Tempo-Express: tempo transformations preserving musical expression

Maarten Grachten, Josep Lluís Arcos, and Ramon López de Mántaras
IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia, Spain.
{maarten,arcos,mantaras}@iiia.csic.es

Abstract

Tempo-Express is a case-based reasoning system for tempo transformation of musical performances, that preserves expressivity in the context of standard jazz themes. Expressive tempo transformations are a common manual operation in audio post-processing tasks. A CBR system can be a useful tool for automating this task.

Keywords: Expressive Music Performance, Case Based Reasoning, Tempo Transformations, Performance annotation.

1 Introduction

Tempo-Express is a case-based reasoning system for generating expressive tempo transformations in the context of standard jazz themes. As shown by [Desain and Honing, 1991], 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. When a human performer plays a given melody at different tempos, she does not perform uniform transformations. On the contrary, the relative importance of the notes will determine, for each tempo, the performer's decisions. For instance, if the tempo is very fast, the performer will, among other things, tend to emphasize the most important notes by not playing the less important ones. Alternatively, in the case of slow tempos, the performer tends to delay some notes and anticipate others.

In the development of *Tempo-Express* we are using the experience acquired in developing the SaxEx system [Arcos and López de Mántaras, 2001]. The goal of SaxEx was also to generate expressive music performances but the task was centered on transforming a non expressive input performance into an expressive new sound file taking into account the user preferences regarding the desired expressive output characterized along three affective dimensions (tender-aggressive, sad-joyful, calm-restless). The task of *Tempo-Express* is to perform tempo transformations with 'musical meaning' to an already expressive input performance. That is, a recording has to be re-performed at a user required tempo that can be very different from the input tempo.

Tempo-Express is being implemented in *Noos* [Arcos and Plaza, 1997; 1996], an object-centered representation language designed to support the development of knowledge intensive case based reasoning systems.

The paper is organized as follows: In section 2.1, we will describe the musical analysis that is applied to the musical scores. The annotation of the performances will be explained in section 2.2. In sections 2.3–2.6, the case-based reasoning system and it's phases (retrieval, adaptation, retention) will be explained. In section 3, we present the conclusions.

2 Tempo-Express

In this section we briefly present the main *Tempo-Express* modules and the Inference flow (see Figure 1). The input of *Tempo-Express* is a recording of a jazz performance at a given tempo T_i (a sound file), its corresponding score (a MIDI file) and the desired output tempo. The score contains the melodic and the harmonic information of the musical piece and is analyzed automatically in terms of the Implication/Realization Model (see [Narmour, 1990]). The recording is parsed by the performance analysis module that produces an XML file containing the performance melody segmentation (onset points and durations of notes). Then, the melody segmentation is compared to its corresponding score in the annotation process (see the detailed description in section 2.2). The 'annotated' performance and the desired output tempo T_o is the input for the case-based reasoning. The task of the case-based reasoning modules is to determine a sequence of operations that achieves the desired tempo transformation with a musical expressivity that is appropriate for that tempo, while maintaining as much as possible the expressivity that was present in the original performance. Finally, the output of the system is a new version of the original melody, at the desired tempo, generated by the synthesis process.

The performance analysis and synthesis processes have been implemented by the Music Technology Group (MTG) of the Pompeu Fabra University using signal spectral modeling techniques (see [Serra *et al.*, 1997; Gómez *et al.*, 2003] for a detailed description).

2.1 Musical Analysis: Narmour's Implication/Realization Model

An intuition shared by many people is that appreciating music has to do with expectation. That is, what we have already heard generates expectations of what is to come. Narmour [Narmour, 1990] proposed a theory of cognition of

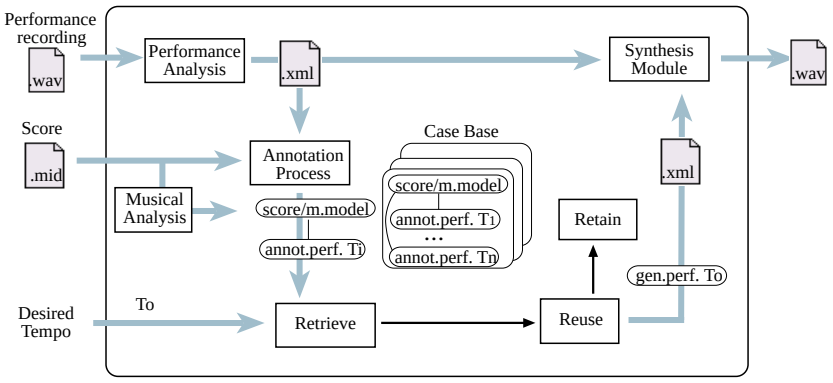

Figure 1: General view of *Tempo-Express* modules.

melodies based on a set of basic grouping structures (see figure 2). These structures characterize patterns of melodic implications (or expectations) that constitute the basic units of the listener's perception. Other resources emphasize or inhibit the perception of these melodic implications. These resources fall into two categories: *bottom-up* influences, such as duration and rhythmic patterns on the one hand, and *top-down* influences, such as the listener's memory (of previously heard musical material). The use of the Implication/Realization (I/R) model provides a musical analysis of the melodic surface of the piece.

