in the musical corpus.

### *3.2.2. Musical score analysis*

The second step in the case acquisition is an analysis of the musical score. This step actually consists of several types of analysis, used in different phases of the case based reasoning process.

Firstly, a metrical accents template is applied to the score, to obtain the level of metrical importance for each note. For example, the template for a 4/4 time signature specifies that every first beat of the measure has highest metrical strength, followed by the third beat, followed by the second and fourth beat. The notes that do not fall on any of these beats, have lowest metrical strength. This information is used in the Implication-Realization analysis, described below, and during melody comparison in the retrieval/adaptation step of the CBR process (see subsection 3.4).

Secondly, the musical score is segmented into groups of notes, using the Melisma Grouper (Temperley, 2001), an algorithm for grouping melodies into phrases or smaller units, like motifs. The algorithm uses rules regarding inter-onset intervals, and metrical strength of the notes, resembling Lerdahl and Jackendoff’s *preference rules* (Lerdahl and Jackendoff, 1993). The algorithm takes a preferred group size as a parameter, and segments the melody into groups whose size is as close as possible to the preferred size. Figure 6 shows the segmentation of phrase A1 of *Up Jumped Spring* as an example. In *TempoExpress*, the segmentation of melodic phrases into smaller units is done as part of the retrieval and reuse steps, in order to allow for retrieval and reuse of smaller units than complete phrases.

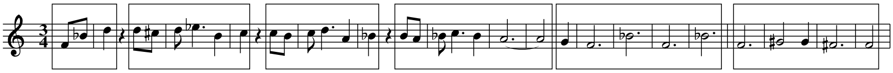


**Fig. 6** An example phrase segmentation (*Up Jumped Spring, A*1)

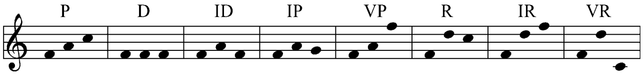
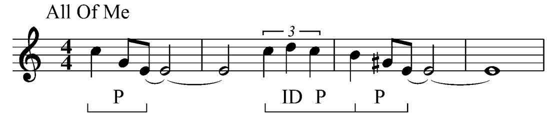


**Fig. 7** Eight basic Implication-Realization structures

**Fig. 8** I-R analysis of an excerpt from *All of Me* (Marks & Simons)



We used a preferred group size of 5 notes (because this value tended to make the segments coincide with musical motifs), yielding on average 4.6 segments per phrase.

Lastly, the surface structure of the melodies is described in terms of the Implication-Realization (I-R) model (Narmour, 1990). This model characterizes consecutive melodic intervals by the expectation they generate with respect to the continuation of the melody, and whether or not this expectation is fulfilled. The model states a number of data driven principles that govern the expectations. We have used the two main principles (*registral direction*, and *intervallic difference*, see Schellenberg (1997)) to implement an I-R parser for monophonic melodies. Additionally, the parser applies the I-R principle of *closure*, that predicts the inhibition of the listener's expectations as a function of rhythmic and metrical aspects of the melody. The output of the I-R parser is a sequence of labeled melodic patterns, so called I-R structures. An I-R structure usually represents two intervals (three notes), although in some situations shorter or longer fragments may be spanned, depending on contextual factors like rhythm and meter (i.e. closure). Eighteen basic I-R structures are defined using labels that signify the implicative/realizing nature of the melodic fragment described by they I-R structure. The I-R structures are stored with their label and additional attributes, such as the melodic direction of the pattern, the amount of overlap between consecutive I-R structures, and the number of notes spanned. Eight basic I-R structures are shown in figure 7. The letters above the staff are the names of the I-R structures the melodic patterns exemplify. Note that only the *relative* properties of registral direction between the two intervals matter for identifying the structure. That is, the melodic patterns obtained by mirroring the shown patterns along the horizontal axis exemplify the same I-R structure. This can be seen in the example I-R analysis shown in figure 8, where *downward P* structures occur.

The I-R analysis can be regarded as a moderately abstract representation of the score, that conveys information about the rough pitch interval contour, and through the boundary locations of the I-R structures, includes metrical and durational information of the melody as well. As such, this representation is appropriate for comparison of melodies. As a preliminary retrieval step, we use it to compare the score from the input problem of the system to the scores in the case base, to weed out melodies that are very dissimilar.

### *3.2.3. Proto case representation*

The data gathered for each phrase in the problem description steps described above are stored together in a proto case. This includes the MIDI score, the I-R analysis, the segmentation boundaries, and for every audio recording of the phrase, it includes the estimated tempo, the performance (i.e. the sequence of performed notes), and the performance annotation. At this point, the MIDI score segment boundaries are used to partition the available performances and their performance annotations. This is largely a straight-forward step, since the performance annotations form a link between score notes and performed notes. Only when non-score-reference events (such as ornamentations or insertions) occur at the boundary of two segments, it is unclear whether these events should belong to the former or the latter segment. In most cases however, it is a good choice to group these events with the latter segment (since for example ornamentation events always precede the ornamented note).

Note that any of the available performances is potentially part of a problem description, or a solution, as long as the source and target tempos have not been specified. For this reason, a proto case holds the information for more than just one tempo transformation. More precisely, when the proto case contains performances at $n$ different tempos, any of them may occur as an input performance paired with any other as output performance. If we exclude identity tempo transformations (where the same performance serves both as input and output performance), this yields $n(n-1)$ possible tempo transformations.

## 3.3. Proto case retrieval

The goal of the proto case retrieval step (see figure 3) is to form a pool of relevant cases that can possibly be used in the reuse step. This is done in the following three steps: Firstly, proto cases whose performances are all at tempos very different from the source tempo and target tempo are filtered out. Secondly, the proto cases with phrases that are I-R-similar to the input phrase are retrieved from the proto case base. Lastly, cases are constructed from the retrieved proto cases. The three steps are described below.

### *3.3.1. Case filtering by tempo*

In the first step, the proto case base is searched for cases that have performances both at source tempo and the target tempo. The matching of tempos need not be exact, since we assume that there are no drastic changes in performance due to tempo within small tempo ranges. We have defined the tempo tolerance window to be 10 BPM in both upward and downward directions. For example, a tempo transformation from 80 BPM to 140 BPM may serve as a precedent for tempo transformation from 70 BPM to 150 BPM. This particular tolerance range (which we feel may be too nonrestrictive), is mainly pragmatically motivated: In our corpus, different performances of the same phrase are often at 10 BPM apart from each other. Therefore, a $<10$ BPM tempo tolerance will severely reduce the number of available precedents, compared to a $\geq 10$ BPM tempo tolerance.

