E-005

# リズム練習支援のためのリアルタイム発音検出手法の検討
# Real-time Onset Detection for Rhythm Practice Supporting System

松尾 章弘†，原 佑輔†，土江田 織枝†，山田 昌尚†
Akihiro MATSUO, Yusuke HARA, Orie DOEDA, Masanao YAMADA

## 1. はじめに

音楽を演奏する上でリズムと音高は基礎的で重要な要素である．演奏した音高を数値や視覚的に知る方法としてはチューナが普及しているが，リズムに関してはそのような外部的な補助手段で客観的に数値化，可視化する方法は一般化していない．関連研究として，ピアノ演奏[1]やドラム演奏[2]に関するものがあるが，これらは MIDI 楽器などを発音検出の手段としており，管楽器などには適用できない．また，発音検出に関する研究において，様々な方法でリアルタイムな発音検出が行われてきたが，それを実際に何らかのシステムに応用している例は少ない．そこで本研究は，単音の楽器を対象として，リアルタイムな発音検出手法について検討するものである．最終的には演奏された発音タイミングをリアルタイムに可視化することでリズム感向上のための練習の補助となるシステムの構築をすることを目指す．

## 2. 発音検出

音響信号から発音のタイミングを得ることを発音検出という．発音検出は一般的に，前処理，検出関数，ピーク抽出の 3 段階からなる[3]．第 1 段階の前処理としては正規化を施す．第 2 段階の検出関数として，周波数ごとの信号スペクトル強度を利用するスペクトルフラックスや HFC (High Frequency Content)，位相の変化を利用する方法など，各種が提案されている．本研究では，周波数ごとの信号スペクトル強度の変化が大きい場合に発音となるピークが現れるスペクトルフラックスを使用する．スペクトルフラックスは次式で表される．

$$SF(n) = \sum_{k=1}^{\frac{N}{2}-1} \max(0, |X(n,k)| - |X(n-1,k)|)$$

ここで$N$は FFT フレームのデータ数，$n$は時間，$k$は周波数であり，$X(n,k)$はスペクトログラム (時間周波数信号) を表す．0 とのmaxをとることでスペクトル強度が増加する場合のみを対象としている．第 3 段階のピーク抽出では，しきい値を超える局所最大値 (local maxima) 検出し，その時刻を発音時刻とする．しきい値として次式による動的しきい値を用いる．

$$TH(n) = \delta + \lambda \cdot \mathrm{median}(SF(n-v_1 : n-v_2)) + \alpha \cdot \mathrm{mean}(SF(n-v_1 : n-v_2))$$

ここでδはしきい値の定数項，$\lambda$ および$\alpha$はそれぞれ中央値，平均値に対する重みであり，$v_1, v_2$は動的しきい値の対象幅を表す．このしきい値を用いて，検出関数$DF(n)$および発音時刻$OD(n)$を次のように求める．

$$DF(n) = SF(n) - TH(n)$$

$$OD(n) = \begin{cases} 1, & DF(n) > 0 \ and \ \underset{w_1<m<w_2}{\mathrm{argmax}} \ DF(m) = n \\ 0, & otherwise \end{cases}$$

ここで$w_1, w_2$は local maxima の探索範囲である．

## 3. システム構成

図 1 にシステム構成を示す．構築環境は Windows である．電子メトロノーム音の生成とマイク入力からの信号処理を ChucK で行い，表示には Processing を用いる．ChucKはプリンストン大学で開発された実時間音楽処理用のプログラミング言語である．後述する OSC を用いることができる点や，音の生成，信号処理を並列処理できることからこの言語を用いた．

マイクから入力した信号の A/D 変換にはフリーの ASIO エミュレーションドライバである ASIO4ALL を使用する．これは A/D 変換に Windows 標準のオーディオエンジンを使用すると 100～200ms のレイテンシーが生じるためである．サンプリング周波数は 44.1kHz でモノラル入力とし，この信号を 1024 サンプルのフレーム幅でハニング窓をかけて短時間フーリエ変換 (STFT) により時間周波数情報に変換した．STFT のホップ幅は 10ms であり，これがシステムの時間解像度となる．この信号を前節で述べた方法で発音検出した．

