A Case Based Approach to the Generation of Musical Expression

Taizan Suzuki Takenobu Tokunaga Hozumi Tanaka
Department of Computer Science
Tokyo Institute of Technology
2-12-1, Oookayama, Meguro, Tokyo 152-8552, Japan.
E-mail: {taizan, take, tanaka}@cs.titech.ac.jp

Abstract

The majority of naturally sounding musical performance has musical expression (fluctuation in tempo, volume, etc.). Musical expression is affected by various factors, such as the performer, performative style, mood, and so forth. However, in past research on the computerized generation of musical expression, these factors are treated as being less significant, or almost ignored. Hence, the majority of past approaches find it relatively hard to generate multiple performance for a given piece of music with varying musical expression.

In this paper, we propose a case-based approach to the generation of expressively modulated performance. This method enables the generation of varying musical expression for a single piece of music. We have implemented the proposed case-based method in a musical performance system, and, we also describe the system architecture and experiments performed on the system.

1 Introduction

Almost all musicians play music with musical expression (varying of tempo, volume, etc.). They consider how the target pieces should be played, and they elaborate upon it with tempo curves and change in volume. Thus, musical expression is a highly significant element in making performance pleasant and attractive.

Many past research efforts have focused on the computerized generation of musical expression. The majority of them employ musical expression rules, which define relations between phrase characteristics and musical expression (Figure 1). Past approaches have used rules of musical expression manually acquired by human researchers ([Fryden and Sundberg, 1984], [Friberg and Sundberg, 1986], [Friberg, 1991], and [Noike *et* a/., 1992]). Here, expressively modulated performance is generated by applying these rules to the target piece. Some recent research efforts have introduced learning mechanisms into the acquisition of rules ([Bresin *et* a/., 1992 , [Chafe, 1997], [Widmer, 1993b], [Widmer, 1993a], and [Widmer, 1995]). These approaches extract rules of musical expression from sample performance data played by human musicians. Since the above methods generate and apply rules of musical expression, they are called *rule-based approaches.*

One advantage of rule-based approaches is, once the rule set is established, it is applicable to any piece of music. Another advantage is transparency in that users can access rules for musical expression used in the generation process. These rules are useful for cognitive research.

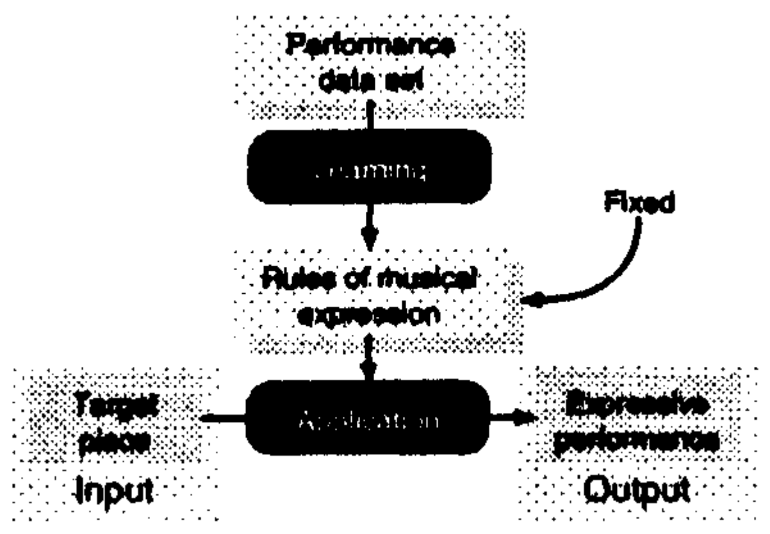

Figure 1: The basic mechanism employed by rule-based approaches

On the other hand, rule-based approaches have some drawbacks. The most serious one is that these approaches are hard to adapt to the generation of performances with different styles.

Generally, musical expression has vast freedom and a broad range of tolerance. Musical expression varies according to various factors, for instance, the performer, style (e.g. "Baroque", "Romantic", etc.), mood (e.g. "lively", "calm", etc.), and so forth. We call these factors *performance conditions.*

To generate suitable musical expression by computer, these performance conditions must be taken into consideration. However, as was seen for rule-based approaches, it is hard to introduce these factors into the process of generation. Besides, performance conditions are difficult to describe in term of rules of musical expression, since they consist of various elements and each element continuously changes. Thus, there is little research which has considered such factors ([Canazza *et* a/., 1997]).

On the other hand, there is very little research which has employed non-rule-based approaches. Arcos, et al. applied case based reasoning (CBR) to the generation of musical expression ([Arcos *et* a/., 1997]). Their approach uses a performance data set as a musical expression knowledge base. For each note in the target piece, it retrieves similar notes from the knowledge base, analyzes musical expression in these similar notes, and applies to the target piece. However, Arcos, et al. do not take any kind of performance conditions into account, such that, their approach cannot generate performance variety, similarly to rule-based approaches.

We aim to develop a method of computerized generation of natural musical expression which incorporates a range of performance conditions. To overcome the problems faced by conventional methods, we propose a new case-based method for the generation of musical expression. The advantage of this method is that it can easily consider performance conditions, to be able to generate various kinds of musical expression for a single piece of music in accordance with performance condition settings. We have implemented the case-based method proposed in this paper in a music performance system. In the remainder of this paper, we present our case-based method for the generation of musical expression, and discuss the architecture of the performance system incorporating this method, and experiments on it.

2 Case-based method for musical expression generation

2.1 Concept

Figure 2 shows a rough outline of our method. Our method uses a *performance data set* consisting of pre-analyzed musical pieces, from which an *example data set* is extracted for use as the musical expression knowledge base. An example data set is acquired for each inputted target piece. Moreover, we evaluate the significance of each example piece to the input piece by considering the structural resemblance of the two pieces and similarity between performance conditions for the input and example piece. The resultant performance is generated based on the example data set and the various significance values. Hence, even if the example pieces are fixed, the generated performance will change according to the input performance conditions. This mechanism realizes our aim of generating varying musical expression.

