2013-Interaction (1EXB-51)
2013/2/28

旋律概形を用いた旋律編集: 概形レベルと音符レベルの編集をシームレスに行えるインターフェース

土屋 裕一[1,a)]　北原 鉄朗[2,b)]

概要：本研究では作曲スキルを持たないユーザに旋律編集を支援するシステムを提案する. 音楽の聴き方には, 旋律全体を１つの流れとしてとらえる聴き方と, 個々の音符を個別にとらえる聴き方があり, 旋律編集でもこの２つの認知レベルに対応していることが望ましい. 本システムは旋律概形とピアノロールを統合することで, この２つの手段による旋律編集をシームレスに行えるようにしたシステムである. 試用の結果から, 楽曲を仕上げていく思考錯誤を手軽に行えることが明らかになった.

Melody Editing Using Melodic Outline: Seamless Interface Integrating Outline-level and Note-level Editing

Yuichi Tsuchiya[1,a)]　Tetsuro kitahara[2,b)]

Abstract: In this paper, we proposed a system for supporting melody editing for users without music composition skills. It is known that there are two different levels in music cognition, melodic-outline-level and note-level, therefore melody editing systems should support the editing at these two different levels. Our system enables users to edit melodies at these two different levels seamlessly by integrating piano-roll and melodic-outline representations. Trial uses show that the proposed method facilitates the user's trial-and-error in melody editing.

1. はじめに

自動作曲システム [1], [2], [3], [4], [5] は, 作曲スキルのないユーザにオリジナルの楽曲を得る手段を与えてくれるものである. しかし, 自動作曲システムが出力する楽曲は, 必ずしもユーザの欲するものと合致するとは限らない. 自動作曲システムが出力した楽曲にユーザが満足しない場合, 満足いくものにするには, ユーザが自ら MIDI シーケンサなどを用いて楽曲を編集する必要がある. しかし, 自動作曲システムの主たるターゲットである, 作曲スキルを持たないユーザにとって, MIDI シーケンサなどを用いて楽曲を編集するのは, 困難な作業である.

音楽の聴き方には, 旋律全体を 1 つの流れとしてとらえる聴き方と, 個々の音符を個別にとらえる聴き方がある. そのため, 音楽の編集に対しても, 旋律全体の音の流れに対する編集要求と, ある特定の 1 音に対する編集要求との両方が存在すると考えられる.

本研究では, こういった異なる 2 つの認知レベルに基づく旋律編集をシームレスに行えるシステムを提案する. 自動作曲システムなどが出力した旋律をこのシステムに入力すると, 我々が以前提案した手法 [6], [7] に基づいて旋律概形が計算され, 画面に表示される. その際, その背後に薄い色でピアノロールと同様の音符レベルの旋律情報が表示される. ユーザがマウスで旋律概形を描き直すと, 描き直された旋律概形に基づいて旋律の音符列が即座に再生成され, 画面が更新される. 音符列を直接編集することもでき, 音符列を直接編集するとそれに合わせて旋律概形が自動的に更新される. このように, 旋律概形と音符レベルの旋律表現とが常に同期し, 一方の編集結果が即座に他方に反映さ

[1] 日本大学大学院総合基礎科学研究科
Graduate School of Integrated Basic Sciences, Nihon University
[2] 日本大学文理学部
College of Humanities and Sciences, Nihon University
[a)] tsuchiya@kthrlab.jp
[b)] kitahara@chs.nihon-u.ac.jp

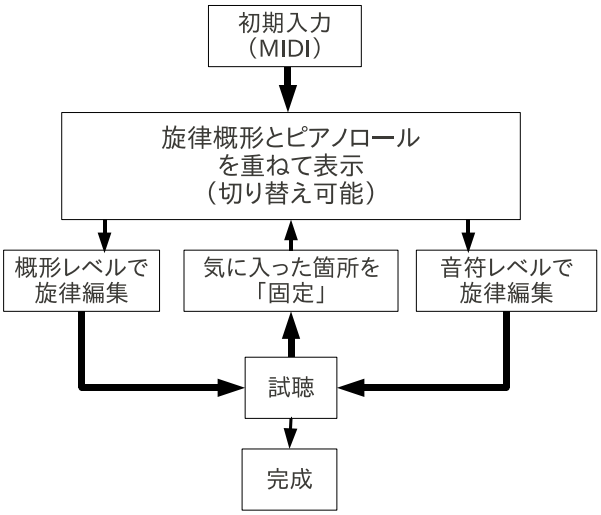

図 1 編集の流れ図

れることで, 2 つの編集手段を自由に行き来しながら旋律を編集することができる.

