Ha-Hi-Hun: Performance Rendering System of High Controllability

Keiji Hirata
NTT Communication Science Laboratories
email: hirata@brl.ntt.co.jp

Rumi Hiraga
Bunkyo University
email: rhiraga@shonan.bunkyo.ac.jp

Abstract

This paper presents the design principle and algorithm of a new performance rendering framework, and a prototype system based on the framework. We believe that a practical performance rendering system should be able to refine and improve a generated performance interactively, incrementally, and locally through direct instructions issued by a user. In addition, the generated performance must reflect the user's intention properly. For these purposes, we propose a new framework called two-stage performance rendering. The first stage translates a user's instruction into the deviations of the onset time, duration, and amplitude of structurally important notes. The second stage spreads the deviations over surrounding notes. Lastly, we introduce a performance rendering system called "Ha-Hi-Hun" that we are developing.

1 Introduction

The authors believe that a practical performance rendering (PR) system should have a high level of controllability. Consider the following scenario of a piano lesson. A tutor gives a student direct instructions, such as "more passionate, here", "even brighter for this phrase" or "play like me", whenever he/she thinks it best to refine the student's performance. Ideally, the student follows the instructions and changes his/her performance accordingly. The tutor, after having listened to the refined performance, may then issue other instructions. Observations of this scenario have given the authors the idea that a user and a PR system should work as in the piano lesson, where the user becomes the tutor and the PR system corresponds to the student. That is, it should be possible to refine and improve a generated performance interactively, incrementally and locally through direct instructions given by a user. In addition, the performance should sound natural throughout when heard. Note that a PR system discussed in this paper is for piano.

To achieve a high level of controllability, a PR system should be able to properly interpret the user's instructions. If instructions are given in a natural language, they are usually subjective, equivocal, and even time-varying. The system should be customizable or personalizable and be context-sensitive. A method where a user gives some sample performances instead of a natural language would be suitable for these purposes.

The system must also be able to synthesize a natural performance that reflects the user's instructions. Let us suppose a case in which a user gives the system an instruction to play a note Q louder in a particular part of a piece. If the system naively increases only the amplitude of Q, the generated performance may become unnatural because of a musical imbalance. Considering the role of Q in the piece, the surrounding notes should also be played either louder or softer and even their agogics may have to be adjusted according to the amplitude change of Q. Thus, to keep a generated performance natural, a PR system must maintain a certain musical consistency, which is represented in the form of the constraints regarding the agogics and dynamics for Q and the surrounding notes.

This paper proposes a new framework called *two-stage performance rendering* that gives the system the abilities to properly interpret the user's instructions and synthesize a natural performance based on them.

This paper is organized as follows. In Section 2, issues of conventional PR methods are discussed. Then, a method of representing polyphony and expression is introduced in Section 3, propose two-stage PR framework followd by two examples in Section 4, and describe a prototype system in Section 5. Last, we put concluding remarks and future work in Section 7.

2 Conventional Systems and Problems

Given a score, a conventional PR system calculates the agogics and dynamics of all notes in the score all at once using rules, mathematical expressions and/or cases (which we call performance knowledge), extracted from real sample performances beforehand or taken from research results in musicology (Bresin 2001; Widmer 2001; Arcos, de Mántaras, and Serra 1997; Suzuki, Tokunaga, and Tanaka 1999; Friberg 1991).

The conventional PR systems have two problems.

First, they can not easily change performance locally. As a result, they can not produce a cycle that includes the feedback of listening to the output and modification of a certain parts of the output that a user does not prefer. The relationships between the sample performances for extracting performance knowledge beforehand and the generated output is unclear. That is, a user can hardly know which sample performances should be used to achieve a desired output; it is almost impossible to select appropriate sample performances that will modify only a relevant part and leave the others unchanged.

Second, conventional PR systems can not interpret the user's subjective and qualitative instructions in the manner that the student in the piano lesson does. Usually, instructions issued by a tutor including expression marks[1], expression words[2], or only sample performances themselves, are interpreted differently student by student. Moreover, they contain more or less quantitative expression words, such as brighter, even brighter, and even even brighter. Conventional PR systems can not interpret such instructions properly. Hence, it is almost impossible to build a universal body of performance knowledge to generate a desired output.