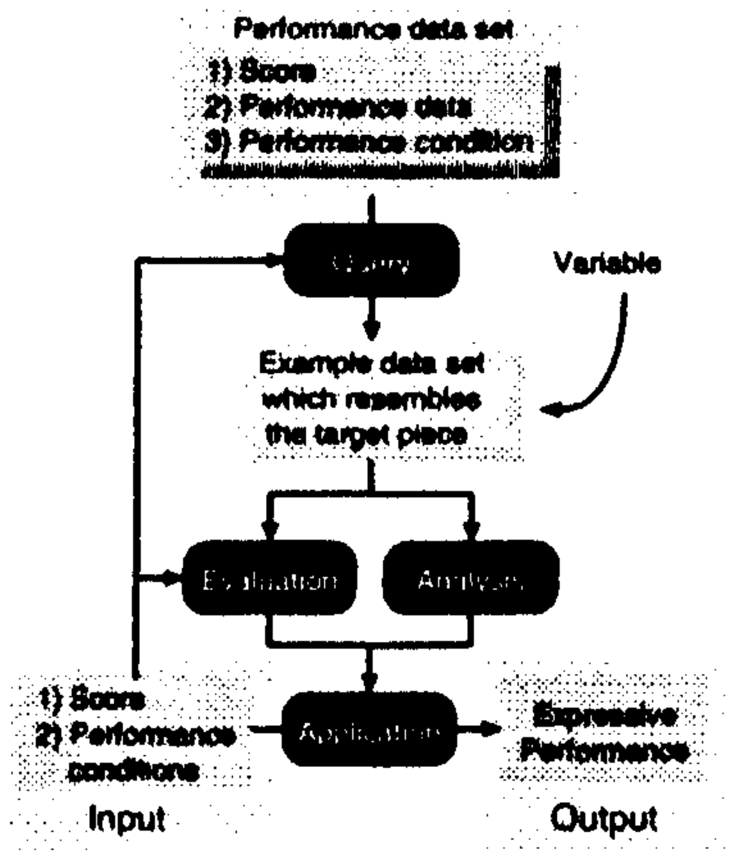

Figure 2: Rough outline of our case-based method for musical expression generation

2.2 Algorithm

This section describes the basic architecture used in our case-based method. Figure 3 shows the algorithm of our method.

Our method requires a performance data set, which is a set of musical performance data performed by human musicians. Each data component has not only a record of the event sequence (note on, note off, pedal control, etc.) but also the musical score of the performed piece and the performance conditions under which the data was

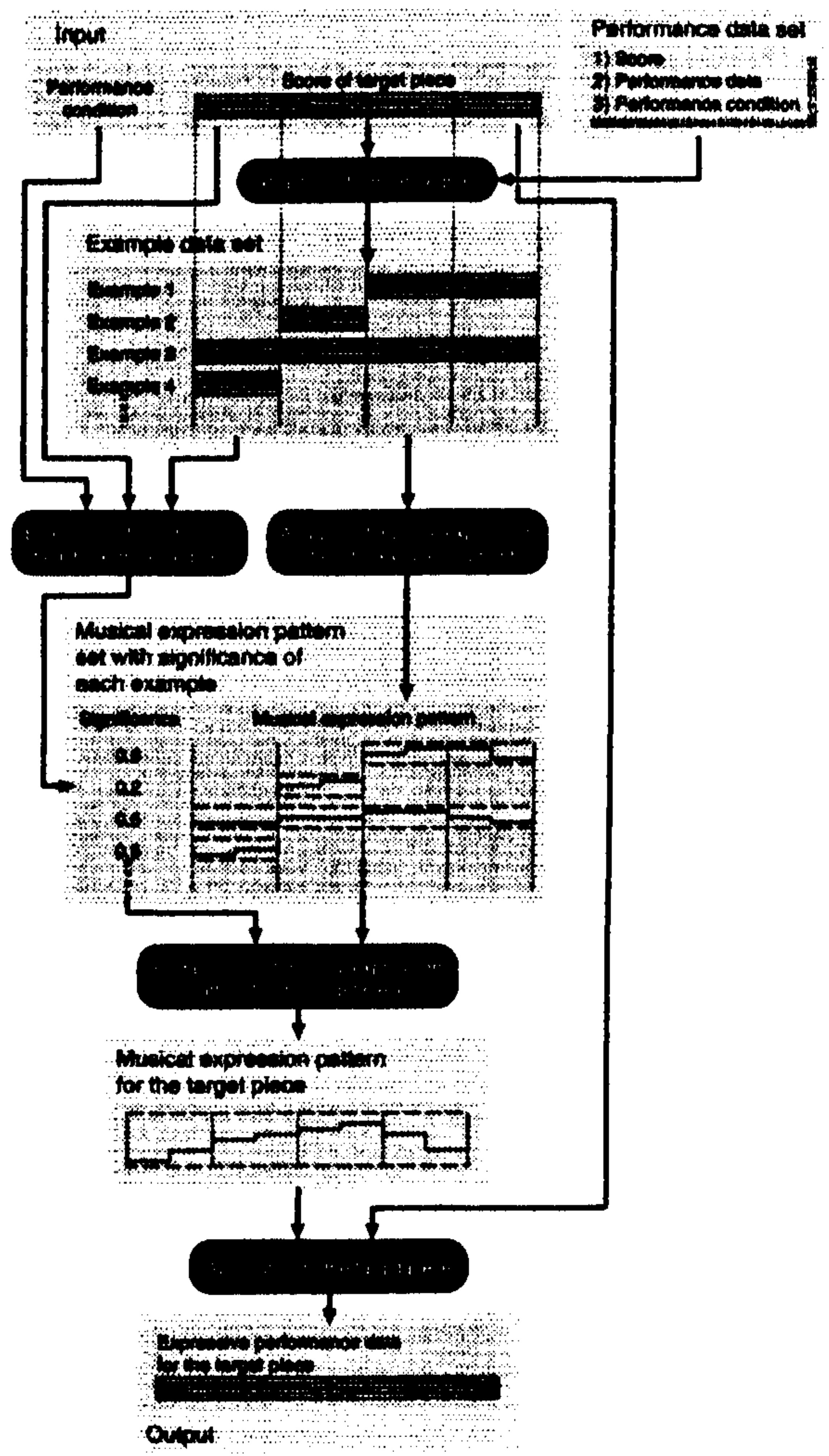

Figure 3: Overview of the case-based method for musical expression generation

recorded. This performance data set must be collected beforehand.

