Vol.2021-MUS-132 No.13
2021/9/16

接続コストに基づく後続メロディ候補の提示による メロディ打ち込み支援インタフェース

平井 辰典[1,a)]　澤田 隼[2,b)]

概要：本稿では，メロディ素片間の接続コストに基づいて後続メロディ候補を提示することによって，メロディの打ち込みを支援するインタフェースを提案する．著者らが提案した BiLSTM に基づくメロディ素片間の接続コスト [1] を活用して，実際にメロディの打ち込みを行う際に必要に応じて後続メロディや後続音符の候補を提示するインタフェースを開発した．具体的には，ピアノロール上にメロディを打ち込んでいくようなメロディ制作のシチュエーションを想定し，ユーザが入力済みのメロディを基に，後続のメロディや音符の候補を，既存のメロディによって構成されるデータベース内から探索し，ユーザに提示するようなインタフェースである．開発したインタフェースを使って実際にメロディの打ち込みを行うユーザスタディを実施し，本提案インタフェースの有効性についての評価を行った結果，その有効性が確認できた．さらに，本提案インタフェースを使った音楽制作の可能性についても議論する．

1. はじめに

音楽のメロディは，楽曲を象徴する重要な要素であり，メロディを作ることは音楽制作において重視される．メロディのあり方は多様なもので，単純なモチーフをただ何度も繰り返すようなメロディもあれば，曲の最初から最後まで同じメロディが登場しないような複雑なメロディもある．多くの楽曲に共通して言えることは，メロディを構成するためにはそれなりの長さが必要になるということである．作曲を行う際には，1 曲分のメロディを 0 から紡ぎあげていく作業が必要となる．しかし，一度に思いつくメロディの長さには限界があり，楽曲全体に対するほんの一部ずつしか制作できないことが多い．多くの例外はあるが，一般的なメロディ制作では，短いフレーズ単位のメロディをいくつも考えながら繋げていく作業が行われている．

短いメロディのフレーズは，鼻歌を歌うような感覚で比較的容易に思い浮かべることができ，この行為自体は作曲に精通している人でなくてもできるものと考えられる．一方で 1 曲分のメロディを最初から最後まで作り上げることは誰もが簡単にできることとは言い難い．以上のことから，メロディ制作における困難な作業は，短いメロディ素片同士をうまく繋げる部分にあるのではないかと考えた．そこで，本研究ではメロディ素片同士の繋がりの自然さを測るための尺度としてメロディの接続コストを提案し，それを用いたメロディ打ち込み支援インタフェースを提案する．メロディ素片同士の接続コストの算出については以前著者らが [1] にて提案した手法を用いることとしており，本稿はその応用についての続報に位置する．

自身が考えた短いメロディ素片を集めて繋ぎ合わせることでより長いメロディを作り上げることができれば，何気なく鼻歌を歌うような行為をより本格的な音楽制作へと繋げることができると考えられる．他にも，自身が考えたメロディに限らずに既存のメロディを対象に，複数の曲からなるマッシュアップ音楽などを制作する際にどのメロディ同士が自然に繋がりやすいかを測り，利用することができれば制作支援に繋がると考えられる．本稿ではこのような音楽制作支援の可能性を追究する一歩としてメロディの打ち込みを支援するインタフェースについて検討する．

近年，Music Transformer[2] や MusicVAE[3] などのように深層学習モデルを用いたメロディの生成モデルが多く提案されており，その生成結果は非常に高い品質になってきている．一方で，これらの手法の多くはいわゆる「自動作曲」に分類されるようなモデルであり，ユーザが創作に使おうと考えた場合に，機械が担う役割の比重が大きいものとなっている．本研究では，機械による創作物への関与をあくまでも支援のレベルに留める形でメロディ制作の支援ができないかを考えた．著者らが [1] にて提案したメロディ素片同士の接続コストの算出手法では，BiLSTM を用

[1] 駒澤大学
Komazawa University
[2] 東京理科大学
Tokyo University of Science
[a)] thirai@komazawa-u.ac.jp
[b)] sawada@rs.tus.ac.jp

いており，ネットワークの構成を少し変えればメロディの自動生成にも応用できるモデルとなっているが，自動生成という形はとらず，あくまでも既存のメロディ同士の繋がりやすさのみを評価するモデルとして活用する．これにより，過去に制作したメロディを素材として，現在制作中のメロディの後続メロディとして繋がりやすいメロディを必要に応じて提示できるようなシステムを目指す．

本インタフェースでは機械による生成は行わず，必要に応じて過去に打ち込んだことがある既存メロディを提示する．それによって，文字入力の予測変換機能のように必ず使わなければいけない機能ではないけれど，良い候補が挙げられたときに使うと便利であるというような立ち位置の支援インタフェースとする．本稿で提案するインタフェースにおいて，後続メロディ候補のメロディは生成されたものではなく過去に人手で打ち込まれたメロディである点と，候補の並び順こそ算出された接続コストを基準に変わるものの最終的に後続メロディ候補を選ぶ作業がユーザに委ねられているという点で，機械が担う役割の比重は小さいものとなっている．

本稿は，メロディの接続コストを応用したメロディ打ち込み支援インタフェースを開発し，その有用性に関する簡易的なユーザスタディを行ったことに関する報告である．

2. 関連研究

メロディの接続コストに基づいて既存メロディを再利用することで新たなメロディを生成する手法として，Bretan らは Deep Learning モデルによるメロディの Unit selection（素片選択）手法を提案している [4]．Bretan らの手法では，本研究で行うメロディの制作支援に近いアプローチによって，自動で既存メロディの再利用を行い，メロディを生成する．本研究との差分としては，Bretan らはあくまでも音楽生成の文脈でメロディの接続コストを利用しており，ユーザ主体の制作の支援やそれを実現するインタフェースの開発にまでは言及されていない点にある．また，Bretan らの手法では，メロディの接続コストだけでなく，メロディ同士の意味的な関連性も用いて探索対象の絞り込みを行っている．

既存のメロディを繋ぎ合わせることで新たな音楽を生成するというアプローチは Cope によっても提案されており，楽曲を細かくセグメンテーションし，それぞれの素片の特性に基づいてラベル付けをすることで，再利用，再結合をしながら新しい楽曲を制作する手法を提案している [5]．Cope による試みは，本稿で提案する手法や前述の Bretan らによる手法とは違い，機械学習等によるモデル化を行わずにルールに基づいてメロディの再構築を行うというアプローチとなっている．

