E-031

DP マッチングを用いた演奏の現在位置解析手法の提案
Proposal of score following using DPmatching

赤本 仁史[†]　　武田 正之[‡]
Hitoshi Akamoto　　Masayuki Takeda

1. はじめに

近年, キーボードのほかにもギター型, トランペット型のような計算機との情報交換が容易な MIDI 規格の電子楽器も普及してきており, 計算機による演奏の需要は高まっている.
それに伴い, 人間が楽譜に基づいて曲を演奏する時, 計算機に伴奏させて仮想的なアンサンブルを実現する自動伴奏の研究が行われている.

人間の演奏者に計算機が伴奏を付けるという処理を考えたとき, 演奏者の演奏が曲中のどの位置に対応しているのかを求め(楽譜追跡), その楽譜追跡の結果に基づいてどのように伴奏を再生するかを考えるという大きな 2 つのプロセスが必要である.

本研究では自動伴奏に必要な楽譜追跡, 再生方法の検討のうち前者の楽譜追跡に注目した.

実演奏においては, 演奏された音を単純に楽譜と対応させるだけでは両者の正しい対応をとることはできない. なぜなら, 実演奏には演奏音の間違いによる音の不一致, 演奏音の欠落や挿入, 演奏タイミングの不一致といった誤りが発生するからである. さらに曲の練習の際にはフレーズの弾きなおしや演奏箇所のスキップといった可能性も想定しなくてはならない.

演奏の位置解析の手法としては,HMM(隠れマルコフモデル)を用いたもの [1] や,DP マッチングを用いたもの [2], DP マッチングを用いた手法を拡張したもの [3] がある.
DP マッチングを用いたものはフレーズの弾き直しや跳躍に対応できない, その手法を拡張したもの, HMM を用いたものについては, 事前に演奏データを収集し, それをもとに解析を行っている.

そこで本研究では楽曲の構成要素のみを事前に準備し, 事前のデータ採取を一切行わずに位置解析を実現する手法を提案する.

2. 提案手法

2.1 概要

本手法の特徴は現在位置の可能性のある地点について,DP マッチングで演奏誤りを判断し, その結果からその地点のスコアを決定し, すべての地点の中でスコアが一番低い地点を現在位置とするものである.

このとき, ある 1 地点が現在の演奏位置であるという仮定を 1 つの解釈と呼ぶことにする.

準備として, システムにはあらかじめ演奏されるべき音の音高と演奏される時間を読み込ませておく.

まず, 音の入力があると入力された音と同じ音がある楽譜上の地点を見つけ出し, 解釈とする.

その後, 各解釈について,DP マッチングを行い, 最適経路とコストを計算する.

最後に, 最適経路から誤りの種類を判定し, ペナルティを与える.
なお, 本研究で考慮する演奏誤りは,

- 演奏音の誤り
- 演奏音の欠落
- 演奏音の挿入
- 演奏のタイミングのずれ

の 4 種類とする.
この誤りを楽譜表記にすると, 以下のようになる.

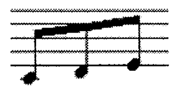

図 1: 正しい演奏

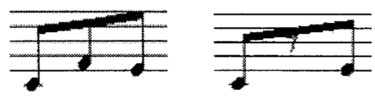

図 2: 演奏音の間違い (左), 演奏音の欠落 (右)

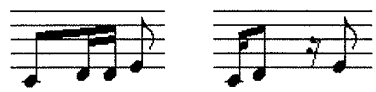

図 3: 演奏音の挿入 (左), 演奏タイミングのずれ (右)

このコストとペナルティを合わせたものを各解釈におけるスコアとし, 考慮されているすべての解釈の中で一番低いスコアを持つ解釈を採用しその解釈が示す現在位置を演奏の現在位置と判断する.

この処理を音の入力ごとに行う.

2.2 DP マッチングによる間違いの判定

入力音に対して, 楽譜中で現在位置である可能性のある地点が抽出されるのでその各地点について DP マッチングを行い, スコアと最適経路を計算し, 最適経路から演奏誤りを判定する.
DP マッチングは縦軸に読み込んである演奏情報, 横軸に実際に入力された演奏をとり,1 マスの間隔は 16 分音符 1 つ分としている.