### *3.3.2. I-R based melody retrieval*

In the second step, the proto cases that were preserved after tempo filtering are assessed for melodic similarity to the score specified in the problem description. In this step, the primary goal is to rule out the proto cases that belong to different styles of music. For example, if the score in the problem description is a ballad, we want to avoid using a bebop theme as an example case.

We use the I-R analyses stored in the proto cases to compare melodies. The similarity computation between I-R analyses is based on the edit-distance. Figure 9 illustrates how

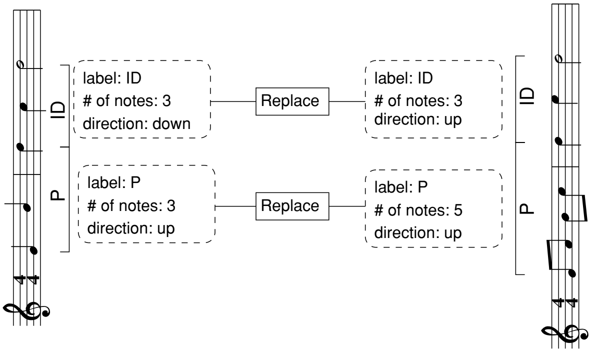


**Fig. 9** Comparison of melodies using an I-R based edit-distance

melodies are compared using their I-R analyses, for two fictional score fragments. The I-R analyses are represented as sequences of I-R structure objects, having attributes like their I-R label, the number of notes they span, and their registral direction. The edit-distance employs a replacement operation whose cost increases with increasing difference between the I-R structures to be replaced. When the cost of replacement is too high, a deletion and insertion are preferred over a replacement. The parameters in the cost functions were optimized using ground truth data for melodic similarity (Typke et al., 2005), analogous to the edit-distance tuning approach explained later in this paper (section 4.1.2). More details can be found in Grachten et al. (2005). The optimized I-R edit-distance performed best in the MIREX 2005 contest for symbolic melodic similarity (Downie et al., 2005), which shows it can well compete with other state-of-the-art melody retrieval systems.

With this distance measure we rank the phrases available in the proto case base, and keep only those phrases with distances to the problem phrase below a threshold value. The proto cases containing the accepted phrases will be used as the precedent material for constructing the solution.

### *3.3.3. Case construction from proto cases*

From the selected proto cases, the actual cases are constructed. First, the input performance and the output performance and their corresponding performance annotations are identified. The input performance is the performance in the proto case with the tempo closest to the source tempo, and the output performance is the performance closest to the target tempo. Then, the data for each segment in the proto case is stored in a new case, where the input performance and its performance annotation are stored in the problem description of that case, and the output performance and its performance annotation are stored as the solution. The problem description of the case additionally contains the MIDI score segment, that will be used to assess case similarity in the constructive adaptation step.

## 3.4. Constructive adaptation

In this step a performance of the input score is generated at the target tempo, based on the input performance and the set of matching cases. Constructive adaptation (CA) (Plaza and Arcos, 2002) is a technique for case reuse that constructs a solution by a search process through the space of partial solutions. In *TempoExpress* the partial solution to a phrase input problem is defined as a state where zero or more of the (segment) input problems have a

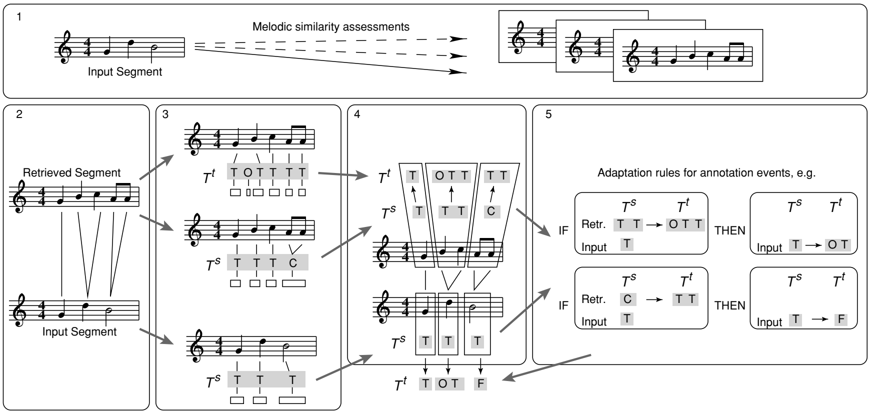


**Fig. 10** Example of case reuse for a melodic phrase segment; $T^s$ and $T^t$ refer to source and target tempo, respectively; The letters T, O, C, and F in the performance annotations (gray bars), respectively represent transformation, ornamentation, consolidation, and fragmentation events

corresponding solution (see figure 3). In the initial state, none of the input problems have a solution. This state is expanded into successor states by generating solutions for one of the input problems. Typically, more than one solution can be generated for an input problem, by reusing the solutions from different retrieved cases. When a state is reached that satisfies the goal state criteria, the solutions are concatenated to form the solution for the phrase problem specification. Otherwise, the state expansion is repeated, by solving one of the remaining unsolved input problems.

The goal state criteria require that a state has a solution for every input problem, and that the overall estimated *solution quality* of the solutions is maximal. The quality of a solution is estimated as the proportion of notes in the problem score segment for which performance events could be inferred based on the retrieved case. This proportion depends on the matching quality between problem score and retrieved score segment, and the availability of a matching adaptation rule, given the performance annotations in the problem and the case.

Independence is assumed between the solution qualities of the input problems, and thus the solution quality of the solution to a phrase input problem is defined as the average quality of the segment solutions, weighted by the segment length. Therefore, a best first search that expands states in the order of their solution quality is guaranteed to find the solution with maximal quality.

Although as of now no constraints have been defined for regulating interdependence of segment solutions, note that such constraints can easily be incorporated through CA. A constraint can take the form of a rule that prescribes a decrease or increase of the overall solution quality, based on some (probably high level) description of two or more segment solutions. Of course this may introduce local maxima in the search space, and the search strategy employed will become more crucial.