既存のメロディを利用するアプローチは，これまでにも音楽生成の文脈で提案されてきている．Pachet が提案した The Continuator は既存のメロディを基に新たなメロディ生成を実現するシステムである [6]．The Continuator は，メロディを細かい素片に分割し，素片から素片への遷移を木構造のマルコフ連鎖によりモデル化することで，入力されたメロディに続くようなメロディとして適したものを学習データの中から探索するインタラクティブなシステムである．Kitahara らが提案した JamSketch では，ユーザが入力したメロディの概形に基づいて，既存のメロディを基に遺伝的アルゴリズムによって即興演奏のメロディをリアルタイムで生成している [7]．JamSketch では既存のメロディをそのまま使用しているわけではないが，既存のメロディをメロディ生成に活用している一つの事例と言える．JamSketch ではユーザの入力に応じて生成が行われるため，ここまでに挙げた他のメロディ生成手法に比べてユーザの担う役割が比較的大きいシステムである．

既存のコンテンツを再利用することによって新たなコンテンツを生成するという試みは，メロディに限らず様々なドメインで試行されているアプローチである．例えば，画像合成手法である PatchMatch は，画像補間の処理として，補間領域に対応する小さなパッチ領域を同一画像内から探索して組み合わせることで補間された新たな画像コンテンツの合成を実現している [8]．他にも，平井らによる音楽動画の自動生成手法では，既存の動画データベースから，音楽に合うような動画素片を探索してそれらを繋ぎ合わせることで新たな音楽動画の自動生成を実現している [9]．Nakano らによる DanceReProducer も既存の音楽動画を再利用することによって新たな音楽動画を自動生成するシステムとして提案されているが，ダメ出しインタフェースという機能が実装されており，自動生成結果の中でユーザが気に入らない生成結果を他の候補と置き換えることが可能となっている [10]．DanceReProducer の例のように，自動生成を行って終わりとするのではなくユーザが出力結果に積極的に関与する余地を残すことで，創作における人と機械との協調に繋がることが期待できる．

本研究では，メロディの制作行程に注目し，メロディ素片同士の接続コストを活用することで，入力中のメロディの後続メロディ候補として既存のメロディを必要に応じて提示するインタフェースを開発する．これにより，ユーザ主体のメロディ制作作業において，ユーザが行き詰まったときや他のアイデアが欲しいときなどを想定した支援をするインタフェースを目指す．

3. メロディ素片同士の接続コストの算出

本章では著者らが [1] にて提案したメロディ素片同士の接続コストの算出手法について記述する．手法の大筋については [1] から変更はないが，ネットワークの学習時に Early Stopping を導入して追加の評価実験を行ったことで精度の向上が確認できている．

3.1 前処理

3.1.1 データセット

接続コストの算出モデルの学習には，Web から収集された 176,581 曲分の MIDI ファイルにより構成されている The Lakh MIDI dataset[11] から抽出したメロディデータを使用する．このデータセットの中から，[12] で実施された前処理に則って 10,736 曲分のメロディのみを抽出する．この前処理では，取得したメロディの調を推定し，必要に応じて移調することでデータセット内のメロディの調の統一を図っている．

3.1.2 メロディデータの表現方法

メロディは音高と音価との組み合わせからなる音符によって構成されるものであり，音符列によって表現される．抽出して調を統一したメロディデータを，接続コスト算出モデルで処理しやすいようにテキスト形式の音符列のデータに変換する．

著者らが [1] で提案した接続コストでは，小節レベルでの接続の妥当性と音符（ノート）レベルでの接続の妥当性の二つの要素に注目して，それらを統合するようなモデルを構築している．小節レベルの接続の妥当性を評価するモデルのデータ表現に関しては，1 小節の長さを 4 分音符 4 つ分（4 分の 4 拍子）と仮定し，メロディデータの先頭から 4 分音符 4 つ分の長さ毎にメロディをセグメンテーションする．さらに 1 小節分の音符列は，16 分音符毎に分割して，音名のみを記録した One-hot vector で表現する．この One-hot vector は，12 音の音名と休符を含んだ 13 次元のベクトルとなっており，これが 1 小節あたり 16 個あるため，13×16 の行列となる．小節レベルの接続の妥当性を評価するモデルのデータ表現については，オクターブの情報を無視して音名のみを利用している．

ノートレベルの接続の妥当性を評価する際のメロディのデータ表現については，オクターブ情報や音符の長さについても重要となるため，音名だけでなくオクターブ情報，さらに音の長さも考慮した音符（音高＋音価）によるデータ表現を採用している．メロディ内に登場する多種多様な音符を扱うために，メロディを文章と見立てて音符を単語としてボキャブラリー化して，データセット内の全音符に ID を付与する．特定の音符から次の音符に遷移する情報を ID から ID への遷移として表現し，候補となる音符の遷移パターンがデータセット内に多く含まれていれば含まれているほど，妥当な遷移であると判定できるようにしている．

3.2 小節レベルの接続の妥当性

3.1.2 で述べた小節単位の One-hot vector を入力とした小節レベルの接続の妥当性を判定するモデルを構築する．実装するのは，任意の 2 つの小節のメロディが入力されたときにその 2 つのメロディが元々繋がっているメロディ同士なのかを判定するような Deep Learning モデルである．

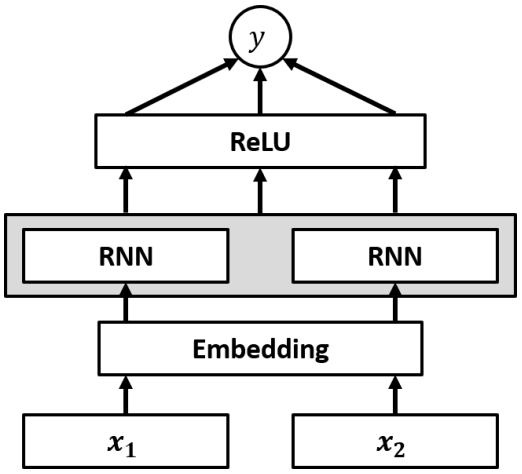

図 1 小節レベルの接続妥当性を判定するネットワークの枠組み

メロディに関するモデルを構築する際には，時系列情報を考慮可能な RNN をベースとしたモデルが望ましいため，複数種類の RNN モデルを実装し，判定精度を比較して最終的に採用するモデルを決定する．