縦軸のある地点について, その時間に音が弾かれる場合には,

[†]東京理科大学大学院理工学研究科情報科学専攻
Dept. of Information Sciences, Graduate School of Science and Technology Tokyo University of Science
[‡]東京理科大学理工学部情報科学科
Dept. of Information Sciences, Tokyo University of Science

その場所には音高情報が入り, なにも弾かれない場合には $null$ が格納されている.

横軸に関しては音が入力されると対応された位置に入力された音が格納されるようになっている.

横軸を x,x に格納されている音を S_x, 縦軸を y,y に格納されている音を S_y とすると,DP マッチングの際に S_x と S_y を比較したときのコストの増加分を以下のように設定する.

- $S_x \neq S_y$ かつ ($S_x \neq null$ または $S_y \neq null$) $\rightarrow 1$
- $S_x \neq S_y$ かつ ($S_x, S_y \neq null$) $\rightarrow 3$

さらに最適経路 (x, y) に注目したとき, 入力音 (S_x) と演奏誤りの対応付けは以下のようになる.

なお $judgex$ は音 S_x について誤り判定がされたかどうか, $judgey$ は音 S_y について対応する音 S_x が入力されたかを示す.

- 演奏音の誤り
  $S_x \neq S_y$ かつ $S_x, S_y \neq null$ かつ $judgex, judgey = false$

- 演奏音の挿入
  $S_x = S_y$ かつ $S_x, S_y \neq null$ かつ $x \neq y$ かつ $judgey = true$
  $S_x \neq S_y$ かつ $S_x, S_y \neq null$ かつ $judgex = false$ かつ $judgey = true$
  $S_x \neq null$ かつ $S_y = null$ かつ $judgex = false$

- 演奏音の欠落
  $S_x \neq S_y$ かつ $S_x, S_y \neq null$ かつ $judgex = true$
  $S_x = null$ かつ $S_y \neq null$ かつ $judgey = false$

- 演奏タイミングのずれ
  $S_x = S_y$ かつ $S_x, S_y \neq null$ かつ $x \neq y$ かつ $judgey = false$

2.3 スコアの決定

ある解釈について,DP マッチングで得られたコストに, 以下の条件でペナルティを加える.

2.3.1 過去の入力をどれだけ考慮しているか

解釈が発生する時間によって, どれだけ過去の入力を考慮してマッチングするかが変わってくる.

たとえばある 2 つの解釈のスコアが同じになったとしても,1 つは過去 5 回の入力にさかのぼってマッチングをした結果のスコアで, もう 1 つは過去 1 回の入力にさかのぼってマッチングをした結果のスコアだとしたら, そのスコアの妥当性は前者のほうが高いと考えられる.

今回は約 1 小節分の長さまでさかのぼっていればスコアが妥当であるという仮定の下でこのペナルティの計算を行う.

この条件を考慮するためのペナルティを p_{count} として,

$$p_{count} = average^2 - matchingcount^2$$

で計算する. このとき $average$ は曲において 1 小節に入っている音符の数の平均,$matchingcount$ は何回分過去の入力を考慮しているかを示す.

また,p_{count} の値は最小で 0 とする.

2.3.2 演奏音の間違いに対するペナルティ

フレーズの弾き直し, 跳躍が発生したとき, 今まで採用してきた解釈は採用されるべきではなく, そのためにその解釈のスコアを高くする必要がある.

フレーズが飛んだとき, 今まで採用してきた解釈については演奏音の間違いが蓄積されるので, これに関するペナルティを与えればよい.

このペナルティを p_{miss} として,

$$p_{miss} = misscount^2$$

で計算する. このとき $misscount$ は入力音が正しいと-1, 間違っていると+1 される 0 以上の値である.

2.3.3 演奏タイミングのずれに対するペナルティ

DP マッチングによる誤りの判定において, 演奏タイミングのずれの誤りがある解釈とない解釈で 2 つの解釈のコストが必ずしも変化するわけではない.

しかし誤りが発生しているわけであるから, 比較に用いる両解釈のスコアは誤りがある解釈のほうが大きくなる必要がある.

したがって、この誤りに対するペナルティを p_{gap} として,

$$p_{gap} = gapcount$$

で計算する. このとき $gapcount$ は誤り判定で演奏タイミングのずれと判定された回数である.