3 Representing Music Expression

We think that the generative theory of tonal music (GTTM) (Lerdahl and Jackendoff 1983) is the most promising music theory in terms of computer implementation and the deductive object-oriented database (DOOD [3]) (Yokota 1992; Kifer, Lausen, and Wu 1995) is a knowledge representation method with a theoretical foundation and is thus tractable.

3.1 DOOD

The DOOD framework is motivated by the introspection that things in the real world can be represented as the combination of a basic (atomic) object and a set of its attributes. Hereafter, we identify an object term with an object itself. We write an object term as $o(\cdots, l : v, \cdots)$, where o is an atomic symbol that stands for a basic object, $l : v$ is an attribute, l an attribute name (label), and v an attribute value.

The most fundamental relation in the real world is the "is_a" relation, and it is modeled as the subsumption relation defined by the deductive rule in the DOOD framework. That is, the subsumption relation (written as $\sqsubseteq$) represents the relation "a more informative object $\sqsubseteq$ a less informative object". In other words, it represents "an instantiated object $\sqsubseteq$ an abstract object" or "a special object $\sqsubseteq$ a general object".

This definition of the subsumption relation $o_1 \sqsubseteq o_2$ means that if for all attributes of o_2, the values of these attributes of o_1 are more instantiated, then o_1 is considered more instantiated than o_2 (or o_2 is more abstract than o_1). Thus, an object that has less attributes is more abstracted. For example, an object for a note whose pitch is C in the fifth octave is more instantiated than an object for a note C. This relationship is presented as *note*(*pitch* = C, *octave* = 5) $\sqsubseteq$ *note*(*pitch* = C)

Definition of Least Upper Bound (lub): Suppose there are two objects x and y. Operation $lub(x, y)$ is defined as $min(\{z|x \sqsubseteq z \wedge y \sqsubseteq z\})$ as in the standard way. Intuitively, $lub(x, y)$ extracts the largest common part of x and y; it is accompanied by the image of conjunction.

3.2 Representing Polyphony by DOOD

Consider a score for piano pieces. The score is likely characterized by polyphony, which is texture formed by the interweaving of several melodic lines that are independent but sound together harmonically. We design an object term for representing polyphony and expression based on DOOD and time-span reduction in GTTM(Hirata and Aoyagi 2002). Since the subsumption relation of DOOD is generally used for representing the instantiation-abstraction relation (the is_a relation), we assert that the counterpart of the subsumption relation in GTTM is the time-span reduction. We think that this correspondence is the most natural.

Time-Span Tree of Polyphony Time-span reduction represents the intuitive idea that if we remove grace notes or less important notes from a long melody, we obtain a simple melody that sounds similar. The operation of removing notes is called reduction. An entire piece of music can eventually be reduced to a key note or a tonic triad. The application of reduction to a melody is depicted by a time-span tree (the upper half of Fig.1). In the figure, $e_1 \sim e_6$ are events. A time-span tree is a binary tree. We refer to an important branch as primary and the other as secondary. The time span covered by a primary and secondary branches is represented by a single note, called a salient note or a head. Then, the following object term represents the subtree of the branch with $\bigcirc$ in the figure:

[1]The marks that indicate articulation and dynamics, such as accent, slur, staccart, *f*, *p*, $<$ and $>$.

[2]The words that describe expressions, such as *appassionata*, *dolce*, and *espressive*.

[3]DOOD was originally the name of an international conference and a research field. Here, it only refers to the name of a knowledge representation method.

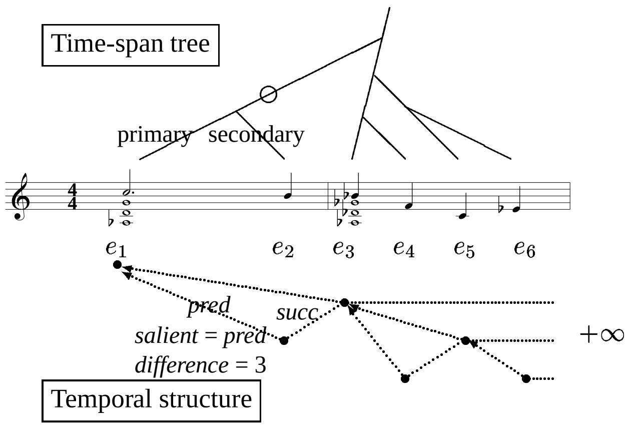

Figure 1: Time-span tree and temporal structure

ts(*head* = e_1,
at = t_1,
primary = *ts*(*head* = e_1,
at = t_1),
secondary = *ts*(*head* = e_2,
at = t_2))

e_1 = *chord*(*notes* = {56,62,67,72},
duration = 360,
velocity = 60)

Here, the *head* attribute describes a salient note. The value of the *notes* attribute is the set of MIDI note numbers. The *duration* attribute is calibrated with one measure being 480 ticks long. Object term t_n ($n = 1 \sim 6$) represents the onset time of event e_n, which is explained next.