構築した RNN ベースのモデルのネットワーク図を**図 1** に示す．ここで，RNN 層については，複数種類の RNN モデルについて比較を行っており，モデルによって図 1 における RNN 層の箇所が LSTM や BiLSTM などに置き換わる．入力データは，16 分音符単位で表現された 12 音+休符の計 13 次元の One-hot vector（13×16 の行列データ）を 2 小節分（13×32）である．構築する RNN モデルの枠組みにおけるデータの流れは以下のようなものである．

(1) 入力された 1 小節目のデータ x_1 を Embedding した後に RNN 層に入力し，最後の隠れ層の出力 h_1 を得る
(2) 2 小節目のデータ x_2 についても同じく Embedding した後に RNN 層に入力し，最後の隠れ層の出力 h_2 を得る
(3) h_1 と h_2 を結合し，全結合層（2〜3 層）に通して二値分類の出力 $\{0, 1\}$ を得る
(4) 入力された 2 つの小節が実際に繋がっていた小節の組み合わせである場合には 1，そうでない場合には 0 を教師データとしてネットワークを学習する

上記の枠組みで，3.1 のデータセットのメロディを対象として小節同士の接続判定を行うような RNN モデルの学習を行う．また，(1)，(2) の RNN 層を単純な RNN から，LSTM，2 層の LSTM，BiLSTM などに変えながら判定精度がどのように変化するかについても実験を行う．

学習にあたって，データセットの 10,736 曲分のメロディデータを 9:1 の割合で学習データとテストデータに分割し，データセットのメロディ 1 曲毎に，元のメロディのままの正例のデータと偶数番目の小節をすべてランダムなメロディに置き換えた負例（前後の小節同士が接続関係にない例）のデータを用意した．ランダムなメロディへの置

き換えにあたっては，完全にランダムなメロディを使用するのではなく，データセット内の別の曲の別の小節のメロディと置き換えている．

いずれの RNN モデルにおいても，Embedding の次元数は 25，RNN の隠れ層の出力の次元数は 50 とした．BiLSTM の場合のみ，隠れ層からの出力の次元数が 2 倍となる都合上，最後の全結合層の数を一層分増やしている．どのモデルでも損失関数を二値交差エントロピーとして，Early Stopping を導入して損失が収束するまで学習を行った．

以上のように構築したモデルに対して，任意の 2 小節のメロディデータを入力すると，$[0, 1]$ のスケールで表現される小節同士の接続の妥当性が求められる．これを小節レベルの接続スコア S_m とする．

3.3 音符レベルの接続の妥当性

3.2 節で述べた小節レベルの接続スコアのみでは，小節という粒度での接続の自然さを考慮できるものの，小節境界における音符の接続の滑らかさまでは考慮しきれない．また，個々の音符に関する情報は一部失われてしまっているため，音符の接続についても別途考慮したい．そこで，小節境界における音符の接続の妥当性を評価する尺度を導入する．

具体的には，学習データに含まれる全てのメロディを小節単位に分割した際の，前の小節の最後の音符と後の小節の最初の音符が何であるのかを基に，小節をまたぐ際の音符遷移としてどのような遷移であれば尤もらしい接続であるかを記録する．3.1.2 で述べた音符の ID 表現を基に，小節境界において，どの ID からどの ID に遷移するのかをカウントし，音符の種類毎に，遷移の最頻値で除算し，遷移の妥当性に対応する $[0, 1]$ スケールの値を得る．これを，ノートレベルの接続スコア S_n とする．この尺度は，音符の遷移確率に対応するような尺度となっているが，スケールが最大値で正規化されているため，最もありがちな遷移の場合にスコアの値は 1 となるように設計されている．学習データに含まれない音符の遷移の場合にはスコアは 0 となるように設計されており，接続コストを算出する上で，学習データに事例がないような音符同士の接続が起こりづらくなるようにしている．

3.4 メロディの接続コスト

3.2 節および 3.3 節の手順で算出した小節レベルの接続スコア S_m とノートレベルの接続スコア S_n を組み合わせることでメロディの接続コストを算出する．S_m と S_n はともに $[0, 1]$ のスケールで，スコアが高いほど接続が妥当であることを示している．そこで，重み係数 α を用いてメロディの接続コスト C を，

$$C = 1 - \{\alpha S_m + (1 - \alpha) S_n\} \tag{1}$$

と定義した．ここで，α は，$[0, 1]$ の重み係数で，値が大きいほど小節レベルの接続を重視し，小さいほどノートレベルの接続の方を重視するパラメータとなっている．本稿では，小節レベルの接続を重視することを目的として，$\alpha = 0.75$ としている．

表 1 小節レベルのスコアによるメロディ接続箇所の判定精度，学習エポック数，学習の所要時間の比較

モデル	RNN	LSTM	二層 LSTM	BiLSTM
正解率 [%]	82.82	84.08	84.34	**84.38**
学習エポック数	459	533	459	362
所要時間 [秒]	38,984	52,434	59,312	50,300

3.5 メロディの接続コストに関する評価

算出したメロディの接続コストを評価するために，入力した 2 小節分のメロディが接続されているかどうかを判定するタスクとしてメロディ接続箇所の判定精度を算出して比較を行う．本節では，まず小節レベルの接続スコア算出時の RNN モデルの比較を行った後に，小節レベルの接続スコア，ノートレベルの接続スコア，両者を考慮した接続コストによる判定精度の比較を行う．

3.5.1 RNN モデルの比較

まず，小節レベルの接続スコアの算出時に，どの RNN モデルを採用すべきかを検証するために，各 RNN モデルを用いた際のメロディ接続箇所の判定精度を比較する．データセットの 90% のデータ（小節数ではなく全曲数の 90% ）をモデルの学習に用い，残りの 10% のデータについて，2 小節分のメロディが元々繋がっていたものなのか別のメロディが接続されたものなのかを 3.2 節に記述したモデルにより判定する．RNN 層として，単純な RNN，LSTM，2 層の LSTM，BiLSTM を使った場合の判定精度（正解率）を**表 1** に示す．精度は，各モデルで出力された 0 から 1 の間のスコアを 0.5 で閾値処理し，閾値以上であれば元々繋がっていた小節同士であると判定しているものとした．学習には，Early Stopping を用い，学習が収束するまで試行しており，表 1 には，各モデルで学習したエポック数と学習完了までの所要時間も示している．

表 1 に示したように，RNN 層として BiLSTM を採用することで，最も高い 84.38% の精度で元々繋がっていた小節同士なのか，ランダムに接続された小節なのかを判定できた．ランダムな接続の中には，妥当な小節同士の接続がなされている箇所も存在することが考えられるため，この精度は必ずしも 100% に達するとは限らないものである．負例の中にも偶然自然な接続となった小節の組み合わせが含まれている可能性が十分に考えられるためである．ランダムな予測を行った場合の精度は 50% 前後となることから，本手法によって判定する小節レベルの接続スコアはある程度妥当な尺度として使用できるものであるといえる．