### *3.4.1. Case adaptation at the segment level*

In this subsection we will explain in detail how a solution is obtained for a segment input problem. This is the process that constitutes the state expansion mentioned before. Figure 10 shows an example of the reuse of a retrieved case for a particular input segment. We will briefly explain the numbered steps of this process one by one:

The *first step* is to find the case from the case base that is most similar to the input problem. The similarity is assessed by calculating the edit-distance between the score segments (the edit-distance now operates on notes rather than on I-R structures, to have a finer grained similarity assessment).

In the *second step*, a mapping between the input segment and the best matching retrieved segment is made, using the optimal alignment found through the calculation of the edit-distance.

In the *third step*, the performance annotations corresponding to the relevant tempos are extracted from the retrieved segment case and the input problem specification (both the source tempo $T^s$ and the target tempo $T^t$ for the retrieved segment case, and only $T^s$ from the input problem specification).

The *fourth step* consists in linking the performance events of the retrieved segment at $T^s$ and $T^t$ to the performance events of the input segment at $T^s$. The mapping between the input segment and the retrieved segment is used to determine which performance events from the retrieved case belong to which performance event from the input problem, as shown in box 4 from the figure. For every subsequence of one or more notes from the input segment **n** that was mapped to a subsequence of one or more notes **r** of the retrieved segment, this yields an annotation triple $\langle P_{T^s}(\mathbf{n}), P_{T^s}(\mathbf{r}), P_{T^t}(\mathbf{r})\rangle$, where the notation $P_T(\mathbf{s})$ is used to refer to the sequence of performance events corresponding to a sequence of notes **s** at tempo $T$. In the example, there are three one-note subsequences of the input segment.

In case the mapping is not perfect and a note of the input segment is not matched to any note of the retrieved segment, that note has no corresponding annotation triple. Such gaps are filled up by resorting to UTS. That is, $P_{T^t}(\mathbf{n})$ is constructed from $P_{T^s}(\mathbf{n})$ by scaling the duration (in seconds) of the events in $P_{T^s}(\mathbf{n})$ by the proportion $\frac{T^s}{T^t}$, leaving all other expressive features unchanged.

An annotation triple $\langle P_{T^s}(\mathbf{n}), P_{T^s}(\mathbf{r}), P_{T^t}(\mathbf{r})\rangle$ can be read intuitively as saying: a score fragment (usually just a single note) **r** was played as $P_{T^s}(\mathbf{r})$ at tempo $T^s$, and played as $P_{T^t}(\mathbf{r})$ at tempo $T^t$, while a melodically similar score fragment **n** was played as $P_{T^s}(\mathbf{n})$ at tempo $T^s$. In order to infer from this how to play **n** at tempo $T^t$, i.e. $P_{T^t}(\mathbf{n})$, two potential difficulties must be overcome. Firstly, it is possible that although **r** and **n** were similar enough to be matched, the number of notes in **r** and **n** differs (as occurs in the example in figure 10). Secondly, even when **r** and **n** are identical, it still may occur that $P_{T^s}(\mathbf{r})$ and $P_{T^s}(\mathbf{n})$ are very different. For example, in one performance a note may be prolonged, and preceded by an ornamentation, whereas it is deleted in the other. This suggests that although the input problem and case are similar with respect to their score, their performances are very different.

To deal with these situations, we have defined a set of *adaptation rules* (applied in the *fifth step*), that given an annotation triple $\langle P_{T^s}(\mathbf{n}), P_{T^s}(\mathbf{r}), P_{T^t}(\mathbf{r})\rangle$, determine $P_{T^t}(\mathbf{n})$. The rules are intended to capture the *perceptual similarity* of the performances. We will illustrate this using the example adaptation rules that are shown in figure 10.

The lower rule infers the fragmentation event ($F$). This rule states that if you have an annotation triplet $\langle T, C, TT\rangle$, you may infer $F$. The motivation for this is that from a perceptual point of view (ignoring the score), changing a performance from a consolidation event ($C$) to two transformation events ($T$) amounts to changing from one performed note to two performed notes. To obtain this effect when the initial performance is a single performed note ($T$), a fragmentation event ($F$) is needed, so that two notes occur in the performance.

The upper rule infers $OT$, based on the annotation triple $\langle T, TT, OTT\rangle$. The annotation triple indicates that in the retrieved case two notes were performed as two transformation events at tempo $T^s$ and similarly at tempo $T^t$, but with an ornamentation preceding the first transformation event. The net result is thus the introduction of a ornamentation in front. Since the performance of the input problem at tempo $T^s$ is $T$, the inferred performance at tempo $T^t$ is therefore $OT$.

The examples provided above are actually instantiations of abstract rules. The abstract rule system is defined as follows: First, a symmetric and reflexive *perceptually-similar*-relation $PS$ on performance event sequences is defined on the alphabet of correspondence events $\mathcal{A} = \{T, C, F\}$. We defined $PS = \{\{T, C\}, \{TT, F\}\} \cup \{\{\mathbf{X}, \mathbf{X}\} \mid \mathbf{X} \in \mathcal{A}\}$. This relation is then used to specify three abstract adaptation rules that infer an output performance from an annotation triple:

$$\langle \mathbf{X}, \mathbf{Y}, O\mathbf{Y} \rangle \rightarrow O\mathbf{X} \quad \Leftrightarrow \{\mathbf{X}, \mathbf{Y}\} \in PS \qquad (2)$$

$$\langle \mathbf{X}, \mathbf{Y}, I\mathbf{Y} \rangle \rightarrow I\mathbf{X} \quad \Leftrightarrow \{\mathbf{X}, \mathbf{Y}\} \in PS \qquad (3)$$

$$\langle \mathbf{X}, \mathbf{Y}, \mathbf{Z} \rangle \rightarrow \mathbf{V} \Leftrightarrow \{\mathbf{X}, \mathbf{Y}\} \in PS \wedge \{\mathbf{Z}, \mathbf{V}\} \in PS \qquad (4)$$

The first rule governs the introduction of ornamentation events, the second rule governs the introduction of insertion events, and the last rule allows the introduction of any correspondence event that is perceptually similar to the performance event of the retrieved case at tempo $T^t$, whenever the performance events of the input problem and the retrieved case at tempo $T^s$ are perceptually similar. Since the rules are based on a general and theoretic conception of perceptual similarity, we believe them to be general, that is, not specific to the domain of jazz music we currently deal with.

