MELODY EXTRAPOLATION IN GTTM APPROACH

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

We developed a melody-morphing method in which we input melodies A and B and not only interpolate but also extrapolate other melodies between those two melodies. This is done in a systematic order according to a certain numerical measure, based on the parameters which reflect the influential features of two input melodies. The main advantage of our method is that a time-span tree is used, which is acquired from the music surface using a music theory called Generative Theory of Tonal Music (GTTM). By using our defined primitive operations of time-span trees, we can manipulate melodies like numerical expressions.

1. INTRODUCTION

The purpose of this study is to construct an interactive melody generator for professional as well as novice composers. Because professional composers want to reflect their intensions accurately, they want to lower the abstraction level of the objects to manipulate the melody. On the other hand, musical novices tend to raise the abstraction level so as to handle the melody more easily. Therefore, there are trade-off relationships, which are difficult to satisfy both professionals and novices needs. For example, commercial music sequence software only operates on the surface structure of a melody, that is, the pitch and note-on timing of each note, and thus it is difficult for novices to generate melodies. On the other hand, Mozart's "Musical Dice Games" [1] generates agreeable melodies, even if a novice configures the parameters. However, it is difficult to sufficiently reflect the composers' intension.

To solve this trade-off problem, there is a framework, in which a user indicates his or her intension by representing an instance. For example, suppose that a user intends to arrange melody A by adding some musical flavor to it and he or she knows that melody B has such taste. Then, if he or she could use a command such as "add the nuance of melody B to melody A", he or she would be able to accurately convey his or her intension to a system. We call this operation, adding the nuance of melody B to melody A, 'melody morphing'.

Figure 1 is the layout of our interactive melody generator in which the melodies on the pop-up menu are morphing result of a selected melody on the editing score with another melody chosen by the user. We can play, copy, paste, or displace the morphed melodies on the pop-up.

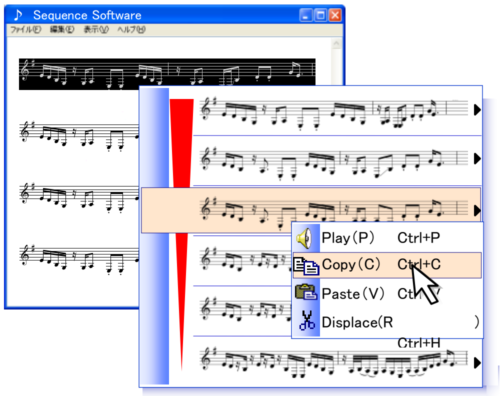


Figure 1. Interactive melody generator.

We have developed a melody-morphing method by using a time-span tree, which is acquired from the music surface using the Generative Theory of Tonal Music (GTTM) [2]. 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 ornamental notes.

Our previous melody-morphing method only generates interpolative melodies and thus the variation of the output melodies is small [6]. In interpolative melody C generated from melody A and B, the similarity between A and C is higher than that of A and B, and the similarity between B and C is higher than that of A and B. On the other hand, in extrapolative melody C from melody A and B, a melody B is an interpolative melody of melodies A and C, or a melody A is an interpolative melody of melodies B and C (Figure 2).

We propose a melody-morphing method, which enables the extrapolation of melodies that generates melody C, which emphasizes the character of melody A or melody B. As part of this overall method, we devised a melody divisional reduction and melody divisional augmentation methods to reduce or increase the notes of melody A in the differential branch of the time-span tree of melodies A and B.

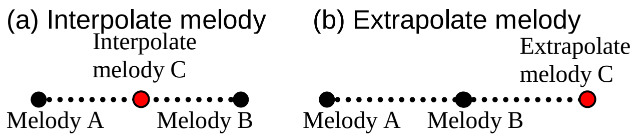


Figure 2. Interpolative and extrapolative melodies.