表 2　各スコア/コストによるメロディ接続箇所の判定精度の比較

モデル	小節レベルの接続スコア	ノートレベルの接続スコア	接続コスト
正解率 [%]	**84.38**	77.99	84.13

3.5.2　メロディ接続箇所の判定精度の比較

次に，ノートレベルの接続スコアによる判定精度，さらに両者を考慮した式 (1) の接続コストによる判定精度を算出し，比較する．ノートレベルの接続スコアについても，小節レベルのスコアと同じくデータセットの 90% のデータで音符遷移に関するデータを収集し，残りの 10% のデータで精度を求めている．ノートレベルの接続スコアに基づく接続の判定については，小節境界の 2 つの音符間の接続スコアが 0.1 以上だった場合に自然な接続であると判定するものとした．閾値を 0.1 とした理由は，単にその周辺の閾値における推定精度が高かったからである．

本稿で提案した接続コストに基づく接続箇所判定については，閾値を 0.5 として，接続コストが閾値以下だった場合に自然な接続であると判定し，データセットの 10% のテストデータを対象として判定精度を求めた．ここで，小節レベルの接続スコアについては，BiLSTM によるモデルを採用し，接続コストを算出する上での重み係数 α は 0.75 としている．**表 2** に，小節レベルの接続スコア，ノートレベルの接続スコア，接続コストに基づく接続箇所判定精度を比較した結果を示す．

表 2 に示したように，純粋に小節同士の接続判定を行うという観点では，小節レベルの接続スコアを用いた場合が最も高精度となる．これは，小節レベルの接続スコアが接続判定を行う際の損失を最小化するような学習によってモデル化されていることから，自然な結果であると言える．ノートレベルの接続スコアを考慮し，接続コストを導入した場合，メロディ同士の接続判定タスクの精度は下がってしまうが，それと引き換えにノートレベルの接続の妥当性も考慮できるため必ずしも効果がないとは言い切れない．前の小節の最後の音符と後の小節の最初の音符の遷移の自然さを考慮できることになるため，小節境界部の局所的な自然さが考慮できるという点で提案接続コストの導入にはメリットがあるといえる．また，接続コスト算出の際の重みを調整することで，ユーザがどこに重きを置いて接続コストを算出するかを決めることも可能であり，その調整については，今後インタフェースの実装により柔軟に行えるようにする予定である．

今回の比較は，各スコア，コストに対して閾値処理を行って評価したものである．そのため，閾値やパラメータの値を変えることで結果も容易に変わってしまう．実際に，ノートレベルの接続判定の閾値を 0.5 にすると，ノートレベルの判定精度は 67.32% にまで下がり，接続コスト算出時の α の値を 0.5 にすると，接続コストによる判定精度は 77.63% に下がる．データについては同じものを使用してはいるが，判定精度の値については完全に同じ条件で比較できているわけではないことに注意されたい．

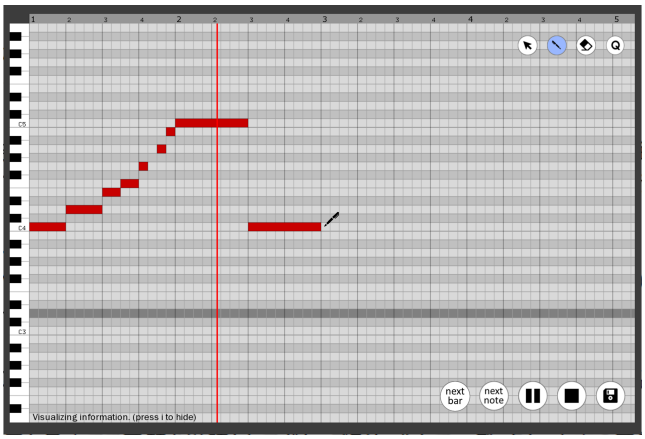

図 2　提案インタフェースを一般的なピアノロールとして使用した画面例

4.　接続コストに基づくメロディ打ち込み支援インタフェースの構築

3 章で説明したメロディ素片同士の接続コストを応用して，メロディ制作支援を目的とした打ち込み支援のインタフェースを構築する．一般的に，コンピュータを使ってメロディを入力する際にはピアノロールが用いられることが多い．そのため，本研究における打ち込み支援インタフェースもピアノロールを用いたメロディ制作のシチュエーションでの支援を想定する．

4.1　メロディ打ち込みインタフェースの基本構成

本提案インタフェースは一般的なピアノロールの上に追加の機能として実装する．**図 2** に，提案インタフェースを一般的なピアノロールとして使用した際の画面例を示す．本インタフェースでは，ペン型の入力ツールを使用し，ピアノロール上をドラッグ＆ドロップをすることによって音符の入力ができる．一般的なピアノロールと同様に，音符のオンセットタイミングや音符の長さを揃えるためのクオンタイズ（図中：Q ボタン）機能や入力した音符を削除する機能，入力したメロディを再生，一時停止，停止する機能などが実装されている．横軸（時間方向）のグリッドは 16 分音符を最小単位とし，4 分の 4 拍子を前提として 4 拍を 1 小節としてメロディを打ち込んでいくようなインタフェースとなっている．

図 2 に示したインタフェースは一般的なピアノロールと同様な機能が実装されているため，このインタフェース単体でメロディの打ち込み作業は一通り実現できるものとなっている．ただし，一般的にピアノロールは DAW ソフトの中で利用されることが多く，他のトラックの音と同時にメロディを再生させたり，MIDI コントローラなどでリ