メトロノーム音は ChucK で生成し，スピーカから出力する．強拍と弱拍は周波数で区別する．強拍は 880Hz，弱拍は 440Hz とした．システムのユーザはこのメトロノーム音を聴きながら楽器を演奏する．ChucK から Processing へメトロノーム音と発音検出結果のタイミングを送るために OSC (Open Sound Control) を用いる．OSC は電子楽器およびコンピュータ間で音楽演奏データ等を送受信するための通信プロトコルである．OSC では URL 形式でデータを送るため，メトロノームのタイミングとオンセットを容易に区別して情報伝達をできる．

Processing でのメトロノームと発音検出のユーザインタフェイスはいくつかのパターンを用意した．実際のメトロノームに近い振子状のもの，針が円状に回転するもの，拍ごとに画面を分割して表示するものを作成した．例として拍ごとに画面を分割して表示したユーザインタフェイスを図 2 に示す．演奏はホルンで行ったときのものである．画面を横に 4 等分し，1 拍間かけて左端から右端へと黒い棒を動かす．この棒がメトロノームの視覚的役割を持つ．次の拍でもう一段下から黒い棒が動き，4 段目も終わると 1 段目に戻って，同様の動作をする．縦にある線は拍を 4 等分していて 16 分音符の長さに相当する．ChucK から発音検出した時刻を受信したとき，その時点のメトロノームの棒の位置に点を打つ．この点が一定の数を超えた場合，点をクリアする．

## 4. 実験

発音検出の動作を確認，評価するために 2 種類の実験を実施した．なお実験は室内残響の影響を排除するため無響

†釧路工業高等専門学校　National Institute of Technology, Kushiro College

室で測定している．1 つめの実験は，スピーカから ChucK で生成したメトロノームの音を出力し，そのままマイク入力，リアルタイムで発音検出するものである．その結果，約 20ms の一定した遅延があるものの，100%の発音検出ができた．2 つ目の実験として，フルート (Fl)，トランペット (Tp)，ホルン(Hr) の 3 種類の楽器で録音した音の発音検出を実施した．録音したパターンは変ロ長調で 1 オクターブのタンギングとレガートの演奏 (48 音，約 15 秒) である．前節で述べたパラメータは$v_1$ = 100ms，$v_2$= 0ms，$w_1$ = 50ms，とした．$w_2$を 0～100ms の間で変化させ，検出結果の違いを比較した．正解とする発音タイミング (ground truth data)は入力信号の振幅エンベロープとスペクトログラムに基づいて 3 名でアノテーションした時刻の平均を用いた．一般的には，この ground truth data に対して検出した発音が前後 25～30ms の範囲に入っているかどうかで評価することが多いが，本研究では前に 25ms，後に 55ms の範囲で発音検出されていれば True Positive と判断した．管楽器では，息を吹き込み始めた瞬間が ground truth として判断されるのに対して，実際に演奏者や聴衆に音として知覚されるタイミング，つまり音楽演奏として意味のある発音時刻はそれよりも遅くなることを考慮したためである．ただし，検出された発音時刻と聴覚上の発音と判断されるタイミングの関係は現時点では明らかではない．