Our method comprises the following stages: 1) input the musical score of the target piece and performance condition settings, 2) extract similar parts (called the example segment set) from the performance data set, 3) analyze musical expression in each example segment, 4) evaluate the significance of each example segment, 5) compose the musical expression pattern for the target piece, and, 6) apply the musical expression pattern to the target piece.

Input data consists of information about the target piece taken from the musical score and performance condition settings. The musical score information is not only the information about note sequence but also accompanying information (e.g. beats, bars, repetitions, pauses, etc.). The performance condition settings are parameters which decide the characteristics and mood of the generated performance. A description of the performance condition settings is presented in Section 3.2.

In the extraction stage, our method divides both the target piece and each example piece in the performance data set into segments (e.g. parts, phrases, bars, etc.) (see Section 3.1 for details). Then, the similarity between each segment of the target piece and all sample segments is evaluated, and similar sample segments are obtained as the *example data set* for the target piece.

In the analysis stage, our method compares the record of the performance sequence and the musical score for each example segment, and analyzes variances in tempo, volume, and so on. Variance in musical expression is represented as a curve of the relative diversity, called the *musical expression pattern (MEP)* (see Section 3.3 for details). Patterns for all example data segments are stored in the *MEP set*

In the evaluation stage, our method calculates a significance score for each example segment. This score indicates how useful the example data is expected to be in the generation of musical expression for the target piece. It is determined principally from similarity in musical score and performance conditions.

As a result of the analysis and evaluation stages, an MEP set with significance scores is obtained. In the composition stage, these MEPs are integrated into a musical expression value for the whole target piece (see Section 3.3 for details). The first step of this stage is the calculation of the MEPs for each segment of the target piece. This is achieved through the average of example MEPs for that segment. The average is weighted by the significance of each MEP. The second step is the integration of segments MEPs. In this step, averaged MEPs for each target piece segment are unified as the integrated MEP.

Finally, in the application stage, our method applies the integrated MEP to the musical score of the target piece, and outputs the resultant performance data.

3 Component technologies

3.1 Segmentation of musical pieces

Generally speaking, one possible serious problem faced by the case-based method is shortfalls in the example data set. Our methods extracts available example segments from the example data set, analyzes them, and applies them to the target piece. Thus, if the size of available example data is insufficient, the proceeding processes will not function satisfactorily.

Arcos et al. used single notes as their segment granularity, and introduced cognitive musical structure to relate neighboring notes. This is based on Narmour's implication/realization model ([Narmour, 1990]), and Lerdahl and Jackendoff's generative theory of tonal music ([Lerdahl and Jackendoff, 1983]). This is a good way to avoid shortfalls in the example data set. However, such an approach is insufficient to generate musical expression variance over longer stretches of music. Therefore, as mentioned above, our method extracts sequence of notes instead of single notes as the example data, and does not rely on the cognitive musical structure. It is obvious that the cognitive structure has a good effect on the generation of musical expression. However, since there may be individual differences between some of these structures, it is undesirable to rely solely on this knowledge type. Moreover, we think that the cognitive structure can equally be acquired with a case-based method similar to that proposed here. So, in this research, we chose the more challenging path, that is the generation without cognitive musical structure.

In our method, the most desirable example type is performance data on the target piece. However, it is unrealistic to expect that such examples can be obtained, and close-fitting examples for all portions of the target piece are also rarely found. In other words, it is likely that enough examples could not be found simply by querying a piece which is similar throughout.

To avoid this problem, as briefly mentioned above (cf. Section 2.1), we divide the target piece into segments, and extract an example data set for each segment.

So as to extract examples extensively for all parts of the target piece, queries should be made at various granularities of division. Ideally, all possible methods of division should be tried. However, the number of plausible segment lengths reaches exponential order on the number of notes which appear in the target piece, making such exhaustive computation unrealistic from the viewpoint of computational cost. Hence, our method uses a number of consistent approaches to division, which are based on the musical structure described in the musical score.

Most music pieces have a hierarchical musical structure consisting of musically meaningful segments (e.g. motives, repetitions, phrases, bars, etc.). The musical structure mentioned in this paper is not cognitive, but a combination of parts which constitute musical pieces. This structure consists of multiple layers of variably sized segmentation units. The segmentation unit at the bottorn layer is the smallest sized segments, i.e. the single note. The segmentation unit at the next layer up is one size larger, which is usually a beat. Still higher layers consist of much larger segments, such as a half bar, bar, phrase, sequence of phrases, repetition, motive, and so on. The top layer is composed of the whole piece. The segmentation of a musical piece is described in the musical score to some extent, and likely to be unaffected by factors other than musical score information. In dividing the target piece into segments, the possibility of finding appropriate examples increases.

3.2 Performance conditions

This section explains performance conditions and the associated method of comparison.

Performance conditions are described as a set of *features.* Each feature is made up of *key* and *value.* The key indicates the particular feature in the form of a keyword. The value is the degree of the feature, and given as a real number in the range -1 to 1. For instance, "an elegant and bright performance" has two features. One feature has the key "elegant", and the other has the key "bright". The value of each feature is between 0 and 1. In the case of "elegant and very bright performance", the value of the feature "bright" is close to 1. In contrast, in the case of "somewhat bright performance", the value of the feature "bright" is close to 0. In the case of "non-bright performance", the feature "bright" has a negative value. If the feature "bright" is not given, it is considered that this performance implicitly has the feature "bright" with value 0.

