MELODY MORPHING METHOD BASED ON GTTM

Masatoshi Hamanaka
University of Tsukuba
hamanaka@iit.tsukuba.ac.jp

Keiji Hirata
NTT Communication Science Laboratories

Satoshi Tojo
Japan Advanced Institute of Science and Technology

ABSTRACT

This paper describes a melody morphing method that generates an intermediate melody between a melody and another melody with a systematic order according to a certain numerical measure. Conventional music sequence software only operates on the surface structure of music, such as its notes and rests. The time-span tree, which is acquired from the music surface by using a music theory called GTTM (Generative Theory of Tonal Music), enables us to analyze the deeper structure. Our method makes it possible for one melody to be morphed into another melody by using a melody divisional reduction that applies meet and join operations to the GTTM time-span trees. Experimental results show that our morphing method makes it possible to generate intermediate melodies between two melodies.

1. INTRODUCTION

Our goal is to create a system that will enable a musical novice to manipulate a piece of music, which is an ambiguous and subjective media, according to his or her intentions. We believe that it is important to make it possible to create a musical system for musical novices that can:

1) easily manipulate a piece of music, and
2) mirror the user's intentions.

Note that the higher the abstraction level of the objects for manipulating the music is, the more difficult it becomes to reflect the user's intentions. For example, it is difficult for musical novices to manipulate music with commercial music sequence software that only operates on the surface structure of music, that is, the pitch and note-on timing of each note. On the other hand, Garageband [1] can create a piece of music though simple manipulations, i.e., by just concatenating pre-stored phrases. However, when we want arrange a portion of a melody in a phrase, we have to manipulate the surface structure of the music, and a musical novice would find it difficult for the software to mirror his or her intentions in such a case.

We constructed a system that will enable a musical novice to manipulate a piece of music by using a music theory called the Generative Theory of Tonal Music (GTTM) [2]. We have previously developed a system of music analysis based on GTTM called FATTA [3-4]. FATTA can generate a time-span tree as the result of a GTTM analysis. GTTM consistently represents multiple aspects of music in a single framework. This feature is important when the musical system is to assist a musical novice in manipulating musical structures. For instance, if we imagine a simple operation that splits a melody, the split operations may vary depending on the relevant musical structure. Therefore, it is preferred that the splitting position of the melody and that of the ornamented melody be identical. Unless a system of consistent operations in terms of melody, rhythm, and harmony is developed, the resulting splitting position may be different.

With regard to how the software should reflect the user's intentions, we take an example of composing a melody, where it is supposed that a user wants to arrange melody A by adding some musical nuances to it and he/she knows melody B has such nuances. If a user could use a system accepting a morphing command, issuing a simple command, for example, "add the nuance of melody B to melody A", can accurately convey a user's intention to a morphing system. As a result, the system generates multiple melodies approaching from melody A to B by little and little. The advantage of morphing is not only its simple and accurate transferability and manipulability but also its ease of understanding the relationship between inputs and outputs. Our system exploits melody morphing to make it possible for novices to create melodies reflecting their intensions.

Previous music systems [5-7] have their own way of music analysis, from which it is difficult to acquire deeper musical structures, and thus these systems are difficult to manipulate according to the user's intention. On the other hand, Hirata [8] defined a representation method and primitive operation for polyphony, and this development indicates the potential for constructing a melody arranging algorithm.

In this paper we developed a melody morphing method in which we input monophonies A and B and then generate intermediate melodies between melodies A and B with a systematic order according to a certain numerical measure by configuring the parameters that determine the level of influence of the features of melodies A and B. As part of this overall method, we devised the melody divisional reduction method to reduce the notes of melody A in the difference branch of the time-span tree of melodies A and B.

2. GTTM

Melody morphing uses time-span trees acquired by analysing the results of the Generative Theory of Tonal Music (GTTM). In this section, we briefly describe GTTM. GTTM is composed of four modules, each of which assigns a separate structural description to a listener's understanding of a piece of music. These four modules output a grouping structure, a metrical structure, a time-span tree, and a prolongational tree, respectively (Figure 1). The time-span tree is a binary tree, which is a hierarchical structure describing the relative structural importance of notes that differentiate the essential parts of the melody from the ornamentation.