2. システムの概要

本研究では, 次の要件を満たすシステムを実現する.

- 旋律の概形と音符レベルの情報とが, 重ね合わされて表示される.
- 概形を編集すると音符レベルの情報が, 音符レベルで編集すると概形が即座に更新される.
- 音符レベルで編集すると, その後概形を編集しても, その音符は変化しない（「音符が固定される」という）.
- 各音符の固定は, 手動で設定・解除することができる.

この 4 つの要件により, たとえば次のような使い方が実現できる (図 1).

(1) 編集対象の旋律に対して, 旋律概形を描き直して, その結果生成される旋律を試聴するというプロセスを何度も繰り返す (図 2).

(2) 繰り返し編集する過程で, 旋律全体ではなく, 特定の音符を修正したい場合には, 適宜編集対象を「音符レベル」に切り替えて音符レベルでの編集を行う (図 3).

(3) 旋律概形あるいは音符レベルの旋律を編集・試聴し, 気に入った箇所があれば, その音符を「固定」する. これにより, 旋律概形を編集しても, その音符は変化されなくなる (図 4).

この (1)~(3) を自由に組み合わせながら繰り返すことで, 徐々に所望の旋律に近づけていくことができる.

ただし, 現状の実装では, 簡単のため, 音符レベルでの編集においても, 音高の編集のみ対応している.

3. 実現手法

本システムは, 入力旋律から旋律概形を求める部分, 編集された旋律概形からその概形に沿った音符列を求める部分, 編集された音符列から改めて概形を求める部分に分けることができる. 以下, 各々の手法について述べる.

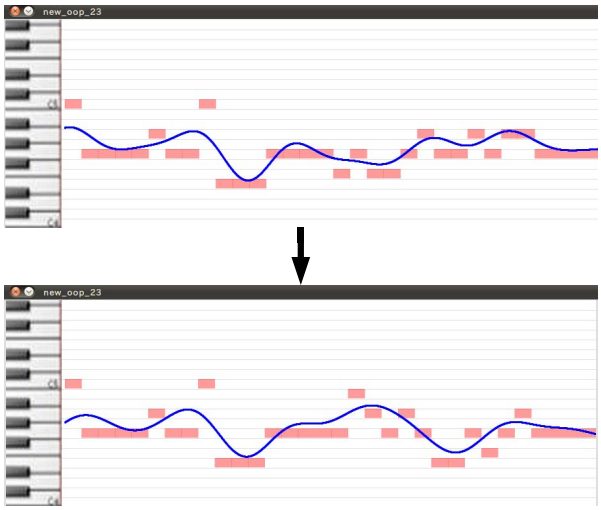

図 2 概形レベルの旋律編集の例. 青色で表示された旋律概形を描き直すと, それに合わせて音符列が自動的に更新される.

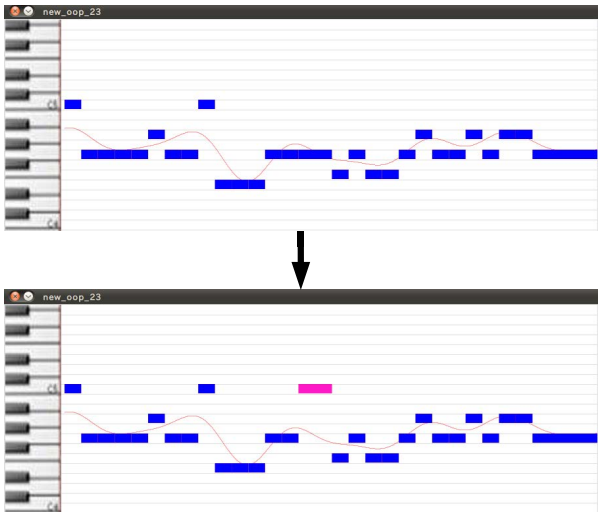

図 3 音符レベルの旋律編集の例. 特定の音符を編集すると, その音符が「固定」され, 色が変わる. 固定されれば音符は, 以降旋律概形を編集しても変更されない.

