情報処理学会論文誌 Vol.54 No.4 1338–1349 (Apr. 2013)

任意箇所への弾き直し・弾き飛ばしを含む演奏に追従可能な楽譜追跡と自動伴奏

中村 栄太[1,a)] 武田 晴登[2] 山本 龍一[3] 齋藤 康之[4] 酒向 慎司[3] 嵯峨山 茂樹[1]

受付日 2012年7月4日, 採録日 2013年1月11日

概要：人間の演奏に対して，楽譜に基づき，自動的に同期して伴奏することは自動伴奏と呼ばれる．これは，合奏曲の練習や生演奏と電子音楽の融合に応用可能な技術であり，その効用は大きい．特に，練習時においては，弾き誤りや弾き直し，弾き飛ばしを含む演奏が起こりうるが，従来の自動伴奏はそのような演奏に対して追従が困難であるか，または特定箇所への弾き飛ばしのみ対応している．本稿では，任意箇所への弾き飛ばしを含む演奏に対しても素早く追従可能な自動伴奏を議論する．まず，隠れマルコフモデルを用いて演奏の生成過程をモデル化し，その確率的逆問題を解く形で，楽譜追跡アルゴリズムを設計する．そして，この際に起こる処理時間増大の問題を解決するために，最尤状態推定の高速アルゴリズムである $\alpha\beta\gamma$ 法を考案する．この楽譜追跡手法の有効性を確認するために，実際の演奏を用いて評価を行う．さらに，これをもとに自動伴奏システム Eurydice を構築し動作を確認する．

キーワード：自動伴奏，楽譜追跡，隠れマルコフモデル，高速 Viterbi 探索，テンポ推定

Score Following Handling Performances with Arbitrary Repeats and Skips and Automatic Accompaniment

Eita Nakamura[1,a)] Haruto Takeda[2] Ryuichi Yamamoto[3] Yasuyuki Saito[4]
Shinji Sako[3] Shigeki Sagayama[1]

Received: July 4, 2012, Accepted: January 11, 2013

Abstract: Automated performance of an accompaniment score, synchronized to human performances, is called automatic accompaniment. It is applicable for practicings of ensemble music and live electronics, and its utility is large. Particularly in music practicings, performances with errors, repeats and skips are expected. However, conventional automatic accompaniment systems have difficulties in following those performances, or may handle only skips to specific locations in scores. In this paper, we study a robust automatic accompaniment capable of following performances with skips to arbitrary locations. We construct a hidden Markov model which describes the generation process of performances and propose a score following algorithm, which solves the probabilistic inverse problem of the model. To solve the problem of increasing processing time, a fast algorithm for estimating most likely states is proposed. To confirm the validity of the proposed score following algorithm, an evaluation test for the algorithm with human-played performances is performed. Furthermore, we construct an automatic accompaniment system *Eurydice* and confirm its operation.

Keywords: automatic accompaniment, score following, hidden Markov model, fast Viterbi search algorithm, tempo estimation.

[1] 東京大学
The University of Tokyo, Bunkyo, Tokyo 113–0033, Japan
[2] ソニー株式会社
Sony Corporation, Shinagawa, Tokyo 141–0001, Japan
[3] 名古屋工業大学
Nagoya Institute of Technology, Nagoya, Aichi 466–8555, Japan
[4] 木更津工業高等専門学校
Kisarazu National College of Technology, Kisarazu, Chiba 292–0041, Japan
[a)] enakamura@hil.t.u-tokyo.ac.jp

1. はじめに

人間の演奏に対して，機械が与えられた楽譜に基づく伴奏を自動的にリアルタイムで同期して演奏することは，自動伴奏と呼ばれる．人間の演奏においては，テンポや強弱を含む多くの演奏要素における不確定性が存在し，演奏結果はそのつど異なる．このような，楽譜から一意に予測できない演奏の不確定性に対して頑健に対応し，必要な要素はリアルタイムに取り入れて伴奏演奏に反映するという点が，自動伴奏の特徴である．このような技術は，奏者数を軽減した合奏を可能にするほか，合奏曲などの練習の際における効用も期待される．また，電子音楽やリアルタイムで録音，処理された音源を伴奏として用いることも考えられ，新しい作曲や演奏形態の可能性を与えることができる．このように，自動伴奏は広い応用を持ち，その効用は大きいと期待される．こうした背景から，自動伴奏はDannenberg [1] や Vercoe [2] の研究以来，活発に研究されている（この分野の発展については，たとえば文献 [3] を参照）．

実際の演奏においては，上に述べた意図された演奏要素の不確定性のほかに，音抜け，弾き間違いなどによる不確定性が生じる．さらに，練習における演奏では，弾き直しや大きく離れた箇所への弾き飛ばしが生じうる．また，演奏会においても，多義形式を持つ楽曲の演奏においてこうした状況が生じる．弾き直しや弾き飛ばしを考慮した従来の自動伴奏システム [4], [5] や連弾支援システム [6] では，小節や楽句など曲の構造や演奏者の傾向をシステムに入力する必要があるうえ，特定箇所以外への弾き直しや弾き飛ばしに対する追従性は保証されていない．これに対し，演奏楽譜が MIDI で与えられた場合のように楽曲の構造が特定できていない場合や，不特定多数の奏者による多様な演奏に対応するためには，任意箇所への弾き飛ばしに対応する必要がある．本稿では，弾き間違いのほかに任意箇所への弾き直しや弾き飛ばしを含む演奏に対して追従可能な自動伴奏の実現を目標とする．

自動伴奏には入力信号として，音響信号を用いるものと MIDI 信号を用いるものがある．音響信号による自動伴奏システムには，Raphael [7] や Cont [8] により考案されたものがある．これらのシステムは特に単旋律楽器による演奏に対しては優れた性能を示すものの，ピアノなどの多声楽器の演奏に対しては依然として大きな困難を有している [9], [10]．このような楽器の演奏に対しては MIDI 信号を入力とする自動伴奏システムが実際的である [9], [11]．また，鍵盤楽器の場合には電子ピアノの普及により直に MIDI 信号を得ることが可能な場合も多い．本稿では，特に需要の大きいと考えられるピアノなどの鍵盤楽器の演奏に対する自動伴奏システムの実現を目的とし，多声 MIDI 演奏を入力とする自動伴奏を議論する．

2. 自動伴奏問題の定式化

2.1 演奏の楽譜位置推定問題

自動伴奏を実現するためには，まず奏者による演奏が楽譜上のどこの位置にいるのかを推定する必要がある．通常の楽譜では，各音符の楽譜上での位置は音符間の相対的な長さをもとに指定されている．この方法による楽譜上の位置のことを，以下，**楽譜時刻**（単位は，たとえば，四分音符）と呼ぶことにする．また，楽譜時刻を表す変数として τ を用いる．これに対し，通常の時刻のことを**実時刻**（または単に，**時刻**）と呼び，変数 t で表す．実際の演奏では，それぞれの時刻の差（楽譜時間と実時間）が局所的に一定の比率を持って実現されるが，この比率 $r = \delta\tau/\delta t$ のことを**テンポ**と呼ぶ．また，各音符の発音開始点（Onset time）を**発音時刻**や**発音楽譜時刻**（または，単に**発音点**）と呼ぶ．

演奏される楽譜（以下，**演奏譜**）は，発音楽譜時刻を与えられた音符の列であるので $X = \{(\tau_i, c_i)\}_{i=1}^{I}$ と表される．ここで，I は演奏譜上の「音符」の個数，i はその添え字である．τ_i は i 番目の「音符」の発音楽譜時刻，c_i は「音符」の音価や音高，強弱などの各情報を表しているものとする．単旋律の楽譜の場合，c_i は単音を表すが，多声音楽の楽譜の場合，c_i は同楽譜時刻に発音される音全部の情報を表している．以下，c_i を和音と呼ぶこととする[*1]．

奏者による演奏は，演奏イベントの時系列 $S = \{(t_m, s_m)\}_{m=1}^{M}$ で表される．ここで，m は各演奏イベントを表す添え字であり，t_m は演奏イベントの到達時刻，s_m は演奏イベントの種類を表す．これを，**演奏イベント列**と呼ぶことにする．本稿では，MIDI イベントを演奏イベントとして用いるため，s_m は MIDI メッセージを表していると考えてよい．また，演奏イベント列およびその長さ M はオンラインで演奏イベントが到達するたびに更新していくものとする．