When an annotation triple matches none of the adaptation rules, this implies that the performances of the retrieved case is too different from the performance of the input problem to be reused. In this case, UTS will be applied as a default transformation.

The quality of the solution found in this way, is defined as the proportion of notes in the input segment for which a matching adaptation rule was found.

## 4. Experimental results

In this section we describe experiments we have done in order to evaluate the *TempoExpress* system that was outlined above in comparison to default tempo transformation, that is, uniform time stretching (see section 1). We have chosen to define the quality of the tempo transformation as the distance of the transformed performance to a target performance. The target performance is a performance played at the target tempo by a human player. This approach has the disadvantage that it may be overly restrictive, in the sense that measuring the distance to just one human performance discards different performances that may sound equally natural in terms of expressiveness. In another sense it may be not restrictive enough, depending on the choice of the distance metric that is used to compare performances. It is conceivable that certain small quantitative differences between performances are perceptually very significant, whereas other, larger, quantitative differences are hardly noticeable by the human ear.

To overcome this problem, we have chosen to model the distance measure used for comparing performances after human similarity judgments. A web based survey was set up, to gather information about human judgments of performance similarity. In the rest of this section we will explain how the performance distance measure was derived from the survey results, and we will give an overview of the comparison between *TempoExpress* and uniform time stretching.

## 4.1. Obtaining the evaluation metric

The distance measure for comparing expressive performances was modeled after human performance similarity judgments, in order to prevent the risk mentioned above, of measuring difference between performances that are not perceptually relevant (or conversely, failing to measure differences that *are* perceptually relevant).

### *4.1.1. Obtaining ground truth: a web survey on perceived performance similarity*

The human judgments were gathered using a web based survey.[4] Subjects were presented a target performance *A* (the nominal performance, without expressive deviations) of a short musical fragment, and two different performances *B* and *C* of the same score fragment. The task was to indicate which of the two alternative performances was perceived as most similar to the target performance. Thus, subjects were asked questions of the form:

$$\underline{A} \text{ is most similar to } \frac{\underline{B}}{\underline{C}}$$

The underlined items could be clicked to play the corresponding performance, and listeners were asked to mark their answer by selecting either *B* or *C*, through radio-buttons.

The two alternative performances were systematically varied in the expressive dimensions: fragmentation, consolidation, ornamentation, note onset, note duration, and note loudness. One category of questions tested the proportionality of the effect quantity to perceived performance distance. In this category, versions *B* and *C* contained variations in the same expressive dimension, but to a different degree. In the case of numerical parameters like duration and dynamics, this means that in version *C* the same notes were lengthened/shortened, loudened/softened as in version *B*, but to a lesser degree. In the case of discrete parameters such as ornamentation or consolidation, version *C* would have a smaller number of those events than version *B*. Another category measured the relative influence of the type of effect on the perceived performance distance. In this category version *B* contained deviations in a different dimension than version *C* (e.g. dynamics changes vs. ornamentation, or fragmentation vs. consolidation).[5]

Ten different score fragments were used for constructing the questions (i.e. triples of performances). The fragments were manually selected motifs (varying in length from six to nine notes) from eight different jazz standards (*All of Me*, *Body and Soul*, *Black Orpheus*, *Like Someone in Love*, *Once I Loved*, *How High the Moon*, *Sophisticated Lady*, and *Autumn Leaves*). More than one score fragment were used, because in initial tests, subjects reported losing their attention after answering several questions that employed the same score fragment. Therefore, care was taken to prevent the use of the same score fragment in two subsequent questions. The three performance variants *A*, *B*, and *C* were rendered into audio using a sampled saxophone, based on manually generated specifications of the expressive deviations from the score. The deviations were defined so as to comply to the question categories mention above.

A total of 92 subjects responded to the survey, answering on average 8.12 questions (listeners were asked to answer at least 12 questions, but were allowed to interrupt the survey).

[4] The survey is available on line at: http://musje.iiia.csic.es/survey/introduction.html.

[5] In both types of questions, the performances of the labels *B* and *C* were interchanged occasionally, to prevent familiarization.

From the total set of questions (66), those questions were selected that were answered by at least ten subjects. This selection was again filtered to maintain only those questions for which there was significant agreement between the answers from different subjects (at least 70% of the answers should coincide). This yielded a set of 20 questions with answers, that is, triples of performances, together with dichotomous judgments, conveying which of the two alternative performances is closest to the target performance. The correct answer to a question was defined as the mode of all answers for that question (the filtering implies that this is the answer on which at least 70% of the subjects were in agreement). This data formed the ground truth for modeling a performance distance measure.

### *4.1.2. Modeling a performance distance measure after the ground truth*

An edit-distance metric was chosen as the basis for modeling the ground truth, because the edit-distance is flexible enough to accommodate for comparison of sequences of different lengths (in case of e.g. consolidation/fragmentation) and it allows for easy customization to a particular use by adjusting parameter values of the edit-operation cost functions. In this subsection we will explain how the distance was fit to the human performance similarity judgments by optimizing parameter values.

The distance is intended to assess the similarity between different performances of the same score. Notice that this time, we are not interested in the optimal alignment, as in the performance annotation process, where a score and a performance were matched. Moreover, since one performance is not an interpretation or variation of the other (as in the case of performance vs. score), it is conceptually inappropriate to speak of the differences between the performances in terms of e.g. ornamentation, or consolidation. To avoid confusion, we will call the edit operations in this context by their edit range, e.g. 1-0 or 1-N. To compute the distance, using equation (1), we defined the following cost functions for 1-0, 0-1, 1-1, N-1, and 1-N edit-operations, respectively:

$$w(\mathbf{s}_i, \emptyset) = \alpha_1 \left(\delta \mathcal{D}(s_i) \ + \ \epsilon \mathcal{E}(s_i)\right) + \beta_1 \tag{5}$$

$$w(\emptyset, \mathbf{t}_j) = \alpha_1 \left(\delta \mathcal{D}(t_j) + \ \epsilon \mathcal{E}(t_j)\right) + \beta_1 \tag{6}$$