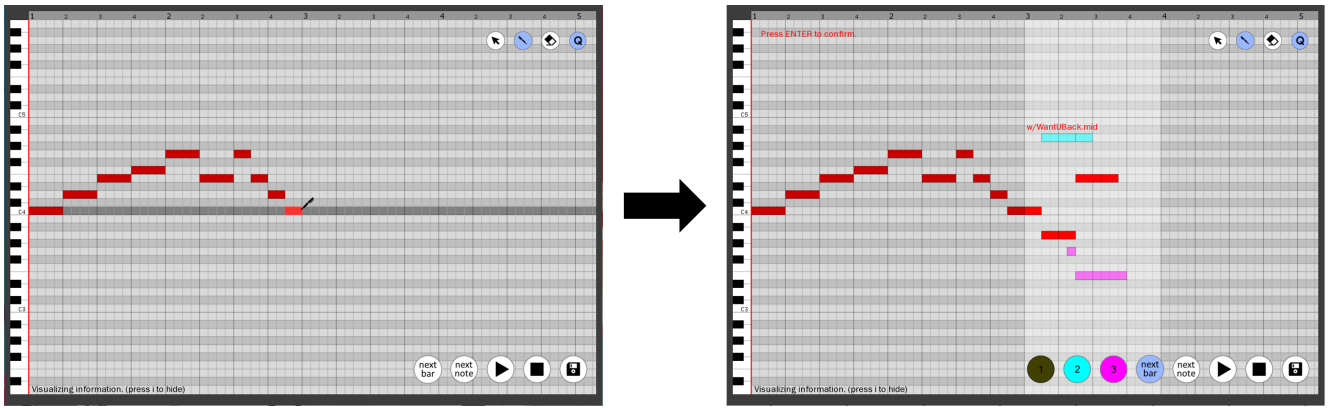

図 3 接続コストに基づく後続メロディ候補の提示

アルタイムに演奏しながらメロディを入力することなども可能であり，本インタフェースはその中の最も基本的な部分のみを押さえたものとなっている．

このインタフェースに，接続コストに基づいて情報提示を行うような機能を加えることでメロディの打ち込み支援を行う．

4.2 接続コストに基づく後続メロディ候補の提示

3 章で記述したメロディ素片同士の接続コストを用いて，ユーザがすでに打ち込んでいるメロディに最も繋がりやすい（つまり，最も接続コストが低い）既存のメロディ素片をデータベースから検索して提示するような後続メロディ候補の提示機能を提案する．ユーザが図 2 の一般的なピアノロールを使ってメロディの打ち込みを始め，1 小節分以上のメロディを打ち込んだ上で，インタフェース内の「next bar ボタン」を押すと，ユーザが入力したメロディの最後の 1 小節を入力とし，データベース内の任意の数の既存メロディ（1 小節分の長さ）との間で接続コストを算出していき，その中から接続コストが低かった上位 3 つのメロディを後続メロディ候補として提示する．接続コストに基づく後続メロディ候補の提示機能を使った様子を**図 3** に示す．図 3 左側がユーザが自力で打ち込んだメロディであり，図 3 右側が「next bar ボタン」を押した後の結果画面である．ピアノロールの背景が明るい部分が後続メロディ候補が提示されている小節にあたる．ユーザは提示された 3 つの後続メロディ候補を自身が入力したメロディに繋げて試聴し，もし気に入ったメロディがあればそれをそのまま自身の制作中のメロディの一部として採用することができる．

ここで，候補となったメロディがデータベース内のどの MIDI ファイルから抜き出されたものなのかはピアノロール上に表示され，実際に提示された候補を採用した場合には，該当小節のメロディ上に表示され続ける．提示された候補から一つのメロディを採用した後には，一般的なピアノロールと同様に再度編集することができる．そのため，仮に候補メロディの一部が気に入らない場合には元のテイストを残したまま，自身のイメージに合うように修正を加えることができる．この際，ユーザは必ず提示されたメロディを自身の制作中のメロディに採用しなくてはいけないわけではないため，あくまでも制作に行き詰まった時の参考程度に使うことができる．むしろ，この機能自体を必ずしも使わなければいけない機能とはしていないため，ユーザが必要としたときのみの支援という立ち位置のものとなっている．

このように後続のメロディを提示（生成）するような自動作曲手法はこれまでにも提案されてきているが，本研究に関して特筆すべき点は，提示されるメロディが既存のメロディであるということである．つまり，自動生成の結果ではなく人手で作られたメロディが候補として提示される．この機能を使えば，自身が作ったメロディ以外のデータもデータベースに加えておき，数多のメロディの中から現在のメロディに繋がりやすいメロディを探索し，自身のメロディの一部として採用することもできる．その際に元のメロディがどこからきたものなのかを表示できているため，N 次創作のような他者の創作物を基に創作の連鎖を繋げていくような制作スタイルも可能となる．

「next bar ボタン」を押すと，入力メロディを基に，接続コスト算出モデルを使ってデータベース内のメロディとの間で推論が行われるため，探索対象のメロディが多ければ多いほど候補が提示されるまでの待ち時間が長くなる．現状では，メモリ 32.0GB, Intel Core i9-1088H 2.40GHz, NVIDIA GeFource RTX 2060 のマシンで，10,000 小節の候補を対象として探索を行う場合の所要時間は 20 秒程度である．このような機能を使用する上では，より短時間でフィードバックが得られれば，より多くの試行を試すことができるため待ち時間は短いほど好ましい．この待ち時間は，実装の方法などによりある程度短縮可能であり，その短縮は今後の課題である．

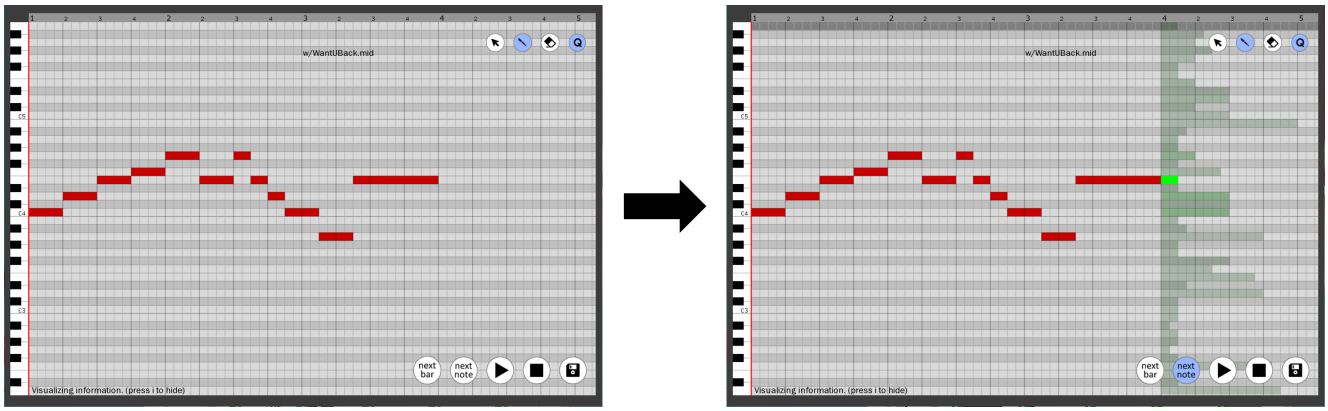

図 4 後続音符候補の可視化

4.3 後続音符候補の可視化

メロディの接続コストの算出にあたって，音符単位の接続の妥当性を考慮しており，それをメロディの打ち込みの際に可視化することで，次に入力する音符として適したものは何かを調べることができる．インタフェース内の「next note ボタン」を押すと，ユーザが入力した最後の音符の後にどのような音符がどれだけ出現しやすいか，つまり後続音符候補を可視化することができる．**図 4** に，後続音符候補を可視化した様子を示す．