3.1 旋律概形の抽出

入力旋律から旋律概形を求めるのは, 文献 [6] で提案したように, 音高の時系列にフーリエ変換を適用することで実現する. まず, 与えられた旋律 (図 5 (a)) を音高の時系列 (図 5 (b)) に変換する. 次に, この音高の時系列に対してフーリエ変換を行う. このとき, 低次のフーリエ係数が旋律の大まかな流れを表し, 高次のフーリエ係数が旋律の細かな動きを表す. そこで, 低次のフーリエ係数のみを取り出して, 逆フーリエ変換をすることで, 旋律の大まかな流れ (旋律概形, 図 5(c)) を得る.

3.2 旋律概形からの音符列の生成

旋律概形からの音符列の生成についても, 基本的には文

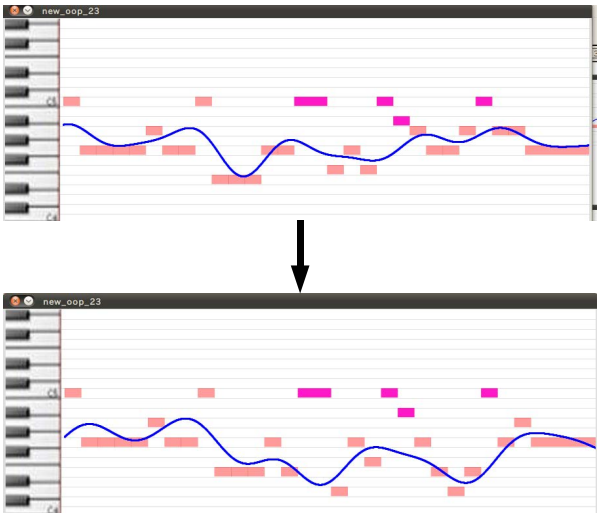

図 4 いくつかの音符が固定された状態 (ピンク色) での概形レベルの旋律編集の例. 旋律概形を書き換えても, 固定された音符は変更されないことがわかる.

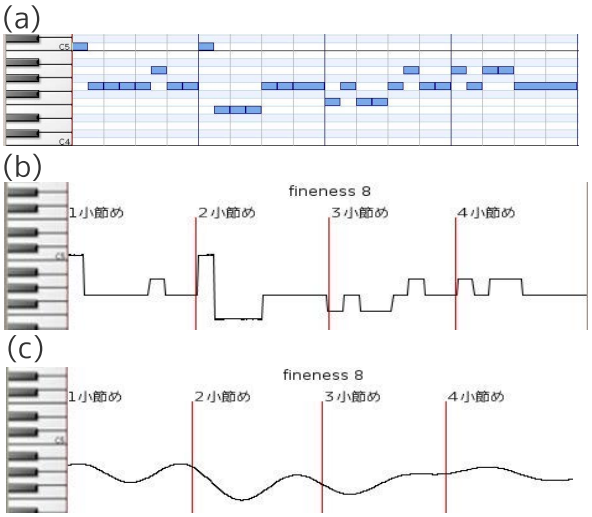

図 5 音符列から旋律概形を抽出する手法の概要. (a) 旋律の MIDI シーケンス, (b) 音高の時系列, (c) 旋律概形.

献 [6] で提案したものと同様の手法で行う. まず, 編集後の旋律にフーリエ変換を行って得られた低次のフーリエ係数と, 編集前の高次のフーリエ係数を結合して逆フーリエ変換をすることで, ユーザが入力した全体の流れに沿いつつ, 元の旋律の微細な特徴を反映させた音高の時系列を得る. 次にこの音高の時系列から, 音楽的に適したノートナンバー系列を隠れマルコフモデル（HMM）で推定する. この HMM では各状態 s_n が各ノートナンバー n に対応し, 平均 n, 分散 σ^2 の正規分布に従って音高の値を出力する. 分散 σ^2 の値はすべての状態で共通であり, 実験的に定める.