Not only the information on the feel to the performance but also information on the performer and performance style are also described in this form. For example, performance data from musician "A" has feature "performer A". The value of this feature is 1. In the case of musician A imitating musician B, the performance conditions consist of feature "performer A" and "performer B", with values slightly closer to 0 than in the previous case.

Now, considering the key and value of each feature as unit vector and the norm of that vector, respectively, performance conditions are the sum of vectors on a vector space which covers each unit vector key. This summation of the vectors is named the *performance condition vector.* Equation 1 shows performance condition vector *v*

$$v = \sum_{i \in V} a_i \cdot i \tag{1}$$

where *V* is the set of keys of features which constitute the performance conditions, and *ai* is the value of key t.

By introducing the concept of the performance condition vector, similarity in performance conditions can be evaluated through the distance between the performance conditions vectors. Equation 2 shows the resemblance value of performance condition vectors *v* and u. The numerator is the dot product of the performance vectors. The denominator is the square of the length of the larger vector, hence normalizing the degree of resemblance.

$$\text{Resemblance}(v, u) = \frac{v \cdot u}{\max\{|v|^2, |u|^2\}} \tag{2}$$

3.3 Musical expression pattern

This method uses *musical expression patterns (MEPs)* in the generation process. This section describes analysis and composition of MEPs.

Analysis of MEPs

This method uses the ratio of "the musical expression value (tempo, volume, and so on) of the target example segment" to "the average value of the next segment up (parent segment)" as a representation of variance in musical expression. The MEP of an example segment is the set of the ratios for each type of musical expression (tempo, volume, etc.). Equation 3 shows this calculation, $P_{exp}(s)$ is the MEP of musical expression type *exp* (seconds/crotchet (see below), volume, etc.) for a segment *s, Si j* is a segment of the target piece, t is the depth of the segmentation layer (c.f Section 3.1), *j* and *k* are segment indices within the given segmentation layer, *Si,j* is a sub-phrase of Si-1,k,and *exp(s)* is the musical expression value of segment *s.* The average MEP over all segments composing one segment size up is always 1.

$$P_{exp}(s_{i,j}) = \frac{exp(s_{i,j})}{exp(s_{i-1,k})} \tag{3}$$
$$s_{i,j} \subset s_{i-1,k}$$

The following example shows the calculation of MEP for a performance data segment of a 4 bar phrase (Figure 4). This performance data is played at an average tempo of 120 (0.5 seconds/crotchet). In the case of human performance, the tempo varies with musical expression, so that the tempo of most notes in the phrase will be other than 120. In this example, the average tempo of each bar is, respectively, 115 (0.52 seconds/crotchet), 133 (0.45 seconds/crotchet), 150 (0.4 seconds/crotchet), and 95 (0.63 seconds/crotchet). (Note that the average of these tempos will not be 120, since the average tempo is the reciprocal of total performance time.) As mentioned above, MEP is the ratio of the expression value of the target segment to the value of the next segment up. In this case, target segments are made up of each bar, and the parent will generally be the whole phrase (4 bars) or a half phrase (2 bars). The parent is decided in accordance with the segmentation strategy. In the case of tempo, the MEP is calculated from the seconds/crotchet value, instead of the tempo value, since the tempo value is inconsistent in some calculations (e.g. the mean of tempo values and the average tempo value usually differ).

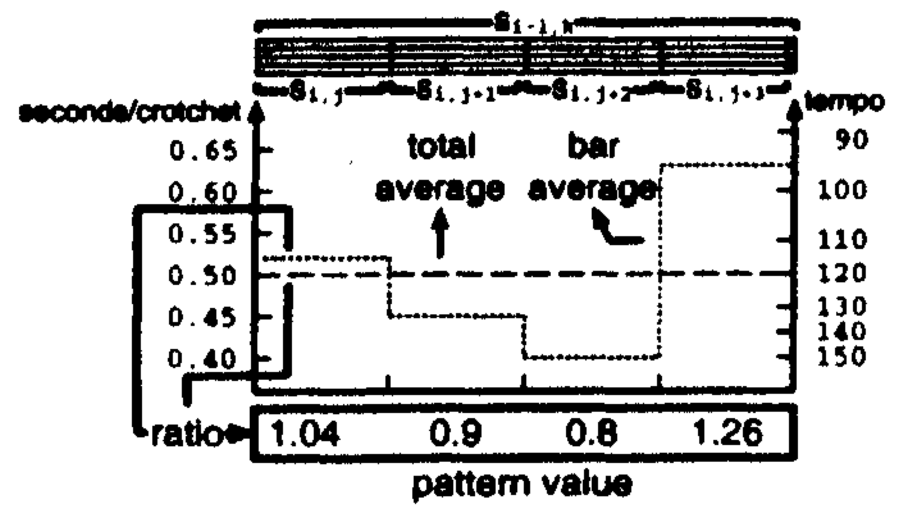

Figure 4: An example of MEP calculation for tempo

Assuming that the next segment up is the whole phrase, the tempo MEP for each segment (each bar) is the ratio of the seconds/crotchet tempo of each bar (0.52, 0.45, ...) to the seconds/crotchet tempo of the whole phrase (0.5). In this way, the MEP for the bars are 1.04, 0.9, 0.8, and 1.26, respectively.