Temporal Structure of Polyphony The lower half of Fig. 1 represents the temporal structure of the melody, which is newly proposed for analyzing and synthesizing note positions and durations.

First, suppose that the onset time of e_1, t_1, is the origin point. Next, in terms of e_3, we can see that e_3 occurs between e_1 and $+\infty$ (the ordering information) and on the fourth beat from e_1 (an absolute timing of onset). The following *temp* object represent the onset time of e_2.

temp(*pred* = t_1,
succ = t_3,
salient = *pred*,
difference = 3)

Event e_2 occurs between e_1 (*pred*) and e_3 (*succ*), and e_1 is more important for e_2 (*salient*= *pred*, represented by ◂┈┈ in the figure), and e_2 occurs on the third beat from e_1 (*difference* = 3).

Note that the values of the *head* attribute and the *salient* attribtes should be set consistently to each other. For the detail of the time-span tree and the temporal structure, please refer to Hirata and Aoyagi (2002).

3.3 Representing Expression

We call the shifts of onset timings and volumes for expressing agogics and dynamics as deviations generically. The deviations are represented by *chord* and *dev* objects as follows:

e_1 = *chord*(*notes* = {56,62,67,72},
duration = 360,
velocity = 60,
deviation = d_1) $\Leftarrow$

d_1 = *dev*(*onset* = 0.0,
duration = 1.0,
velocity = 1.0)

Compared to the *chord* object in the previous subsection, the *deviation* attribute is newly attached to this *chord* object (at the line of $\Leftarrow$), and the *dev* object itself represents the deviation of relevant notes. There are two ways for describing the shift of onset time: the ratio to the duration of a relevant note or the ratio to the duration of one beat. Our prototype system mentioned in Section 5 can adopt either way as a user likes. Similarly, the shifts of duration and velocity (i.e. volume) are described in ratios.

Since we newly add the *deviation* attribute to every salient note, we can describe the deviation at every layer of a time-span tree.

4 Two-Stage Performance Rendering

We examined the scenario of a piano lesson where a tutor gives a student many instructions in Section 1. Here, let us assume that the tutor's instructions for expression refer to salient notes. That is, when a tutor says "play this note carefully", this note means a salient note within a certain time range. The two-stage PR framework is motivated by this assumption (Hirata and Hiraga 2000).

4.1 Architecture

In Fig. 2, the first stage translates a user's instruction into the agogics and dynamics of structurally important notes in a range and the second stage adjusts the surrounding notes. Here, a structurally important note means a salient note in the context of the time-span reduction of GTTM.

The inputs for two-stage PR are a score to be performed, instructions given by the user, and sample performances for extracting performance knowledge, which may be substituted by built-in rules or mathematical expressions derived from musicology. The user's instructions specify an operation and the range of the score to which the operation is applied. The operations include faster, brighter, more passionate, or "imitate this sample performance". The output is an ex-

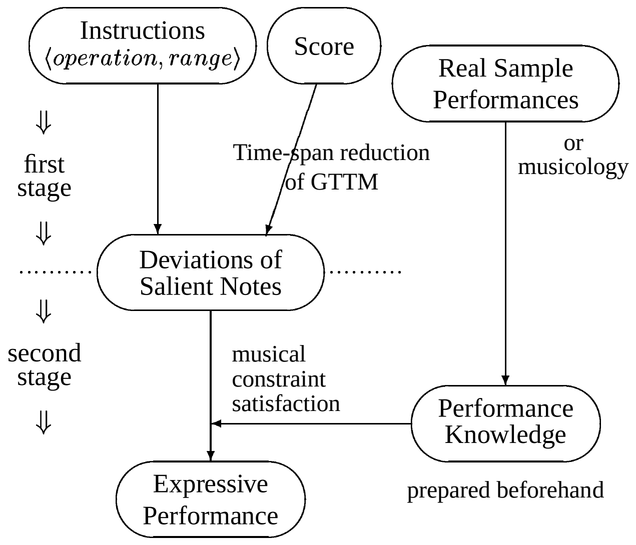

Figure 2: Two-stage performance rendering

pressive performance of the score with the instructions issued.