図 4 のように，ユーザが最後に入力した音符の次に来る音符としてどの音高でどれだけの長さの音を打ち込むと良いかが可視化される．これは，元々のメロディの接続コスト算出モデルの学習に使ったデータセット内のメロディに関する音符遷移の頻度を可視化したものである．そのため，単によくある音符遷移がより濃い緑色で表示されるというものになっている．多くの既存メロディで同じ音高の音符に遷移するケースが多いため，この機能を使うと，同じ音高の音符が最も有力な候補として提示されがちとなる．これはあくまでも情報の可視化をしているだけであり，ユーザがこの情報にしたがって次の音符を入力する必要はない．この情報を参考にして，ありがちでない音符へ遷移させたりすることもでき，基本的にはユーザが主体としてメロディを打ち込む際の参考情報の提示に過ぎない．

このように，本提案インタフェースでは，小節単位での後続メロディ候補の提示と，音符単位での後続音符候補の提示という二つの機能によってメロディの打ち込みを支援するものとなっている．候補の提示はあくまでも任意のものとなっており，どちらの機能も，ユーザが必要とした場面でのみ活用できるようにしている．

5. ユーザスタディ

提案した接続コストに基づくメロディ打ち込み支援インタフェースの有効性をユーザスタディにより評価する．

5.1 条件

ユーザスタディは，4 名のユーザに本インタフェースを使ってもらい，候補の提示に関する機能を使った場合と使わなかった場合でそれぞれ 3 回ずつ，計 6 回メロディの打ち込みを行い，各試行について有効性評価のためのアンケート項目に回答してもらう形式で行った．ユーザスタディ開始前には著者が実際にインタフェースを操作する様子を参加者に見てもらい，基本的な操作方法を覚えた上で試行してもらった．また，試行中に操作方法がわからない場合に見ることができる各ボタンの説明が書かれた書類を用意した．本ユーザスタディの参加者 4 名に事前に聞いた音楽経験，メロディの打ち込み経験，および作曲経験は以下の通りであった．

- ユーザ A：音楽経験 1 年未満，DTM/打ち込み経験なし，鼻歌程度の作曲経験あり
- ユーザ B：音楽経験なし，DTM/打ち込み経験なし，作曲経験なし
- ユーザ C：音楽経験 10 年以上，DTM/打ち込み使用経験あり，鼻歌程度の作曲経験あり
- ユーザ D：音楽経験なし，DTM/打ち込み経験なし，作曲経験なし

打ち込んでもらうメロディは 2 小節以上 4 小節以下の短いメロディとして，複数回の試行でどのような評価結果が得られるかを調査した．候補提示機能を使用する試行においては，4 小節以内に必ず接続コストに基づく後続メロディ候補の提示機能を使用してもらうという制約を課し，後続音符候補の提示機能については必要に応じて任意で使用してもらうこととした．各ユーザ 6 回の試行において，候補提示機能は偶数回目の試行で使用してもらい，使用する試行と使用しない試行を交互に繰り返すようにした．候補提示機能において使用するデータベースは，3 章に記述した接続コスト算出モデルの学習に使用しなかったテストデータからランダムに抽出した 10,000 小節分のメロディであり．この条件で機能を 1 回使用すると約 20 秒の待ち時間が生じる．

インタフェースに関する評価に加え，全参加者がすべての試行が終了した後に，各参加者が制作した 6 個のメロディを制作者以外の 3 名の参加者に聞いてもらい，メロディそのものの評価も合わせて行った．

5.2 評価項目

参加者には，候補提示機能使用の有無で 3 回ずつ，計 6 回のメロディ制作を行ってもらった．その過程で，メロディが 1 つ完成するたびに以下に挙げる 4 つの評価項目に回答してもらった．

(1) 思い通りのメロディが制作できた
(2) 思いがけないメロディが制作できた
(3) 満足のいくメロディが制作できた
(4) 簡単にメロディが制作できた

上記 4 つの評価項目については，すべて，「１：当てはまらない」，「２：どちらかというと当てはまる」，「３：どちらかというと当てはまらない」，「４：当てはまる」の 4 段階で試行毎に評価してもらった．

上記に加え，6 回の試行と評価項目への回答が終わった後に全体的な評価を行うために，後続メロディ候補の提示機能と後続音符候補の提示機能のそれぞれについて，「１：有効でなかった」，「２：どちらかというと有効でなかった」，「３：どちらかというと有効であった」，「４：有効であった」の 4 段階で評価してもらった．さらに，後続音符候補の提示機能については，「１：ほとんど使わなかった」，「２：あまり使わなかった」，「３：何度か使った」，「４：多用した」の 4 通りで各ユーザが使用を任意とした後続音符候補の提示機能をどの程度使ったかを回答してもらった．最後に，全体を通しての感想やコメントを自由記述形式で挙げてもらった．

ユーザの各試行の様子はすべてスクリーンキャプチャで記録しておき，各試行に要した時間も記録しており，機能の有無でメロディ制作に要する時間がどう変わるかについても評価した．

また，ユーザスタディ参加者が制作したメロディについての評価は，各メロディに対して「１：良いメロディではない」，「２：どちらかというと良いメロディではない」，「３：どちらかというと良いメロディである」，「４：良いメロディである」の 4 段階で評価してもらった．評価を行ったのは残りの 2 名のユーザスタディ参加者で，各メロディがどのように作られたものであるかは隠した状態で聴いてもらった．

すべての評価項目は 4 段階評価で，評価値が高いほど良い評価であることを示している．中間の値は 2.5 で，2.5 より高い評価結果が得られれば好意的な結果が得られたことを示す．

表 3 メロディ制作に関する評価結果

	評価項目			
	（1）	（2）	（3）	（4）
候補提示機能使用なし	2.17	2.25	2.08	2.42
候補提示機能使用あり	3.25	3.58	3.17	3.75