演奏譜 X と，演奏開始から現在までの演奏イベント列 S が与えられたときの，現在の演奏位置の推定問題は，X と S に対して，現在の楽譜位置 i_M を推定するという問題として定義される．ここで，i_M は M 番目の演奏イベントに対応する楽譜上の「音符」の番号を表している．その楽譜時刻は，τ_{i_M} である．順次更新されていく，推定楽譜位置の時系列 $Q = \{(t_m, i_m)\}_{m=1}^{M}$ を以下では，**楽譜位置情報列**と呼ぶことにする．

2.2 テンポ推定問題

自動伴奏を可能にするためには，一般的に演奏の楽譜位置だけでなく，演奏のテンポを同時に推定する必要がある．

[*1] ここでの「和音」は，一般的な意味で用いる和音とは異なることに注意されたい．単旋律の場合，ここでの和音は 1 音からなるものである．また，複数の旋律音が同時に鳴る場合もここでは和音と呼ぶことにする．

これは，伴奏の音符が演奏の音符よりも細かく，演奏の発音点間に複数の伴奏音符が存在する場合には，これらの音符の再生には発音時刻間で自動的に楽譜時刻を進める必要があるためである．また，伴奏の音源に何らかの原因により発音の遅延が起きてしまう場合には，あらかじめ伴奏の発音時刻を予想しておく必要があるが，この場合にもテンポ情報が必要となる．

実際には，テンポ情報が必要かどうかは，自動伴奏をする楽曲やその箇所，またシステムの使用状況に依存する．しかし，なるべく一般性を失わずに議論するため，本稿ではつねにテンポ情報の推定を行うことを考え，以下では，楽譜位置とテンポの両方を推定することを**楽譜追跡**（Score following）と呼ぶこととする．よって，楽譜追跡問題とは，現在までの演奏イベント列 S に対して，(t_M, i_M, r_M) を推定する問題として定義される．推定結果の時系列 $R = \{(t_m, i_m, r_m)\}_{m=1}^{M}$ のことを以下では，**演奏情報列**と呼ぶことにする．

ここで定式化した楽譜追跡は，自動伴奏を実現するために最小限の情報を与えるためのものといってよい．実際には，強弱やアーティキュレーション，その他の様々な演奏情報を伴奏再生に反映することが考えられ，状況に応じてこれらの演奏情報の抽出・推定を行う必要がある．

2.3 伴奏再生問題

伴奏の再生[*2]に最小限必要な情報は，各時刻での伴奏再生位置を示す楽譜時刻と伴奏再生のテンポである．よって伴奏再生は，随時，伴奏再生情報 (τ_ℓ, r_ℓ) を与えることにより行われる．時系列 $O = \{(t_\ell, \tau_\ell, r_\ell)\}_{\ell=1}^{L}$ を，**伴奏再生情報列**と呼ぶことにする．ここで，L は系列長であり，t_ℓ は ℓ 番目の情報を与えた時刻を表している．伴奏再生問題は，楽譜追跡の結果得られた演奏情報列 R から，伴奏再生情報を順次与える問題として定義される．

3. 演奏生成の確率モデル

3.1 演奏における不確定要素

1 章でも述べたとおり，楽譜追跡が難しい問題であるのは，同じ楽譜に基づく演奏でも毎回様々な差異があるからである．これは演奏の不確定性と呼ばれる．演奏の不確定性は意図されるものとされないものがあるが，代表的なものは次に列挙されるものである．

a) 発音時刻の微妙な変動

音楽的意図，奏者の技術的制約，楽器の物理的な制約などにより，各音の発音時刻には微小な変動がある．特に，和音は楽譜上では同時に発音される音の集まりであるが，実際の演奏においては和音構成音の発音時刻は厳密には，ずれている．また，その順序も不定である．

b) テンポ

テンポの絶対値や変化の仕方は奏者に任される場合が多い．また，その制御には奏者の技術や物理・身体的制約が関わっている．また，フェルマータも局所的なテンポ変動ととらえることができる．

c) 強弱やアーティキュレーション

強弱やアーティキュレーションは，記譜される場合も多いが，それ以外でも即興的に演奏に付されることが多い．また，いずれの場合もその度合いの詳細は奏者に委ねられている．

d) 装飾音などにおける，音数や音高

トリルや前打音などを含む装飾音には，音価や音数，また，ときには音高などに不確定性があるものが多い．

e) 弾き誤り

弾き誤りには演奏技術の制約に起因するものと奏者の読譜誤りによるものがある．その結果，音符の挿入，脱落，置換，音長誤りなどの誤りが生じる．また，慣習により正しい演奏法が唯一でない場合もあるが，これもここでは弾き誤りの一部と考えることにする．

f) 弾き直しや弾き飛ばし

特に，練習の際には同じ箇所の弾き直しや遠く離れた場所への弾き直し・弾き飛ばしが起こることがある．また，繰返し記号の無視や挿入も起こることがある．楽曲によっては，意図された弾き飛ばしが行われることがあるが，このような構造を持った形式は多義形式と呼ばれる．

これらの，楽譜からは一意に予測できない演奏の不確定性は，奏者の意図や即興性によるものとそうでないものを分別することは困難であるが，自動伴奏ではそれらを伴奏演奏に反映することが望ましい．それを可能とするためには，これらの不確定的要素を含む演奏に対して，頑健に楽譜追跡を行いながら，必要な演奏要素の情報を抽出・推定することが要求される．

3.2 演奏生成の確率モデルと楽譜追跡

前節で述べた，不確定的要素を含んだ演奏は確率的に生じると解釈できる．このように，演奏の生成過程を確率モデルとして記述できれば，楽譜追跡の問題は与えられた観測に対する確率的逆問題ととらえることができる．この方法は，人間による楽譜追跡の方法を模倣していると考えられる．本稿では，この理念に立脚して楽譜追跡を議論する．

確率モデルに基づいた楽譜追跡の利点として，モデルに含まれる各パラメータに物理的意味を与えることが容易である点がある．さらに，確率モデルに基づいたアルゴリズムでは，各パラメータの値を機械学習によって最適な値に調整可能であるという点が長所である．

[*2] 以下では，伴奏の「演奏」を「再生」と表現することがある．本稿においては，両者は同義語として用いられることを了解されたい．

3.3 HMM による演奏生成のモデル化

以下，演奏譜 X が与えられたときの，演奏イベント列 $S = \{(t_m, s_m)\}_{m=1}^{M}$ の生成を確率モデルとして記述する．

MIDI による演奏イベントでは，各音符の音高と強弱の情報のほかに発音開始時刻と発音終了時刻の情報が得られる．このうち，発音終了時刻の情報は実際に鍵盤などを離した時刻を表すものであるが，実際の演奏では，奏者の技術的制約やアーティキュレーションによって発音終了時刻は不確定の度合いが大きい．また，ピアノ演奏では，ペダルの効果によりこの発音終了時刻と実際の音の消音時刻は一致するとは限らない．また強弱についても，不確定の度合いが大きいうえ，通常の楽曲での演奏位置推定においては重要度は低いと考えられる．そこで，以下では MIDI イベントのうち発音開始イベントの音高情報のみを用いた楽譜追跡を議論する[*3]．よって以下では，演奏イベント (t_m, s_m) は MIDI の発音開始イベントを表し，t_m は発音時刻，s_m は音高を表すものとする．

人間による演奏の過程は，まず演奏行為の準備として楽譜上の演奏位置や強弱などが意図され，その次にその意図に基づいて演奏行為が実行される，と 2 段階に分けて行われると考えられる．この過程は，隠れ状態を持った確率モデルとして記述される．時刻 t_m において，楽譜位置 i_m にある和音 c_{i_m} の演奏が意図される確率を $p(i_m; t_m)$，c_{i_m} を弾く際に s_m が発音される確率を $p(s_m|i_m)$ とすると，音高 s_m が発音される確率 $p(s_m; t_m)$ は

$$p(s_m; t_m) = \sum_{i_m} p(s_m|i_m) p(i_m; t_m) \tag{1}$$

と表される．

現在時刻 t_M において演奏譜中の和音 c_{i_M} が演奏される確率は過去の演奏に依存していると考えられる．過去の演奏に対する依存関係がマルコフ的であると仮定すると，

$$\begin{aligned} &p(i_M; t_M | i_{M-1}, \cdots, i_1; t_{M-1}, \cdots, t_1) \\ &\quad = p(i_M; t_M | i_{M-1}; t_{M-1}) \end{aligned} \tag{2}$$