$$w(\mathbf{s}_i, \mathbf{t}_j) = \alpha_2 \begin{pmatrix} \pi \ |\mathcal{P}(s_i) - \mathcal{P}(t_j)| + \\ \delta \ |\mathcal{D}(s_i) - \mathcal{D}(t_j)| + \\ o \ |\mathcal{O}(s_i) - \mathcal{O}(t_j)| + \\ \epsilon \ |\mathcal{E}(s_i) - \mathcal{E}(t_j)| \end{pmatrix} + \beta_2 \tag{7}$$

$$w(\mathbf{s}_{i-K:i}, \mathbf{t}_j) = \alpha_3 \begin{pmatrix} \pi \sum_{k=0}^{K} |\mathcal{P}(s_{i-k}) - \mathcal{P}(t_j)| + \\ \delta \ |\mathcal{D}(t_j) - \sum_{k=0}^{K} \mathcal{D}(s_{i-k})| + \\ o \ |\mathcal{O}(s_{i-K}) - \mathcal{O}(t_j)| + \\ \epsilon \sum_{k=0}^{K} |\mathcal{E}(s_{i-k}) - \mathcal{E}(t_j)| \end{pmatrix} + \beta_3 \tag{8}$$

$$w(\mathbf{s}_i, \mathbf{t}_{j-L:j}) = \alpha_3 \begin{pmatrix} \pi \sum_{l=0}^{L} |\mathcal{P}(s_i) - \mathcal{P}(t_{j-l})| + \\ \delta \ |\mathcal{D}(s_i) - \sum_{l=0}^{L} \mathcal{D}(t_{j-l})| + \\ o \ |\mathcal{O}(s_i) - \mathcal{O}(t_{j-L})| + \\ \epsilon \sum_{l=0}^{L} |\mathcal{E}(s_i) - \mathcal{E}(t_{j-l})| \end{pmatrix} + \beta_3 \tag{9}$$

**Table 2** Optimized values of edit-operation cost parameters

| $\alpha_1$ | $\alpha_2$ | $\alpha_3$ | $\beta_1$ | $\beta_2$ | $\beta_3$ | $\pi$ | $\delta$ | $o$ | $\epsilon$ |
|---|---|---|---|---|---|---|---|---|---|
| 0.031 | 0.875 | 0.243 | 0.040 | 0.330 | 0.380 | 0.452 | 1.000 | 0.120 | 0.545 |

Where $\mathbf{s}_i = \langle s_i \rangle$, and $\mathcal{P}(s_i)$, $\mathcal{D}(s_i)$, $\mathcal{O}(s_i)$, and $\mathcal{E}(s_i)$ respectively represent the pitch, duration, onset, and dynamics attributes of a note $s_i$. Each attribute has a corresponding parameter ($\pi$, $\delta$, $o$, and $\epsilon$, respectively), that controls the impact of that attribute on operation costs. The $\beta$ parameters control the absolute cost of the operations. The $\alpha$ parameters control the partial cost of the operation due to (differences in) attribute values of the notes. Note that the same $\alpha$ and $\beta$ parameters occur in the 1-0 and 0-1 cost functions, and also in the 1-N and N-1 cost functions. This ensures that the distance will be symmetric.

Fitting the edit-distance to the ground truth is a typical optimization problem, and an evolutionary optimization was used as a local search method to find good values for the ten parameters in equations (5)–(9).

The fitness function for evaluating parameter settings was defined to be the proportion of questions for which the correct answer was predicted by the edit-distance, using the parameter settings in question. A correct answer is predicted when the computed distance between the target performance and the most similar of the two alternative performances (according to the ground truth) is lower than the computed distance between the target and the remaining alternative performance. More precisely, let $Q = \{q_1, \cdots, q_n\}$ be the questions for which the ground truth is known, where $q_i$ is a triple $\langle t_i, s_i, r_i \rangle$ containing the target performance $t_i$ of question $q_i$, the alternative performance $s_i$ that was selected by the subjects as being most similar to $t_i$, and the remaining (less similar) alternative performance $r_i$ for that question. The fitness of a distance $d$ is then defined as:

$$\mathit{fitness}(d) = \frac{|\{\, q_i \in Q \mid d(t_i, s_i) < d(t_i, r_i) \,\}|}{n} \tag{10}$$

Using this fitness function a randomly initialized population of parameter settings was evolved using an elitist method for selection. That is, the fittest portion of the population survives into the next population unaltered and is also used to breed the remaining part of the next population by crossover and mutation (Goldberg, 1989). A fixed population size of 40 members was used. Several runs were performed and the fitness tended to stabilize after 300 to 400 generations. Typically the percentages of correctly predicted questions by the best parameter setting found were between 70% and 85%. The best parameter setting found (shown in table 2) was employed in the edit-distance that was subsequently used to evaluate the tempo transformed performances generated by *TempoExpress*.

### 4.2. Comparison of *TempoExpress* and uniform time stretching

In this subsection we report the evaluation results of the *TempoExpress* system on the task of tempo transformation, and compare them to the results of uniformly time stretching the performance. As said before, the evaluation criterion for the tempo transformations was the computed distance of the transformed performance to an original performance at the target tempo, using the edit-distance optimized to mimic human similarity judgments on performances.

A leave-one-out setup was used to evaluate the CBR system where, in turn, each proto case is removed from the case base, and all tempo transformations that can be derived from

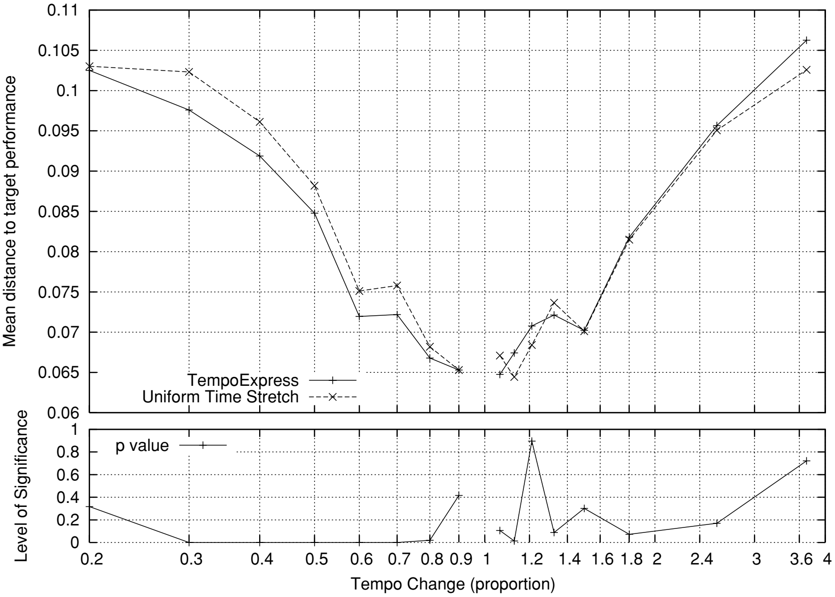


**Fig. 11** Performance of *TempoExpress* vs. uniform time stretching as a function of tempo change (measured as the ratio between target tempo and source tempo). The lower plot shows the probability of incorrectly rejecting $H_0$ (non-directional) for the Wilcoxon signed-rank tests