これらのペナルティと DP マッチングで得られたコスト ($cost$) を踏まえて, ある地点におけるスコア S を,

$$S = cost + p_{count} + p_{miss} + p_{gap}$$

で計算し, この値が一番低い地点を現在位置として認識する.

3. 実装と評価

今回提案した手法を Java により実装し, 演奏誤りを含んだ演奏において, 現在位置を正しく追従できるか, フレーズの弾き直し, 跳躍を含んだ演奏においてどの程度の時間で正しい位置を認識できるかを評価した.

3.1 実験 1:演奏誤りのある演奏における評価

評価のサンプルとして,7～10 小節程度の MIDI ファイルを 5 つ用意し, 曲中に含まれる間違いの割合を 10%,20%,30%とした時の現在位置の認識率を調べた. 結果は表 1 のとおりである.

表 1: 演奏誤りのある演奏における現在位置の認識率

誤り出現率	A	B	C	D	E
10%	100.0%	97.3%	92.5%	91.6%	97.4%
20%	100.0%	94.5%	97.5%	94.2%	94.8%
30%	97.4%	91.8%	95.0%	91.4%	92.3%

3.2 実験 2:フレーズの弾きなおし, 跳躍があるときの評価

サンプルの MIDI ファイル 5 つについて, フレーズの弾き直しを発生させたもの, フレーズの跳躍を発生させたものをそれぞれ用意し, 弾き直し, 跳躍の発生地点から何回目の入力で正しい位置を認識できたかを調べた. 結果は表 2 のとおりである.

表 2: 誤りが発生してから位置を認識するまでの時間

誤りの種類	A	B	C	D	E
フレーズの弾きなおし	4 回目	1 回目	4 回目	5 回目	5 回目
フレーズの跳躍	5 回目	1 回目	2 回目	10 回目	4 回目

本手法では, 音の入力があると現在位置解析をするシステムになっているため, 入力回数で評価している.

4. 考察

4.1 実験結果について

実験 1 において, いずれの状況でも 90%以上の認識率を得ることができたので, この手法は演奏誤りを含む演奏についても位置解析を行うことが可能であるといえる.

この実験において、入力があった時に正しい位置を認識できなかった時には大きく 2 つの状況が見られた.

1 つは正しい位置と実際に認識した位置が少しだけ離れていたという状況である.

この状況が発生した原因としては，演奏のタイミングがずれて入力されたときに発生した解釈が最適なものとして選択され，その時のずれの分だけ認識位置が離れてしまったからであると考えられる.

この状況で発生した解釈のスコアは時間が経つにつれて大きくなっていくが，値が大きくなる前に入力音の誤りがいくつか発生してしまうと本来採用されるべき解釈よりもこちらの解釈の値のほうが低くなってしまうという計算結果になっていた.

この状況を改善する対策としては，タイミングがずれて音が入力された際に発生した解釈についてはそのずれ分を記録しておき，その解釈が示す現在位置を補正するということが考えられる.

もう 1 つは正しい位置と実際に認識した位置がかなり離れてしまった状況である.

この状況が発生してしまった原因としては，間違って入力された音が偶然他の位置の正しい演奏と同じになってしまったため，その位置を現在位置と認識してしまったからであると考えられる.

現在の位置解析の手法では、楽譜の情報と入力された音の情報を比較し，そこにずれが生じたら値を操作するという手法のため, この状況ではどうしても解釈のスコアが低くなる傾向にあり, 誤認識を 0 にするというのは難しい.

しかし, その後の 1,2 回入力によって正しい位置を認識することはできているので, 一度誤認識が起こってから誤認識がずっと続くといった状況は起こらない.

実験 2 の結果については,1 つの状況を除き,5 回以下の入力で位置を認識できた. フレーズの弾き直しや跳躍が発生したとき, 瞬時にそれを判断して対応するのは, 人間同士の演奏でも非常に困難であり, 約 1 小節程度の時間は必要であると考えられる.

この実験結果で得られた入力回数は, 各サンプルにおける 1 小節以下の長さに相当する入力であるので, 弾き直し, 跳躍に対応できているといえる.

サンプル B の 1 回目の入力で認識できたという結果については, 弾き直し, 跳躍が発生する前にそれらが発生した場合を考慮していたためと考えられる.　サンプル D において、認識に時間がかかった理由としては，サンプルの曲中に全く同じフレーズが複数存在しており，認識してほしい位置とは別のところを認識してしまったためである.

