オーディオビジュアルパフォーマンスのための自由発話に対応したライブインタフェース

大谷泰斗[†1]　越智景子[†2]　大淵康成[†2]

概要：近年，ラップトップをステージ上に持ち込み映像と音響表現からなるオーディオビジュアルパフォーマンスが行われるようになった．これらのパフォーマンスは，高次元のパラメータ操作による映像・音響表現を可能にする．一方で，伝統的な楽器演奏の持つ即興的な表現力を持たない．そこで我々は，以前の研究で伝統的な楽器演奏にオーディオビジュアル表現を持ち込むためのマイク・ギター・フットペダルを用いたライブインタフェースを提案した．このインタフェースは，音声発話を用いる事で，音楽ソフトウェアや映像ソフトウェアの持つ高次元のパラメータの操作を可能にする．これにより，従来のフットペダル等によるパラメータ制御より多くのパラメータ制御を可能にする．しかしながら，従来のシステムではボイスコマンドを正確に覚える必要があり，一般的な楽器演奏家にとっては，直感的な取り扱いが難しいといった問題がある．

本稿では，このアイディアを拡張し，大語彙連続音声認識と自然言語理解により，フレーズから意図解釈することで，演奏者は自由発話によるボイスコントロールが可能になり，より直感的な操作感覚を持つインタフェースを目指す．

1. はじめに

近年，電子音楽や実験音楽に限らず，シーケンサやシンセサイザ・サンプラといった音楽アプリケーションや映像アプリケーションをステージ上に持ち込み，高度な音響と映像によるオーディオビジュアルパフォーマンスが行われている．これらのシステムは，大量のパラメータをノブやスライダーといったインタフェースを使用して制御したり，演目に合わせて，あらかじめプログラミングする事で制御される．これらのインタフェースと伝統的な楽器を共存させることは難しい．そのため，伝統的な楽器演奏の持つ直感的な表現力を持たせることは難しく，即興的なパフォーマンスは難しい．

そこで，我々は以前の研究[1]で，伝統的なエレキギター演奏のスタイルを基に，マイク・ギター・フットペダルを組み合わせることにより，伝統的なエレキギター演奏にオーディオビジュアルパフォーマンスを持ち込むインタフェースを提案し，音楽表現の拡張を目指した．これは音楽・映像アプリケーションの持つ高次元のパラメータをボイスコマンドにて指定し，ギター入力やマイク入力と組み合わせることで操作できるインタフェースである．これにより，伝統的なギター演奏スタイルに基づき，高次元のパラメータの操作を可能にした．しかしながら，このインタフェース使ったパフォーマンスでは，パラメータに割り当てられた正確なボイスコマンドを発話する必要があった．そのため，ボイスコマンドを記憶する必要があったり，言い間違いや即興的な発話フレーズに対応できないといった問題があった．また，日常的なボイスコマンド方式のインタフェースでは，多くのユーザーはマニュアルに記載されたコマンドに目を通さずに，直感的な発話を行うこと多いとされている[2]．この問題に対して，あるパラメータあたりに，複数のボイスコマンドを割り当て，演奏者の発話表現の揺れに対応させることができるが，多種多様な表現の違いを事前に網羅することは難しい．そこで，この多種多様な演奏者の発話に対応するため，大語彙連続音声認識を用い，その書き起こし結果を自然言語理解させる．これにより，演奏者の負荷を軽減し，より直感的なインタフェースとして，演奏者の意図した音楽・映像アプリケーションの機能を呼び出せるようにする．

本論文で提案するシステムでは，以前の研究で提案したインタフェースを基に，この多種多様な演奏者の発話に対応するため，大語彙連続音声認識を用い，その書き起こし結果を自然言語理解させる．これにより，演奏者の負荷を軽減し，より直感的なインタフェースにより，演奏者の意図した音楽・映像アプリケーションの機能を呼び出せるようにする．本論文では，音声認識モジュールを拡張したプロトタイプの設計と実装について述べる．