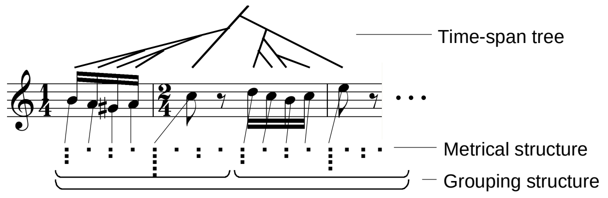


Figure 1. Time-span tree, metrical structure, and grouping structure.

2.1. Abstraction of melody

Figure 2 is an example of abstracting a melody by using a time-span tree. In the figure, there is a time-span tree from melody D, which embodies the results of the GTTM analyses. In the time-span tree, the important notes are connected to a branch nearer the root of the tree. In contrast, the un-important notes are connected to the leaves of the tree. We can obtain an abstracted melody E by slicing the tree in the middle and omitting notes that are connected to branches under line E. In the same manner, if we slice the tree higher up at line F, we can get a more abstracted melody F. We can regard the abstraction of melody as a kind of melody morphing, because melody E is an intermediate melody between melody E and melody F.

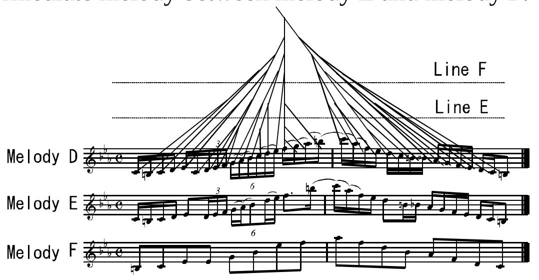


Figure 2. Abstraction of melody.

2.2. Primitive operations using time-span trees

In order to realize the melody morphing, we use the primitive operations of the subsumption relation (written as $\sqsubseteq$), meet (written as $\sqcap$) and join (written as $\sqcup$), as proposed in Hirata [8]. The subsumption relation represents the relation "an instantiated object" $\sqsubseteq$ "an abstract object" (Figure 3a). For example, the relationship among T_D, T_E and T_F, which are the time-span trees (or reducted time-span trees) of melodies D, E and F in Figure 2, can be represented as follows:

$$T_F \sqsubseteq T_E \sqsubseteq T_D$$

The meet operator extracts the largest common part or the most common information of the time-span trees of two melodies in a top-down manner (Figure 3b). The join operator joins two time-span trees in the top-down manner as long as the structures of two time-span trees are consistent (Figure 3c).

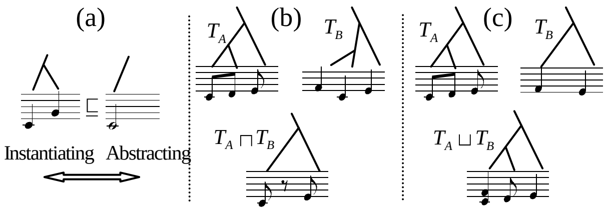


Figure 3. Examples of subsumption $\sqsubseteq$, meet $\sqcap$ and join $\sqcup$.

3. MORPHING METHOD BASED ON GTTM

Initial melody A, target nuance melody B, morphing result melody C , and the morphing method must meet the following conditions. 1 and 2 are conditions for melody C, 3 and 4 are conditions for the method.

1. The similarity between A and C is closer than that of A and B and the similarity between B and C is closer than that of A and B.
2. When B is the same as A, C will be the same as A.
3. The output of multiple melodies C depend on the parameters that decide the level of influence of the features of melodies A and B.
4. C will be a monophony if A and B are monophonies.

3.1. Overview of melody morphing method

The meaning of morphing is to change one image into another through a seamless transition. For example, a morphing method for a face picture can create intermediate pictures through the following operations.

1) Link characteristic points such as on the eyes, nose, etc, in the two pictures (Figure 4a).
2) Rate the intensities of shape (position), colour, etc, in each picture.
3) Combine the pictures.

Similarly, our melody morphing method creates intermediate melodies with the following operations.

1) Link the most common information of the time-span trees of two melodies (Figure 4b).
2) Abstract the notes of a melody in the difference branch of the time-span tree by using the melody divisional reduction method.
3) Combine both melodies.

The melody morphing method is illustrated in Figure 5.