MEP composition

In the composition stage, these MEPs are integrated into a single MEP for the whole target piece. As mentioned in Section 2.1, the composition stage consists of two steps. The first step is the calculation of the MEP for each segment of the target piece. The MEP of each segment of the target piece is the weighted mean of MEPs of all examples for that. Equation 4 is a formalization of this process. In this equation, *Si,j* refers to a segment of target piece, *Ei,j* is the overall example data set for segment si,j and *W(s)* is the weight of example segment s, which is calculated from the significance of each segment.

$$P_{exp}(s_{i,j}) = \frac{\sum_{s \in E_{i,j}} W(s) \cdot P_{exp}(s)}{\sum_{s \in E_{i,j}} W(s)} \tag{4}$$

The second step is the integration of the individual MEPs. In this step, for each note of the target piece, the MEPs of all ancestral segments are multiplied. An ancestral segment of a note is any segment which contains that note. Equation 5 shows the integrated MEP for the mth note n_m. *Si* is the set of segments in tth layer, and *n* is number of layers, where the segmentation unit of the nth layer is a single note (i.e. sn,m = n_m).

$$P_{exp}(n_m) = \prod_{1 < i \leq n} P_{exp}(s_{i,j}) \quad \{j | n_m \in s_{i,j}\} \tag{5}$$

Figure 5 shows a simple example of this calculation. The MEP for a half bar segment is the ratio of the value of the half bar to the value of full length containing bar, and the MEP for a bar segment is the ratio of the bar value to the whole 4 bar phrase value. Thus, the integrated MEP indicates the ratio of the value of each note to the value of the whole phrase.

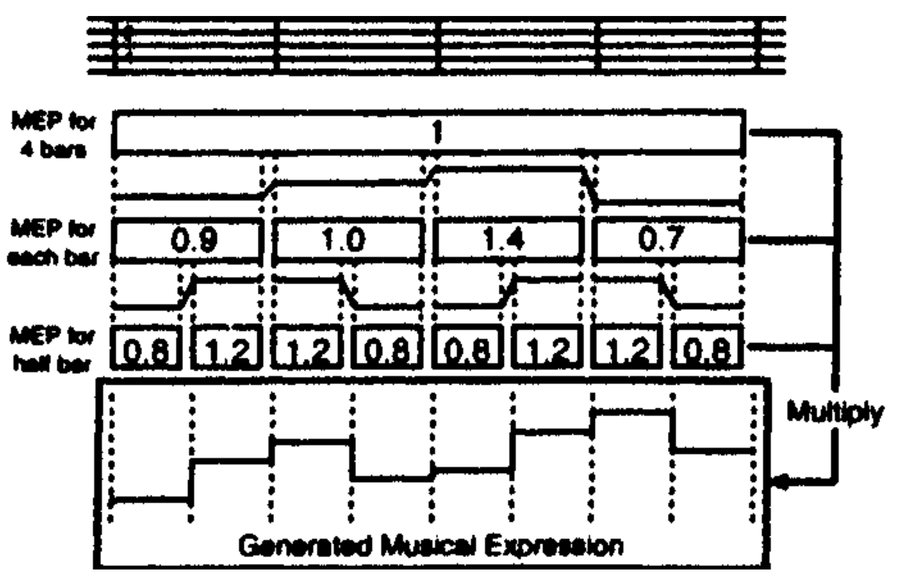

Figure 5: An example of MEP generation for a 4 bar phrase

4 Musical expression generation system

4.1 Outline

We have been developing a musical expression generation system called *Kagurame,* which uses the case-based method described above. Kagurame Phase-I, the first stage of Kagurame, is intended to estimate the system capability and possibilities of our method. For the sake of simplicity, the types of musical pieces and performance conditions the system can handle have been limited. For example, the target piece and sample data are limited to single note sequences.

4.2 Architecture

Figure 6 shows the architecture of Kagurame Phase~-I. The following section describes the basic mechanism and algorithm for each component.

Input

As input, this system uses: 1) musical score information of the target piece, 2) the musical structure of the target piece, and 3) performance condition settings. The musical score information is a sequence of detailed parameters for each note (e.g. position, beat length, key value, etc.). The musical structure is information on segment boundaries, used to divide the target piece into segments (cf. Section 3.1). The performance condition settings are given in the form of a performance condition vector (cf. Section 3.2). This combined information is given in an originally formatted text file.

Performance data set

Each performance data set consists of: 1) musical score information, 2) musical structure, 3) performance conditions, and 4) performance data. The musical score information, musical structure, and performance conditions are given in the same format as described for the system input. The performance data is a sequential record of a human performance. It is given as a standard MIDI format file (SMF). The SMF is a sequence record of note event information, which consists of the time, key value, and strength ("velocity"). This format file is easily obtained from a computer and electronic keyboard. Each data type is divided into segments beforehand for convenience of calculation at the extraction stage.

Extraction of examples

In the extraction process, first of all, the target piece is divided into segments according to the musical structure information. The similarity score between a given target segment and each performance data segment is then