となる．状態遷移確率 $p(i_M; t_M | i_{M-1}; t_{M-1})$ は，2 音間の発音時刻間隔（Inter-onset interval, IOI），$\delta t_{M-1} = t_M - t_{M-1}$ に依存していると考えられる．しかし，特に練習などにおける演奏においては IOI は必ずしも楽譜から予測されるものとはならず，奏者による変動も大きいと考えられる．このような多様な演奏が生成されるようなモデルでは，状態遷移確率の IOI 依存性は低くなると考えられる．以下，IOI 依存性を取り除き[*4]，$a_{i_{M-1}, i_M} = p(i_M; t_M | i_{M-1}; t_{M-1})$ と記すこととする．

[*3] たとえばフェルマータの終了時のように，消音時刻の情報が楽譜追跡に重要である状況が考えられる．しかし，これらの状況は特殊であることから，別枠で扱うこととし，以下では発音開始イベントに集中する．

[*4] ただし，次節に述べる和音の発音を記述する自己遷移においては，IOI 依存性が特に重要であるので考慮する．

以上より，演奏譜 X が与えられたときの演奏イベント列 S の生成確率は，演奏位置情報列 $Q = \{(t_m, i_m)\}_{m=1}^{M}$ を用いて，

$$p(S) = \sum_{Q} p(S|Q) p(Q) \tag{3}$$

$$= \sum_{Q} \left[\prod_{m=1}^{M} b_{i_m}(s_m) a_{i_{m-1}, i_m} \right] \tag{4}$$

と表される．ここで，$b_{i_m}(s_m) = p(s_m|i_m)$ と記した．ただし，$a_{i_0, i_1} \equiv p(i_1)$ は初期状態確率である．式 (4) は，演奏イベント生成が，状態遷移確率 a_{i_{m-1}, i_m} と出力確率 $b_{i_m}(s_m)$ を持つ隠れマルコフモデル（Hidden Markov model, HMM）で記述されることを表している．このモデルを，**演奏 HMM** と呼ぶことにする．以下では，和音間の状態遷移確率 a_{i_{m-1}, i_m} と和音構成音の出力確率 $b_{i_m}(s_m)$ について詳しく述べる．

3.4 演奏 HMM における和音の記述

3.1 節の項目 a) で述べたように，和音の構成音は実際の演奏では，異なる時刻に順序不定で発音される．我々はこの状況をふまえ，和音構成音は 1 つの HMM 状態から自己遷移によって出力されるものとしてモデル化を行う．人間による鍵盤楽器演奏における 1 つの和音に属する構成音の IOI の分布は約 30 msec まで広がっていること，そしてこの程度の IOI は通常分離して聴かれないことが知られている [12], [13]．この時間間隔[*5]は通常の曲における和音間の IOI に比べて非常に小さい値である．そこで，閾値 Δt_{limit} を設けて，IOI が Δt_{limit} 以内の HMM 状態遷移は，和音構成音の出力に対応する自己遷移のみであると近似する．以下では，$\Delta t_{\text{limit}} = 35\,\text{msec}$ とする．よって，$t_m - t_{m-1} < \Delta t_{\text{limit}}$ のとき，

$$a_{i_{m-1}, i_m} = \delta_{i_{m-1}, i_m} \tag{5}$$

となる．ただし，$\delta_{i,i'}$ は Kronecker のデルタである．

音高 s_m が和音 c_{i_m} の構成音である場合には，出力確率 $b_{i_m}(s_m)$ は正の値を持っていると考えられるが，弾き間違えが起こることを考慮すると，構成音以外の s_m に対しても 0 でない確率を与える必要がある．具体的な出力確率の値は，演奏データから統計的な学習により設定することが可能である．また，演奏データが与えられない場合には値を経験的に設定することもできる．ここでは後者の手段をとり，和音構成音のほか，構成音から半音あるいは全音だけずれた音高に対しても小さな出力確率を与えることにより，ミスタッチを含む演奏のモデル化を行う．

3.5 演奏 HMM における状態遷移確率

状態遷移確率 a_{i_{m-1}, i_m} は和音の演奏順序を確率的に表

[*5] テンポ 120（四分音符/分）のとき，これは 64 分音符の長さにほぼ等しい．

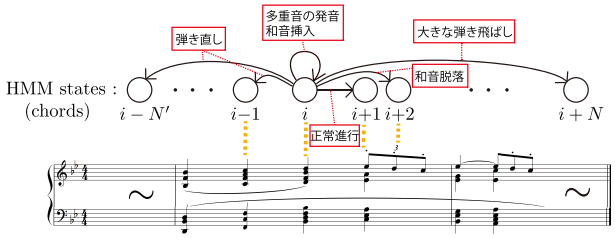

図 1 演奏生成の HMM の状態遷移トポロジ．i 番目の状態は演奏譜の i 番目の和音に対応している

Fig. 1 Topology of state transition probability for the performance HMM. The i'th state corresponds to the i'th chord of the performace score.

している．誤りのない演奏の際には $a_{i_{m-1},i_m} = \delta_{i_{m-1}+1,i_m}$ である．しかし，これ以外の遷移を考えることにより弾き間違いや即興性を含んだ演奏のモデル化が可能である．また，遠く離れた和音間の状態遷移確率に対しても 0 でない値を与えることにより，弾き直しや大きな弾き飛ばしを含む演奏をモデル化できる．

具体的には，弾き間違えによる 1 つの和音の脱落の確率は a_{i_{m-1},i_m} において $i_m = i_{m-1} + 2$ の値で表される．また，ミスタッチによる音の挿入誤りの確率は $i_m = i_{m-1}$ の場合の値により与えられる．さらに，$i_m < i_{m-1}$ の場合の値は弾き直しの確率であり，$i_m = i_{m-1} + d$ で $d \gg 1$ の場合は大きな弾き飛ばしの確率を表している．演奏 HMM の状態遷移トポロジを**図 1** に示す．

4. 演奏の楽譜位置推定

4.1 Viterbi アルゴリズムによる楽譜位置推定

演奏の楽譜位置推定問題は，前章で述べた演奏イベント列の生成モデルの逆問題として扱うことができる．演奏譜 $X = \{(\tau_i, c_i)\}_{i=1}^{I}$ に基づく演奏イベント列 $S = \{(t_m, s_m)\}_{m=1}^{M}$ が与えられたとき，演奏位置情報列 $Q = \{(t_m, i_m)\}_{m=1}^{M}$ は，Bayes の定理により

$$\underset{Q}{\operatorname{argmax}}\, p(Q|S) = \underset{Q}{\operatorname{argmax}}\, [p(S|Q)p(Q)] \tag{6}$$

$$= \underset{i_1,\cdots,i_M}{\operatorname{argmax}} \left[\prod_{m=1}^{M} b_{i_m}(s_m) a_{i_{m-1},i_m} \right] \tag{7}$$

を求めることにより，最尤推定される．楽譜追跡においては，演奏情報列を順次リアルタイムで最尤推定する必要がある．これは最新の M 番目の演奏イベント s_M が観測された際に，演奏情報列の最尤推定を更新し，演奏位置 i_M を得ることにより行われる．

HMM の最尤推定は（前方）Viterbi アルゴリズムを用いて行うことができる．時刻 t_{M-1} における各状態 i_{M-1} での最尤値を

$$\hat{p}_{i_{M-1}} = \max_{i_1,\cdots,i_{M-2}} \left[\prod_{m=1}^{M-1} b_{i_m}(s_m) a_{i_{m-1},i_m} \right] \tag{8}$$

とすると，

$$\hat{p}_{i_M} = \max_{i_{M-1}} \left[b_{i_M}(s_M) a_{i_{M-1},i_M} \hat{p}_{i_{M-1}} \right] \tag{9}$$

と帰納的に求めていくことができる．この最尤値の更新には演奏譜に含まれる和音の数 I に対して $\mathcal{O}(I^2)$ 回の確率計算を行う必要があることが分かる．これは演奏に用いる楽曲の長さが大きくなるにつれて最尤推定に要する計算量が大幅に増加することを意味している．特に練習時における自動伴奏では，伴奏再生を演奏イベントに瞬時に合わせて行うことが有効であると考えられるが（6.2 節参照），この際には特に演奏位置推定を高速に行う必要があり，最尤推定の計算コストを抑えることが重大である．