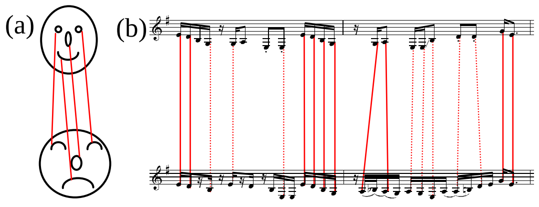
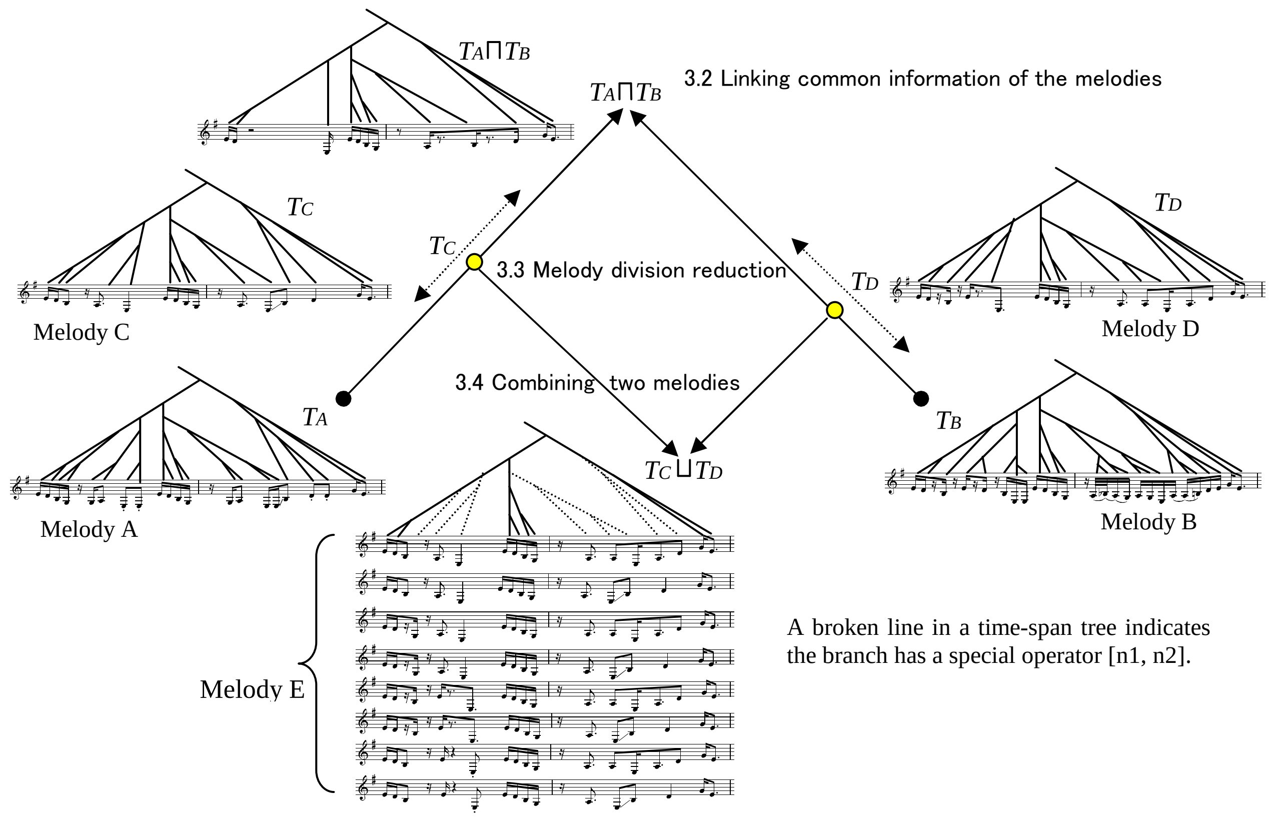


Figure 4. Examples of linking two pictures / melodies.

3.2. Linking common information of the melodies

By using the time-span trees T_A and T_B from melodies A and B, we can calculate the most common information $T_A \sqcap T_B$ which is not only the essential parts of melody A but also those of melody B. The meet operation $T_A \sqcap T_B$ are abstracted from T_A and T_B, and those discarded notes are regarded to be the *difference* information of T_A and T_B.

We use FATTA [4] to generate a time-span tree from a score automatically. We restrict the music structure to monophony, because FATTA only allows monophony input. From now on, we use a word 'melody' only as it pertains to monophony.