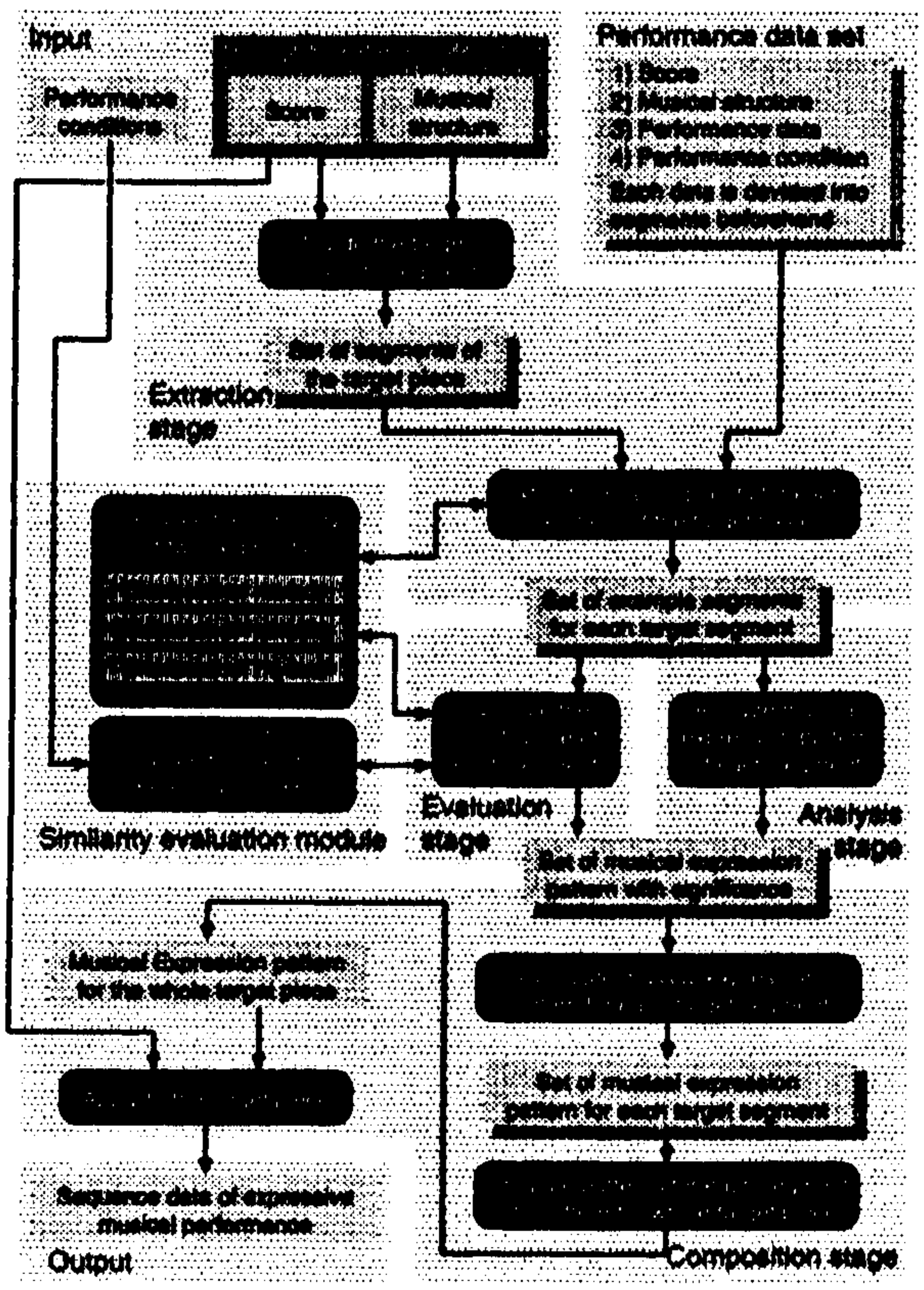
Input
Performance conditions
Score
Musical structure
Performance data set
1) Score
2) Musical structure
3) Performance data
4) Performance condition
Each data is divided into segments beforehand
Set of segments of the target piece
Extraction stage
Set of example segments for each target segment
Similarity evaluation module
Evaluation stage
Analysis stage
Set of musical expression pattern with significance
Musical Expression pattern for the whole target piece
Set of musical expression pattern for each target segment
Sequence data of expressive musical performance
Output
Composition stage

Figure 6: The system architecture of our performance system

calculated, and high scoring segments are used as the example data set for the target segment. This extraction process is carried out for all segments of the target piece.

Evaluation of similarity

The similarity score used at the extraction stage is calculated by the similarity evaluation module. This estimation is based on the resemblance of the characteristics of the concerned segments. The system currently uses three factors as segment characteristics: melody, harmony, and rhythm. Melody is the tendency for fluctuation in the melody. It is calculated as the difference between the average pitch of the first half of the segment and that of the latter half. Equation 6 shows the melody characteristic function $C_m(s)$ for segment s, where N is the set of notes in the first half of the segment, *N1* is the set of notes in the latter half, and *p(n)* is the pitch of note n. The characteristic of harmony is the chord component of the segment. This is a set of 12 values. Each value is a count of given pitch note. Equation 7 shows the ith value of the harmony characteristic function $C_{h,i}(s)$. The characteristics of rhythm is the beat length of the segment.

$$C_m(s) = \frac{\sum_{n \in N_l} p(n)}{|N_l|} - \frac{\sum n \in N_f p(n)}{|N_f|} \tag{6}$$

$$C_{h,i}(s) = \frac{|\{n | n \in N, p(n) \text{ is } i\text{th pitch on the scale}\}|}{|N|} \tag{7}$$

$$D_h(s_1, s_2) = \sum_{1 \le i \le 12} |C_{h,i}(s_1) - C_{h,i}(s_2)| \quad (8)$$

For each factor, the system evaluates the characteristic parameters of target segments, calculates the resemblance of these parameters, and normalizes them. Resemblance of melody is the difference between $C_m(s)$ for two segments. Resemblance of harmony is the summation of the difference of $C_{h,i}(s)$ for each i (Equation 8). Resemblance of rhythm is the ratio of beat length. If this value calculates to less than 1, the inverse is used. The summation of these resemblances is used as the similarity between segments.

Analysis of MEPs
Then, this system analyzes the MEP of each example segment. Details of this process are given in Section 3.3.

Evaluation of significance
The significance of each example segment is the product of similarity with the target segment, similarity with neighbor segments, similarity with ancestral segments, and resemblance of performance conditions. The similarity of target segments is calculated in the same way as for the extraction process, and likewise for the similarity of neighbor or ancestral segments. Resemblance of performance conditions is the dot product of the performance condition vectors in question (cf. Section 3.2).

Composition of musical expression
The application process consists of: 1) calculation of MEP for the each segment of the target piece, 2) integration of segment MEPs for the whole target piece, and 3) generation of expressive performance data for the target piece. Details of the calculation and integration process are given in Section 3.3. The weight for the calculation of MEP of each segment *(W(s)* in Equation 4) is an exponential function on the significance of that segment. As a result of this process, the integrated MEPs for the overall target piece are generated.

In the generation process, the system multiplies the integrated MEPs by the average for each musical expression over the whole piece. For example, in the case of tempo, the average seconds/crotchet value for the piece is multiplied in its entirety with each integrated MEP. The overall average value is based on example data for segments of the overall piece and notation on the musical score of target piece. All types of musical expression are generated in same way. Finally, the system applies the overall musical expression to each note of the target piece, and outputs the resultant performance data as an SMF file.