4.2 高速 Viterbi 探索アルゴリズム（$\alpha\beta\gamma$ 法）

弾き直しや大きく離れた箇所への弾き飛ばし（以下，合わせて「弾き飛ばし」と省略）を含まない演奏に対しては，遠く離れた状態間の遷移確率は 0 としてよいので，計算コストを抑えることができる．しかし，本稿では任意箇所への弾き飛ばしを含む演奏にも追従可能な楽譜追跡を議論しているため，こうしたことはできない．

HMM の状態間の遷移は，楽譜上の正常な進行に対応する 1 つ先への遷移（ここでは α 遷移と呼ぶ）が大部分を占めると仮定できる．その確率 α は 1 に近い．ついで，音の抜けや 1 音の弾き直しなどに対応する近傍への遷移（ここでは β 遷移と呼ぶ）の確率が高い．反復練習や部分スキップなどに対応する近傍以外の離れた状態への遷移（ここでは γ 遷移と呼ぶ）は，実際はフレーズの先頭などにジャンプすることが多いが，参照している楽譜データにフレーズ先頭などの情報が付与されていない場合は，どの状態への遷移確率も同等とせざるをえない．一般にその遷移確率 γ は，近傍の状態間の遷移確率に比べて小さいと仮定できる．

すべての遷移先状態 i_M に関して，もし γ が不均一であれば，すべての遷移元 i_{M-1} の尤度に状態遷移確率 a_{i_{M-1},i_M} を掛けた中の最大値をとることが必要なのに対し，γ が均一であれば，すべての遷移元の最大尤度の状態からの遷移のみ考えればよく，その後に α 遷移および β 遷移を考えればよい．i_M の中から尤度最大の状態を選べば，それが楽譜上の現在位置に対応する．そして，その尤度最大値は次の γ 遷移の計算に用いられる．

以上のような仮定のもとに，以下のような効率的な Viterbi 探索アルゴリズム（$\alpha\beta\gamma$ 法と仮称する）を導くことができる．上の議論により，状態遷移確率 $a_{i,i'}$ は

1) α $(i' - i = 1)$
2) β $(0 < |i' - i - 1| \leq d)$
3) γ $(d < |i' - i - 1|)$

と表される．ここで，d は近傍を表す正の数であり，$\gamma < \alpha, \beta$

である[*6]．このとき，$M-1$ 番目から M 番目への最尤値の更新式

$$\hat{p}_{i_M} = \max_{i_{M-1}} \left[b_{i_M}(s_{i_M}) a_{i_{M-1},i_M} \hat{p}_{i_{M-1}} \right] \tag{10}$$

の右辺は

$$b_{i_M}(s_{i_M}) \cdot \max \left\{ \gamma \ \hat{p}_{i_{M-1}^{\max}}, \ \alpha \hat{p}_{i_M-1}, \right. \\ \left. \beta \max_{0<|i_M - i_{M-1}-1| \le d} \left[\hat{p}_{i_{M-1}} \right] \right\} \tag{11}$$

となる．ここで，$i_{M-1}^{\max}$ は $M-1$ 番目の最尤状態を表す．

式 (11) により，最尤状態更新は $\mathcal{O}(dI)$ 回の確率計算で行えることが分かる．楽譜追跡の場合，HMM の状態数 I は演奏譜の和音の個数であり，多くのピアノ曲ではおよそ 50 から 10,000 である[*7]．また，近傍を表す d の値は通常の演奏では曲の長さよりも十分に小さいと考えられるため，$d \ll I$ と仮定してよいであろう．よって，楽譜追跡に必要な処理時間は $\alpha\beta\gamma$ 法により大幅に軽減されることが期待される．最後に，近傍内での β の値は個別に設定することが可能であるが，これによる計算回数の増加はないことを注意しておく．

4.3 $\alpha\beta\gamma$ 法の有効性検証

$\alpha\beta\gamma$ 法の有効性を示すために，楽譜追跡の処理時間を測定した．測定実験では Rachmaninoff のピアノ協奏曲第三番第一楽章のソロピアノパート全体を繰り返して 2 倍の長さにしたものを用意し，この中から必要な長さの部分を切り取って演奏譜とした．演奏譜の長さとしては 100 から 10,000 まで 7 種類を用いた．この演奏譜に対して，最尤状態更新にかかった処理時間を測定した．この処理時間には演奏イベント列の詳細は無関係であるので，測定には，ランダムな音高列（音符数 163）を用い，各更新にかかった処理時間の平均を求めた．演奏 HMM のパラメータ α，β，γ には次節で議論されている現実的な値を用いたが，これらは確率計算回数を変えないため，具体的な値による処理時間への影響は無視できると考えられる．また，近傍を表す上記パラメータ d の値として $d=3$ を用いた．また，比較のため，通常の Viterbi アルゴリズムを用いた場合の処理時間も測定した．なお，測定に用いた計算機の動作環境は，CPU Intel(R) Core(TM) i5-2540M，RAM 8 GB，OS は Windows 7 Professional 64 bit である．また，処理時間測定には標準 C ライブラリ `time.h` 内で定義されている `clock` 関数を用いた．

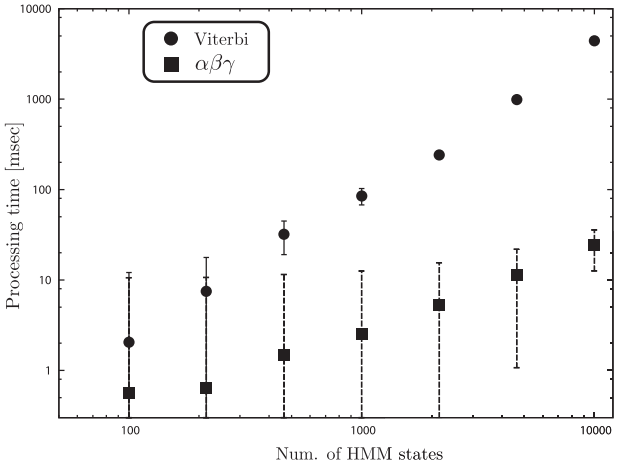

図 2 HMM の最尤状態更新に必要な処理時間の測定結果．丸点は Viterbi アルゴリズムを用いた場合の結果であり，四角点は $\alpha\beta\gamma$ 法を用いた場合の結果を示している．誤差棒は 1σ の誤差を表している

Fig. 2 Processing time for updates of most likely HMM states. The filled circles show results using the forward Viterbi algorithm and the squares show results using the $\alpha\beta\gamma$ algorithm.

図 2 に実験結果を示す．図中，丸点は通常の Viterbi アルゴリズムを用いた場合の結果であり，四角点は $\alpha\beta\gamma$ 法を用いた場合の結果を表している．測定誤差は，データの標準誤差のほか，系統誤差としてシステムの時間分解能 10 msec を含んでいる．実験結果より，Viterbi アルゴリズムを用いた場合に比べ，$\alpha\beta\gamma$ 法を用いた場合では状態数の多さに対する処理時間の増加が抑えられることが確かめられた．特に演奏譜の長さが 1,000 以上になると，Viterbi アルゴリズムを用いた場合は処理時間が急激に増加する一方，$\alpha\beta\gamma$ 法を用いた場合には，楽譜追跡で要求される時間分解能である，数 10 msec にとどまっていることが分かる．また，前節で議論したように，d を大きくするほどモデルの記述能力は増加する一方で，処理時間はほぼ線形で増大する．実験結果より，長い楽曲に対してもこの時間分解能以内の処理時間を得るためには，$d=3$ が d に関する実際的な上限に近いことが分かる．

4.4 楽譜位置推定の評価

提案する楽譜位置推定手法の有効性を確かめるために，上記アルゴリズムを実装して評価実験を行った．本節では，弾き飛ばしに対する追従性を系統的に評価するために，打ち込みの MIDI 演奏と人間による演奏の両方を用いた評価結果を示す．これは，人間による演奏では実際の弾き誤りや弾き飛ばしを含んだ演奏に対して現実的な評価が行える一方で，均質な演奏を大量に系統的に得ることが難しく，楽譜位置推定の性能の詳細な定量評価が得にくいという理由による．以下，打ち込みの MIDI 演奏を用いて弾き飛ば

