Melody Slot Machine:
Melody Morphing by Using Time-span Tree of GTTM

Masatoshi Hamanaka
RIKEN

Mizuki Kobayashi
RIKEN/Toho Gakuen College Music Department

Kyoko Otagawa
RIKEN

Mayumi Shimada
RIKEN

masatoshi.hamanaka@riken.jp

ABSTRACT

This paper describes an overview of our developed interactive music system called a "Melody Slot Machine," which is an application based on the generative theory of tonal music (GTTM). Generally, melody manipulation requires expert knowledge and is difficult for musical novices. Therefore, we developed the Melody Slot Machine, which provides an experience of manipulating melodies by allowing users to freely switch two original melodies and morphing melodies. We conducted an experiment with a musicologist who understood the melody morphing method based on GTTM and attempted to use it for composition. When a melody generated with the method was unnatural, the musicologist added notes to correct it. The experimental results showed that 78.5% of the notes in the composed pieces were generated with the melody morphing method, while the remaining 21.5% was added by the musicologist.

1. INTRODUCTION

This paper describes a system called the "Melody Slot Machine," which enables novices without musical knowledge to manipulate melodies. In the Melody Slot Machine, melody fragments are displayed on the dial, and the melody to be played can be switched by rotating the dial. The valuations of melody fragments are composed on the basis of the generative theory of tonal music (GTTM) [1], and thus, switching the melody fragments does not change the overall structure of the melody.

Various melody generation systems based on deep learning are being constructed [2, 3]. For example, DeepBach enables users to generate three voices from one voice by merging the learned networks of two long short-terms memories (LSTMs) and one neural network [2]. In another example, Celtic Melody Generation makes it possible to generate melodies of Celtic-style ethnic music using a recurrent neural network (RNN) [3]. Although those systems performed well in specific tasks, they are not compatible with various music operations and are difficult at this stage for people to use to compose music [2, 3].

We believe that a computer that understands music deeply must be developed to realize a composition support system that enables various music operations. Therefore, for 15 years, we have been implementing the GTTM, proposed by Fred Lerdahl and Ray Jackendoff in 1983 [4-7]. The performance of GTTM analysis by computers has recently been dramatically improved by introducing deep learning [7].

The time-span tree, which is the deep structure obtained as a result of GTTM analysis, shows the relationship between the main notes and the ornamatation notes structurally. The main advantage of using the time-span tree is that it is possible to reduce the notes of a melody by decreasing the ornamentation notes step by step while maintaining the main structure of the melody. One example of a melody operation by using the time-span tree is melody morphing, which generates an intermediate melody from two input melodies. Previously, we proposed a melody morphing method to generate an intermediate melody of two melodies as a melody operation that uses the time-span tree [8]. We are also studying an operation called flip to invert the time-span tree [9]. We are planning to realize the operation of various melodies in the future by using time-span trees.

We believe that melody manipulation by the time-span trees can give pleasure to musical novices by enabling them to manipulate music and can improve the composition efficiency of professional composers. To assess whether the former is feasible, we construct a system called a Melody Slot Machine. We conducted an experiment to verify the effectiveness of the melody morphing method. In the experiment, a musicologist who understood the method used it to manually generate an intermediate melody. If the intermediate melody was unnatural, the musicologist added notes to correct it. We found that 78.5% of the notes were generated with the melody morphing method, while the remaining 21.5% was manually added by the musicologist.

2. MELODY SLOT MACHINE

The Melody Slot Machine, which enables users to manipulate melodies, has three features.

2.1 User friendly interfaces

To allow anyone to easily manipulate melodies, we adopted a dial type interface that makes it possible to replace a part of the melody (Figure 1). There is a rectangular hole in a part of the acrylic board sandwiching the score, and the dial interface on a tablet can be operated with fingers through the hole. When the red lever on the right side of the score is pulled down, all the dials rotate, and one of the melody fragments on the dial is randomly selected.

