

音響入力に対する自動伴奏システムの開発 -楽器演奏者の練習支援に-

1. 背景

合奏曲の演奏者にとって、個人練習とは必要不可欠なものである。しかし、単純な個人練習は、自分の演奏パート音しか聴こえなくて味気ないものであり、演奏者の練習へのモチベーションを低下させることにも繋がりがかねない。

このような必要な個人練習に対し、演奏者の練習意欲を増大させる安直な解決策として、第一に毎回合奏練習を行う、第二に伴奏音トラックを背景で流す、という案が浮かばれる。しかし、前者は可能であれば理想的な解決策だが、実際に毎回個人の練習の度にすべてのパートを集めて練習を行うのは非現実的である。後者は一見実現可能な妥協案だが、演奏者が独自のテンポで演奏できず、伴奏音に引っ張られてしまうというのは不愉快な練習となる可能性が高く、練習効果も高いとは言えない。

2. 目的

本プロジェクトの目的として、背景で挙げた第二の解決策のように手軽で、かつ第一の解決策のように全パートを集めたような臨場感のある練習を行うことができるようにするため、演奏者の演奏箇所を自動的に追跡し、対応する伴奏音を再生する自動伴奏システムの開発を挙げる。このようなシステムは、巨大な合奏曲の演奏者に対する補佐として活躍することが期待できるのみならず、未だに合奏曲を体験したことが無い楽器演奏者にも手軽な体験ツールとして使用されることも考えられる。さらに、本来ソロ演奏で使われることのない楽器に対しても、初心者に対する導入のし易さを増大させることが期待される。

3. 開発の内容

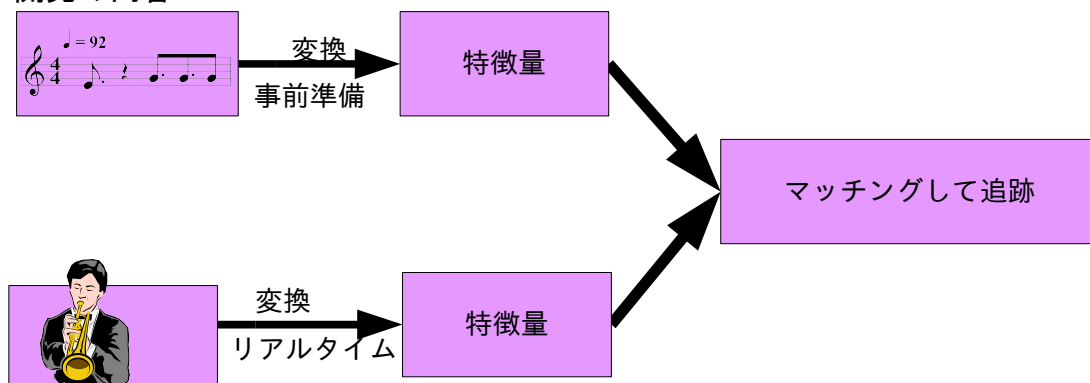


図 1: 本システムの処理の流れ

開発したシステム全体の処理の流れを図 1 に示す。本システムの実装では

ActionScript を用いた。図 1 の処理を実現するための内部機能は、リアルタイム音響信号入力・音響特徴量抽出・楽譜追跡の 3 つに分けられる。

図 1 左下で示すユーザの演奏データをリアルタイムで取得するために必要となるリアルタイム音響信号入力機能の実装は、ActionScript の EventListener でイベントの変化と処理の実行を連続的に組み合わせることで実現した。リアルタイムに音響信号データを処理し続けるためには、マイクロフォンから入ってくる音響信号データのバッファと音響信号データの処理のための関数を結合してやればよい。音響信号データの処理のための関数の中に、音響特徴量抽出処理を記述することで、リアルタイムに音響演奏から特徴量への変換を行うことが可能となった。

音響演奏と楽譜は全く別の次元の存在であるため、共通の土俵で比較、追跡を行うために、音響特徴量抽出機能が必要となる。本プロジェクトで採用した手法は、まずシステムは用いる楽器に固有の特徴量を前もって学習しておき、それに伴って事前に楽譜を特徴量の系列に変換しておく（図 1 上）。次に、マイクロフォンから逐次入力される音響演奏信号を、フーリエ変換した後特徴量に変換する（図 1 下）。この処理をリアルタイムに続けることで、演奏の特徴量の系列がリアルタイムに得られる。

最後に、楽譜追跡部分において音響演奏の特徴量の系列と楽譜の特徴量の系列を比較し、演奏者が現在楽譜上のどの位置にいる可能性が一番高いかを検索する（図 1 右）。単純に毎時間全ての楽譜上の位置を調べるのは計算量的に困難な問題になる。そこで、前の時間までどの位置にいたかの可能性を記憶しておくことにより計算量を削減し、次の時間の計算に活かす手法を用いた。これは動的計画法の原理に基づいた効率的なアルゴリズムである。現在時刻までの音響演奏の特徴量の系列と楽譜の特徴量の系列の差異を最小化するような位置を、現在演奏者が演奏している可能性が一番高い楽譜上の位置として返し、伴奏を再生するようにする。