2. 先行事例

2.1 オーディオビジュアルパフォーマンス

まず，音楽と密な関係性を持った芸術的な映像提示によるライブパフォーマンスのことをオーディオビジュアルパフォーマンスと呼ぶ[3]．録音音楽を芸術的にミックスするディスクジョッキー(DJ)に文字って，映像を芸術的にミックスすることをビジュアルジョッキー(VJ)と呼ぶ．VJ は，主

†1 東京工科大学大学院バイオ情報メディア研究科メディアサイエンス専攻
†2 東京工科大学メディア学部メディア学科

にテクノクラブなどで音楽パフォーマンスに合わせて，映像素材をマッシュアップしていたのが始まるとされる．VJには映像素材を混ぜ合わせるものや，音響解析を行い，音量などの音響的な特徴をトリガーに映像をジェネラティブに生成したり，映像に効果を与えるパラメータを動かしたりする手法がある[4]．例えば，Live Writing[5]は，詩をリアルタイムで観客へ提示するオーディオビジュアルパフォーマンスである．タイピングにより打ち出される文章と音によって，詩とタイピング音によるオーディオビジュアルパフォーマンスである．イギリス出身のアーティストであるMax Cooperは，EMERGENCE[6] など，多くのオーディオビジュアル作品を公開している．また，坂本龍一とアルバノトによる``UTP_"は，ピアノ・室内楽団・エレクトロに抽象的な映像表現を加えた作品[7]を発表している．紹介したこれらのパフォーマンスは，パフォーマの背景へディスプレイやプロジェクターによる投影により映像提示をしている．また，音楽的な要素や音楽を奏でるための動作（ジェスチャ）を視覚情報に変換したり，密に連動して映像内のオブジェクト変形や色の変化，映像シーンの切り替えなどによる映像提示との相互作用によるパフォーマンスである．

2.2 パフォーマンスのための音声インタフェース

ライブコンサートのための音声インタフェースの実用例は少ない．Fasciani ら[8] は，物理的に制限のあるサウンドシンセサイザの音色をボイスジェスチャにより，制御するためのボイスインタフェースを提案している．Igarashi ら[9]は，声の高さや大きさといった簡単な特徴量を使った非言語インタラクションを提唱している．音声入力は，自然なインタフェースとして，日常の様々な環境で利用されている．しかしながら，音楽インタフェースの分野では，遅延の問題のため，ほとんど利用されてこなかった．しかしながら，Deacon ら[10]は，音声認識とボイスジェスチャを組み合わせたシステムを提案している．これは，音声入力の長さにより，制御する対象が変化するインタフェースで，短い発話はボイスコマンドとして利用される．中程度の長さの発話は，シンセサイザの音色の選択に使われ，長い発話は音高やノート（音の長さ）の制御に使われるなどの工夫がされている．

我々が以前開発した伝統的な楽器演奏とオーディオビジュアルパフォーマンスの共存を可能にしたインタフェースでは，音声認識による遅延問題があった．これには，ペダル操作を加えることで対応した．しかしながら，高次元のパラメータのコントロールが可能とする．一方で，演奏者には正確なボイスコマンドの発話が要求されるといった問題がある．

2.3 音声認識システム

ライブパフォーマンスにおける音声認識インタフェース実用例は少ないが，カーナビゲーションシステムにおける音声認識インタフェースの研究事例は盛んに議論されてきた．車載機器では，運転操作が最優先タスクであり，走行中の手を使った操作はできないという点が挙げれる．これは，ライブパフォーマンス中は，楽器演奏が最優先タスクであり，手による他の機器の操作が難しいという制約と共通している．カーナビゲーションシステムにおいてもボイスコマンド方式が多く用いられてきた. 倉田らの研究では，大語彙連続音声認識と自然言語理解を組み合わせることで，ユーザの意図する機能を呼び出すカーナビゲーションシステムが提案されている[11].