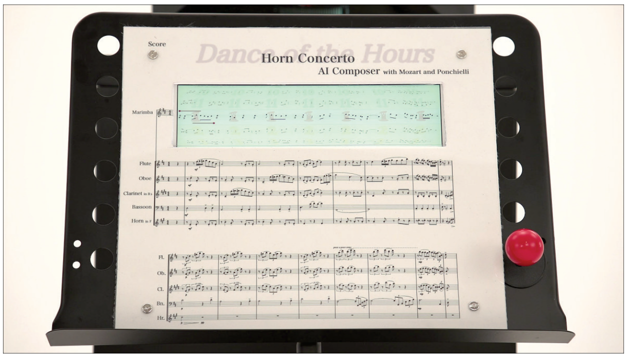

Figure 1. Slot dial and lever

2.2 Easy-to-understand operation results

We prepare a display showing a performer so that the result of the operation can be confirmed visually as well as aurally. A holographic display was used for show the performer so as to increase the feeling of presence (Figure 2). The users can feel like they are controlling a performer by operating a melody.

Figure 2. Holographic display

2.3 Improve the feeling of presence

We recorded all the performance sounds so as to increase the feeling of presence. For the recording, we used a studio with very little reverberation because otherwise only the reverberation of the preceding sound enters the beginning of the melody fragment due to the melody splitting into fragments. Reverberation is added when the melody is played. Three pairs of speakers were installed, and panpot and reverb were set for each direction so that the holograph seemed to be a real performer (Figure 3). When putting your head between two pairs of speakers, both the sound and video enhance the presence.

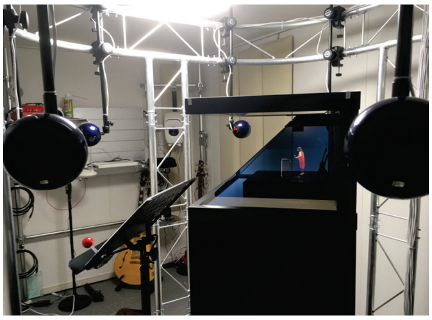

Figure 3. Three pairs of speaker

3. TIME-SPAN TREE OF GTTM

Melody morphing uses time-span trees obtained from the results of GTTM analysis. The GTTM consists of four modules, each of which assigns a separate structural description to a listener's understanding of a piece of music. As shown in Figure 4, the four modules output a grouping structure, metrical structure, time-span tree, and prolongational tree. The time-span tree is a binary tree, which is a hierarchical structure, representing the relative structural importance of notes that differentiate the essential parts of a melody from ornamentation.

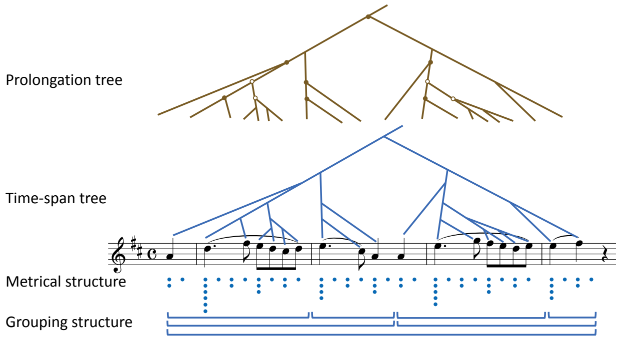

Figure 4. Prolongational tree, time-span tree, metrical structure, and grouping structure obtained from GTTM analysis

3.1 Abstraction of melody

Figure 5 shows an example of abstracting a melody by using a time-span tree. The figure includes a time-span tree from melody, D, which embodies the results of GTTM analysis. In the time-span tree, important notes are connected to branches nearer the root of the tree, whereas unimportant notes are connected to leaves. We can obtain an abstracted melody, E, by slicing the tree in the middle (line E) and then omitting notes whose branch connections are below line E. In the same manner, if we slice the tree higher up at line F, we can obtain an even more abstracted melody, F. We can regard this abstraction of melody as a kind of melody morphing because melody E is an intermediate melody between melodies D and F.