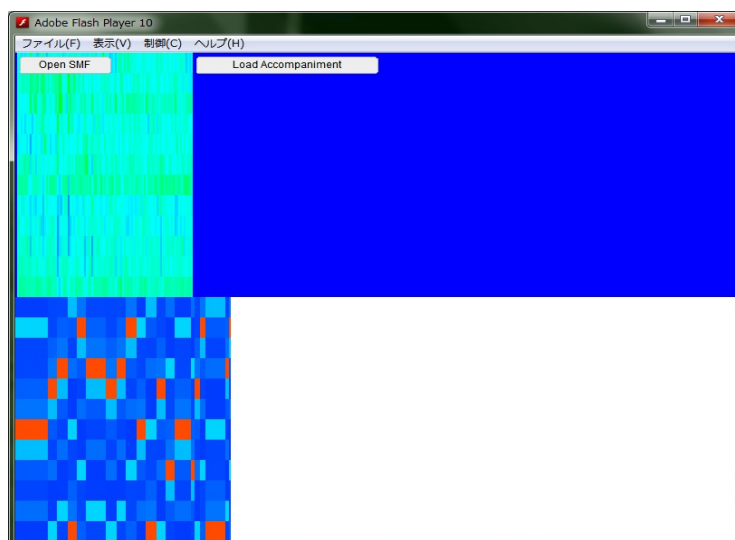


図 2: システムの GUI

開発したシステムの GUI を図 2 に示す。上段が演奏者の演奏からリアルタイムに抽出したクロマベクトル、下段が楽譜から事前に抽出しておいたクロマベクトルである。演奏者が正しく演奏信号が抽出できているかを容易に視認できるように、リアルタイムでクロマベクトルを描画するシステムとなっている。ユーザーは演奏したい Standard MIDI Format (SMF) の楽譜ファイルをシステムに読み込ませる。SMF はトラック毎にデータが格納され、トラック内は発音・消音・テンポ変動などのイベントの列で埋められている。演奏トラックの音名さえ判明すれば、楽譜追跡のための楽譜のクロマベクトル系列を作成することができるので、演奏トラックから音名を抽出している。SMF における時間単位は実時間ではなく小節単位となっているため、テンポから伴奏時間を自動で計算し、演奏者のクロマベクトル系列とフレームの大きさを合わせるようにしている。

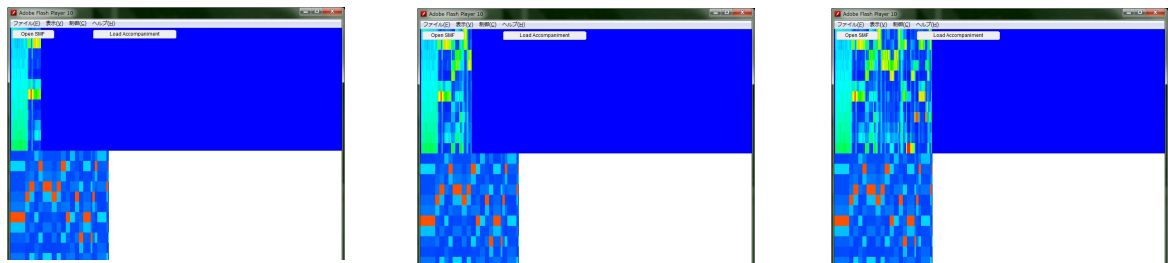


図 3: クラリネットを用いた伴奏での GUI

図 3 に、実際にクラリネットで楽譜を演奏している際の GUI 上の追跡をしている様子を示す。図 3 の例では、楽曲に Mozart の Clarinet 5 重奏 (K. 622) の一部を利用した。上段の演奏者のクロマベクトルが、演奏に合わせて順に描画されていく様子が示されている。

4. 従来の技術（または機能）との相違

既存の研究などでは、単一の楽器のみにしか対応できなかったり、MIDI 信号に対応した電子楽器のみにしか対応できていなかったが、演奏したい楽器があればそこから特徴量を学習することが容易なため、豊富な楽器の種類に対応することが可能になった。

5. 期待される効果

本プロジェクトの結果により、合奏曲演奏者への練習支援が効果的に行われる。それに伴って、時間の取りづらいビジネスマンなどのオーケストラ参加などを支援することが可能になると考えられる。

また、クラリネットやトロンボーンなど、ソロ演奏には普通用いないが合奏曲に頻

繁に用いられる楽器に対して、初心者の導入のハードルを大幅に下げることができると考えている。従って、本ソフトウェアの存在を同時にアピールすることで、楽器産業にも貢献できると考えられる。

6. 普及（または活用）の見通し

flash ソフトウェア開発コミュニティサイト Wonderfl を通してソフトウェアを公開することで、ユーザー及びエンジニアに同時にアピールすることができる。インターネットさえあれば手軽に利用できることから、PC 初心者のユーザー層までユーザー獲得できるのではないかと考えている。また、ユーザーからのフィードバックやエンジニアによる改善なども容易に受けることのできる見通しが立つ。

7. クリエータ名（所属）

鈴木 孝輔（東京大学大学院 情報理工学系研究科）

（参考）関連 URL

<http://wonderfl.net/c/hPty/>