Handling of musical expression
Kagurame Phase I handles three types of musical expression: local tempo, duration, and volume. Local tempo is the tempo of each note. Duration is the ratio of the time from note on until note off to the given length of the note. Duration of 1 means the note is played for its full length, (there is no pause or overlap). In the case of staccato, the duration will be close to 0, and in the case of legato, it will exceed 1. Volume is a measure of the strength of sound. These parameters are easily accessible from the SMF file.

5 Evaluation

We generated some expressive performance data with Kagurame Phase-I, and evaluated the resultant performance. This section describes the experiments and evaluation of the performance generated by Kagurame Phase-I.

5.1 Experiments

A relatively homogeneous set of 21 short etudes from Czerny's "160 Kurze Ubungen" and "125 Passageniibungen" were used for the experiment. Performance data was prepared for each piece. All performance data was derived from a performance by an educated human musician, and each piece was played in two different styles: 1) Romantic style and 2) Classical style. The performance conditions for each piece has the single feature of "Romantic" or "Classical" with a value of 1.

Out of the 21 pieces, one piece was selected as test data, and performance data for all the remaining pieces (20 pieces) was used as the performance data set. As such, the human performance data for the test piece was not included in the sample data set (i.e. evaluation is open). Two styles (those described above) of performance data were generated for the test piece by Kagurame Phase~I based on the performance data set. The test piece was varied iteratively (similar to cross-validation), and performance data was generated for all the pieces. All generated SMF data was played with a YAMAHA Clavinova CLP 760 and recorded on an audio tape for the listening experiments.

5.2 Evaluation of performance results

We evaluated the resultant performance through a listening test and numerical comparison. In the listening test, the resultant performances were presented to several human musicians for their comments. Some of them were players of sample data. In the numerical comparison, the difference between human performance and the generated performance was calculated, and rating was also made of the difference between performance data for the two styles.

The following are comments from the listeners. From the viewpoint of satisfaction of performance, the resultant performances sounded almost human-like, and musical expression was acceptable. There were some overly weak strokes caused by misplay in the sample data, but these misplays were not obvious in the resultant performance. It is hard to determine which performance (human or system) is better, since it relies heavily on the listener's taste. But, if forced to say one way or the other, human performance was better than the system one.

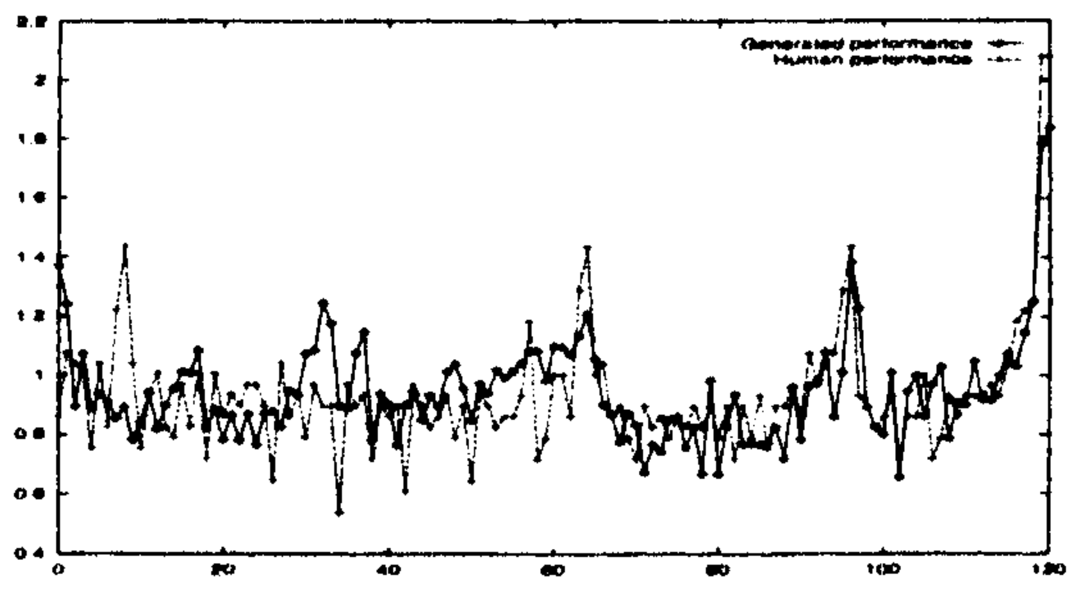

Figure 7: The tempo curve of the system and human performance of "No. 1, 160 Kurze Ubungen"

Human listeners pointed out that the curve of the generated performance tended to be similar to that of the human performance particularly at characteristic points. (e.g. the end of each piece). Numerical comparison between the human performance and generated performance also showed that fluctuations in musical expression for the system performance resembled human performance in many respects. Figure 7 shows the comparative tempo curves for the generated performance and human performance of "No. 1, 160 Kurze Ubungen" in the "Romantic" style (Of course, this is not the best resultant data but an average case). In this graph, it observable that the peaks of the graph coincide (e.g. around 65, 100, the ending, and so on). In some portions, however, differences in the curve behavior are noticeable. Human listeners judged some of these differences to be permissible and not critical errors. They seem to represent variance of musical expression within the same style.

The difference between the generated performance for the two styles was clear in each case. In the listening test, very high percentages of correct answers were obtained when listeners were asked to identify the performance style of the piece. Figure 8 shows the tempo curve of the "Romantic" and "Classical" styles for the generated performance. The target piece is "No. 1, 160 Kurze Ubungen". This graph also evidences differences in the generated tempo curve. The range of fluctuation for the "Romantic" style is much broader than the "Classical" style. Since a broad range of rubato is known as a typical characteristic of the "Romantic" style, the broader fluctuation seen for the "Romantic" performance seems to be appropriate. Based on this result, at least these two styles were discriminated in performance.