[*6] この仮定は，以下の式 (11) を導く際に用いているが，この仮定がない場合も，式中で $\gamma \hat{p}_{i_{M-1}^{\max}}$ を $\gamma \max_{|i_M - i_{M-1}-1|>d} \left[\hat{p}_{i_{M-1}} \right]$ とした式は成り立つ．この最大値の計算は $\hat{p}_{i_{M-1}}$ の $2d+2$ 番目までの最大値を求めることにより効率的に行える．よってこの場合でも，計算量は大きくなるが，最尤推定は I に関して線形で行うことができる．この指摘は，中村友彦氏によりなされた．

[*7] たとえば，Chopin の Prélude Op. 28-7 A-dur では $I=48$ であり，Rachmaninoff のピアノ協奏曲第三番 d-moll の第一楽章のソロピアノパートの場合は，I は約 5,000 である．

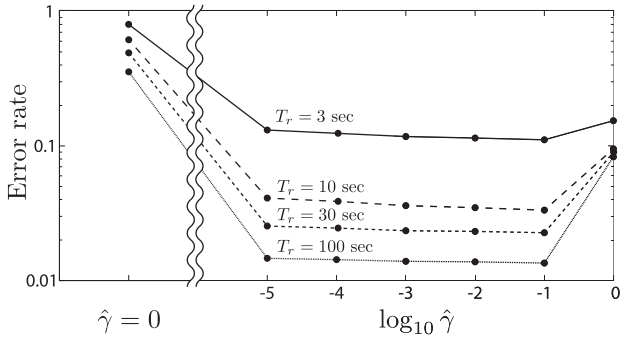

図 3 弾き飛ばしを含む演奏に対する楽譜位置推定の誤り率

Fig. 3 Error rates of score position estimation for performance with large skips.

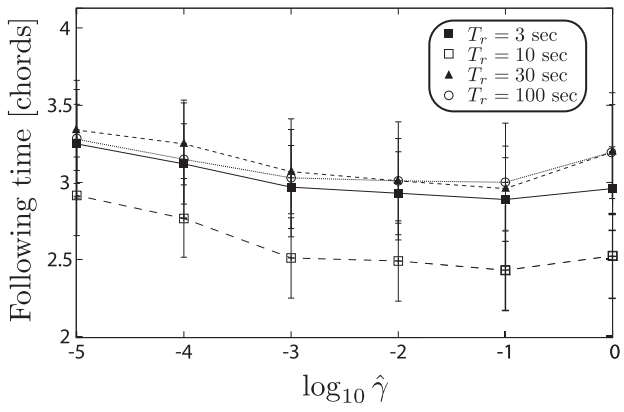

図 4 弾き飛ばしに対する平均追従時間（和音数）

Fig. 4 Required times in chords for correctly following the score position after large skips.

しに対する追従性に関する系統的な評価を行い，パラメータ依存性を議論したうえで，人間による実際の演奏をもとに現実的な演奏での楽譜位置推定の性能を評価する．

まず，打ち込みの MIDI 演奏をもとに時間 T_r ごとに楽曲の任意箇所への弾き飛ばしを含むような演奏を作成し，全体の楽譜追跡の精度および弾き飛ばしへの追従に要した時間を測定する実験を行い，弾き飛ばしの頻度を表す T_r や演奏 HMM のパラメータ α，β，γ に関する依存性を調べた．人工的に人間の弾き誤りを合成することは難しいため，ここではこれを含めず，誤りを含んだ演奏に対する評価は後述する人間の演奏により行うことにする．

実験結果を**図 3**，**図 4** に示す．実験に使用した楽曲は Mozart の 2 台のピアノのためのソナタから第一楽章提示部（第一ピアノパート）である．弾き飛ばしの時間間隔を $T_r = 3, 10, 30, 100$ sec とし，弾き飛ばしを各 100 回含む演奏を作成した．ただし，用いた MIDI 演奏の総演奏時間は約 150 秒であり，弾き飛ばし後 T_r 以内に曲の終わりに達した場合は，その時点で任意箇所への弾き飛ばしを行うようにした．図 3 はこれらの演奏に対する，楽譜位置推定の発音単位の誤り率を表し，図 4 は，各弾き飛ばしの後楽譜位置推定が正しく行われ始めるまでにかかった時間

表 1 弾き飛ばしを含む演奏に対する楽譜位置推定の誤り率と平均追従時間の β 依存性．値は $T_r = 10$ sec，$\hat{\gamma} = 0.1$ のときのものを示している

Table 1 The β dependence of error rates (ER) of score position estimation and the averaged following time (FT) for performance with large skips, for $T_r = 10$ sec and $\hat{\gamma} = 0.1$.

$-\ln \beta$	2.30	2.59	3.28	4.30	5.33	6.51
ER (%)	4.71	3.42	3.34	3.49	3.85	4.30
FT [chords]	2.51	2.49	2.46	2.46	2.43	2.44

（以下，**追従時間**）を和音数で表したときの平均とその誤差を示している．この際，連続した 2 和音に関して正しい楽譜位置推定が行われることを正しい追従の条件とした．図中，各点は $\hat{\gamma} = (I - 2d - 1)\gamma$ として，$\hat{\gamma} = 0.99$ から 10^{-5} まで変化させた場合の値を示している．また，弾き飛ばしに対応していない場合 [9], [11] に相当する $\hat{\gamma} = 0$ としたときの値も示した．演奏 HMM の他ののパラメータ α と β は，$\alpha + 2d\beta + \hat{\gamma} = 1$ を満たし，α/β を指定すれば定まる．ここでは，各 $\hat{\gamma}$ に対して α/β を 1 から 1,800 までの 8 段階で用意し，それらのなかで $\alpha \geq \beta \geq \gamma$ を満たすものについて最適化を行った結果を示した．また，d の値としては前節で得られた上限値である $d = 3$ を用いた．

図 3 より，$\hat{\gamma} > 0$ の導入により弾き飛ばしを含む演奏への楽譜位置推定精度が大きく向上していることが分かる．すべての T_r の場合に最適な $\hat{\gamma}$ の値は $\hat{\gamma} = 0.1$ であり，$\hat{\gamma}$ が小さくなると徐々に誤り率が大きくなっている．また，図 4 では弾き飛ばしに対する追従時間は，いずれの T_r でもおよそ 2.5 から 3.5 和音であることが分かる．図には $\gamma = 0$ の場合の値を載せていないが，この場合，次の弾き飛ばしまでの間に正しく追従が行われた割合が $T_r = (3, 10, 30)$ sec のときにそれぞれ約 (35, 62, 71)%と低く，この割合が 100%であった $T_r = 100$ sec の場合でも，追従時間は最低でも 127.7 和音と大きかった．この結果により，誤りのない演奏の場合，適切な α，β，γ を設定すれば任意箇所への弾き飛ばしに平均約 3 和音以内での追従が可能であることが分かる[*8]．この値は特定箇所への弾き飛ばしに対応した文献 [6] の手法による追従性能とほぼ等しい．

両方の測定において，最適な β の値はおよそ 0.005 から 0.04 であったが，いずれの場合もこの周辺での変化は比較的小さかった．**表 1** では，この周辺での $T_r = 10$ sec，$\hat{\gamma} = 0.1$ の場合の β に対する誤り率を示している．この結果から，パラメータ d により定義される近傍内において β の値を個別に設定しても追従性能の向上は比較的小さく，すべて共通の値でも相応の性能が得られることが期待される．

[*8] 弾き飛ばしに対する追従性能は楽曲の構造に依存すると考えられる．たとえば，繰返し構造が多い楽曲では必然的に追従が困難である．これに関しては本節後半で議論されている．

表 2 実験で用いた人間による演奏に含まれている音符と弾き誤り，弾き飛ばしの数

Table 2 The number of skips and performance errors contained in the human performance used in the experiment.

楽曲	弾き飛ばし	音符数	音高誤り	脱落・挿入
Mozart1	0	1,387	27	24
Mozart2	22	1,621	18	41
Mussorgsky1	0	2,066	60	16
Mussorgsky2	12	2,120	78	12
Debussy1	0	1,210	17	10
Debussy2	15	2,032	36	21

表 3 演奏位置推定の結果

Table 3 Results of performance position estimation.