表 4 各機能の有効性の評価結果

	評価値の平均
後続メロディ候補の提示機能の有効性	3.75
後続音符候補の提示機能の有効性	2.67
後続音符候補の提示機能の使用頻度	1.5

5.3 評価結果

ユーザスタディの結果を**表 3**，**表 4**，**表 5** に示す．表 3 はメロディ制作に関する試行毎に行われた評価結果を，表 4 は全体的な試行を踏まえての候補提示機能の有効性に関する評価結果を，表 5 は各参加者が制作した 6 つのメロディに対する他の 3 名の参加者による評価結果を示している．表 3 の評価値は候補提示機能の有無に分けて評価値の平均を算出して示している．表 4 の評価結果についても各項目の評価値の平均を示しており，表 5 にはすべての評価値と機能使用有無に分けた際の平均を示している．評価値はすべて 1 から 4 までの数値をとり，値が高いほど高い評価が得られていることを示す．

表 3 の結果によると，「（１）思い通りのメロディが制作できた」，「（２）思いがけないメロディが制作できた」，「（３）満足のいくメロディが制作できた」，「（４）簡単にメロディが制作できた」，のすべての評価項目について後続メロディ候補提示機能を使用しなかった場合に比べ，使用した場合の方がより高い評価結果が得られた．意外な結果として，後続メロディ候補提示機能を使った方が，すべて自身で打ち込む場合よりもより思い通りのメロディが制作できたという評価が得られた．この結果から，提示された候補がユーザのイメージするメロディに近いものとなっていると考えられる．実際，初めてメロディを制作する参加者は，音楽的に聴きやすいメロディを作ることに苦心している様子が伺えたが，候補提示機能によって提示されるメロディは音楽的に成立している実際のメロディであるため，それがユーザにとって痒いところに手が届くような支援となったのではないかと考えられる．

表 4 の結果，後続メロディ候補の提示機能の有効性に関する評価値の平均は 3.75 で，4 名の参加者全員が有効だと回答した．後続音符候補の提示機能の評価値の平均は 2.67 であり，後続音符候補の提示機能はそれほど有効なものとは評価されなかった．使用頻度についても，4 名中 3 名の参加者が「ほとんど使わなかった」と回答しており，メロディ制作支援には繋がらない結果となった．

表 5 の実際に制作された 4 名 ×6 個のメロディに対する評価結果によると，機能を使用して制作されたメロディの

表 5 各ユーザが制作したメロディに対する評価結果

制作者	評価値					
	候補提示機能使用なし			候補提示機能使用あり		
ユーザ A	3.33	2.33	2.00	3.67	3.33	2.67
ユーザ B	1.67	2.33	2.33	2.33	2.33	3.33
ユーザ C	3.00	3.33	3.00	3.00	3.67	3.33
ユーザ D	1.67	1.67	3.33	3.00	2.67	3.00
平均	1.88			2.27		

表 6 制作に要した時間についての評価

ユーザ	候補提示機能使用なしの試行の所要時間 [秒]			候補提示機能使用ありの試行の所要時間 [秒]		
ユーザ A	681	184	233	450	663	551
ユーザ B	138	134	244	208	261	231
ユーザ C	166	124	193	181	222	216
ユーザ D	218	191	181	297	238	368
平均	223.9			322.2		

方が，機能を使用せずに制作されたメロディよりも全体的に高い評価が得られた．制作するメロディの質にはユーザ毎のばらつきがあり，例えば最も音楽経験があるユーザ C によるメロディはすべて 3 以上の評価値を得られている．それを踏まえてユーザ毎に機能の使用有無で評価値がどのように変化しているかに注目すると，どのユーザも機能の使用によってより高い評価値が得られるメロディ制作ができている．この結果から，本インタフェースの支援により制作されるメロディの質も向上していることが確認できる．

試行終了後に自由記述欄を通じて得られた意見には以下のようなものがあった．

- 音符の推薦機能は同じ音ばかりを推薦していた
- 提案されたメロディが気に入らなかったときには，他のメロディを提案してほしい
- 自分では思いつかなかったメロディを提案してくれるため，作った曲がワンパターンにならなくて良かった
- 回数を重ねるごとにメロディとして成立するような感覚があり，自分がメロディを成立したものとして作れると，提示されるメロディも良くなるように感じる

上記の意見からも，本インタフェースがある程度有効な支援となっていたことが確認できる．一方で，後続音符の提示機能については改善の余地があると考えられる．得られた意見を基に今後のインタフェースの改善に繋げていく予定である．

最後に，機能の使用有無によって，メロディの制作にかかった時間がどのように変化したかについて評価する．**表 6** に，全ユーザの全 6 回のメロディ制作の所要時間を示す．候補提示機能使用ありの場合は，候補提示機能の待ち時間（1 回の使用で約 20 秒）も所要時間に含めている．経験が少ないユーザほど候補提示機能を何度も使用する傾向にあり，そのたびに待ち時間が生じるため全体的な所要時間は長くなってしまった．また，待ち時間だけでなく，提示される 3 つのメロディ候補を聴き比べるためにも再生時間の分だけ時間がかかるため，本インタフェースは，現状ではメロディ制作の効率化には繋がらないと考えられる．この点については，今後システムの高速化を図るとともに，よりユーザのニーズに合った候補を提示できるようにすることで解決していきたいと考えている．

本ユーザスタディを通して，試行を重ねるごとに，ユーザがメロディ制作の感覚を掴んでいく傾向にあり，より評価の高いメロディをより短い時間で作ることができるようになっていた．同様に，どのようなメロディを入力すればよりよい候補が提示されるかといった感覚を掴んでいったユーザもいた．従来の音楽制作ツールと同様，何度も使用して慣れることでより便利に使いこなすことができるツールになっていることが期待できる．

6. 議論

本章では，4.2 節で紹介した本研究の特徴である既存のメロディの再利用を可能とするインタフェースの可能性と懸念点について議論する．

1 章で述べたように，短いメロディのフレーズを鼻歌感覚で打ち込む行為を，より長い 1 曲分のメロディ制作に繋げられれば，誰もがより手軽に音楽制作ができるようになるのではないかという考えの下，本インタフェースはそのような支援を行うことを想定して開発されている．実際に，5 章のユーザスタディの結果からも，メロディ制作におけるある程度の有効性が示されている．