$T_A \sqcap T_B$
$T_A \sqcap T_B$
3.2 Linking common information of the melodies
T_C
T_C
3.3 Melody division reduction
T_D
T_D
Melody C
Melody D
3.4 Combining two melodies
T_A
T_B
$T_C \sqcup T_D$
Melody A
Melody B
Melody E
A broken line in a time-span tree indicates the branch has a special operator [n1, n2].

Figure 5. Overview of melody morphing method.

When calculating $T_A \sqcap T_B$ by extracting the largest common part of T_A and T_B in a top-down manner, the result may change depending on whether or not the octave notes such as C4 and C3 can be distinguished. If we discriminate octave notes, C4 $\sqcap$ C3 will be empty $\perp$. On the other hand, if we do not discriminate octave notes, the result is just C, which abstracts the octave information. We regard a note and the octave note to be different notes, because processing is difficult if the octave information is not defined.

3.3. Melody division reduction

We consider that there are features without the other melody in the difference information of T_A and T_B. Therefore, we need a method for smoothly increasing or decreasing the features. The melody divisional reduction method abstracts the notes of the melody in the difference branch of the time-span tree by applying the abstraction described in Section 2.1.

In the melody divisional reduction method, we can acquire melodies *Cm* (*m*=1,2,…,n) from T_A and $T_A \sqcap T_B$, by following algorithm. The subscript *m* of *Cm* indicates the number of notes in the difference information of the time-span trees that are included in T_{Cm} and not included in $T_A \sqcap T_B$.

Step 1: Decide the level of abstraction
A user decides the parameter *L* that determines the level of abstraction of the melody. *L* is from 1 to the number of notes in the difference information of the time-span trees that are included in T_A but not included in $T_A \sqcap T_B$.

Step 2: Abstraction of notes in the difference information
This step selects and abstracts a note which has the fewest number of dots in the difference information. The numbers of dots can be acquired from the GTTM analysis results [3]. If two or more notes have the fewest dots, we select the first one.

Step 3: Iteration
Iterate step 2 *L* times.

Subsumption relations hold as follows for the time-span trees T_{Cm} constructed with the above algorithm.

$$T_A \sqcap T_B \sqsubsetneq T_{Cn} \sqsubsetneq T_{Cn-1} \sqsubsetneq \ldots \sqsubsetneq T_{C2} \sqsubsetneq T_{C1} \sqsubsetneq T_A$$

In Figure 5, there are 9 notes included in T_A but not included in $T_A \sqcap T_B$. Therefore, the value of n is 8, and we can acquire eight kinds of melodies *Cm* (*m*=1,2,…,n) between T_A and $T_A \sqcap T_B$. Hence, melody *Cm* is attenuates features that only have melody A without melody B (Figure 6).

In the same way, we can acquire melody D from T_B and $T_A \sqcap T_B$ as follows.

$$T_A \sqcap T_B \sqsubsetneq T_D \sqsubsetneq T_B$$

3.4. Combining two melodies

We use the join operator to combine melodies C and D, which are results of the divisional reduction using time-span tree of melodies A and B. The simple join operator is not sufficient for combining T_C and T_D, because $T_C \sqcup T_D$ is not always a monophony if T_C and T_D are monophonies. In other words, the result of the operation has chords when the time-span structures are overrides and the pitches of the notes are different; therefore, the result violates condition 4 in section 3.

In order to solve this problem, we introduce a special operator [n1, n2], which indicates note n1 or note n2, as a result of n1 $\sqcup$ n2. Then, the result of $T_C \sqcup T_D$ is all combinations of monophonies made by the operators.