The first stage maps the user's subjective instructions in a natural language or with a sample performance itself to the deviations of onset time, duration, and amplitude (velocity) for every salient note. (Fig. 3). In the figure, the small circle stands for the *head* attribute, the big circle the *ts* object, p within the box the *primary* attribute, and s the *secondary* attribute. The *deviation* attribute is written outside of the big circle.

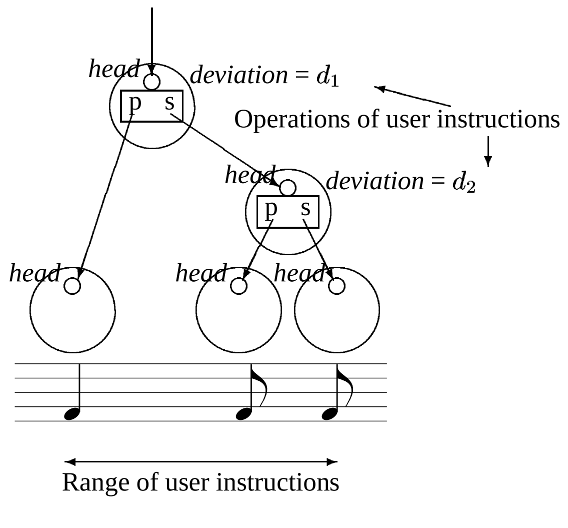

Figure 3: Setting of deviations to salient notes

At the first stage, there are several ways for calculating the values of deviations d_1 and d_2 from user's instructions. Since the values are derived from heuristic values provided a priori, case-based reasoning, and/or learning on a user behavior and the operation environment, they highly depend on the user's subjectivity and context in general.

On the other hand, notes included in a score are grouped hierarchically according to the time-span reduction. Then, the reduction identifies salient notes in groups at every level. The range of an instruction is mapped to the time span of a group.

The second stage propagates the deviations set up to salient notes at the first stage to their surrounding notes (Fig. 4). At that time, the deviations of salient notes are

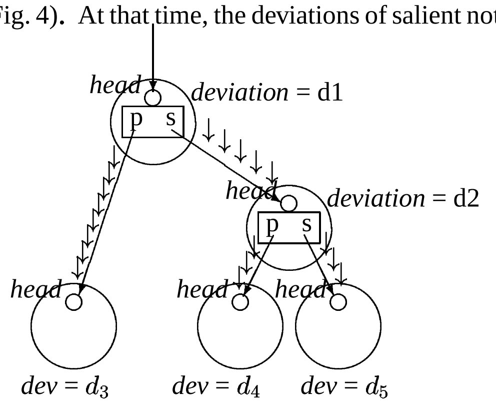

Figure 4: Musical constraint satisfaction

unchanged, and only agogics and dynamics of the surrounding notes are adjusted. This stage is introduced to bring about a musically natural performance. The performance knowledge for the propagation should be obtained in advance, and may be acquired from the analysis of sample performances with some musical theories, such as GTTM and the I-R model (Narmour 1990).

The arrows $\downarrow$ in the figure show the propagation of deviations d_1 and d_2 to the lower *ts* objects. The values of d_1 and d_2 are unchanged, whereas those of d_3, d_4, and d_5 are changed. The performance knowledge used here can be considered a constraint regarding the agogics and dynamics for salient notes and their surrounding ones.

4.2 Sample 1: Vivace

Fig. 5 depicts how the system generates the output when a user issues a sample instruction ⟨Vivace, bars 1 to 2⟩ on a score. The score has been analyzed based on time-span reduction beforehand; it contains three groups, α and β at a lower layer and γ at a higher layer, and α is subsidiary and β is primary. The salient notes (chord) of α are **p**, and those of β are **r**. Note that the results of the analysis greatly depend on a user's intention and interpretation of the score and are not unique.