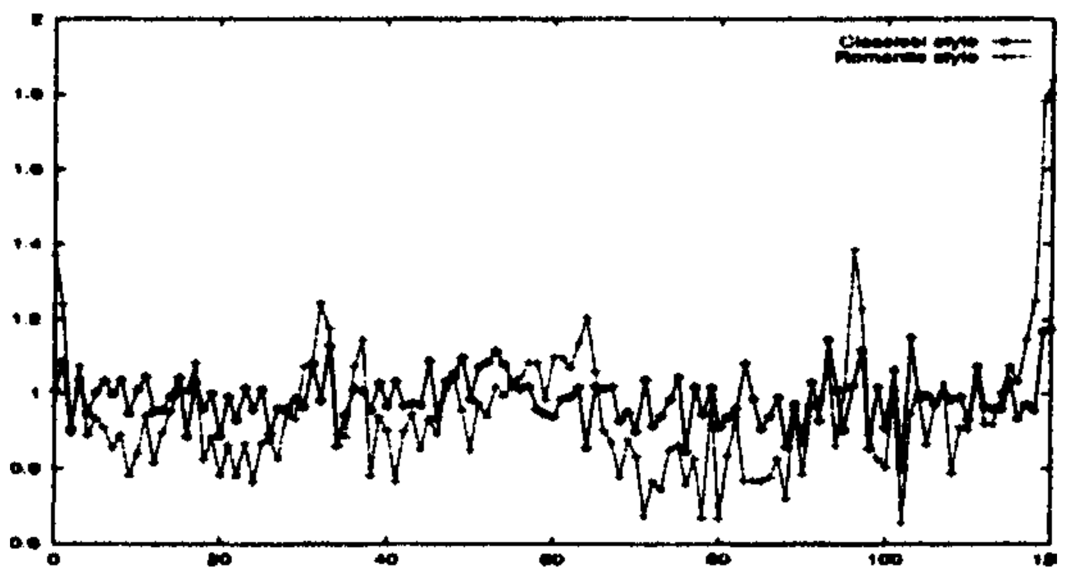

Figure 8: The tempo curve of the "Romantic" and "Gassical" style performances of "No. 1, 160 Kurze Ubungen"

6 Conclusion

This paper described a case-based method for the generation of musical expression, and detailed a music performance system based on the case-based method proposed in this paper. The advantage of the proposed method is that it can model performance conditions during the generation process. This makes it easy to generate various kinds of musical expression for a single piece of music in accordance with the performance condition settings.

According to a listening test, the resultant performance of the described system was judged to be almost human-like and acceptable as a naturally expressed performance. Particularly, at characteristic points of the target piece, musical expression tended to be remarkably similar to human performance. By testing different styles of system performance, it was proved that our system can generate different musical expression for a given piece of music. Moreover, most of the generated musical expression was judged to be appropriate for the given style.

As a result of these experiments on the system, the case-based method presented in this paper can be seen to be useful for the generation of expressive performance. It was also confirmed that this method can generate varying musical expression for a single piece of music through changing the performance condition settings.

References

[Arcos *et al.,* 1997] J. L. Arcos, R. L. de Mantaras, and X. Serra. SaxEx : a case-based reasoning system for generating expressive musical performances. In *Proceedings of the 1997 International Computer Music Conference,* pages 329-336. International Computer Music Association, 1997.

[Bresin *et al.,* 1992] R. Bresin, G. De Poli, and A. Vidolin. Symbolic and sub-symbolic rules system for real time score performance. In *Proceedings of the 1992 International Computer Music Conference,* pages 211-214. International Computer Music Association, 1992.

[Canazza *et al.,* 1997] S. Canazza, G. De Poli, A. Roda, and A, Vidolin. Analysis by synthesis of the expressive intentions in musical performance. In *Proceedings of the 1997 International Computer Music Conference,* pages 113-120. International Computer Music Association, 1997.

[Chafe, 1997] C. Chafe. Statistical pattern recognition for prediction of solo piano performance. In *Proceedings of the 1997 International Computer Music Conference,* pages 145-148. International Computer Music Association, 1997.

[Friberg and Sundberg, 1986] A. Friberg and J. Sundberg. A lisp environment for creating and applying rules for musical performance. In *Proceedings of the 1986 International Computer Music Conference,* pages 1-3. International Computer Music Association, 1986.

[Friberg, 1991] A. Friberg. Generative rules for music performance: a formal description of a rule system. In *Computer Music Journal,* volume 15, number 2, pages 56-71. MIT Press, 1991.

[Fryden and Sundberg, 1984] L. Fryden and J. Sundberg. Performance rules for melodies, origin, functions, purposes. In *Proceedings of the 1984 International Computer Music Conference,* pages 221-224. International Computer Music Association, 1984.

[Lerdahl and Jackendoff, 1983] F. Lerdahl and R. Jackendoff. *A Generative Theory of Tonal Music.* MIT Press, 1983.

[Narmour, 1990] E. Narmour. *Analysts and Cognition of Basic Melodic Structures.* The University of Chicago Press, 1990.

[Noike *et al.,* 1992] K. Noike, N. Takiguchi, T. Nose, Y. Kotani, and H. Nisimura. Automatic generation of expressive performance by using music structures. In *Proceedings of the 1992 International Computer Music Conference,* pages 211-214. International Computer Music Association, 1992.

[Widmer, 1993a] G. Widmer. The synergy of music theory and AI: Learning multi-level expressive interpretation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence,* pages 114-119. American Association for Artificial Intelligence, 1993.

[Widmer, 1993b] G. Widmer. Understanding and learning musical expression. In *Proceedings of the 1993 International Computer Music Conference,* pages 268-275. International Computer Music Association, 1993.

[Widmer, 1995] G. Widmer. Modeling the rational basis of musical expression. In *Computer Music Journal,* volume 19, number 2, pages 76-96. MIT Press, 1995.