## 5. 結果

3 種類の楽器で，それぞれ$w_2$を変化させ検出を行った結果を図 3 に示す．$w_2$はある時刻がピークかどうかを求めるためにその時刻よりも後の時刻まで探索範囲を広げるものである．どの楽器においても$w_2$が 30ms を超えると F 値が 0.6 を超えるようになった．探索範囲の$w_2$を変化させることにより検出結果は大きく変わることがわかった．しかし，リアルタイムな処理で ピークかどうかを求めたい時点よりもあとのデータを使うと，その分だけデータを待つことになるので遅れが生じる．その遅れの対処として，Processing が ChucK から OSC により発音となったタイミングを受け取り，ディスプレイに描画するときにはその遅れの分を考慮して描画を行う．この遅れは 0～数 10ms の遅れであるが，実際にシステムをユーザが使う場合，自分の発音したタイミングをメトロノームに比べてどれくらいずれているかを知るという目的からみて問題のない遅れといえる．また，図 2 は Hr のデータで発音検出を行ったときの結果を示しており，False Positive に印を付けている．次に，$w_2$を変化させたとき，それぞれの楽器で最も F 値の高かった発音検出結果を表 1 に示す．トランペットで F 値が 0.980，ホルンで 0.970 となった．再現率も高い．False Positive が数個まで下がることに成功している．フルートは 48 個の発音で，False Negative が 7 個，False Positive は 6 個あるものの，F 値は 0.863 となった．$w2$が 0 のときと比較すると F 値が 0.662 から大きく上がった．

## 6. まとめ

発音検出を行い，その精度の向上と評価を目的に実験を行った．今回の実験で，発音かどうかを判断するには事前の情報だけではなく，事後の情報も大きく精度に関係することがわかった．いくつかの誤検出や未検出があるものの良い結果が得られた．また，メトロノームと同期した発音検出システムの構成を行い，複数のユーザインタフェイスを作成した．今後，検出方法に関して検出精度を高めるために別な手法で実験すること，ユーザインタフェイスに関しては，ユーザにとって使いやすくするために，多くの人に実際に使用してもらい，検出結果やメトロノームの表示方法を改善していくことが課題である．

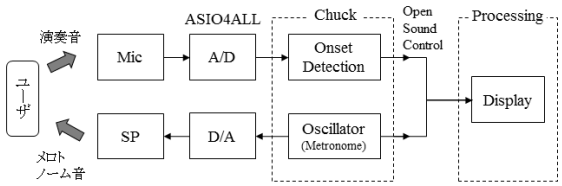


図 1 システム構成図

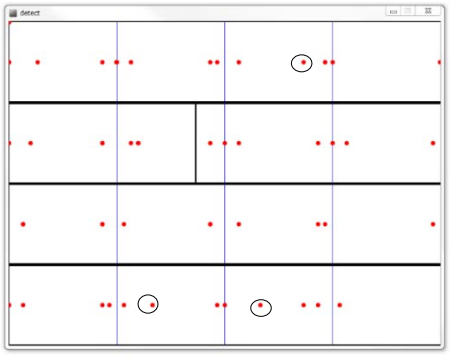

図 2 発音検出実行画面

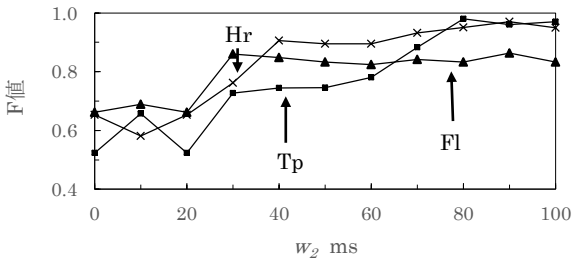


図 3 探索範囲を変化させたときの F 値

表 1 楽器での発音検出結果

| 楽器 ($w_2$) | 適合率 | 再現率 | F 値 |
|---|---|---|---|
| Fl (90) | 0.872 | 0.854 | 0.863 |
| Tp (80) | 0.980 | 0.980 | 0.980 |
| Hr (90) | 0.941 | 1.000 | 0.970 |

### 参考文献

[1] 竹川佳成，寺田努，塚本昌彦，“リズム学習を考慮したピアノ演奏学習支援システムの設計と実装”, 情報処理学会論文誌,第 54 巻, 第 4 号, pp.1383-1392 (2013)
[2] 岩見直樹, 三浦雅展，“MIDI 楽器を用いたドラム演奏 練習支援システムの提案”, 情報処理学会研究報告[音楽情報科学] pp.85-90 (2007)
[3] Bello, Juan Pablo, et al., “A tutorial on onset detection in music signals”, Speech and Audio Processing, IEEE Transactions, Vol.13, No.5, pp.1035–1047 (2005)