Vol.2021-MUS-130 No.26
Vol.2021-EC-59 No.26
2021/3/17

CNNによる関節座標を用いたボディパーカッションの特徴抽出とデジタル効果音を付与する動画編集システムの提案と実装

角谷 康太[†1]　片山 泰輔[†1]　豊田 麻友[†1]　久野 文菜[†2]　濱川 礼[†3]

概要：本論文ではユーザが所有しているボディパーカッションの動画の関節座標から CNN を用いて特徴抽出し，手で叩いた音を自動で変換した様々なデジタル効果音を付与する動画編集システムについて述べる．最近 YouTube や Tik Tok などの SNS の普及により，誰でも動画を用いたエンターテイメントを発信できるようになった．しかし発信する動画を制作する際，動画編集の経験が少ないためクオリティの高い作品を作ることが難しい場合がある．そこで自動で動画編集ができれば誰もが高いクオリティの動画の制作が可能になると考え，動画編集の中でもその瞬間の印象付けに役立つ効果音の付与を自動で行うシステムを提案する．ボディパーカッションの動画をシステムの入力とし，Open Pose で手の動きを追うことで体を叩いた時間領域を取得する．その後，時間領域内にある動画の音源データをメルスペクトログラム画像に変換し，CNN で真偽判定することでその時間領域内にデジタル効果音を付与するかを判断する．最後に，動画内の人が着ている服装の種類を認識し，対応する効果音を付与する．

1. 背景

最近では SNS を通じて動画や写真を投稿し，シェアする機会が多くなってきている．特に Youtube に関しては，世界で 20 億人以上の人が視聴し，2020 年での日本の 19〜64 歳の月間 Youtube ユーザは 6,500 万人いると言われている．さらに無くなったら最も寂しいプラットフォームで YouTube は 1 位を獲得している [1]．動画を使用したエンターテイメントが発展していく中で, YouTuber として無名の人が有名になったり，芸能人が参入するケースが増えてきている．実際に, YouTuber 専用の事務所である UUUM で契約するチャンネル数は, 2020 年 4 月に 10,000 チャンネルを突破している [2].

また, 動画媒体の需要が伸びていく中で動画編集の需要も伸びてきている．現在無料・有料の動画編集ソフトやテロップを自動で編集できるものなど多くのものが存在する [3][4]．こうしたソフトが開発されているため，今後も動画編集の需要は続いていくと考えられる．

これから自分自身で動画制作を行おうとするのに動画編集の経験がない場合がある．簡単に編集するだけでも何をしているのかは分かりやすくなるが，どんなに面白い動画が撮れたとしても編集のスキルがなければ面白く見せることが出来ず, 上手く人に伝えられない可能性がある．そこで我々は, 動画編集の経験がなくても質の高い動画制作ができる方法を模索した．動画編集には，様々なサイトや自身の編集経験から主に動画の切り取り，効果音の付与，字幕付与の 3 つが編集で必要な工程であると考える [5][6][7]．3 つの中で動画のカット，字幕付与については編集者が必要か不必要かを判断すれば良いため時間はかかるが難しい作業ではない．一番大変なのが，効果音の付与だと我々は考える．効果音や BGM などは動画の印象を左右する大事な要素であり，場面に適したものを一致させないと印象が変わってしまう．実際，昔から音と感情が関係していると考えている人がいる [8][9]．その瞬間のアクセントとして有効な効果音は，経験者になるほど個性のある動画を作り上げることができるはずである．そのため本システムは，数ある動画ジャンルから比較的短い時間で何をしているのか伝えやすいダンスに絞って, 効果音に特化した自動編集を提案する．

2. 動画効果音自動付与手法

動画の効果音を付与するにあたって, 動画内の人の手の

†1 現在，中京大学工学部情報工学科
Presently with School of Engineering Chukyo University
†2 現在，中京大学大学院工学研究科情報工学専攻
Presently with Department of Information Engineering, Graduate School of Engineering Chukyo University
†3 現在，中京大学工学部
Presently with School of Engineering Chukyo University

© 2021 Information Processing Society of Japan

動きを追うことでどのタイミングで体を叩いたのか，その瞬間に実際に体を叩いたのかを判定して判断する (以下動画内の人をダンサーと定義する)．ダンサーが実際に手を叩いたかを判定するため，本システムではダンサーの動画を一枚ずつ画像変換し，一枚ごとに手の座標がどこにあるのかを関節座標で判断する．その後，手の動きから叩いた可能性のある時間を求め，予め手で叩いた音を集めた学習データを使用して実際に叩いたかを判定する．最後にダンサーが着ている服装に合わせて効果音を付与することで, 自動で編集した動画を出力する．こうすることで本システムを使用したユーザは動画編集のスキルがなくとも, 質の高い効果音付与ができると考える．手の特徴抽出，手で叩いた音の判定，服装を使用した効果音付与の手法は以下に示す．