2. GTTM AND MELODY MORPHING

Our melody morphing method uses time-span trees acquired by analysing the results of the GTTM. Figure 3 is an example of abstracting a melody by using a time-span tree. There is a time-span tree from melody D, which embodies the results of the GTTM analyses. 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 regard the abstraction of a melody as a kind of melody morphing because melody E is an intermediate melody between melody D and melody F.

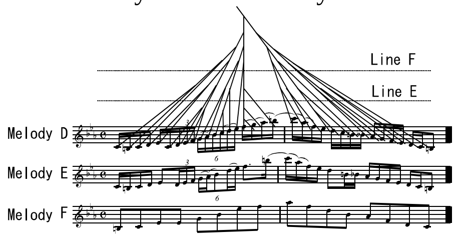


Figure 3. Abstraction of melody.

2.1. ATTA

We have previously constructed an automatic time-span tree analyzer (ATTA), which derives a time-span tree based on an extended GTTM (exGTTM) that we proposed. The exGTTM re-formalizes the rules of the GTTM and establishes an algorithm for acquiring a time-span tree [3, 4].

2.2. Primitive Operations Using Time-Span Trees

For 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 by Hirata [5]. The subsumption relation represents the relation "an instantiated object" $\sqsubseteq$ "an abstract object" (Figure 4a). For example, the relationship among TD, TE and TF, which are the time-span trees (or reduced time-span trees) of melodies D, E, and F in Figure 3, can be represented as follows:

$$T_F \sqsubseteq T_E \sqsubseteq T_D$$

The meet operator extracts the maximally common part of two time-span trees of two melodies in a top-down manner (Figure 4b). 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 4c).

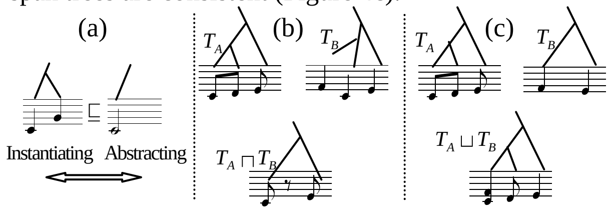


Figure 4. Examples of subsumption $\sqsubseteq$, meet $\sqcap$ and join $\sqcup$.

2.3. Melody Morphing Algorithm

Generally, morphing is the changing of one image into another through a seamless transition. For example, a morphing method for a face picture is used to create intermediate pictures through the following operations.

1) Link characteristic points such as eyes and nose, in the two pictures (Figure 5a).
2) Rate the intensities of shape (position), color, etc… in each picture.
3) Combine the pictures.

Similarly, our melody morphing method is used to create intermediate melodies with the following operations.

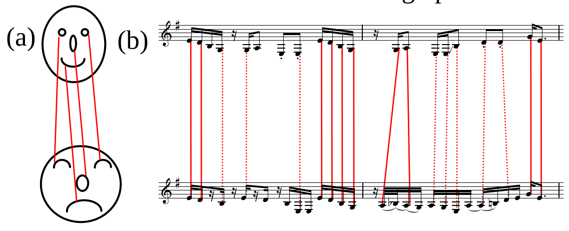


Figure 5. Examples of linking two pictures/melodies.