本研究では, 大語彙連続音声認識と自然言語理解をモジュールに追加することで，演奏者の自由な発話により音楽・映像アプリケーションの内の意図した機能を呼び出すインタフェースを提案する．これにより，直感性と即興的なフレーズの変化に対応した音声コントロールインタフェースを目指す．

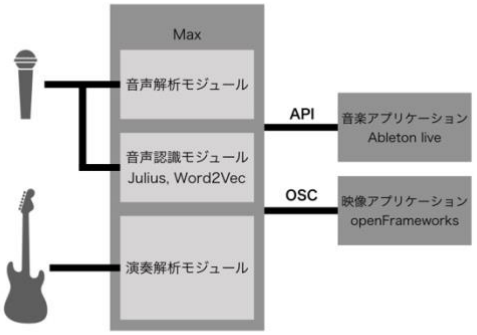

図 1 システム概要図

3. 設計と実装

本章では，音声フレーズの意味解釈により，ライブパフォーマンスシステムの操作が可能なインタフェースのプロトタイプの設計と実装について述べる．

本システムは，全体を統合するメインモジュール・音声解析モジュール・音声認識モジュール・演奏解析モジュールから構成される．図 1 にシステム概要図を示す．メインモジュールは Max で実装し，音声解析モジュール・音声認識モジュール・演奏解析モジュールを持つ．マイクからの入力は, 音声解析モジュールと音声認識モジュールに送られ，エレキギターからの入力は演奏解析モジュールに送られる．

音声解析モジュールは, マイクに入力される音声中の音の高さや大きさといった音響的な特徴を取得する．音声認識モジュールは，入力音声から言語情報を取り出す．さらに取り出された言語情報はあらかじめ登録された全てのフレ

ーズとの類似度を計算し，最有力候補のコマンドを呼び出し，映像や音楽アプリケーション内のパラメータの制御を可能にする. 演奏解析モジュールは, 演奏状態を解析する．

3.1 音声認識モジュール

音声認識モジュールは音声を入力すると, フレーズの書き起こしと自然言語処理を行い，入力フレーズに最も意味の近いコマンドを出力する．図 2 に音声認識モジュールの処理の流れを表す.

音声認識エンジンにはオープンソースの音声認識エンジンである Julius[12]を利用し，音声入力からフレーズを抽出する. Julius はマイクからの音声入力を受けると, 認識結果をフレーズとして出力する．この認識結果は，Max 内のシェルインタフェースを通して，文字列として受信される．その後，分かち書きを行う．正規化された文字列は，word2vec モデルにより，辞書に登録されたすべてのフレーズとの類似度を計算する．辞書に登録されたコマンドは，本インタフェースの起動時に分かち書きされる．辞書登録された全てのフレーズとの類似度が計算された後に最有力候補を探索し，Max へソケットを通して送られ，最有力候補のフレーズを送信する．フレーズ同士の類似度の計算の処理は，python で記述し，Max とは別スレッドで呼びだしている. 音声認識結果と自然言語処理のための入力は UDP 通信され，自然言語処理の計算結果もまた，直ちに UDP 通信でメインモジュールに送信される.

3.2 大語彙連続音声認識

大語彙連続音声認識には, オープンソースの音声認識エンジンである Julius を利用し，音声入力からフレーズの書き起こしを行う．Japanese Newspaper Article Sentences(JNAS) と Corpus of Spontaneous Japanese(CSJ)を基にした GMM-HMM 音響モデルを利用した．これらは Julius Dictation-Kit として公開されている. 音声区間検出は, Julius 側で行う.

3.3 自然言語理解

メインモジュールを実装している Max とは別に python スクリプトで実装した．Julius の認識結果は一度 Max で開発されたメインモジュールを通り，OSC(Open Sound Control) で文字列を受け取る．文字列は MeCab[13]によって分かち書きされ，単語ごとにフレーズを分割する．その後，word2vec を用いて各単語をベクトル化する．加えて，全ての単語の特徴ベクトルの平均を求める．これらの処理を入力フレーズと登録コマンド間で行い，すべてのコマンドと入力フレーズとのコサイン類似度を算出する．その後，コサイン類似度が最大のものを算出し，メインモジュールへ OSC で送信する. 今回は, 2017 年 1 月 1 日時点の Wikipedia のボキャブラリーを学習させたモデルを使用した．モデルは 300 次元の特徴ベクトルより生成されたものである.