ただし, 文献 [6] では実現されていなかった, 音符の固定機能を実現する必要がある. そこで, 固定された音符が上書きされないように, 状態 s_i から状態 s_j への状態遷移確率 $P(s_j|s_i)$ を次のように定義する.

$$P(s_j|s_i) = p_1(s_j)\ p_2(s_j)\ p_3(s_j - s_i)$$

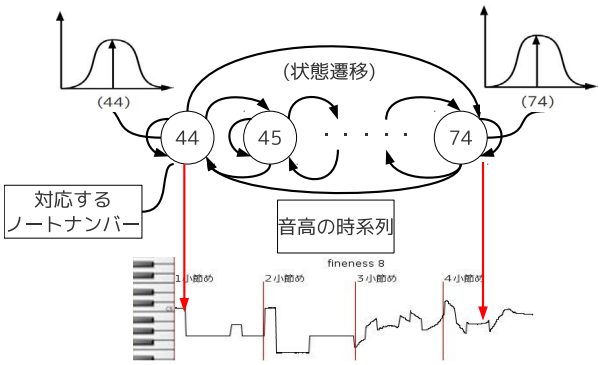

図 6 音高の時系列から音符列を推定する HMM の概要

ここで, $p_1(s_j)$ は遷移先の音符が固定されているときに, その音符が書き換えられなくするためのもので, 遷移先の音符が固定されているときは,

$$p_1(s_j) = \begin{cases} 1 & (\text{遷移先の音符がノートナンバー } j \text{ で固定されているとき}) \\ 0 & (\text{遷移先の音符が } j \text{ 以外のノートナンバーで固定されているとき}) \end{cases}$$

と定義し, 遷移先の音符が固定されていないときには, $p_1(s_j)$ は等確率として定義する. $p_2(s_j)$ は, 音符列ができるだけ特定のスケールに収まるようにするためのもの, $p_3(s_j - s_i)$ は, 音高が極端に跳躍するのを防ぐためのものであり, 現状では手動で実験的に定めた値を用いている.

3.3 編集された音符列からの旋律概形の再抽出

旋律が音符レベルで編集された場合は, 3.1 と同じ手法により旋律概形を再抽出する.

4. 提案手法の試用

提案した手法に基づき, 旋律を編集するシステムを実装し試用した. 入力は自動作曲システム Orpheus[5] で作曲した 4 小節からなる MIDI データである. 編集の一例を図 7 に示す. 編集箇所は 3 小節めから 4 小節めの最後にかけてである. 初めに, 旋律概形を用いて 3 小節めのはじめあたりから徐々に音を低くし, 4 小節めのはじめから終わりにかけて音を徐々に高くするイメージで編集を行った (図 7 (c)). その出力結果を試聴したところ, 3 小節めのはじめから 2 つめまでの音と 4 小節めの最後の 2 つの音が編集後のイメージと合致したので, 固定機能を用いてこれらの音符を固定した. そして 3 小節めの 3 つめの音が編集時のイメージと合致しなかったので, 3 小節めの初めから 3 番目の音をソからラへ変更した (図 7 (d),(e)). 最後に試聴を行い 3 小節目の 4 番目から最後の音と 4 小節目の最初から 3 つめの音にかけて, 旋律概形を用いた編集と音符レベルの編集を併用して微調節を行った (図 7 (f)). このように, 本システムを使うことで概形レベルの旋律編集と音符レベルの旋律編集とを任意の順序で組み合わせて旋律編集ができ