In Figs. 5, (a), (b), and (c) represent the changes of onset time, duration, and amplitude of every note/chord included in the score fragment in the style of a piano roll (except for pitch information). The horizontal axis represents time, and the amplitude of a note is represented by the thickness of a corresponding line segment. In the figure, (a) shows a mechanical (literal) performance that follows the score exactly; (b) shows

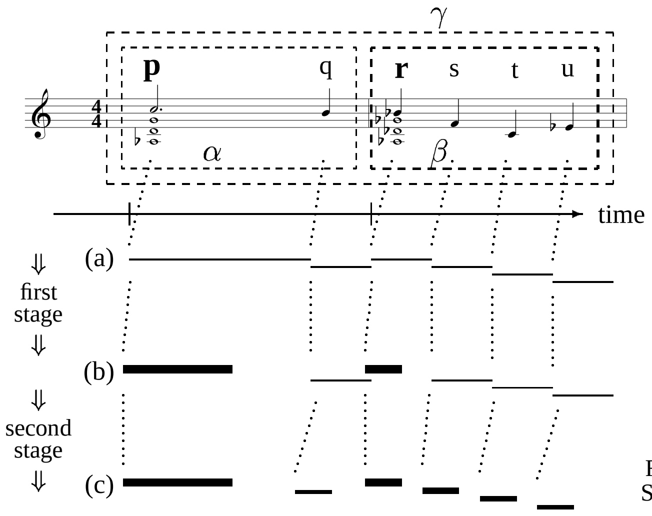

Figure 5: Sample 1: Vivace

the situation after the first stage is finished and the deviations of only salient chords of α and β are calculated; and (c) shows the situation after the entire calculation is finished and the deviations of salient chords are propagated to the surrounding notes.

In (b), since the instruction is vivace, the onset times of **p** and **r** are shifted forward by 10%, and their durations are shorten by 40%. The values of 10% and 40% are not always chosen at the first-stage mapping; different values may be used for a different user.

At the second stage, the deviations of **p** set in (b) are propagated to note q, and similarly, those of **r** to notes s, t and u. The process of the propagation is regarded as musical constraint satisfaction. Various performance knowledge for the musical constraint satisfaction can actually be used, and our prototype system currently assumes a uniform reduction of the durations of all surrounding notes by 60%. Meanwhile, in terms of onset time and duration, for simplicity, it assumes that the constraint is represented as a linear function (Fig. 6). Consequently, the deviation of onset time for

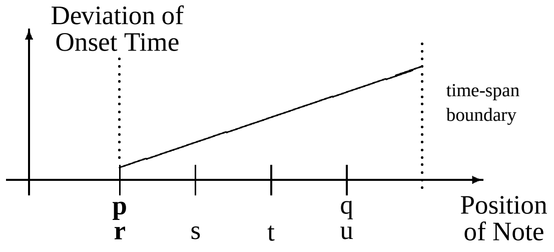

Figure 6: Function for constraint

each note is proportional to the position of the note. Inversely, the amplitudes of notes decrease proportionally to their positions.

Of course, the constraints for performance knowledge are not limited to such a simple function; more complicated expressions or algorithms can be used.

4.3 Sample 2: Graceful

Fig. 7 shows an example where a user gives an instruction ⟨Graceful, bars 1 to 2⟩ on the same score fragment. Here, (a), (b), and (c) represent the same snap-

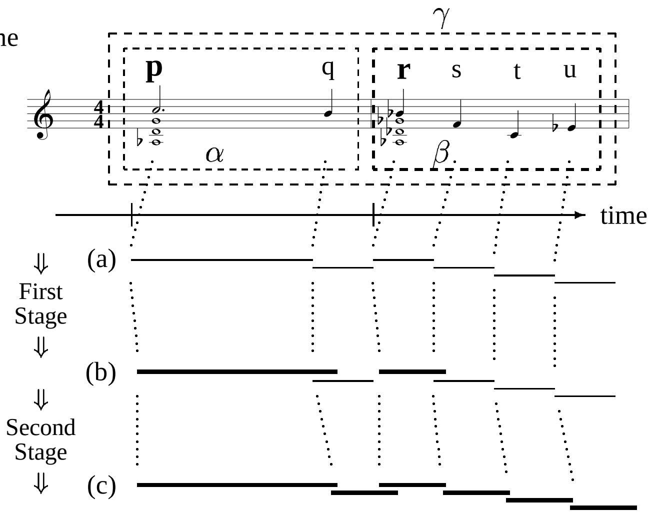

Figure 7: Sample 2: Graceful

shots. In order to realize a graceful performance, the durations of notes get longer throughout, and onset times are slightly shifted backward. These modifications correspond to the style of playing known as legato.