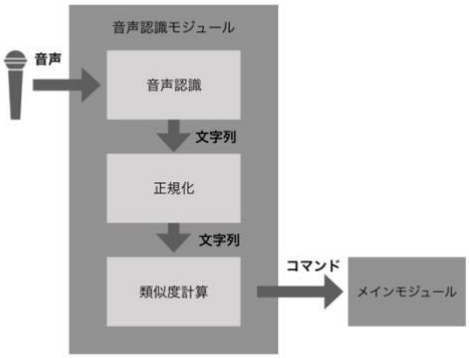

図 2 音声認識モジュールの処理の流れ

3.4 ギタージェスチャ

ギタージェスチャは以前の研究と同じモジュールを利用した．ギター演奏を通じてインタフェースを制御することをギタージェスチャと呼ぶ．このモジュールはピッチ・弦を弾く強さ・持続的な強いストロークを検出することができる． ギタージェスチャは，ギター入力の信号解析を行うことで取得する．ピッチは，フレームサイズ内の基本周波数によって定義される．今回のプロトタイプにおけるフレームサイズは約 43ms(理論値)とした．アタックは，ギター入力が閾値以下の状態から閾値以上になった場合に検出される．長さはアタックを検出してから，信号レベルが閾値以下になるまで区間により定義される．弦を弾く強さは，ギター入力のフレームサイズ内のピークの値から定義される．持続的な強いストロークは，ギター信号解析モジュールだけで定義せず，0．8 以上の強度になるとオブジェクトが生成される．一定以上の個数になった場合にトリガーが送信され，持続的な強いストロークとなる．オブジェクトは, 約 5 秒経過すると消滅する. 今回のプロトタイプでは，50 個以上のオブジェクトが生成された場合にトリガーが送信されるように設定した.

また, エレキギターからの信号レベルはピックアップの種類の影響を大きく受けるため，メインモジュールからそれぞれの検出器の閾値を調整できる．それぞれの検出されたパラメータは，ギタージェスチャとしてメインモジュールに送られる.

3.5 ボイスジェスチャ

ボイスジェスチャは演奏者の発話から音量や音高といった音響的な特徴を取得するものである．マイク信号はマイクの種類や演奏者の声質，パフォーマンス環境の影響を大きく受けるため，音量とピッチ検出の閾値を調整する必要がある． これらの値は，メインモジュールから調整できるようにした．また，ハイパスフィルターを通すことで低域

のノイズ成分を調整できるようにしている．ボイスジェスチャは，マイクからの入力信号の音量と音高から定義される．音量はフレーム毎のピーク値によって定義される． 音高はフレーム毎の基本周波数によって定義される．このモジュールの出力は，ボイスジェスチャのパラメータとして統合モジュールへ送られる． 音量は入力信号から最低値 0, 最高値 1 として扱う．量子化ビット数は，マイクの接続されたオーディオインタフェースに依存する．今回のプロトタイプでは，16 ビットで処理をしている．検出されたピーク値は，最低値 0，最大値 127 の整数値として変換し出力する．

3.6 音楽アプリケーション

今回のプロトタイプでは，Ableton Live10 をホストアプリケーションとした.アプリケーション内のパラメータには，本インタフェースを Max for Live デバイスとして読み込むことで Live API にアクセスし，制御できるようにした．

Live10 では，ギターおよびマイク入力用にオーディオトラックを 2 つ作成し，マイク入力用のオーディオトラックに今回開発した Max for Live デバイスをセットした(図 4). また，ギタージェスチャを習得するため，ギター用のオーディオトラックに，ギターのオーディオシグナルを送信するためのデバイスをセットした．マイクとギターは，オーディオインタフェース(今回は，Steinberg UR-22 を使用)から Ableton Live10 に送られる．セッションビューにインスツルメントやドラムシーケンス・オーディオサンプル等をセットすることで，ボイスコマンドにより制御することができる．表 1 に音楽アプリケーション内のコマンドとパラメータの割当て関係の例を示す．