この実験結果から同じフレーズが複数ある場合に，どのようにしてそのフレーズを差別化するかが精度向上への課題であるといえる.

4.2 既存手法との比較

楽譜追跡を行うための手法として,HMM を用いたもの [1],DP マッチングを用いたもの [2][3] がある.

HMM を用いた手法については, ユーザの演奏データをもとに次の動作を予測していくものなので事前にデータを集める手間が必要である.

DP マッチングを用いる手法については, 曲の音高情報のみ用いて現在位置を判断することが可能なので, 汎用性が高い手法と言える. しかし単純に DP マッチングを用いただけではフレーズの弾き直しや跳躍に対応できず,1 度誤認識をしてしまうと正しい位置を認識できないという問題点がある.

この DP マッチングを用いた手法を拡張し, 弾き直しに対応できるようにした手法 [3] も提案されている.

この手法は練習時の演奏の記録から弾き直しに戻りやすい場所を調べ重みづけを施して現在位置の候補に加えることで弾き直しの誤りに対応している.

今回提案した手法は演奏者からのデータの採取を行わず，楽曲の構成要素のみを利用し，位置解析を行っていることから，ユーザや楽曲に依存することなく解析ができるということが利点としていえる.

音楽は曲のジャンルも様々であり, それを演奏する人間も様々であるから, 自動伴奏を実現させるうえこの利点は重要であると考えられる.

4.3 今後の展望

4.3.1 自動伴奏システムの作成に向けて

今回考慮した演奏誤りのほかに, 位置解析をするうえで考慮する必要があるのが, 演奏のテンポの変化の追従である.

テンポの認識は楽譜情報から得られる音の並びと, 実際に入力された音の並びとを照らし合わせればある程度の認識は可能であると考えられるが, ここで問題となるのが, 入力された音が演奏すべき音なのか, そうでない音なのかということである,

今回, 演奏誤りを含んだ演奏についても頑健な演奏位置解析

を提示できたことから, それを利用してこの問題を解消し, テンポの変化も追従することが可能になると考えられる.
また, 今回演奏誤りを想定した MIDI ファイルを演奏者と仮定して評価を行い, 高い認識率を得ることができたので, 今後は実演奏を対象とした評価を行っていくことで問題点を見つけ出し精度を上げる必要がある.

4.3.2 ツールとしての応用

より人間の演奏に近い自動伴奏のシステムにするために, 人間の演奏モデルを考慮した伴奏システムの研究がなされている.

なかでも演奏者が複数となったときは, 演奏者間でのずれなどが生じ, 演奏者が 1 人のときとはまた違った問題を考慮する必要がある.

このような状況を想定した自動伴奏システムの研究 [4] がされているが, 具体的にどのような条件下でどのような振る舞いが適切なのかということを明示的に示しているわけではない.

そこで, 現在作成しているシステムをツールとして利用し, 複数の演奏者が存在するときにどういった挙動をすれば聴衆にいい印象を与える事ができるのかということを調査していきたい.

5. おわりに

今回,DP マッチングを用いて, 事前のデータ採取を行わず, 楽曲の構成要素のみを情報とした演奏の現在位置解析を試みた.

楽譜中の現在位置の可能性のある地点について,DP マッチングを用いて演奏誤りを判断, 誤りの種類からスコアを計算し, そのスコアの比較によって演奏の現在位置を解析した. その結果, 演奏誤りを含んだ演奏や, フレーズの繰り返し, 跳躍を含む演奏に対しても現在位置を解析できることを示した.

参考文献

[1] 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,2006-MUS-86,pp.109-116,2006

[2] Bloch,J.J and Dannenberg, R.B. : Real-Time Computer Accompaniment of Keyboard Performances, In Proc. of ICMC, pp.279-289,1985

[3] 大島千佳, 西本一志, 鈴木雅実:家庭における子供の練習意欲を高めるピアノ連弾システムの提案, 情報処理学会論文誌,vol.46,No.1,pp.157-171,2005

[4] 堀内靖雄, 藤井敦, 田中穂積:複数の人間と強調する演奏システム, コンピュータソフトウェア,vol12,no5,pp.63-71,1995