3. MELODY MORPHING BY EXTRAPOLATION

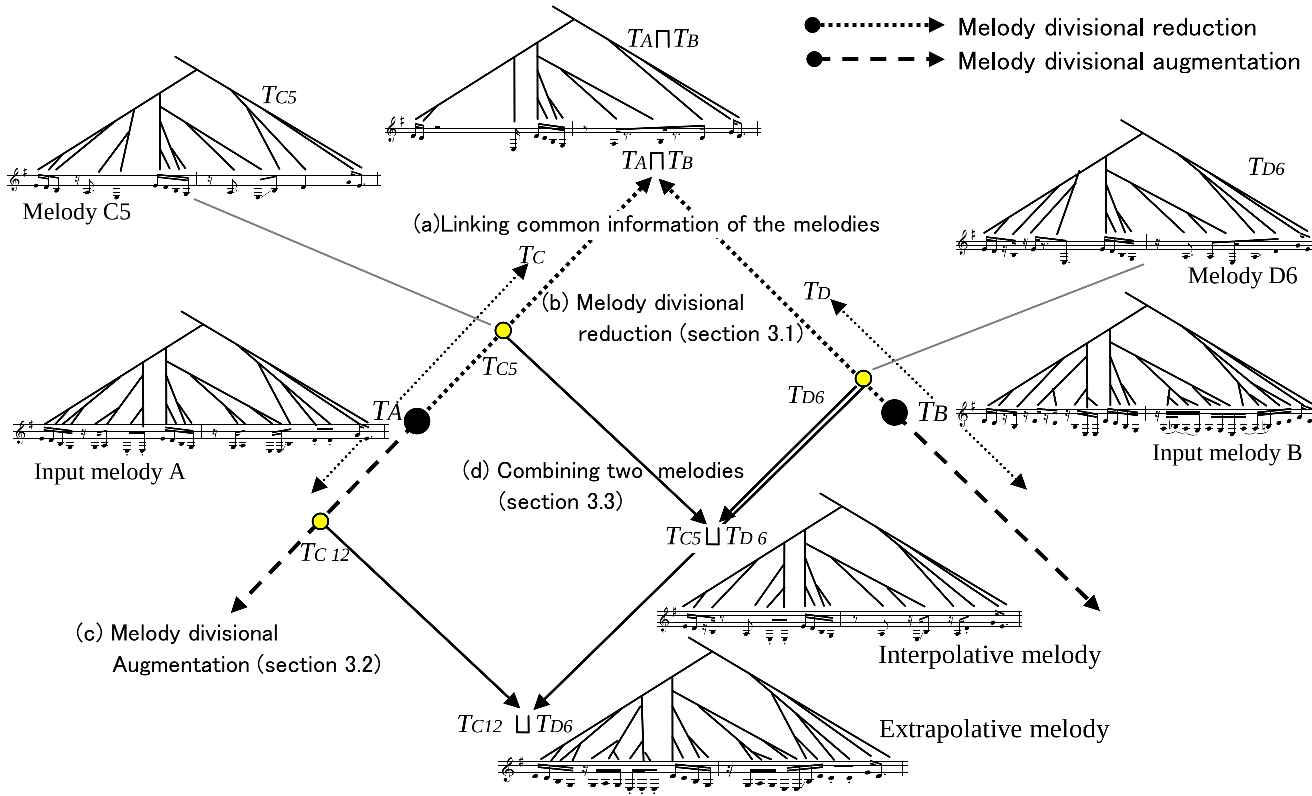
By using the time-span trees T_A and T_B from melodies A and B, we can calculate the most common information $T_A \sqcap T_B$ which are the essential parts of melody A as well as those of melody B. The meet operations $T_A \sqcap T_B$ are abstracted from T_A and T_B, and those discarded notes are regarded as the *difference information* of T_A and T_B (Figure 6a). 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 these features. The melody divisional reduction method can abstract the notes of the melody in the differential branch of the time-span tree [6]. On the other hand, the melody divisional augmentation method can increase the notes of the melody in the differential branch of the time-span tree which is an inverse process of abstraction. Before explaining melody divisional augmentation, we will explain melody divisional reduction.

3.1. Melody Divisional Reduction

In the melody divisional reduction method [6], we can acquire melodies *Cm* (*m*=1, 2, …, n-1) from T_A and $T_A \sqcap T_B$, which holds the subsumption relations as follows (Figure 6b).

$$T_A \sqcap T_B \sqsubsetneq T_{C1} \sqsubsetneq T_{C2} \sqsubsetneq \ldots \sqsubsetneq T_{C\,n-2} \sqsubsetneq T_{C\,n-1} \sqsubsetneq T_A$$

The value of subscript *m* of *Cm* indicates the number of notes in the difference information of the time-span trees that are included in T_A and not included in T_{Cm}. In Figure 7a, there are nine notes included in T_A but not included in $T_A \sqcap T_B$. Therefore, the value of *n* is eight, and we can acquire eight kinds of interpolative melodies *Cm* between T_A and $T_A \sqcap T_B$. Hence, melody *Cm* attenuates features that only have melody A without melody B.