表 1 音楽アプリ内のコマンドとパラメータの割当て関係

パラメータ	コマンド	機能
is_playing	再生	シーケンスを再生
is_playing	停止	シーケンスを停止
tempo	中くらいのテンポ	BPM を 120 に設定
tempo	速いテンポ	BPM を 170 に設定
tempo	遅いテンポ	BPM を 86 に設定
solo	ソロギター	ギターのみを再生
Clip is_playing	次のクリップ	次のクリップを再生
clip is_playing	前のクリップ	前のクリップを再生

3.7 映像アプリケーション

映像アプリケーションは openFrameworks を使って作成した．opernFrameworks は C++をベースとしたツールキットである．OSC の送受信機能により映像シーンの切り替え機能を持つ．それぞれのシーンは，自然現象をテーマにしたジェネラティブに幾何学的な模様を生成するシーンを制作した．プリレンダリングされた映像ファイルは最大 99 個まで割り当てが可能で，読み込みフォルダ内の映像ファイルを変更することで自由に編集することができるようにした．ジェネラティブに幾何学的な模様を生成するシーンには，「気泡」「雨」「風」「石」を用意した．それぞれのシーンには，「/Bubble」のような名前をつけ，コマンドから直接呼び出すことができるようにした．また，ジェネラティブに幾何学的な模様を生成するシーンではパラメータ制御することで生成オブジェクトの個数や形状を変形させることができ,それらの状態によって映像が変化するようにした．

表 2 に映像アプリケーション内のコマンドとパラメータの割り当て関係の例を表す.

表 2 映像アプリ内のコマンドとパラメータの関係

パラメータ	コマンド	機能
/bubble	気泡	映像シーンを切替
/rain	雨	映像シーンを切替
/wind	風	映像シーンを切替
/stone	石	映像シーンを切替
/spark	破裂	映像シーンを切替
/bubbledown	気泡を落とす	気泡シーンを変形
/Next	映像シーンを次へ	映像シーンを切替
/Back	映像シーンを前へ	映像シーンを切替

4. 動作の問題点

4.1 操作テスト

自由発話による操作テストを実施した(図 3)．オーディオインフェースには Steinberg UR-22 を使用した．マイクは Sennheiser e835 を使用した．パフォーマンス用にセッティングした音楽・映像アプリケーションを制作した．音楽アプリケーションには，Jazz・Techno・Hip-Hop・Jungle などのジャンルを彷彿させるオーディオループを 10 個用意した．映像アプリケーションには，自然現象をテーマとした演出に加えて，幾何学的な模様や動物，天体などを示した映像ループを用意した．このようなパフォーマンスシステムで，エレキギターを即興的に演奏し，自由発話を行い操作を実施した．約 10 分間で 33 回フレーズを入力した．

4.2 考察

今回は word2vec モデルを用いて，入力フレーズ間とコマンドとのコサイン類似度の高いコマンドを演奏者の意図したパラメータ操作として，動作するようなインタフェースのプロトタイプを開発した．ここでこのインタフェースを使った動作テストによる考察を行う．

例えば次のクリップを再生するために，「次のクリップを再生」「次のクリップを再生して」「次のがいいです」「別のクリップにして」は意図通りの結果が得られた．また，同じ意図の発話の中で「やっぱり次の方がいい」「クリップ変更」は別のコマンドが呼び出された．「前がいい」「前のやつにして」「さっきのクリップ」などは「前のクリップ」が実行され，以前に再生していたオーディオクリップが再生され，意図通りの操作ができた．一方で，「さっきのがいい」「やっぱりさっきのほうがいいの」などの発話フレーズでは別のコマンドが呼び出された．テンポを変えるためのフレーズでは，速いテンポ・中くらいのテンポ・遅いテンポの 3 つのコマンドを用意したが，「ペースを速めて」「もっと速いテンポ」としても「中くらいのテンポ」といった結果が出力された．一方で「テンポを速く」とすると「速いテンポ」のコマンドが実行された．