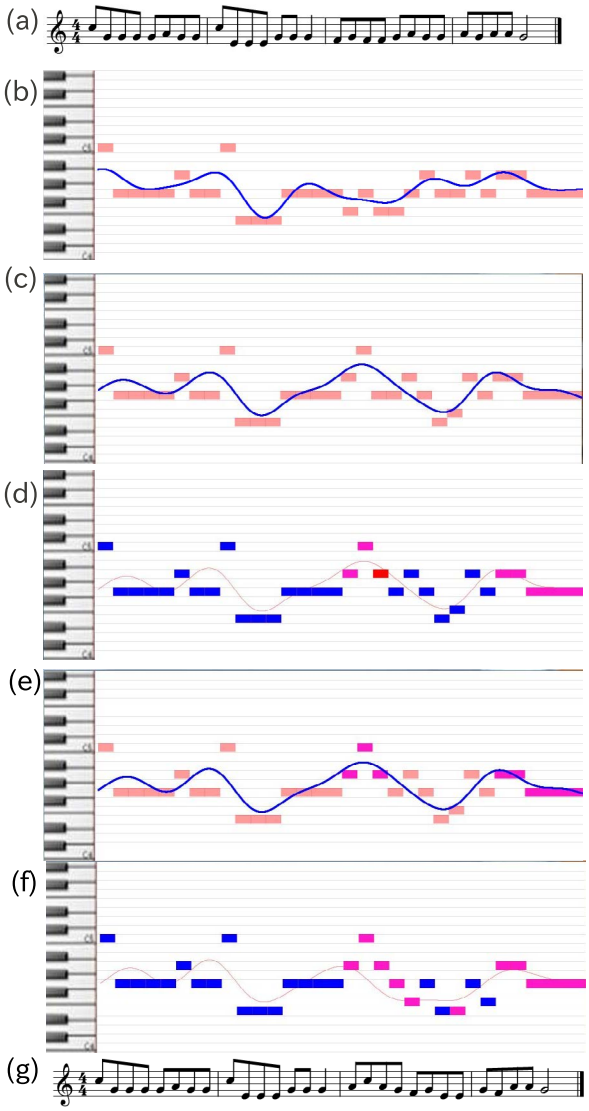

図 7 試用結果, (a) 編集開始前の旋律, (b) 編集開始前の編集画面, (c) 概形レベルで編集を行った図, (d) 音符レベルで編集を行っている図, (e)(d) の編集を行った音符が固定されている図, (f) 仕上げ編集の図, (g) 編集を終了した旋律

るようになった. また出力結果は伴奏と不協和な音を含まない旋律であった.

5. おわりに

本稿では, 旋律概形とピアノロールを併用することで試行錯誤の繰り返しを手軽にできる旋律編集システムを提案した. 旋律概形では, 概形上を描き直すことで旋律を編集でき, ピアノロールでは音符ごとに旋律を編集できる. この２つの方法をシームレスに使い分けられるようにすることによって, 作曲スキルがないユーザが様々な試行錯誤を行えるようになった. 今後は被験者によるユーザビリティ評価に取り組んでいきたい.

参考文献

[1] L. Hiller, L. lsaacson, “Musical composition with a high-speed digital computer”, Journal of Audio Engineering Society (1958).
[2] C. Ames, M. Domino, “Cybernetic composer: An overview,”in Understanding Music with AI, AAAI Press, pp.186-205 (1992).
[3] D. Cope, “Computers and Musical Style”, Oxford University Press (1991).
[4] 安藤 大地, P. dahlstedt, M. G. Nordaxhl, 伊庭 斉志,“対話型 GP を用いたクラッシック音楽のための作曲支援システム”, 芸術科学会論文誌, Vol.4, No.2, pp.77–86 (2005).
[5] 深山 覚, 中妻 啓, 米林 雄一郎, 酒向 慎司, 西本 卓也, 小野 順貴, 嵯峨山 茂樹：“Orpheus：歌詞の韻律に基づいた自動作曲システム”, 情報処理学会研究報告, 2008-MUS-76-30, pp.179–184 (2008).
[6] 土屋 裕一, 北原鉄朗：“旋律概形抽出に基づく直感的な旋律編集手法”, 情報処理学会研究報告, Vol.2012-MUS-95, pp.41–46 (2012).
[7] 土屋 裕一, 北原鉄朗：“誰でも使える旋律編集システム”, EC2012, pp.190–191 (2012).