The basic grouping structures are shown in figure 2 (from [Narmour, 1991]). The two dimensions in the classification of two (or possibly more) contiguous intervals are: the continuation/change of registral direction, and the sameness/difference of interval sizes. For example, the first pattern (called *P*, for *process*) of figure 2 is a continuation of registral direction, with similar (small) interval sizes. The *P* structure can also manifest itself through only one of the dimensions: *VP* has the same registral direction, but dissimilar interval sizes, and *IP* has similar interval sizes, but a reversed registral direction. The *R* (*reversal*) structure is a basic structure that changes in both direction and interval size. Just like *P*, *R* has two variants (*VR* and *IR*) where the characteristic applies to only one of the dimensions. Two special cases are firstly the *D* structure where both interval sizes are zero (this implies that the registral directions are lateral (rather than upward or downward), and secondly the *ID*, where registral directions is inverted and the interval sizes are equal. Based on these eight basic structures, many more derived structures can be identified. See [Narmour, 1990] for a detailed description.

Based on the I/R model, we have built a parser for monophonic melodies, that automatically generates the corresponding sequences of I/R structures. The algorithm implements most of the *bottom-up* part of the I/R model. The *top-down* part is not implemented at this moment. Nevertheless, we believe that the I/R analyses generated by the parser have a reasonable degree of validity (top-down interferences being only occasional exceptions to the bottom-up rules) and are useful for our purposes.

2.2 The Performance Annotation Process

When comparing a performance of a melody to the score of that melody, a crucial problem is to identify which element in the performance corresponds to each note of the score. Especially in jazz performances, which is the area on which we will focus, this problem is not trivial, since jazz performers often favor a 'liberal' interpretation of the score. This does not only involve changes in expressive features of the score elements as they are performed, but also omitting or adding notes. Thus, one can normally not assume that the performance contains a corresponding element for every note of the score, neither that every element in the performance corresponds to a note of the score. Thus, to make sense of the performance in the light of the melody that was performed, some kind of annotation of the performance will be needed. In particular, a description of a musical performance could take the form of a sequence of operations that are applied to the score elements. These operations could make explicit both the mapping of score elements to performance elements, and the deviation of the expressive values (e.g. timing and duration) of the performance elements with respect to the mapped score elements.

We use a variant of the edit distance (also known as Levenshtein distance [Levenshtein, 1966]) to generate an alignment between scores and performances. The edit distance has been used before in the performance-to-score mapping problem by Dannenberg [Dannenberg, 1984] and Large [Large, 1993], among others. This alignment can contain deletions, insertions (1-to-0 and 0-to-1 mappings respectively), replacements (1-to-1 mappings) and fragmentations and consolidations (1-to-many and many-to-1 mappings respectively). An example of such an alignment is shown in figure 3 (for simplicity, the sequences consist of geometrical figures instead of score and performance elements). To determine a preference among the possible alignments, a cost is assigned to each edit operation. The edit distance mechanism then considers all possible sequences of operations that transform the first sequence into the second, and selects the one that has the lowest total cost. This sequence of operations constitutes the *optimal alignment* between the two sequences under comparison.

Once the alignment between a score and its performance is established, it is easy to derive the expressive deviations

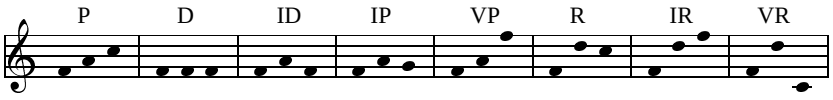

Figure 2: Eight of the basic structures of the I/R model.