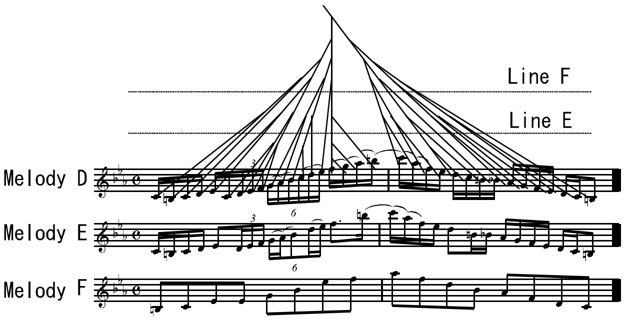

Figure 5. Abstraction of melody

3.2 Primitive operations of time-span trees

To implement melody morphing, we use the primitive operations: the subsumption relation (written as $\sqsubseteq$), meet (written as $\sqcap$), and join (written as $\sqcup$) [9]. As shown in Figure 6a, subsumption represents the relation by which "an instantiated object subsumes an abstract object." 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 5, can be represented as follows:

$$\text{TF} \sqsubseteq \text{TE} \sqsubseteq \text{TD} \quad (1)$$

Figure 6b illustrates the meet operator, which extracts the largest common part or most common information of the time-span trees of two melodies in a top-down manner. Finally, Figure 6c illustrates the join operator, which joins two time-span trees in a top-down manner as long as their structures are consistent.

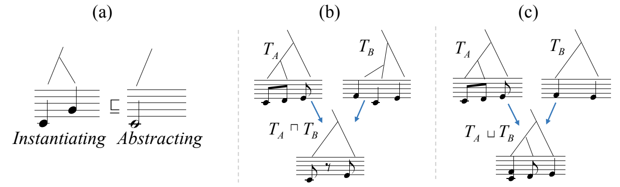

Figure 6. Examples of subsumption ⊑, meet ⊓, and join ⊔ operations

4. MELODY MORPHING METHOD BASED ON GTTM

In this section, we explain the melody morphing method based on GTTM we proposed in 2008 [8]. The initial melody A, target nuance melody B, morphing result melody C, and melody morphing method must meet the following conditions: 1 and 2 for melody C, and 3 and 4 for the method.

1. A must be more similar to C than to B, and B must also be more similar to C than to A.
2. When B is the same as A, C will be the same as A.
3. The output of multiple Cs depends on parameters that determine the levels of influence of the features of A and B.
4. C will exhibit monophony if A and B are monophonic.

4.1 Overview of melody morphing method

Morphing means to change one image into another through a seamless transition. For example, a morphing method for a facial image can create intermediate images through the following operations.

1. Link characteristic points such as the eyes and nose in two images, as shown in Figure 7a.
2. Rate the intensities of shape (position), color, and so forth in each image.
3. Combine the images.

Figure 7. Examples of linking two images and two melodies

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

1. Link the most common information of the time-span trees of two melodies, as shown in Figure 7b.
2. Abstract the notes of a melody in a differing branch of the time-span tree by using the melody divisional reduction step of our melody morphing method.
3. Combine both melodies.

Figure 8 illustrates the melody morphing method.

4.2 Linking of common information

By using the respective 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 includes the essential parts of not only A but also B. The meet operation $T_A \sqcap T_B$ abstracts common notes from T_A and T_B, and the discarded notes are then regarded as the difference information of T_A and T_B.

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 octave notes such as C4 and C3 can be distinguished. If we discriminate octave notes, then C4 ⊓C3 will be empty, denoted as ⊥. On the other hand, if we do not discriminate octave notes, the result is just C, which abstracts the octave information. Here, we regard a note and its octave as different notes, because processing is difficult if the octave information is not defined.