that proto case are performed using the reduced case base. The constraint that restricted the generation of tempo transformation problems from the proto cases was that there must be an original human performance available at the source tempo (the performance to be transformed) and another performance of the same fragment at the target tempo of the tempo transformation (this performance serves as the target performance to evaluate the transformation result). Hence the set of tempo transformation problems for a given proto case is the pairwise combination of all tempos for which a human performance was available. Note that the pairs are ordered, since a transformation from say 100 BPM to 120 BPM is not the same problem as the transformation from 120 BPM to 100 BPM. Furthermore the tempo transformations were performed on a phrase segment basis, rather than on complete phrases, since focusing on phrase level transformations is likely to involve more complex higher level aspects of performance (e.g. interactions between the performances of repeated motifs), that have not been addressed in this paper. Moreover, measuring the performance of the system on segments will give a finer grained evaluation than measuring on the phrase level.

Defining the set of tempo transformations for segments yields a considerable amount of data. Each of the 14 phrases in the case base consists of 3 to 6 motif-like segments, identified using Temperley's Melisma Grouper (Temperley, 2001), and has approximately 11 performances at different tempos (see subsection 3.1). In total there are 64 segments, and 6364 transformation problems were generated using all pairwise combinations of performances for each segment. For each transformation problem, the performance at the source tempo was transformed to a performance at the target tempo by *TempoExpress*, as well as by uniform time stretching (UTS). Both of the resulting performances were compared to the human performance at the target tempo by computing the edit-distances. This resulted in a pair of distance values for every problem. Figure 11 shows the average distance to the target

**Table 3** Overall comparison between *TempoExpress* and uniform time stretching, for upwards and downwards tempo transformations, respectively

| | mean distance to target | | Wilcoxon signed-rank test | | |
|---|---|---|---|---|---|
| | *TempoExpress* | UTS | p <> | z | *df* |
| tempo ↑ | 0.0791 | 0.0785 | 0.046 | 1.992 | 3181 |
| tempo ↓ | 0.0760 | 0.0786 | 0.000 | 9.628 | 3181 |

performance for both *TempoExpress* and UTS, as a function of the amount of tempo change (measured as the ratio between target tempo and source tempo). Note that lower distance values imply better results. The lower graph in the figure shows the probability of incorrectly rejecting the null hypothesis ($H_0$) that the mean of *TempoExpress* distance values is equal to the mean of UTS distance values, for particular amounts of tempo change. The significance was calculated using a non-directional Wilcoxon signed-rank test.

Firstly, observe that the plot in Figure 11 shows an increasing distance to the target performance with increasing tempo change (both for slowing down and for speeding up), for both types of transformations. This is evidence against the hypothesis of relational invariance, which implies that the UTS curve should be horizontal, since under relational variance, tempo transformations are supposed to be achieved through mere uniform time stretching.

Secondly, a remarkable effect can be observed in the behavior of *TempoExpress* with respect to UTS, which is that *TempoExpress* improves the result of tempo transformation specially when slowing performances down. When speeding up, the distance to the target performance stays around the same level as with UTS. In the case of slowing down, the improvement with respect to UTS is mostly significant, as can be observed from the lower part of the plot.

Finally, note that the p-values are rather high for tempo change ratios close to 1, meaning that for those tempo changes, the difference between *TempoExpress* and UTS is not significant. This is in accordance with the common sense that slight tempo changes do not require many changes, in other words, relational invariance approximately holds when the amount of tempo change is very small.

Another way of visualizing the system performance is by looking at the results as a function of absolute tempo change (that is, the difference between source and target tempo in beats per minute), as shown in figure 12. The overall forms of the absolute curves and the relative curves (figure 11) are quite similar. Both show that the improvements of *TempoExpress* are mainly manifest on tempo decrease problems.

Table 3 summarizes the results for both tempo increase and decrease. Columns 2 and 3 show the average distance to the target performance for *TempoExpress* and UTS, averaged over all tempo increase problems, and tempo decrease problems respectively. The remaining columns show data from the Wilcoxon signed-rank test. The p-values are the probability of incorrectly rejecting $H_0$ (that there is no difference between the *TempoExpress* and UTS results). This table also shows that for downward tempo transformations, the improvement of *TempoExpress* over UTS is small, but extremely significant ($p < .001$), whereas for upward tempo transformations UTS seems to be better, but the results are slightly less decisive ($p < .05$).

How can the different results for tempo increase and tempo decrease be explained? A practical reason can be found in the characteristics of the case base. Since the range of tempos at which the performances were played varies per song, it can occur that only one song is represented in some tempo range. For example, for *Up Jumped Spring* the tempos range from 90 BPM to 270 BPM, whereas the highest tempo at which performances of other songs are available is 220 BPM. That means that in the leave-one-out method, there are no