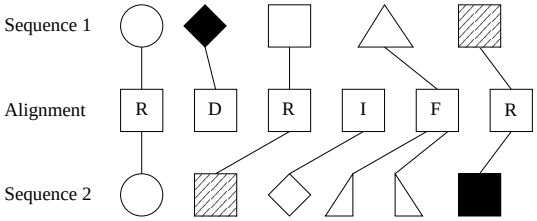

Figure 3: A possible alignment between two sequences of geometrical figures. The operations in the alignment are replacement (R), deletion (D), insertion (I) and fragmentation (F). The way in which the compared sequences are aligned depends on the costs assigned to each of the edit operations.

of the performance elements from the score elements. This alignment, together with the expressive deviations, forms the annotation of the performance, to be stored in the case base. (See [Arcos *et al.*, 2003] for a more detailed description about the annotation process.)

2.3 The Case Base

A case is represented as a complex structure embodying three different kinds of knowledge: (1) the representation of the musical score (notes and chords), (2) a musical model of the score (automatically inferred from the score using Narmour's Implication/Realization model as background musical knowledge as described in Section 2.1), and (3) a collection of annotated performances, as described in Section 2.2 For the case acquisition, saxophone performances were recorded of 5 jazz standards, each consisting of about 4 distinct phrases. The performances were played by a professional performer, at about 11 different tempos per phrase. From this, the initial case base was constructed, containing 20 scores of musical phrases, each with about 11 annotated performances (i.e. more than 5.000 performed notes).

2.4 The Retrieval Step

The retrieval mechanism is organized in three phases:

- In a first phase the input melody is compared with the melodies of the case base using melodic similarity measures for retrieving only those case melodies really similar—For instance, given a slow ballad as input, we are not interested in comparing it with be-bop themes. To this end, similarities are computed between the input melody and each of the melodies in the case base. The similarities can be computed either using the note representations of the melodies, or using other representations such as melodic contour representations, or the musical models derived from the melodies. Combinations of these different measures can also be used to retrieve a subset of melodies. In [Grachten *et al.*, 2002], a comparison of various similarity measures is reported. It turns out that a similarity measure based on note level representations is not very discriminative when applied to melodies that are very different (i.e. all phrases that are not virtually identical, are assessed more or less equally dissimilar). Therefore, it seems more promising to use similarities between musical models or contour representations for retrieving similar melodies from the case base. The final output of this phase is a subset of a melodies of the case base close to the input melody. Only performances from these retrieved melodies will be taken into account in the following phases.

- In a second phase, we try to find similar melodic fragments for segments of the input melody. The input melody is segmented based on the musical model that was constructed for the melody (including musical information such as the metrical strengths of the notes). In particular, I/R structures (or sequences of two or three structures) usually coincide with melodic motifs. For each of these segments, the most similar parts of the retrieved melodies can be selected (again using a similarity measure on either of the melodic representations). The result is that for each segment/motif of the input melody a set of melodic fragments is available, each fragment with one or more performance annotations.

- Finally, in the third phase the performances that were retrieved for each segment of the input melody are ranked using a similarity measure for the performance annotations. The idea behind this step is to use the input performance as a guide for how the input melody should be performed at the desired tempo. For illustration, say that an input performance P_{in} of a melodic segment M at tempo T_{in} was given and a new performance P_{out} of M at the desired tempo T_{out} must be generated. Suppose that there is a retrieved melodic fragment that has a performance P_1 at tempo T_i close to T_{in} and in addition, it has a performance P_2 at T_j close to the desired tempo T_{out}. Then, if performance P_{in} is similar to P_1, we assume that P_2 is a good basis for constructing performance P_{out} at tempo T_{out}. The setting is shown diagrammatically in figure 4. In this way, performances at tempos close to the desired tempo can be selected and ordered according to their expected relevance for constructing the solution.

In conclusion, the output of the retrieval step is an ordered collection of candidate annotations for each segment/motif in the input melody.

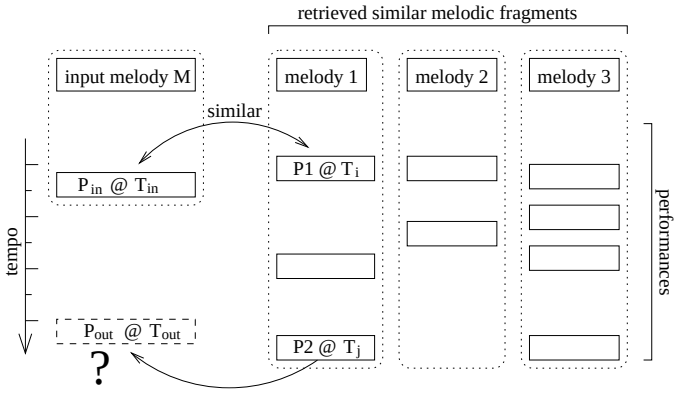

Figure 4: A diagrammatic representation of the third phase of the retrieval step. Melodies 2 and 3 do not contribute to the solution, either because they have no performances at tempos close to T_{in} or T_{out}, or their performances at tempos close to T_{in} are not similar to P_{in}.