Melody divisional reduction
Melody divisional augmentation
TA⊓TB
TC5
Melody C5
TA⊓TB
(a)Linking common information of the melodies
TD6
Melody D6
TC
(b) Melody divisional reduction (section 3.1)
TD
TC5
TD6
TA
TB
Input melody A
Input melody B
(d) Combining two melodies (section 3.3)
TC5⊔TD 6
TC 12
Interpolative melody
(c) Melody divisional Augmentation (section 3.2)
TC12 ⊔TD6
Extrapolative melody

Figure 6. Overview of melody-morphing method.

3.2. Melody Divisional Augmentation

In the melody divisional augmentation method, we can acquire melodies *Cm* (*m*=*n*+1, *n*+2, *n*+3 …), which holds the subsumption relations as follows (Figure 6c).

$$T_A \sqcap T_B \sqsubsetneq T_A \sqsubsetneq T_{C\,n+1} \sqsubsetneq T_{C\,n+2} \sqsubsetneq T_{C\,n+3} \sqsubsetneq \ldots$$

The melody divisional augmentation method increases the number of notes one by one, which is the opposite process that the melody divisional reduction method reduces the number of the notes one by one (Figure 7b).

Step 1: Decide the level of augmentation
A user determines parameter *L*, which determines the level of augmentation. *L* is from 1 to the number of notes, which the user wants to increase in the melody.

Step 2: Divide a note in the difference information
Select a note with the longest duration in the difference information of the time-span tree that are not included in $T_A \sqcap T_B$. If two or more notes have the longest duration, we select the first one. Then, divide the note where the beat is strongest. The value of strength in each beat can be acquired from the GTTM analysis results [2-4].

Step 3: Determine the pitch of divided notes

If a note contains the strongest beat from two divided notes, keep the original pitch before the dividing; otherwise, change the pitch to the most stable one from lowest to highest pitch of the original melody. The value of stability is calculated based on the music theory of Tonal Pitch Space [7].

Step4: Iteration

Iterate steps 2 and 3 *L* times.

3.3. Combining Two Melodies

We use the join operator to combine melodies C and D, which are results of the divisional reduction or augmentation using time-span tree of melodies A and B (Figure 6d).

Our previously proposed melody morphing method only generates interpolative melodies in which the melodies C and D in Figure 6 are both acquired using melody divisional reduction. By using the melody divisional augmentation, we can generate extrapolative melodies.

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 even if T_C and T_D are monophonies. In other words, the result of the operation has chords when the time-span structures override and the pitches of the notes are different, although both input melodies are monophonies.

To solve this problem, we introduce a special operator [n1, n2], which chooses notes n1 or n2 exclusively, as a result of n1 ⊔n2. Then, the result of $T_C \sqcup T_D$ is all combinations of monophonies generated from the operators.