4.3 Melody divisional reduction

We next consider that the difference information of T_A and T_B includes features not present in the other respective melody. Therefore, we need a method for smoothly inserting or removing such features. The melody divisional reduction step of our melody morphing method abstracts the notes of the melody in a difference branch of the time-span tree by applying the abstraction described in Section 3.1.

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

Step 1: Determine the level of abstraction.

The user selects a parameter *L* that determines the level of abstraction of the melody. *L* can range from 1 to the number of notes in the difference information of the time-span trees that are included in T_A but not in $T_A \sqcap T_B$.

Step 2: Abstract notes in the difference information.

The note with the fewest dots in the difference information is selected and abstracted. The numbers of dots can be acquired from the GTTM analysis results [1, 7]. If two or more notes share the fewest dots, we select the first one reading the music left to right.

Step 3: Iterate.

Step 2 is iterated *L* times.

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

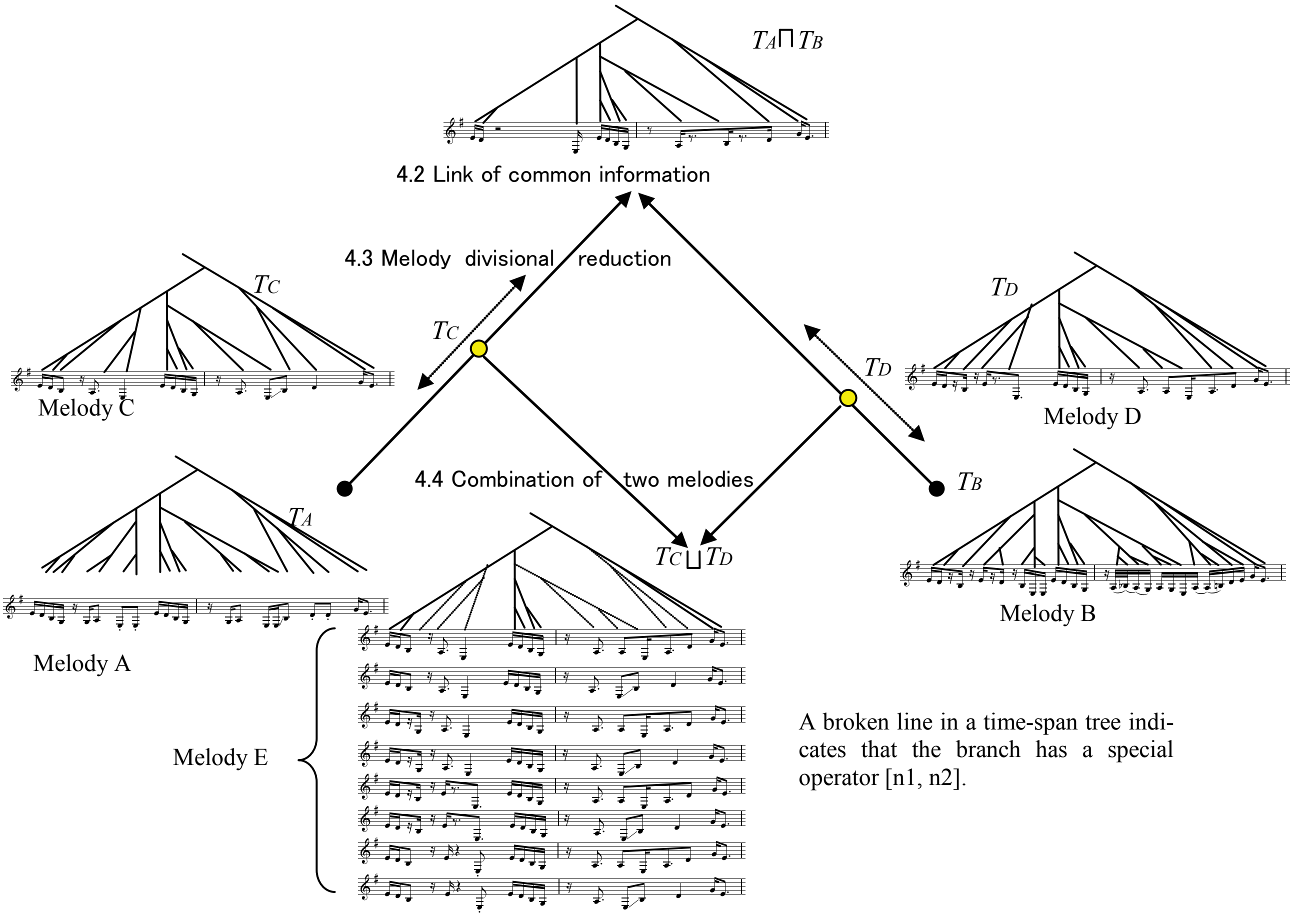

Figure 8. Overview of melody morphing method

$$T_A \sqcap T_B \sqsubsetneq T_{Cn} \sqsubsetneq T_{Cn-1} \sqsubsetneq \ldots \sqsubsetneq T_{C2} \sqsubsetneq T_{C1} \sqsubsetneq T_A. \quad (2)$$

In Fig. 8, nine notes are included in T_A but not in $T_A \sqcap T_B$. Therefore, the value of n is 8, and we can obtain eight intermediate melodies *Cm* (*m*=1,2,…,n) between T_A and $T_A \sqcap T_B$. Hence, melody *Cm* attenuates features that occur only in melody A but not in B. Figure 9 illustrates this process.

In the same way, we can obtain melody D from T_B and $T_A \sqcap T_B$ in the following manner:

$$T_A \sqcap T_B \sqsubsetneq T_D \sqsubsetneq T_B. \quad (3)$$