また，長い入力フレーズでは，文字数が長いコマンドが出力される傾向があることがわかった．発話フレーズにより意図通りにコマンドが実行されないことがあった．この問題に関しては，コマンドが十分にアプリケーションの機能を説明できていないことが考えられる．しかしながら，即興演奏においては「テンポを速くして」との発話に対し，「テンポを遅く」というコマンドが誤って選択されることがあった．これは，テンポという単語に対する類似度が正しく評価された一方で，速さに関する意図理解が正しくされなかったと考えられる．しかし，誤った操作がなされたにもかかわらず，意外なギターフレーズが生まれて好印象になる場合もあった．また，今回は大語彙連続音声認識および，word2vec モデルは，学習済みモデルを使ったため，音楽・映像アプリケーションのパラメータに使われるような専門的な単語を全てカバーできていないという問題があった．そのため，専門的な単語を適応させるなどを検討する必要がある．

図 3 本インタフェースによるパフォーマンス

図 4 音楽アプリケーションの様子．下部画面に開発した Max for Live デバイスを挿入している．

5. おわりに

本稿では，音楽・映像アプリケーションのパラメータをボイスコマンドにより拡張可能にするという以前の研究を拡張し，大語彙連続音声認識と自然言語処理による意図理解を導入した．これにより演奏者は自由発話により音楽・映像アプリケーション内のパラメータを操作することができるインタフェースを提案した．

「次のクリップを再生」「次のクリップを再生して」「次のがいいです」といった発話フレーズでは，意図を読み取り，オーディオクリップを次のパターンに変更して再生することができた．一方で，フレーズによっては，意図通りな結果が得られないことがあった．

今後は，映像・音楽ループの持つジャンルや雰囲気，様々な関連用語と入力フレーズの類似度を算出することで，言葉の持つ表現力から映像・音響の持つ表現力を引き出すインタフェースとしての可能性も探っていきたい．

また，演者と映像・音響システムをつなぐ，より直感的なインタフェースを目指し，認識精度・使いやすさの向上を目指し，オーディオビジュアルパフォーマンスにおける表現力の拡張を目指していきたい．

参考文献

[1] 大谷泰斗，越智景子，大淵康成．JackTopGuitar：オーディオビジュアルパフォーマンスのためのギターと音声入力を使用したライブインタフェース． 芸術科学会論文誌 Vol．17，No．4，pp．83-93 (2018).

[2] 岡本淳，庄境誠．人間の多様な振る舞いを考慮した音声 UI の必要性． 研究報告音声言語情報処理 (SLP), 2009, vol. 2009, no． 10, p． 1- 5.

[3] M．Faulkner，VJ: Audio-Visual Art and VJ Culture: Includes DVD．Laurence King Publishing，2006．

[4] Paul Spinrad and Melissa Ulto． The VJ Book Inspirations and Practical Advice for Live Visuals Performance, A Feral House Book, 2005, pp． 166-180.

[5] S．W．Lee，G．Essl，and M．Martinez，“Live writing : Writing as a real-time audiovisual performance,” NIME，2016，pp．212–217．

[6] Max Cooper，Emergence，http://emergence.maxcooper.net/，2017．

[7] Alva Noto，Ryuichi Sakamoto，Ensemble Modern． UTP_，2008，CD+DVD，Raster-Noton.
[8] S．Fasciani and L．Wyse，“A voice interface for sound generators: adaptive and automatic mapping of gestures to sound,” NIME，2012.
[9] T．Igarashi and J．F．Hughes，“Voice as sound: using non-verbal voice input for interactive control,” in UIST ’01 Proceedings of the 14th annual ACM symposium on User interface software and technology，2001.
[10] J．Deacon，“The development of a software tool that employs vocals for the control of musical elements in a live performance,” Ph．D．dissertation，University of Limerick，2014.
[11] 倉田 岳人, 市川 治, 西村 雅史． ユーザの発話傾向分析に基づく車載機器操作のための音声入力手法の検討． 情報処理学会研究報告, Vol．2009-SLP-78 No．2.
[12] A．Lee and T．Kawahara，“Recent development of open-source speech recognition engine julius,” in Asia-Pacific Signal and Information Processing Association．Asia-Pacific Signal and Information Processing Association， 2009，pp．131–137.
[13] Taku Kudo, Kaoru Yamamoto, Yuji Matsumoto: Applying Conditional Random Fields to Japanese Morphological Analysis, Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004), pp．230-237 (2004).