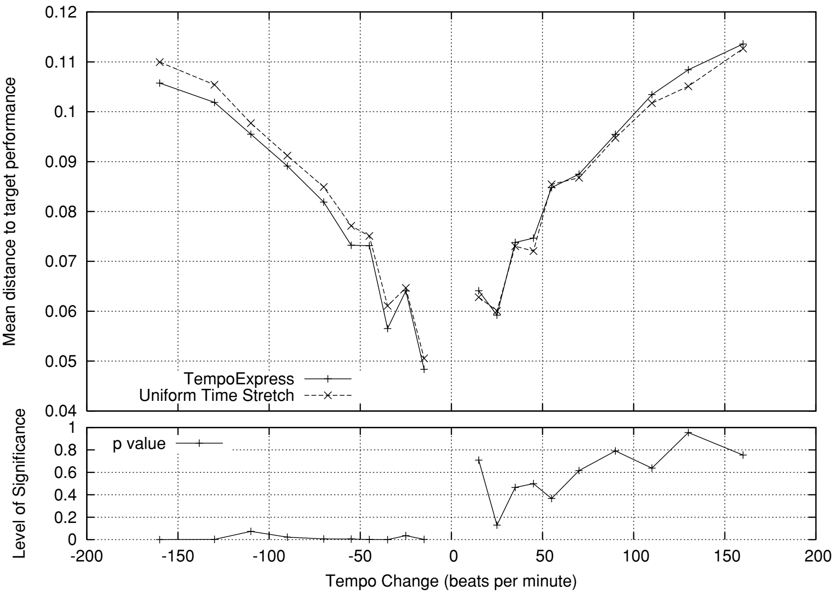


**Fig. 12** Performance of *TempoExpress* vs. UTS as a function of tempo change (measured in beats per minute). The lower plot shows the probability of incorrectly rejecting $H_0$ (non-directional) for the Wilcoxon signed-rank tests

precedents for tempo transformations to tempos in the range from 220 BPM to 270 BPM. This may explain the increasing gap in performance in favor of UTS, towards the end of the spectrum of upward tempo transformations.

## 5. Conclusion and future work

In this paper we presented our research results on global tempo transformations of music performances. We are interested in the problem of how a performance played at a particular tempo can be rendered automatically at another tempo preserving some of the features of the original tempo and at the same time sounding natural in the new tempo. We focused our study in the context of standard jazz themes and, specifically on saxophone jazz recordings.

We proposed a case-based reasoning approach for dealing with tempo transformations and presented the *TempoExpress* system. *TempoExpress* has a rich description of the musical expressivity of the performances, that includes not only timing and dynamics deviations of performed score notes, but also represents more extensive kinds of expressivity such as note ornamentation, and note consolidation/fragmentation. We apply edit-distance techniques in the retrieval step, as a means to assess similarities between the cases and the input problem. In the reuse step we employ constructive adaptation. Constructive adaptation is a technique able to generate a solution to a problem by searching the space of partial solutions for a complete solution that satisfies the solution requirements of the problem.

Moreover, we described the results of our experimentation over a case-base of more than six thousand transformation problems. *TempoExpress* clearly behaves better than a Uniform Time Stretch (UTS) when the target problem is slower than the source tempo. When the target tempo is higher than the source tempo the improvement is not significant.

Nevertheless, *TempoExpress* behaves as UTS except in transformations to really fast tempos. This result may be explained by the lack of cases with tempos higher than 220 BPM. Summarizing the experimental results, for downward tempo transformations, the improvement of TempoExpress over UTS is small, but extremely significant ($p < .001$), whereas for upward tempo transformations UTS seems to be better, but the results are slightly less decisive ($p < .05$).

Future work includes a more thorough investigation of how the quality of the tempo transformations is affected by varying parameters of the system, such as the preferred segment size, the tempo tolerance window, and the I-R-similarity threshold for proto case selection.

In this paper, we have shown that the CBR framework potentially allows for handling inter-dependencies of phrase segment performances. In the future, we intend to take advantage of this by introducing constraints that make such inter-dependencies explicit.

As for evaluation, we wish to extend the experiments to analyze the performance of *TempoExpress* with respect to the complete phrases. Also, instead of evaluating the system by measuring the distance to a target performance, it is preferable that human listeners judge the musical quality of the tempo transformed performances directly.

**Acknowledgments** We wish to thank Gerhard Widmer, and the anonymous reviewers for their valuable feedback on the first draft of this article, and Emilia Gòmez for the analysis and synthesis of the audio.

This research has been partially supported by the Spanish Ministry of Science and Technology under the project TIC 2003-07776-C2-02 "CBR-ProMusic: Content-based Music Processing using CBR" and EU-FEDER funds.

## References

Arcos, J. L., Grachten, M., & López de Mántaras, R. (2003). Extracting performer's behaviors to annotate cases in a CBR system for musical tempo transformations. In *Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03)*, number 2689 in Lecture Notes in Artificial Intelligence, pages 20–34. Springer-Verlag.

Arcos, J. L., López de Mántaras, R., & Serra, X. (1998). Saxex : a case-based reasoning system for generating expressive musical performances. *Journal of New Music Research, 27:3*, 194–210.

Bonada, J. (2000). Automatic technique in frequency domain for near-lossless time-scale modification of audio. In *Proceedings of the International Computer Music Conference*.

Canazza, S., Poli, G. D., Rinaldin, S., & Vidolin, A. (1997). Sonological analysis of clarinet expressivity. In Leman, M., editor, *Music, Gestalt, and Computing: studies in cognitive and systematic musicology*, number 1317 in Lecture Notes in Artificial Intelligence, pages 431–440. Springer.

Desain, P. & Honing, H. (1994). Does expressive timing in music performance scale proportionally with tempo? *Psychological Research*, 56:285–292.

Downie, J., West, K., Ehmann, A., & Vincent, E. (2005). The 2005 Music Information Retrieval Evaluation eXchange (MIREX 2005): Preliminary overview. In *Prooceedings of the 6th International Conference on Music Information Retrieval*.

Friberg, A. (1991). Generative rules for music performance: A formal description of a rule system. *Computer Music Journal, 15:2*, 56–71.

Friberg, A., Colombo, V., Frydén, L., & Sundberg, J. (2000). Generating musical performances with Director Musices. *Computer Music Journal, 24:1*, 23–29.

Friberg, A. & Sundström, A. (2002). Swing ratios and ensemble timing in jazz performance: evidence for a common rhythmic pattern. *Music Perception, 19:3*, 333–349.

Gabrielsson, A. (1987). Once again: The theme from Mozart's piano sonata in A major (K. 331). A comparison of five performances. In Gabrielsson, A., editor, *Action and perception in rhythm and music*, pages 81–103. Royal Swedish Academy of Music, Stockholm.