4.4 Combination of two melodies

Finally, we use the join operator to combine melodies C and D, which are the results of divisional reduction using the time-span trees of melodies A and B. The simple join operator is not sufficient for combining T_C and T_D because $T_C \sqcup T_D$ does not always exhibit monophony even if T_C and T_D are monophonic. In other words, the result of the operation has chords when the time-span structures override, and the pitches of the notes are different; therefore, the result violates condition 4 listed at the beginning of Section 4: C will exhibit monophony if A and B are monophonic.

To solve this problem, we introduce a special operator [n1, n2], which indicates either note n1 or note n2, as a result of n1 $\sqcup$n2. Then, the result of $T_C \sqcup T_D$ is all monophonic combinations given by the operators.

5. MANUAL GENERATION OF INTERMEDIATE MELODY

Because of the problems described below with generating intermediate melodies on a computer with the method discussed in Section 4, we also had a musicologist manually generate a morphing melody.

5.1 Time-span analysis of two input melodies

Before linking the common information of two input melodies, we need their time-span trees. Although we have already developed grouping and metrical structure analyzers based on deep learning [7], we have not yet developed a high-performance time-span tree analyzer [4, 6]. Acquisition of time-span trees thus requires manual analysis by a musicologist.

If two input melodies are completely different, then their common information will be empty, $\perp$, making it difficult to generate intermediate melodies. Therefore, the two input melodies must be carefully chosen by the musicologist. The musicologist selected a variation and theme from Mozart's "Twelve Variations" (K. 265/300e) for the melodies, as explained in Section 6.