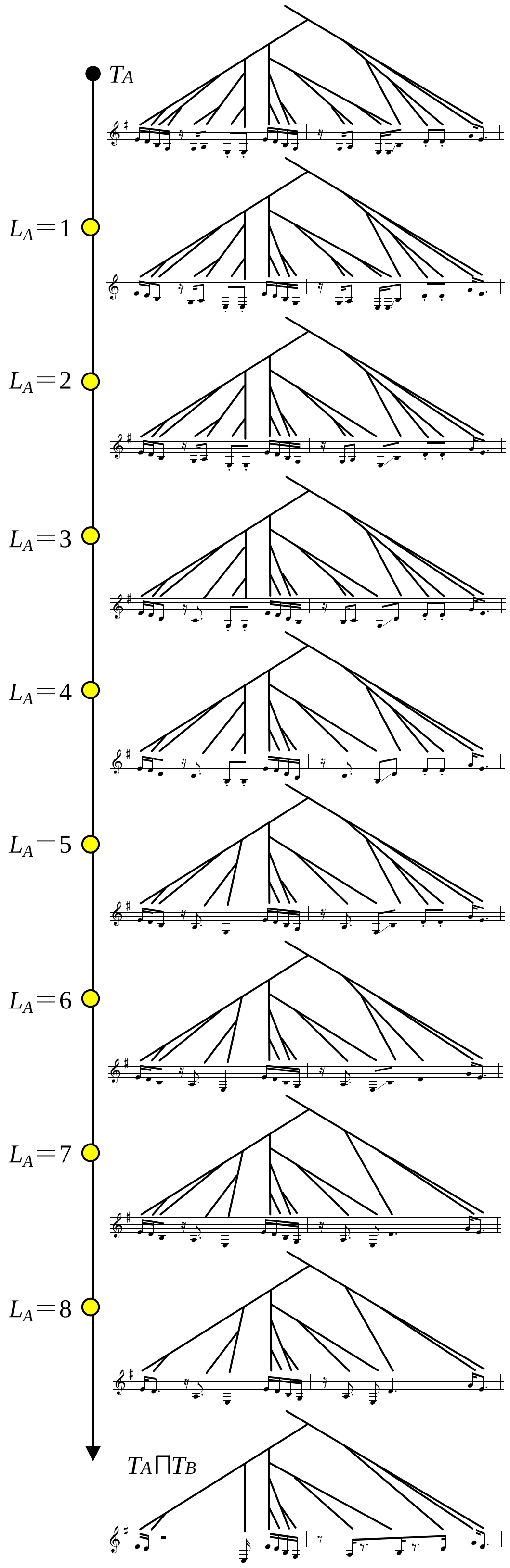


Figure 6. Melody divisional reduction.

4. EXPERIMENTAL RESULTS

We tried to determine whether melody C, which is the morphing result of melodies A and B, is their intermediate. Namely, we tried to determine whether our melody morphing method fulfils condition 1 at the beginning of section 3. To measure the similarity between A and B, A and C, and B and C, we used the following $R_N(A,B)$, defined by Hirata [8], which indicates how much information is lacking from the two melodies as a result of the meet operation.

$$R_N(A,B) = \frac{|meet(A,B)|_N}{max(|A|_N, |B|_N)} \qquad (1)$$

where $|A|_N$ indicates the number of notes in melody A.

We use 10 pairs of sample melodies A and B, and as a result of confirmation, our morphing method meets condition 1 in Section 3 for any parameter set of L_A and L_B.

5. CONCLUSION

We devised a melody morphing method based on GTTM, which makes it possible to construct an intermediate melody between one melody and another melody. Experiment results show that the morphed melodies are intermediates of the input two melodies according to the similarity defined by Hirata [8]. In fact, we do not think that current experimental results are not enough to support our claim that the morphing method proposed meets the conditions listed in Section 3 under the new metric defined in Section 4. Since we however believe that our method has a potential to support the claim, we would like to conduct another experiment to show justification. We are now planning to extend the method to polyphony, because we have thus far restricted it to monophony.

6. REFERENCES

[1] http://www.apple.com/ilife/garageband/.

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

[3] Hamanaka, M., Hirata, K., and Tojo, S. "Implementing 'A Generative Theory of Tonal Music'", *Journal of New Music Research*, 35:4, 249-277, 2006.

[4] Hamanaka, M., Hirata, K., and Tojo, S. "FATTA: Full Automatic Time-span Tree Analyzer", *Proceedings of the International Computer Music Conference*, Vol. 1, pp. 153-156, 2007.

[5] Balaban, M. " The Music Structures Approach to Knowledge Representation for Music Processing", *Computer Music Journal*, 30:2, pp. 96-111, 1996.

[6] Cope, D. *Experiments in Musical Intelligence*. A-R Editions, Inc. 1996.

[7] Dannenberg, R. " Machine Tongues XIX: Nyquist, a Language for Composition and Sound Synthesis", *Computer Music Journal*, 21:3, pp. 50-60, 1997.

[8] Hirata, K., and Aoyagi, T. " Computational Music Representation Based on the Generative Theory of Tonal Music and the Deductive Object-Oriented Database", *Computer Music Journal*, 27:3, pp. 73-89, 2003.