楽曲	弾き飛ばし	誤り率	追従時間（和音数）
Mozart1	無	6.06%	
Mozart2	有	7.09%	2.8 ± 0.3
Mussorgsky1	無	0.68%	
Mussorgsky2	有	3.51%	1.3 ± 0.2
Debussy1	無	2.82%	
Debussy2	有	8.47%	5.3 ± 1.5

次に，人間の演奏を用いた評価を行った．使用した楽曲は，上記 Mozart の楽曲のほか，Mussorgsky の組曲「展覧会の絵」より第一曲 Promenade と，Debussy の前奏曲「亜麻色の髪の乙女」である．1 曲目以外はピアノソロのための楽曲であるが，楽譜追跡は伴奏再生とは独立に行うことができるので，今回は演奏の取得が容易であったためこれらの楽曲を用いた．演奏はアマチュアのピアノ経験者（筆者の 1 人）によるものを MIDI 録音して用いた．録音された演奏は，練習時における弾き誤り・弾き飛ばしを含むものであった．複数回録音した演奏のうち，和音数 3 以上の弾き飛ばしを含むものとそうでないものに分け，前者を「弾き飛ばしあり」，後者を「弾き飛ばしなし」とした．一方，和音数 3 未満の弾き飛ばしは，和音の脱落・挿入とした．**表 2** に演奏に含まれた弾き誤りの種類と数，また弾き飛ばしの数を示す．表中，楽曲欄には各楽曲の作曲者の名前と弾き飛ばしの有無を区別するための数字（1 または 2）を記した．

表 3 に演奏位置推定の結果を示す．演奏 HMM のパラメータは，上記実験の結果をもとに，$\hat{\gamma} = 0.1$，$\beta = 0.04$ とした．誤り率は，演奏データに対して人手で楽譜位置の正解をラベル付けした結果に対するものである．実験結果より，弾き誤りがある場合および，弾き誤りと弾き飛ばしがある場合も高い精度で楽譜位置推定が行われたことが分かる．また，弾き飛ばしに対する追従時間は Mozart の曲に対して，弾き誤りがない場合と同程度であった．このことから，提案手法による楽譜位置推定が弾き誤りを含む演奏に対しても，誤りなしの場合と同様に任意箇所への弾き飛ばしに追従可能であるといえる．

Debussy の曲では追従時間が大きく，弾き飛ばしがある場合に正解率が大きく減少しているが，これは楽曲に含まれる繰返し部分への弾き飛ばしが行われた際に，誤った楽譜位置での追従を続けてしまったことが原因である．このように，弾き飛ばしに関する追従性能は楽曲に含まれる繰返し構造の多さに影響される[*9]．また，Mozart の楽曲は全体的に正解率が低くなっているが，この原因の 1 つに楽曲内にトリルやアルペジオなどの装飾音が存在することがあげられる．また，右手左手同時に速い音形を演奏する場合には同期がうまくとれていない箇所もあり，こうした箇所においても楽譜位置推定が難しいことがあった．

5. 演奏のテンポ推定

5.1 テンポ推定問題

2.1 節で述べたように，自動伴奏を行うためには，演奏位置の推定に加えてテンポ推定を行う必要がある．ここでは，演奏位置の情報列 $Q = \{(t_m, i_m)\}_{m=1}^{M}$ が与えられたときのテンポ推定の方法について論じる．

テンポとは楽譜時間の実時間に対する速度 $r = \delta\tau/\delta t$ のことであるが，特に音符ごとに音価と IOI の比によって与えられる

$$r_m = \frac{\tau_{m+1} - \tau_m}{t_{m+1} - t_m} \tag{12}$$

を**局所テンポ**（Local tempo）と呼ぶことにする．局所テンポ r_m は，m 番目と $m+1$ 番目の発音時刻と楽譜時刻が分かれば式 (12) によって求まる．逆に，r_m と m 番目の発音時刻を知っていれば，$m+1$ 番目の発音時刻を予測することができる．このように，m 番目までの演奏位置情報列が与えられたときに，$m+1$ 番目の発音時刻を予測することと r_m を推定することは等価である．以下では，この問題をテンポ推定問題として議論する．

人間による演奏におけるテンポは楽譜から直接推定することは難しい．これは 3.1 節にも述べたとおり，テンポに関しても演奏ごとに不確定性があるからである．テンポの不確定性は次のものがあげられる．

i) テンポの絶対値

多くの楽曲ではテンポの絶対値の詳細は指定されずに，決定は奏者に委ねられている．また，指定されている場合も，指示どおりに演奏されるとは限らない．

ii) 表情付けなどによるテンポ変化の形

主に表情付けの目的により，演奏では局所テンポに変化を与えることがある．これは，記譜により指定されるもの（*rit.* など）と奏者の即興によるものがあるが，前者の場合においてもその変化の度合いなどの詳細は指定されないも

[*9] ただし，同じフレーズであれば楽譜位置によらず類似した伴奏となることが多いため，多くの楽曲の場合，他のフレーズ箇所と間違えてしまっても違和感のない伴奏の演奏が期待される．

のが多い．

iii) 技術的あるいは物理的要因によるテンポの揺らぎ

i) と ii) にあげた意図されたテンポの変動のほかに，意図されずに起こるテンポの揺らぎも存在する．これは，奏者の技術的制限あるいは物理的な制約より起こるテンポの揺らぎである．発音時刻の変動もテンポ揺らぎの原因となる．

ここにあげた i) や ii) のようにテンポに即興性があることは，テンポ推定を演奏と同時にリアルタイムで行う必要があることを示している．しかし，実際の演奏においては iii) によるテンポの変動が混ざって観測される．そのため，テンポ推定は，iii) によるテンポ揺らぎの影響を吸収しながら，i) や ii) によるテンポの変化を考慮する必要がある．

5.2 テンポ推定アルゴリズム

上述した状況をふまえて，ここでは，観測される局所テンポ r_m は，奏者が意図した理想的な局所テンポ $\tilde{r}_m$ と iii) の原因によるテンポの揺らぎ ϵ_m によって $r_m = \tilde{r}_m + \epsilon_m$ と表されると仮定する．意図された局所テンポ $\tilde{r}_M$ は過去の値 $\tilde{r}_{M-1}, \cdots, \tilde{r}_1$ に依存して決定されると考えられる．この際，直前の値には強く依存するが遠く離れた過去の値に対する依存性は小さいと考えられる[*10]．また，$\tilde{r}_m$ は曲中で変動しうるが，その変動は多くの場合，局所的には緩やかであると考えられる．これらの考察により，ここでは $\tilde{r}_M$ はその直前の値 $\tilde{r}_n$, $n = M-1, \cdots, M-K$ に依存して決定されるとし，局所的には $\tilde{r}_n$ は一定値 $\tilde{r}$ をとると仮定する．テンポの揺らぎ ϵ_m は，意図されないものであり確率的なノイズとしてモデル化できる．

以上より，

$$r_n = \tilde{r} + \epsilon_n, \quad n = M, M-1, \cdots, M-K \tag{13}$$

と表すことができる．ϵ_n は Gauss–Markov 条件を満たす確率変数であると仮定すると，r_M の期待値は $\langle r_M \rangle = \tilde{r} + \langle \epsilon_M \rangle = \tilde{r}$ となる．一方，$\tilde{r}$ は観測値 r_n, $n = M-1, \cdots, M-K$ により推定できるが，その最良線形不偏推定量は最小二乗法により r_n の平均によって与えられる．以上をまとめると，r_M の推定値 $\hat{r}_M$ は

$$\hat{r}_M = \frac{1}{K} \sum_{k=1}^{K} r_{M-k} \tag{14}$$

と与えられることが分かる．K は状況により最適化されるべきパラメータであるが，文献 [14] では，$K = 4$ が最適であるという実験結果が報告されている．

[*10] *a tempo* などによるテンポ指定の場合には，遠く離れた過去の局所テンポが参照される．また，楽節の境界などにおいてテンポの急激な変化が起こる際には，局所テンポが直前の値とは独立に決定されることもある．これらのテンポ変化は通常記譜されるものであり，楽譜により箇所が特定できる場合が多い．ここでは，こうした箇所は楽譜情報をテンポ推定に組み入れることにより対応することを考え，これ以外の場所におけるテンポ推定のみを考えることとする．

表 4 テンポ推定評価の結果．評価には 1 つ先の音符の発音時刻の予測値と実測値の誤差を求め，誤差が許容値以内のものを正解とした．正解率 1 と 2 は，それぞれ発音時刻の誤差の許容値が 30 msec と 100 msec とした場合の正解率を表している

Table 4 Results of evaluation test for tempo estimation. For the evaluation, we calculate the error between the predicted and observed onset times of the following note. The accuracies are listed for allowed errors of 30 msec and 100 msec.