5.2 Time-span reduction

At present, the melody divisional reduction step of the

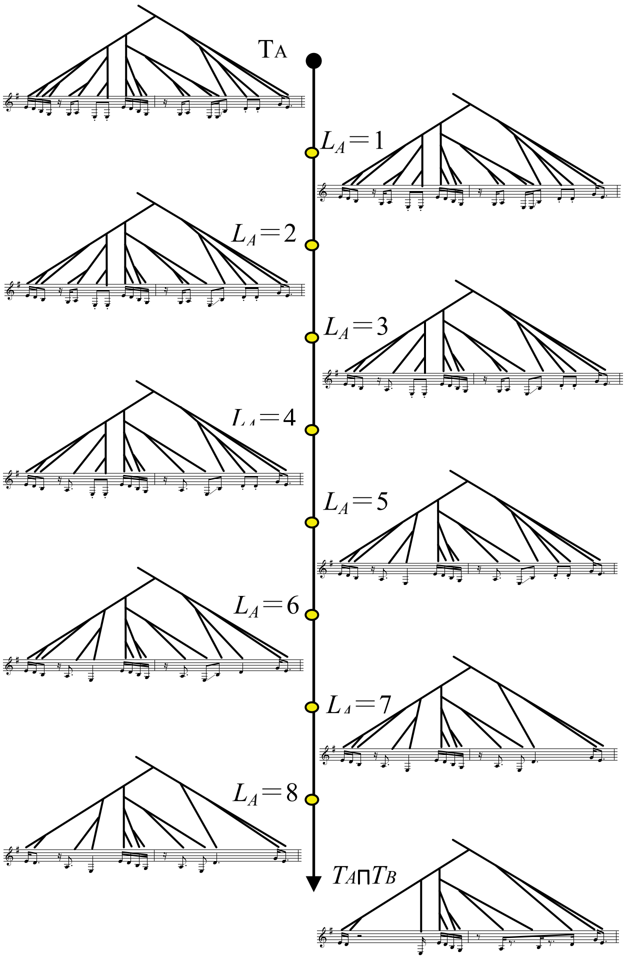

Figure 9. Melody divisional reduction

melody morphing method cannot be automated. This is because the height of each branch in the time-span tree is not given, and the order of the notes to be reduced cannot be determined. A musicologist who can analyze time-span trees, however, can easily generate reduced melodies by considering the inverse process of composition depending on his/her interpretation, as described in Section 3.2. When multiple composition processes are possible, we consider all interpretations and then select one.

5.3 Selection of note from special operators

By selecting a note from each special operator after combining two melodies, the output melody will be monophonic when the two input melodies are monophonic. Therefore, depending on how this note is selected, the melody changes. When the intermediate melody is unnatural, the musicologist adds notes to correct the melody. The details of these added notes are described in the following section on the experimental results.

6. EXPERIMENTAL RESULTS

We asked a musicologist to generate intermediate melodies. The musicologist had over 10 years of experience in GTTM analysis and a deep understanding of the melody-morphing method.

For the input melodies, the musicologist selected the theme and Variation No. 1 of Mozart's Twelve Variations on "Ah vous dirai-je, Maman", K. 265/300e. She constructed nine intermediate melodies between the two melodies. Figures 10 shows the first set of 8 bars for the theme, Variation No. 1, and 9 intermediate melodies, out of 40 bars in total. In the figure, nonparenthetical notes were generated with the melody morphing method, whereas parenthetical notes were added by the musicologist. For each melody, Table 1 lists the numbers of total notes, notes generated with the melody morphing method, and notes added by the musicologist.

Notes were added in five ways.

a) Adding appoggiaturas, auxiliary notes, and passing notes.
b) Borrowing a note from a neighboring branch or swapping the order of notes in a branch.
c) Dividing a note into two notes of the same pitch.
d) Expanding or contracting the melody.
e) Quoting a melody.

In Figure 10, morphing melodies 3 and 5 are examples of adding passing notes, denoted as (a), whereas (b) is an example of swapping the order of notes from Variation No. 1. In Fig. 10, (c) is an example of dividing a note into two notes of the same pitch; (d) is an example of expanding a melody, specifically the last two notes in the fourth bar of Variation No. 1; and (e) is an example of quoting a melody from the fifth and sixth bars of Variation No. 1.

All the notes in morphing melodies 1 and 2 were generated with the melody morphing method because both melodies were close to the common information of the theme and Variation No. 1, so melodies could be generated by simply choosing an appropriate note from each special operator without adding any sound. All the notes in morphing melody 5 were also generated with the melody morphing method because morphing melody 5 was very close to a melody ob-tained by divisional reduction of Variation No. 1, so all the notes were eighth notes.