Gabrielsson, A. (1995). Expressive intention and performance. In Steinberg, R., editor, *Music and the Mind Machine*, pages 35–47. Springer-Verlag, Berlin.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Gómez, E., Gouyon, F., Herrera, P., & Amatriain, X. (2003a). Using and enhancing the current MPEG-7 standard for a music content processing tool. In *Proceedings of Audio Engineering Society, 114th Convention*, Amsterdam, The Netherlands.

Gómez, E., Grachten, M., Amatriain, X., & Arcos, J. L. (2003b). Melodic characterization of monophonic recordings for expressive tempo transformations. In *Proceedings of Stockholm Music Acoustics Conference 2003*.

Gómez, E., Klapuri, A., & Meudic, B. (2003c). Melody description and extraction in the context of music content processing. *Journal of New Music Research*, 32(1).

Gouyon, F., Fabig, L., & Bonada, J. (2003). Rhythmic expressiveness transformations of audio recordings: swing modifications. In *Proceedings of the 6th International Conference on Digital Audio Effects*.

Grachten, M., Arcos, J. L., & López de Mántaras, R. (2004a). Evolutionary optimization of music performance annotation. In *Computer Music Modeling and Retrieval*, Lecture Notes in Computer Science. Springer.

Grachten, M., Arcos, J. L., & López de Mántaras, R. (2004b). TempoExpress, a CBR approach to musical tempo transformations. In *Advances in Case-Based Reasoning. Proceedings of the 7th European Conference, ECCBR 2004*, Lecture Notes in Computer Science. Springer.

Grachten, M., Arcos, J. L., & López de Mántaras, R. (2005). Melody retrieval using the Implication/Realization model. MIREX http://www.music-ir.org/evaluation/mirex-results/article/s/similarity/grachten.pdf .

Hazan, A., Ramirez, R., Maestre, E., Perez, A., & Pertusa, A. (2006). Modelling expressive performance: A regression tree approach based on strongly typed genetic programming. In et al., R., editor, *Proceedings on the 4th European Workshop on Evolutionary Music and Art*, pages 676–687, Budapest, Hungary.

Honing, H. (2002). Structure and interpretation of rhythm and timing. *Tijdschrift voor Muziektheorie, 7:3*, 227–232.

Honing, H. (2006). Is expressive timing relational invariant under tempo transformation? *Psychology of Music*. (in press).

Juslin, P. (2001). Communicating emotion in music performance: a review and a theoretical framework. In Juslin, P. and Sloboda, J., editors, *Music and emotion: theory and research*, pages 309–337. Oxford University Press, New York.

Lerdahl, F. & Jackendoff, R. (1993). An overview of hierarchical structure in music. In Schwanaver, S. M. and Levitt, D. A., editors, *Machine Models of Music*, pages 289–312. The MIT Press. Reproduced from Music Perception.

Lindström, E. (1992). 5 x “oh, my darling clementine”. the influence of expressive intention on music performance. Department of Psychology, Uppsala University.

López de Mántaras, R. & Arcos, J. L. (2002). AI and music: From composition to expressive performance. *AI Magazine, 23:3*, 43–58.

Maestre, E. & Gómez, E. (2005). Automatic characterization of dynamics and articulation of expressive monophonic recordings. In *Proceedings of the 118th Audio Engineering Society Convention*, Barcelona.

Narmour, E. (1990). *The analysis and cognition of basic melodic structures : the implication-realization model*. University of Chicago Press.

Palmer, C. (1996). Anatomy of a performance: Sources of musical expression. *Music Perception, 13:3*, 433–453.

Plaza, E. & Arcos, J. L. (2002). Constructive adaptation. In Craw, S. and Preece, A., editors, *Advances in Case-Based Reasoning*, number 2416 in Lecture Notes in Artificial Intelligence, pages 306–320. Springer-Verlag.

Repp, B. H. (1994). Relational invariance of expressive microstructure across global tempo changes in music performance: An exploratory study. *Psychological Research*, 56:285–292.

Repp, B. H. (1995). Quantitative effects of global tempo on expressive timing in music performance: Some perceptual evidence. *Music Perception, 13:1*, 39–58.

Röbel, A. (2003). A new approach to transient processing in the phase vocoder. In *Proceedings of the 6th International Conference on Digital Audio Effects*.

Schellenberg, E. G. (1997). Simplifying the Implication-Realization model of melodic expectancy. *Music Perception, 14:3,* 295–318.

Sloboda, J. A. (1983). The communication of musical metre in piano performance. *Quarterly Journal of Experimental Psychology*, 35A:377–396.

Sundberg, J., Friberg, A., & Frydén, L. (1991a). Common secrets of musicians and listeners: an analysis-by-synthesis study of musical performance. In Howell, P., West, R., and Cross, I., editors, *Representing Musical Structure*, Cognitive Science series, chapter 5. Academic Press Ltd.

Sundberg, J., Friberg, A., & Frydén, L. (1991b). Threshold and preference quantities of rules for music performance. *Music Perception*, 9:71–92.

Suzuki, T. (2003). The second phase development of case based performance rendering system “Kagurame”. In *Working Notes of the IJCAI-03 Rencon Workshop*, pages 23–31.

Temperley, D. (2001). *The Cognition of Basic Musical Structures*. MIT Press, Cambridge, Mass.

The Real Book (2004). The Real Book: Sixth Edition. Hal Leonard Corporation.

Timmers, R. and Ashley, R., Desain, P., Honing, H., & Windsor, L. (2002). Timing of ornaments in the theme of Beethoven's Paisiello Variations: Empirical data and a model. *Music Perception, 20:1*, 3–33.

Tobudic, A. & Widmer, G. (2004). Case-based relational learning of expressive phrasing in classical music. In *Proceedings of the 7th European Conference on Case-based Reasoning (ECCBR'04)*, Madrid.

Typke, R., Hoed, M., De Nooijer, J., Wiering, F., & Veltkamp, R. (2005). A ground truth for half a million musical incipits. In *Proceedings of the Fifth Dutch-Belgian Information Retrieval Workshop*, pages 63–70.

Widmer, G. (1996). Learning expressive performance: The structure-level approach. *Journal of New Music Research, 25:2*, 179–205.

Widmer, G. (2000). Large-scale induction of expressive performance rules: First quantitative results. In *Proceedings of the International Computer Music Conference (ICMC2000)*, San Francisco, CA. International Computer Music Association.

Widmer, G. (2002). Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research, 31:1,* 37–50.