一方で，既存のメロディを再利用するという行為は，他者の創作物を手軽に再利用できるような創作を促してしまう．創作行為は，少なからず他者の作品からインスパイアを得て行われるもので，例えば音楽においては，コード進行についての著作権というものはなく，他者が作った曲のコード進行を基に作曲を始めるという行為は作曲のスタイルとしても認められている．他にも，サンプリング音楽では，他者が作った音楽や音の一部を自身の楽曲の一部として取り入れるようなスタイルが確立されている．一方でメロディの扱いについては難しく，過去の楽曲のメロディを自身の楽曲の一部にオマージュするような行為は珍しくないものの，そのまま利用するという行為は認められない．しかし，1 小節といった短い単位で見たときに，他の楽曲とメロディが一致するような楽曲の事例は枚挙にいとまがなく，本インタフェースはそのような制作を意識して行うかことはどうなのかという点が議論の焦点となる．

本インタフェースでは，既存メロディを基に後続メロディ候補を提示する機能を使う場合にはピアノロール上に元の楽曲のファイル名が表示される仕様となっている．これを基に，最終的に制作された楽曲に再利用した楽曲の情報をクレジットすることができる余地を与えており，N 次創作文化で見られるような引用を含む作品として制作する

ことを可能としている．しかし，本インタフェースの場合はメロディをそのまま再利用するだけでなく，自身のメロディに合わせて再編集することができるため，もしも取り入れたメロディが原形をとどめていない場合には，そのメロディがどのように扱われるべきかについては難しい問題である．

仮に，他者の作品をデータベースに利用しなくても，自身でたくさんの短いフレーズを蓄積させておくことで文字入力における予測変換の要領で，自身が過去に作ったショートフレーズの引き出しから便利にメロディを取り出して利用することが可能である．この場合，自身で作った素材を再利用する作業となるので，権利の問題が生じることはない．このような使い方をしても十分に有用なインタフェースとなるのではないかと期待している．

本インタフェースで実現しうるより新しい音楽制作のビジョンとして，大量の作者による音楽の共作が考えられる．Twitter に投稿された短文がリツイートされ，スレッド形式で何度も再利用されるような要領で，多くのユーザが投稿した短いメロディフレーズが再利用を重ねられ，たくさんのフレーズの組み合わせで一曲の楽曲が完成させられるようなシステムができれば，誰かの何気ない鼻歌がプロのアーティストの新曲の一部に取り入れられるといった未来も考えられる．実際に，Splice などのサウンドライブラリでは，大量の短い音素材が公開されており，世界中のクリエータが作る作品の一部として他者が作った素材が活用されている．ファンが応募した言葉を集めてアーティストが作詞することで一曲の楽曲が完成する事例があるように，メロディのようなより音楽の中身に踏み入ったような要素においてもそのような制作スタイルが生まれることも可能であると考えられ，本インタフェースはその可能性に近づくための一つの実装例である．

7. まとめ

本稿では，メロディ素片間の接続コストに基づいて後続メロディ候補を提示することによって，メロディの打ち込みを支援するインタフェースを提案した．また，ユーザスタディを実施して本提案インタフェースの有効性や実際にユーザにより制作されたメロディの評価を行い，その有効性を確認した．

今後の課題として，機能を使った際のフィードバックの高速化が挙げられる．後続メロディ候補の提示を行うために，入力メロディとデータセット内のメロディとの間で接続コストを算出するための推論をデータセット内の探索対象の数だけ実施しなければいけない．そのため，現状では 10,000 小節文の探索候補との間で接続コストを算出するために 20 秒ほどの待ち時間が生じてしまう．Bretan らの素片選択型のメロディ生成 [4] では，最初にメロディ同士の意味的な関連性を用いて探索対象の絞り込みを行っているが，高速化のためにはそのような前処理も重要であると考えられる．今後，本インタフェースの完成度を上げてより実用的なものとして，実際の DAW ソフトなどでの使用に耐えうるものとしたい．

謝辞　本研究は JSPS 科研費 JP 19K20301 の助成を受けたものである．

参考文献

[1] 平井辰典 and 澤田隼: LSTM を用いたメロディ素片間の接続コストの算出, **情報処理学会研究報告音楽情報科学（*MUS*）**, Vol. 2021-MUS-131, No. 41, pp. 1–7 (2021).
[2] Huang, C.-Z. A., Vaswani, A., Uszkoreit, J., Shazeer, N., Hawthorne, C., Dai, A. M., Hoffman, M. D. and Eck, D.: Music Transformer: Generating Music with Long-Term Structure, *International Conference on Learning Representations* (2018).
[3] Roberts, A., Engel, J., Raffel, C., Hawthorne, C. and Eck, D.: A hierarchical latent vector model for learning long-term structure in music, *Proceedings of the 35th International Conference on Machine Learning*, pp. 4364–4373 (2018).
[4] Bretan, M., Weinberg, G. and Heck, L.: A unit selection methodology for music generation using deep neural networks, *Proceedings of the International Conference on Computational Creativity 2017* (2017).
[5] Cope, D.: One approach to musical intelligence, *IEEE Intelligent Systems and their Applications*, Vol. 14, No. 3, pp. 21–25 (1999).
[6] Pachet, F.: The continuator: Musical interaction with style, *Journal of New Music Research*, Vol. 32, No. 3, pp. 333–341 (2003).
[7] Kitahara, T., Giraldo, S. and Ramírez, R.: JamSketch: Improvisation Support System with GA-Based Melody Creation from User's Drawing, *International Symposium on Computer Music Multidisciplinary Research*, Springer, pp. 509–521 (2017).
[8] Barnes, C., Shechtman, E., Finkelstein, A. and Goldman, D. B.: PatchMatch: A randomized correspondence algorithm for structural image editing, *ACM Transactions on Graphics*, Vol. 28, No. 3 (2009).
[9] 平井辰典, 大矢隼士 and 森島繁生: 既存音楽動画の再利用による音楽に合った動画の自動生成システム, **情報処理学会論文誌**, Vol. 54, No. 4, pp. 1254–1262 (2013).
[10] Nakano, T., Murofushi, S., Goto, M. and Morishima, S.: DanceReProducer: An Automatic Mashup Music Video Generation System by Reusing Dance Video Clips on the Web, *Proceedings of the 8th Sound and Music Computing Conference (SMC 2011)*, pp. 183–189 (2011).
[11] Raffel, C.: *Learning-based Methods for Comparing Sequences, with Applications to Audio-to-midi Alignment and Matching*, PhD Thesis, Columbia University (2016).
[12] Hirai, T. and Sawada, S.: Melody2Vec Distributed Representations of Melodic Phrases based on Melody Segmentation, *Journal of Information Processing*, Vol. 27, pp. 278–286 (2019).