Among morphing melodies 1, 2, and 3, the three melodies closest to the theme, only 1.5% of the notes were added by the musicologist. In contrast, morphing melodies 8 and 9, the closest to Variation No. 1, had 37.9% of notes added by the musicologist. Notes needed to be add ed to prevent unnatural melodies when generating multiple melodies between a melody with many eighth notes,

Table 1. Numbers of notes in each melody

	Total number of notes	Number of notes generated with melody morphing method	Number of notes added by musicologist
Theme	82	-	-
Morphing melody 1	51	51 (100%)	0 (0%)
Morphing melody 2	51	51 (100%)	0 (0%)
Morphing melody 3	93	90 (96.8%)	3 (3.2%)
Morphing melody 4	121	96 (79.3%)	25 (20.7%)
Morphing melody 5	94	94 (100%)	0 (0%)
Morphing melody 6	157	129 (82.2%)	28 (17.8%)
Morphing melody 7	176	157 (89.2)	19 (10.8%)
Morphing melody 8	253	164 (64.8%)	89 (35.1%)
Morphing melody 9	267	159 (59.6%)	108 (40.4%)
Variation No. 1	271	-	-

such as morphing melody 5, and a melody with many sixteenth notes, such as Variation No. 1.

In total, 78.5% of the notes were generated with the melody morphing method, whereas the remaining 21.5% were added by the musicologist.

7. CONCLUSIONS

We developed the Melody Slot Machine, which enables users to manipulate melodies. For the Melody Slot Machine, we used our melody morphing method based on the generative theory of tonal music (GTTM) for composition. We made melody morphing possible by combining our melody morphing method with a manual process. Experimental results were obtained by having a musicologist compose with the melody morphing method. In the composed pieces, 78.5% of the notes were generated with the melody morphing method, while the remaining 21.5% was added by the musicologist. We confirmed that the arranging processes used in the experiment could be classified into five types.

We plan to develop a support system for adjustment using the melody morphing method.

Acknowledgments

This work was supported by JSPS KAKENHI, Grant Numbers 17H01847 and 16H01744.

8. REFERENCES

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

[2] G. Hadjeres, F. Pachet, and F. Nielsen, DeepBach: a Steerable Model for Bach Chorales Generation, *Proceedings of the 34th International Conference on Machine Learning*, PMLR 70:1362-1371, 2017.

[3] B. Sturm, J. Santos, O. Ben-Tal, and I. Korshunova. Music transcription modelling and composition using deep learning, *CSMC2016*, 2016.

[4] M. Hamanaka, K. Hirata, and S. Tojo: "Implementing 'A Generating Theory of Tonal Music'", *JNMR*, Vol.35, No.4, pp.249-277, 2007.

[5] M. Hamanaka, K. Hirata, and S. Tojo: "FATTA: Full Automatic Time-span Tree Analyzer", *ICMC2007*, Vol.1, pp.153-156, 2007.

[6] M. Hamanaka, K. Hirata, and S. Tojo: "Sigma GTTM III: Learning based Time-span Tree Generator based on PCFG", *CMMR 2015*, pp.303-317, June 16-19, 2015.

[7] M. Hamanaka, K. Hirata, and S. Tojo: "DeepGTTMII: Automatic Generation of Metrical Structure based on Deep Learning Technique," *SMC2016*, pp.221-249, 2016.

[8] M. Hamanaka, K. Hirata, and S. Tojo, "Melody Morphing Method based on GTTM," *ICMC2008*, pp. 155-158, 2008.

[9] K. Hirata and S. Tojo, "Retrograde of Melody and Flip Operation for Time-Span Tree," *CMMR 2016*, pp.298-305, 2016.

Figure 10. First eight bars of theme, Variation No. 1, and nine intermediate melodies