In (c), regarding amplitude, the deviations of **p** and **r** are constantly propagated to the surrounding notes. With respect to onset time, a function similar to the previous sample (Fig. 6) is used.

4.4 Advantages

The two-stage PR greatly owes the structural decomposition of a score to Desain and Honing 1992. However, since the two-stage PR adopts the time-span reduction of GTTM, a salient note becomes available. Thus, it has the following advantages.

- Grouping notes on a score by GTTM and the temporal structure enables a PR system to accept the user's instructions localized to a part of the score.
- The user's subjective knowledge used at the first stage is isolated from the universal musical knowledge used for musical constraint satisfaction at the second stage. While the first stage knowledge allows the system to handle the user's preference and personality flexibly and properly, the second stage one allows it to realize a performance that is musically natural and consistent. Moreover, the isolation of knowledge makes knowledge management easier.

5 Prototype System Ha-Hi-Hun

We implemented a prototype system called “Ha-Hi-Hun” that employs the two-stage PR. The implementation consists of a grouping editor and a PR engine. The system covers the music genre of solo piano tunes. The grouping editor has a simple GUI to manipulate groupings of notes. Besides, since the editor is written in Java and can load and save a score and the associated grouping information in XML, the prototype system is highly portable.

In our current PR engine, during operation, the mapping of the first stage and the constraints of the second stage do not change. Since the structure of a *ts* object is recursive, we can make the granularity of a case vary from a single note to an entire piece. For simplicity, the granularity of a case is fixed to the time span covered by a single measure.

5.1 Copying Deviations based on Lub

As described in Section 4.1, the values of deviations are extracted from user’s instructions at the first stage, and our prototype system obtains these values from a case given as the user’s instruction “play like this performance”, where the performance (case) is represented in a DOOD object term (Section 3). That is, the deviations are extracted from a case to be imitated and are transcripted to salient notes of the polyphony of an input set piece. Here, we use operation lub of case T_c and polyphony T_p in order to identify the salient notes within the case for extracting deviation with ones within the polyphony to be copied to (Fig. 8). Operation

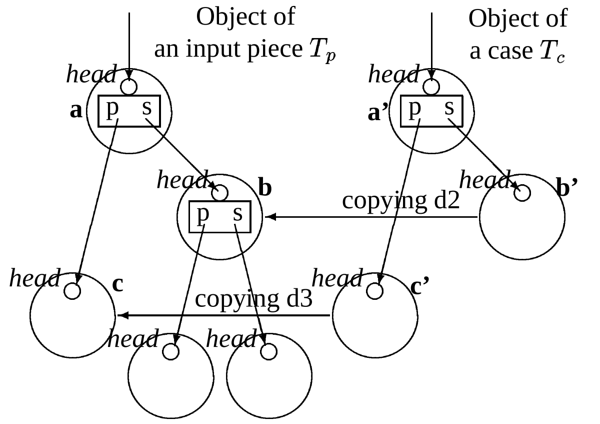

Figure 8: Copying deviation based on lub

$lub(T_p, T_c)$ yields the common part of T_p and T_c, which are the subtree of **a**, **b**, **c**, and that of **a’**, **b’**, **c’**, respectively. The values of deviation are extracted from every terminal *ts* object of the common part (**b’**, **c’**), and the values are transcribed to the corresponding *ts* objects **b**, **c** in the input piece. At the second stage, musical constraint satisfaction is done using the linear function in Fig. 6.

6 Listening Experiments

In this section, we describe two listening experiments.

The first experiment was performed to confirm that the overall atmosphere of a performance is affected by salient notes within the performance. In the experiment, subjects comparatively listened to an original piece and a subpiece that is time-span reduced from the original one. We prepared subpieces of one-level and the two-level pruning. Here, n-level pruning means pruning the secondary branches of the terminal n levels from the original time-span tree. Thus, the more a time-span tree is pruned, the more salient notes remains within it. Actually, two-level pruning decreased the total number of notes by almost half for some pieces. From the results of this experiment, we found that one-level pruning retained more of the atmosphere of the original than two-level pruning, but that even two-level pruning (only about half of the original notes) could retain the original atmosphere to some extent.

The second experiment checked whether the generated performances are heard similar to a sample performance (case) given by a user’s instruction. We made the following two sample performances.