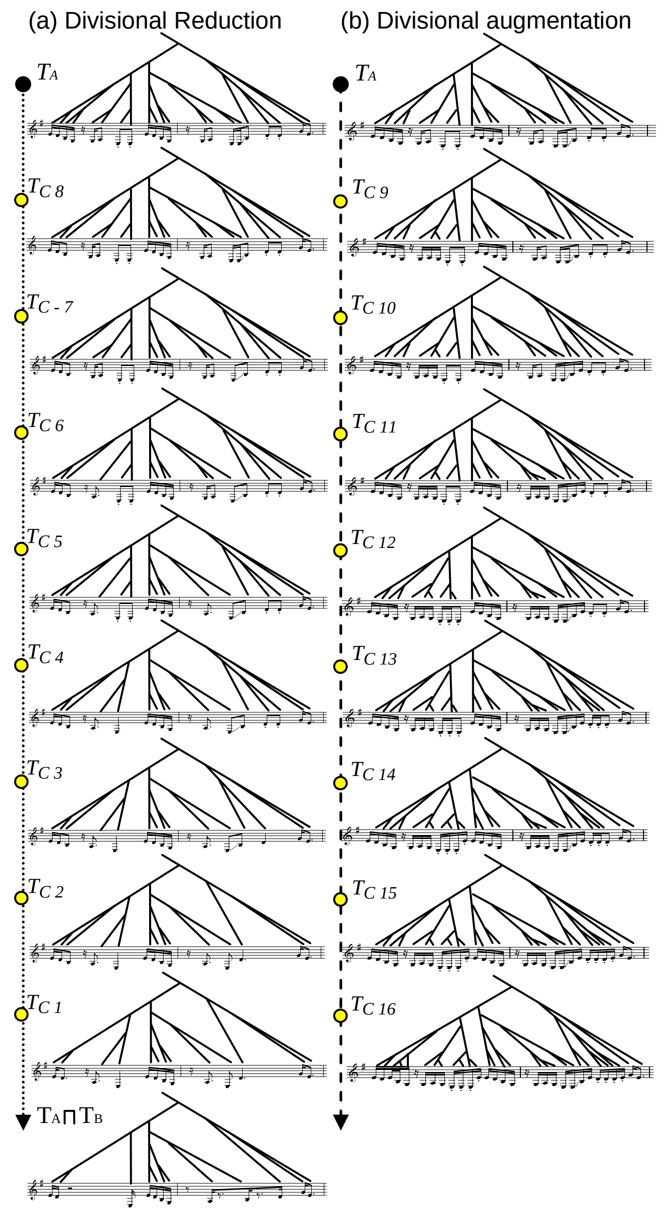


Figure 7. Melody divisional reduction and augmentation.

4. EXPERIMENTAL RESULTS

We tried to determine whether the method can generate the extrapolative melody M from melody A and B which holds following expression.

$$\begin{aligned}&\{R(A,M) < R(A,B)\ and\ R(A,M) < R(B,M)\}, \text{ or}\\ &\{R(B,M) < R(A,B)\ and\ R(B,M) < R(A,M)\}\end{aligned} \quad (1)$$

where $R(X,\mathrm{Y})$ indicates the similarity between melodies X and Y.

The meaning of the expression (1) is that a melody B is an interpolative melody of melodies A and M, or a melody A is an interpolative melody of melodies B and M.

To measure the similarity between melodies X and Y, we used the following $R_N(X, Y)$, defined by Hirata [5], which indicates how much information is lacking from the two melodies as a result of the meet operation.

$$R_N(X,Y) = \frac{|meet(X,Y)|_N}{max(|X|_N, |Y|_N)} \quad (2)$$

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

We use 10 pairs of sample melodies A and B, and as a result of confirmation, all the extrapolative melodies M from melodies A and B, holds the expression (1).

5. CONCLUSION

We constructed a melody morphing method for generating interpolative melodies and extrapolative melodies from two input melodies A and B by using melody divisional reduction and augmentation, which smoothly decreases or increases the difference information of those melodies. To generate, interpolative, or extrapolative melodies, all we need to do is select two input melodies and configure the parameters for controlling the reduction or augmentation level of each melody. We plan to finish developing the interactive melody generator and evaluate whether the melody-morphing method is useful for generating melodies.

6. REFERENCES

[1] Langston, P. "Six Techniques for Algorithmic Composition", Proceedings of the International Computer Music Conference, pp. 153-156, 1989.

[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] 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.

[6] Hamanaka, M., Hirata, K., and Tojo, S. "Melody Morphing Method based on GTTM", *Proceedings of the International Computer Music Conference*, pp. 155-158, 2008.

[7] Lerdahl, F. *Tonal Pitch Space*, Oxford University Press, 2001.