楽曲	音符数	正解率 1	正解率 2
Mozart	1,577	72.3%	98.9%
Mussorgsky	1,007	63.3%	96.4%
Debussy	1,213	31.6%	73.9%

5.3 テンポ推定の評価

前節で述べたテンポ推定の手法の性能を評価するための実験を行った．実験には 4.4 節で用いたものと同じ楽曲と演奏を用いた．評価では，局所的に楽譜位置推定結果が正解と一致しているところに関して，局所テンポ推定値による次の発音時刻の予測値とその実測値との誤差 δ_{onset} を求めた．この誤差が十分小さければ，自動伴奏には十分な精度でテンポ推定ができていると考えられる．そこで誤差の許容値を δ_{th} としたときに，$\delta_{\text{onset}} < \delta_{\text{th}}$ を満たすものの割合を正解率とした．δ_{th} としては 30 msec と 100 msec の 2 つの場合について正解率を求めた．ここで，30 msec は Δt_{limit} に近い値であり，100 msec は通常の合奏では許容されることが多い程度の誤差であると考えられる．

表 4 に実験結果を示す．表中，正解率 1，2 はそれぞれ $\delta_{\text{th}} = 30$ msec，100 msec とした場合の正解率を示している．また，テンポ推定に関しては，演奏中の弾き飛ばしによる影響は小さいと考えられるので，表にはすべての演奏に対する結果を合計したものを記した．曲中にテンポ変化の指示がなく，演奏も標準的なテンポ変動を持つものであった．Mussorgsky と Mozart の楽曲では，高い正解率が得られていることが確認できる．一方，Debussy の楽曲はテンポ変化の指示が多いうえ，その他の箇所においても大きなテンポ変動が期待されるものである．実験に用いた演奏でも多くのテンポ変動があった．正解率は前の 2 つの楽曲に比べると低く，$\delta_{\text{th}} = 100$ msec に対する正解率は約 74%であった．このようなテンポの変化の多い楽曲・演奏に対するテンポ推定法の改良は，今後の課題である．

6. 伴奏再生

6.1 演奏者に追従する伴奏再生

前章までに述べられた楽譜追跡の方法によって得られた演奏情報列 $R = \{(t_m, i_m, r_m)\}_{m=1}^{M}$ が与えられた際の伴奏再生情報列 $O = \{(t_\ell, \tau_\ell, r_\ell)\}_{\ell=1}^{L}$ の構成について述べる．

人間の奏者による伴奏の場合は，演奏者の演奏位置・テンポを随時感知して，少し先の時刻における演奏イベント

を予測しながら伴奏再生が行われていると考えられる．この理由の 1 つとして，人間の奏者には身体能力的な限界により瞬時的な同期をすることが不可能であることがあげられる．また，人間の伴奏者は演奏者の演奏だけでなく自分の演奏にも気を配る必要があり，場合よっては演奏との同期よりも，伴奏の音楽的な自然性を優先することも考えられる．

しかし，機械の場合には演奏イベントに対して瞬時的に反応することが可能な場合も多い．特に，MIDI イベントによる入出力を持つ自動伴奏の場合にはこれが可能である．この場合，機械による伴奏再生では，演奏と同時刻に発音されるべき伴奏音は演奏イベントの到達を待ち，瞬時的に同期するという方法が考えられる．特に，練習時の演奏に対する自動伴奏では，演奏者がテンポについていけずに立ち止まることも多いと考えられるため，この方法が有効であると考えられる．

6.2 伴奏再生アルゴリズム

以下，前節に述べた伴奏再生の方法を記述する．伴奏再生情報列 O は

(1) 演奏情報列 R の更新があるときには，それに従って伴奏再生情報列 O を更新し，次に期待される演奏情報列の更新時刻 t_{exp} を計算する，

(2) もし，この更新時刻になっても演奏情報列が更新されない場合には，伴奏をいったん止める，

という規則に従って更新される．伴奏再生情報列 O の更新を具体的に記すと，

(1) R の更新があった場合，

Set $t_{\mathrm{exp}} = t_M + (\tau_{i_M+1} - \tau_{i_M})/r_M$.

If $i_M = i_{M-1} + 1$, then add (t_M, τ_{i_M}, r_M) to O.

If $i_M \neq i_{M-1}$, $i_{M-1} + 1$, then add $(t_M, -\infty, 0)$ and (t_M, τ_{i_M}, r_M) to O.

(2) $t > t_{\mathrm{exp}} - \varepsilon$ となった場合（ε は任意の微小量），

Add $(t, -1, 0)$ to O and set $t_{\mathrm{exp}} = \infty$.

となる．ここで，伴奏再生情報 $(t_\ell, \tau_\ell, r_\ell)$ において，$\tau_\ell = -1$ と $-\infty$ はそれぞれ現在の再生楽譜時刻と再生中止を表すものとする．

7. 自動伴奏システム Eurydice

7.1 Eurydice の概要と仕様

上述の理論に基づいて自動伴奏システム Eurydice（ユリディス）を構築した．これは，テンポの変化，弾き間違い，弾き直しを含む多様な人間の演奏に対して瞬時に追従して自動的に伴奏を行うシステムである．Eurydice は，大きく分けて楽譜追跡部分と伴奏再生部分の 2 つから成り立つ（**図 5**）．楽譜追跡部分では，逐次的に入力される MIDI 演奏からリアルタイムに演奏位置とテンポを推定する．伴奏再生部分では，楽譜追跡の結果に基づき演奏に同期された伴奏を MIDI イベント列として出力する．両者のアルゴリズムはすでに述べたとおりである．

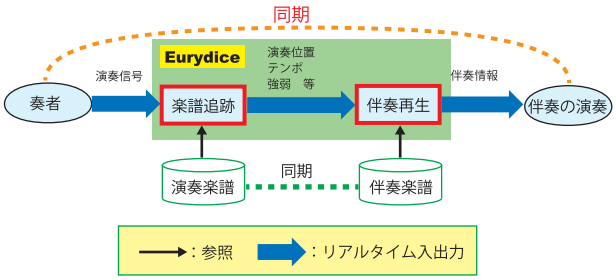

図 5 自動伴奏システム Eurydice の概念図．Eurydice は，演奏イベントをもとに演奏位置やテンポなどの情報を得る楽譜追跡部分とこの情報をもとに伴奏の演奏を行う伴奏再生部分からなっている

Fig. 5 Scheme of the automatic accompaniment system Eurydice. The system consists of the score following part and the accompaniment playing part.

表 5 主観評価実験の結果．

Table 5 Results of the subjective evaluation test.

質問	評価				平均
	1	2	3	4	
A	0	3	18	17	3.4
B	0	1	4	33	3.8
C	0	2	8	28	3.7

システムは，C++ によって実装されている．クロスプラットフォームの GUI 開発フレームワークの Qt [15]，MIDI 入出力ライブラリの RtMidi [16] を用いることで，Windows，MacOS X，Linux など，多くの環境で動作する．演奏の入力，伴奏の出力ともに MIDI を用いるため，使用の際には，MIDI 入力デバイス，MIDI 出力デバイス，および演奏する楽曲の MIDI データを用意する必要がある．楽曲の MIDI データは，楽譜追跡のための演奏 HMM の構築，また，伴奏用の MIDI 演奏イベントとして用いられる．奏者は，GUI を通して演奏する声部および Eurydice によって自動伴奏を行う声部を選択することが可能である．

7.2 Eurydice の主観評価

Eurydice による伴奏の性能を評価するために，主観評価実験を行った．実験では，被験者に練習時に起こる弾き誤り・弾き直し・弾き飛ばしを含んだピアノ演奏をしてもらい，Eurydice による自動伴奏を体験してもらった．被験者は全部で 38 人であり，ピアノ経験歴なしの初級者から 20 年以上の上級者まで含まれていた．Eurydice を使用してもらった後，被験者には質問 A「弾き誤り・弾き直し・弾き飛ばしを含む演奏に適切に伴奏されたか」を 4 段階（1：そう思わない，2：あまり思わない，3：少しそう思う，4：そう思う）で評価してもらった．**表 5** に主観評価の結果を示す．この結果多くの被験者が「そう思う」または「少し