Rendering “All of Me” by case “Autumn Leaves”: “Autumn Leaves” is a normal performance by a jazz piano trio, the famous theme of which was used as a sample performance (case). In the generated performance of “All of Me”, each phrase starts rather later and the sound level (volume) decreases at the end of a phrase. These tendencies are shared with the sample performance of “Autumn Leaves” and normally characterize so-called jazz flavor.

Rendering “Autumn Leaves” by case “a peice by J. Strauss”: J. Strauss’ is played in a standard Vienna-Waltz style, the melody part of which was used as a case. The volume of the generated performance of Autumn Leaves gets lower in the middle of the phrase. Since the generated performance sounds staccato throughout, we think that it succeeded to reflect the feeling of lightness that the case has.

Though our experiments were more or less subjective judgements and incomplete, we think that the generated performances reflected the atmosphere of their input sample performances to some extent.

7 Concluding Remarks

This paper proposed a framework for performance rendering with high controllability. Since this framework has a modular structure, to improve the quality of an output, we can separately examine various modules of the first-stage mapping, the second-stage performance knowledge for musical constraint satisfaction, learning facilities for them and so on. We will consider the other combinations of these modules besides our current implementation.

For better performance, we have to make some improvements to the music representation method and the two-stage PR framework. For a discussion of the former, refer to Hirata and Hiraga 2002. The issues in the latter are as follows:

- A tool to facilitate case acquisition and score analysis is required because it will take quite an effort to extract performance knowledge from sample performances.
- A graphical user interface that can handle a long piece, make a user issue appropriate commands, and visualize a generated performance properly is required.
- A note usually belongs to more than one group, and it is generally unclear and more or less arbitrary to what extent what groups of different layers should contribute to generate the output. Hence, we have to clarify which layer affects the output and by how much.

Last, evaluating our systems is difficult. We think that there are at least four aspects. (1) An overall evaluation for novice users based on observation is presumable one. (2) We are interested in how much user's intentions are reflected in the output performance. Thus, we have to evaluate the capabilities for receiving the user's intentions and reflecting it into the output performance. (3) From the developer point of view, he/she has to identify what modules of the system should be improved. For the purpose, reasoning about the system behavior is required to find out bottlenecks. (4) Our prototype system employs as element technologies DOOD, GTTM, and case-based reasoning. We have to evaluate how well these element technologies work within the system.

References

Arcos, J. L., R. L. de Mántaras, and X. Serra (1997). SaxEx: a case-based reasoning system for generating expressive musical performances. In *Proc. of ICMC*, pp. 329–336.

Bresin, R. (2001). Articulation rules for automatic music performance. In *Proc. of ICMC*, pp. 294–297.

Desain, P. and H. Honing (1992). Towards a calculus for expressive timing in music. In *Music, Mind and Machine*, pp. 173–214. Thesis Publishers.

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

Hirata, K. and T. Aoyagi (2002). Representation method and primitive operations for a polyphony based on music theory GTTM. *IPSJ Journal 43*(2), 277–286. In Japanese.

Hirata, K. and R. Hiraga (2000). Next generation performance rendering – exploiting controllability. In *Proc. of ICMC*, pp. 360–363.

Hirata, K. and Y. Hiraga (2002). Revisiting Music Representation Method based on GTTM. *IPSJ SIG Notes 2002-MUS-45*. In Japanese.

Kifer, M., G. Lausen, and J. Wu (1995). Logical foundations of object-oriented and frame-based languages. *Journal of ACM 42*(3).

Lerdahl, F. and R. Jackendoff (1983). *Generative Theory of Tonal Music*. The MIT Press.

Narmour, E. (1990). *The Analysis and Cognition of Basic Melodic Structures - The Implication-Realization Model*. The Universiy of Chicago Press.

Suzuki, T., T. Tokunaga, and H. Tanaka (1999). A case based approach to the generation of musical expression. In *Proc. of IJCAI*, pp. 642–648.

Widmer, G. (2001). Inductive learning of general and robust local expression principles. In *Proc. of ICMC*, pp. 322–329.

Yokota, K. (1992). Towards an integrated knowledge-base management system: Overview of R&D on databases and knowledge bases in the FGCS project. In *Proc. of International Conference on Fifth Generation Computer Systems 1992*, pp. 89–112. Institute for New Generation Computer Technology.