2.5 The Adaptation Step

The adaptation mechanism is being implemented using *constructive adaptation* [Plaza and Arcos, 2002], a generative technique for reuse in CBR systems. The adaptation mechanism deals with two kinds of criteria: local criteria and coherence criteria. Local criteria deal with the transformations to be performed to each note—i.e. how retrieved candidate annotations can be reused in each input note. Coherence criteria try to balance smoothness and hardness. Smoothness and hardness are basically contradictory: the first tends to iron out strong deviations with respect to the score, while the second tends to favor strong deviations. The resulting expressive performance is a compromise between the smoothness and hardness criteria, with the aim of keeping an overall balance pleasant to the ear.

2.6 The Retain Step

The user decides whether or not each new tempo performance should be added to the memory of cases. The newly added tempo performances will be available for the reasoning process in future problems.

3 Conclusions

We have briefly described a case-based reasoning approach to tempo transformations of musical performances that preserves the expressivity of the input performances. The most fundamental step, the annotation process, has been completed. This process allows to build the case base. We are now developing the remaining case-based reasoning components. As future work we are also planning to consider an alternative functionality of the system consisting in generating an expressive tempo performance just from the score (i.e. without an initial performance) instead of transforming an existing one.

Acknowledgments

This research has been partially supported by the Spanish Ministry of Science and Technology under the project TIC 2000-1094-C02 “Tabasco: Content-based Audio Transformation using CBR”.

References

[Arcos and López de Mántaras, 2001] Josep Lluís Arcos and Ramon López de Mántaras. An interactive case-based reasoning approach for generating expressive music. *Applied Intelligence*, 14(1):115–129, 2001.

[Arcos and Plaza, 1996] Josep Lluís Arcos and Enric Plaza. Inference and reflection in the object-centered representation language Noos. *Journal of Future Generation Computer Systems*, 12:173–188, 1996.

[Arcos and Plaza, 1997] Josep Lluís Arcos and Enric Plaza. Noos: An integrated framework for problem solving and learning. In *Knowledge Engineering: Methods and Languages*, 1997.

[Arcos *et al.*, 2003] Josep Lluís Arcos, Maarten Grachten, and Ramon López de Mántaras. 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)*, 2003. To appear.

[Dannenberg, 1984] R. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the 1984 International Computer Music Conference*. International Computer Music Association, 1984.

[Desain and Honing, 1991] Peter Desain and Henkjan Honing. Tempo curves considered harmful. a critical review of the representation of timing in computer music. In *Proceedings of the 1991 International Computer Music Conference*, San Francisco, 1991. Computer Music Association.

[Gómez *et al.*, 2003] E. Gómez, , A. Klapuri, and B. Meudic. Melody description and extraction in the context of music content processing. *Journal of New Music Research*, 32(1), 2003. (In Press).

[Grachten *et al.*, 2002] Maarten Grachten, Josep Lluís Arcos, and Ramon López de Mántaras. A comparison of different approaches to melodic similarity. In *IInd International Conference on Music and Artificial Intelligen ce*, 2002.

[Large, 1993] E. W. Large. Dynamic programming for the analysis of serial behaviors. *Behavior Research Methods, Instruments & Computers*, 25(2):238–241, 1993.

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

[Narmour, 1990] Eugene Narmour. *The Analysis and cognition of basic melodic structures : the implication-realization model*. University of Chicago Press, 1990.

[Narmour, 1991] Eugene Narmour. The melodic structures of music and speech: Applications and dimensions of the implication-realization model. In Johan Sundberg, Lennart Nord, and Rolf Carlson, editors, *Music, Language, Speech*

and Brain. MacMillan Academic and Professional Ltd, 1991.

[Plaza and Arcos, 2002] Enric Plaza and Josep Ll. Arcos. Constructive adaptation. In Susan Craw and Alun Preece, editors, *Advances in Case-Based Reasoning*, number 2416 in Lecture Notes in Artificial Intelligence, pages 306–320. Springer-Verlag, 2002.

[Serra *et al.*, 1997] Xavier Serra, Jordi Bonada, Perfecto Herrera, and Ramon Loureiro. Integrating complementary spectral methods in the design of a musical synthesizer. In *Proceedings of the ICMC'97*, pages 152–159. San Francisco: International Computer Music Asociation., 1997.