そう思う」と答えており，提案手法による自動伴奏が弾き誤りや弾き飛ばしを含む演奏に対応して伴奏をすることが確かめられた．

また，質問 B「Eurydice の伴奏があることによって，演奏を楽しめたか」に対しては上記 4 段階評価で平均 3.8，質問 C「Eurydice を演奏・練習で使ってみたいか」に対しては平均 3.7 の高い評価が得られ，自動伴奏システムとしての Eurydice の有効性が確認された．「伴奏による違和感はなかった」という意見がある一方で，「人間による伴奏にはおよばない」というような意見もあった．後者の原因として，人間による伴奏と比べたときの本手法のテンポ推定および伴奏再生アルゴリズムによる伴奏の不自然さが考えられる．人間による伴奏を模するという目的のためには，これらの改善が必要であると考えられる．

8. 結論

本稿では，鍵盤 MIDI 楽器の演奏における弾き誤り・弾き直し・弾き飛ばしに柔軟に追従する楽譜追跡アルゴリズムおよび自動伴奏システムを提案した．第 1 に，隠れマルコフモデルを用いて演奏の生成過程をモデル化し，その確率的逆問題に立脚した楽譜位置推定方法について論じた．また，処理コストを低減させるために，最尤状態推定の高速アルゴリズムである $\alpha\beta\gamma$ 法を考案し，その有効性を実験によって確認した．提案する楽譜位置推定アルゴリズムを実装し，人工的な演奏と実演奏を用いて評価を行った．そして，弾き誤りと弾き飛ばしを含む演奏に対しても，高い精度で楽譜位置推定ができ，評価に用いた楽曲において平均約 3 和音で任意箇所への弾き飛ばしに追従が可能なことを確認した．第 2 に，テンポ推定について論じた．テンポ変動の確率モデルに基づいた，テンポ推定アルゴリズムを示し，その推定精度を評価し，有効性を確認した．最後に，自動伴奏システム Eurydice を構築し，動作を確認した．また，主観評価実験を行い伴奏の追従性能およびシステムの有効性を確認した．

今後の課題として，まず，テンポ推定の精度向上があげられる．楽譜にテンポ変化の指示が多く，その他の箇所でもテンポの揺らぎが期待されるような楽曲の場合には，テンポ推定の精度が比較的低く，さらなる改良を行う必要がある．また，装飾音を含む演奏に対応可能な楽譜追跡への拡張も課題である．トリルやアルペジオなどの装飾音は，音符の個数や各音符の音価が一意には定まっていない．こうした装飾音などを含む演奏に対しても頑健に追従する楽譜追跡の方法については，現在検討中であり，稿を改めて報告する．

謝辞 自動伴奏システム Eurydice の開発初期に戦略ソフトウェア創造人材養成プログラムとしてご指導くださった平木敬教授，稲葉真理准教授，金子勇氏，土村展之氏，実装と改良に寄与された東京大学嵯峨山研究室の米林裕一郎氏，Gustav Larsson 氏，金泰憲氏らをはじめとする方々，また，システムの試奏・評価に協力された松井淑恵氏，垣浪文美香氏，饗庭絵里子氏らをはじめとする学外の多数の方々に深く感謝する．

参考文献

[1] Dannenberg, R.: An on-line algorithm for real-time accompaniment, *Proc. ICMC*, pp.193–198 (1984).
[2] Vercoe, B.: The synthetic performer in the context of live performance, *Proc. ICMC*, pp.199–200 (1984).
[3] Orio, N. et al.: Score Following: State of the Art and New Developments, *New Interfaces for Musical Expression*, pp.36–41 (2003).
[4] Tekin, M., Anagnostopoulou, C. and Tomita, Y.: Towards an intelligent score following system: Handling of mistakes and jumps encountered during piano practicing, *CMMR*, pp.211–219 (2004).
[5] Pardo, B. and Birmingham, W.: Modeling form for online following of musical performances, *Proc. 20th National Conference on Artificial Intelligence*, pp.1018–1023 (2005).
[6] 大島千佳，西本一志，鈴木雅美：家庭における子供の練習意欲を高めるピアノ連弾支援システムの提案，情報処理学会論文誌，Vol.46, No.1, pp.157–170 (2005).
[7] Raphael, C.: Music Plus One: A system for expressive and flexible musical accompaniment, *Proc. ICMC*, pp.159–162 (2001).
[8] Cont, A.: ANTESCOFO: Anticipatory synchronization and control of interactive parameters in computer music, *Proc. ICMC* (2008).
[9] Schwarz, D., Orio, N. and Schnell, N.: Robust Polyphonic Midi Score Following with Hidden Markov Models, *Proc. ICMC* (2004).
[10] Lemouton, S. and Nicolaou, V.: Polyphonic audio score following: The Otemo case, Ircam Centre Pompidou Technical Report (2009).
[11] Bloch, J. and Dannenberg, R.: Real-time computer accompaniment of keyboard performances, *Proc. ICMC*, pp.279–289 (1985).
[12] Goebl, W.: Melody lead in piano performance: Expressive device or artifact?, *J. Accoustical Society of America*, Vol.110, No.1, pp.563–572 (2001).
[13] 武田晴登，西本卓也，嵯峨山茂樹：テンポ曲線と隠れマルコフモデルを用いた多声音楽 MIDI 演奏のリズムとテンポの同時推定，情報処理学会論文誌，Vol.48, No.1, pp.237–247 (2007).
[14] 武田晴登，西本卓也，嵯峨山茂樹：HMM による MIDI 演奏の楽譜追跡と自動伴奏，情報処理学会研究報告，MUS, pp.109–116 (2006).
[15] Nokia: Qt - Cross-platform application and UI framework, available from ⟨http://qt.nokia.com/⟩.
[16] Scavone, G.P. and Cook, P.R.: RtMidi, RtAudio, and a Synthesis Toolkit (STK) Update, *Proc. ICMC* (2005).

中村 栄太

2012 年東京大学大学院理学系研究科物理学専攻博士後期課程修了．博士（理学）．現在，東京大学大学院情報理工学系研究科システム情報学専攻研究生．

武田 晴登 （正会員）

2001 年慶應義塾大学理工学部卒業．2003 年東京大学大学院情報理工学系研究科修士課程修了．2006 年関西学院大学理工学部 CrestMuse プロジェクト研究員．博士（情報理工学）．2007 年ソニー株式会社に入社．IEEE 会員．

山本 龍一 （学生会員）

2011 年名古屋工業大学情報工学科卒業．現在，名古屋工業大学大学院工学研究科情報工学専攻博士前期課程に在籍中．2012 年日本音響学会東海支部優秀発表賞受賞．音楽情報処理，音楽信号処理等の研究に従事．日本音響学会会員．

齋藤 康之 （正会員）

1993 年木更津工業高等専門学校電気工学科卒業．1995 年九州工業大学工学部電気工学科情報コース卒業．1997 年北陸先端科学技術大学院大学情報科学研究科情報処理学専攻博士前期課程修了．2000 年同大学院大学博士後期課程単位取得退学．2000 年木更津工業高等専門学校情報工学科助手．現在，同校准教授．博士（情報科学）．画像処理，音楽情報処理等の研究に従事．電子情報通信学会，画像電子学会，日本高専学会，日本福祉工学会，精密工学会画像応用技術専門委員会，日本音響学会，映像情報メディア学会各会員．

酒向 慎司 （正会員）

1999 年名古屋工業大学知能情報システム学科卒業．2004 年名古屋工業大学大学院電気情報工学専攻博士後期課程修了．同年東京大学大学院情報理工学研究科特任助手．2007 年名古屋工業大学大学院情報工学専攻助教．博士（工学）．音楽情報処理，音声情報処理，手話認識の研究に従事．2011 年度電子情報通信学会ヒューマンコミュニケーション賞受賞．電子情報通信学会，日本音響学会，人工知能学会，IEEE 各会員．

嵯峨山 茂樹 （正会員）

1974 年東京大学大学院工学系研究科計数工学専攻修士課程修了．同年から日本電信電話公社武蔵野電気通信研究所にて音声情報処理の研究に従事．1990～1993 年 ATR 自動翻訳電話研究所にて自動翻訳電話プロジェクトを遂行．1998 年に北陸先端科学技術大学院大学を経て，2000 年から東京大学教授，2001 年より同大学院情報理工学系研究科システム情報学専攻．博士（工学）．1996 年科学技術庁長官賞等を受賞．日本音響学会，電子情報通信学会，IEEE 各会員．