2.1 手の特徴抽出

動作認識には Zhe Cao らによって開発された OpenPose[10] を使用した．OpenPose とは, 図 1 の様に 18 の関節点を取得することで姿勢を推定するモデルである．同様にして顔と手も認識することができる．OpenPose の実装については, Cao 自身による実装 [11], Keras による実装 [12], DeNA による Chainer での実装 [13], といったように多くの例が GitHub にアップロードされている [14]．本研究では Python で実装された DeNA による Chainer での実装 [13] を使用した．

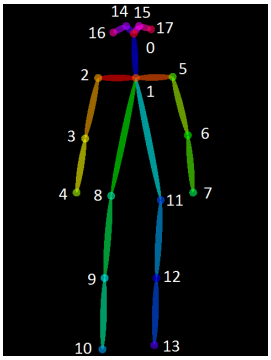


図 1 OpenPose で得られる骨格座標

2.2 手で叩いた音認識

手で叩いた音の認識はメルスペクトログラムを画像変換し，Keras での CNN による機械学習で実装した．

2.2.1 keras

Keras とは Python で書かれた TensorFlow(Google), CNTK(Microsoft)，Theano(モントリオール大学) 上で実行可能な高水準のニューラルネットワークライブラリであり，本システムは最新で複数の GPU を使った処理が可能な TensorFlow をバックエンドとして指定した [15]．Keras はモデルを定義するために Sequential(レイヤの積み上げに使用，最も一般的に使用されている) と，Functiional API(共通レイヤを持つ複雑なアーキテクチャ設計) の 2 つの方法がある．今回は音の判定を真か偽の 2 種類の出力しか行わないため，複雑なモデルやレイヤーを共有するモデルなどを構築する必要がないと考え Sequential を扱った．

2.2.2 CNN 転移学習

CNN とは畳み込みニューラルネットワークの略称で全結合や畳み込み層，プーリング層から構成されている．これは主に画像認識で活用することが多く，画像内のある特徴を抽出することで検出が可能になる．

学習するにあたって使用する枚数が膨大でないため，転移学習により画像枚数の問題を解決した．これは他の画像データを使って学習されたモデルを使うことによって，少ない学習量で新たなモデルを生成することが可能である．Keras ライブラリは ImageNet でトレーニングされた AlexNet，VGGNet，Inception，ResNet，Xception，MobileNet，SqueezeNet など多くのモデルを適用している．この中で画像認識タスクにおいて高い予測性能をもち，ImageNet の SOTA ランキングでも応用モデルが上位にランクインしている ResNet を採用した [16]．

2.2.3 メルスペクトログラム

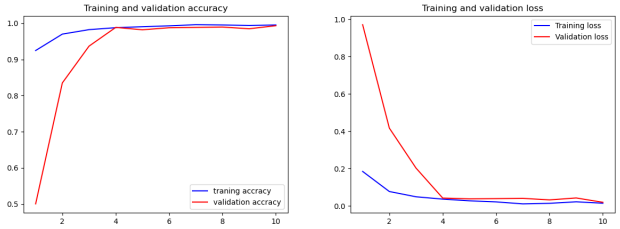
学習用の画像はメルスペクトログラムを使用した．メルスペクトログラムとは, 音の周波数で正弦波を表した時の大きさである振幅スペクトルを，人間の聴覚を考慮するメル尺度で変換したスペクトログラムである．またスペクトログラムとは, アナログ信号の周波数特性を調べるためのフーリエ変換を解析区間を少しずつずらしながら周波数分析を行い，横軸を時間，縦軸を周波数として周波数特性を濃淡表示したものである．

2.2.4 学習

学習では実際に手で叩いた音を 2,600 個 (手，腕，胸，腹，太ももの 5 種類)，ダミーデータを 2,600 個 (無音，外の音，生活音，物) を集めた．Epoch 数と batch 数については 試した中で比較的精度の高かった Epoch 数 16, batch 数 10 回で学習した．以下に学習した正解率、損失関数の結果, 学習画像の例を図 2, 図 3 に示す．10 回目で生成されるモデルデータを使用し，最大値を 1 としたときの訓練正解率は 0.9995，訓練損失関数は 0.0031，精度検証正解率は 0.9877，精度検証損失関数は 0.0437 の結果を得られた．

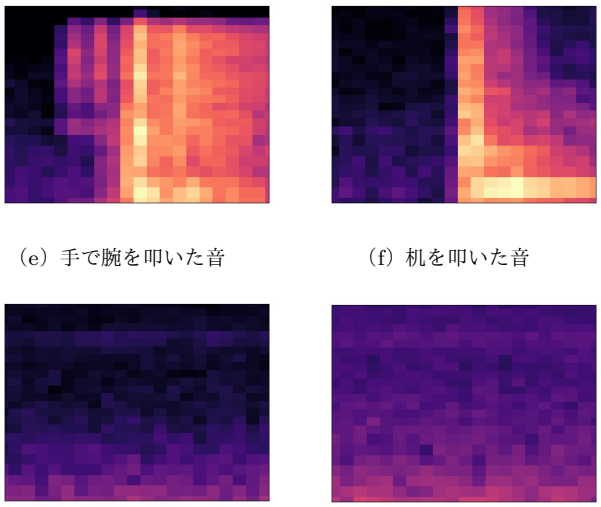
2.3 衣服検出を用いた効果音付与

衣服の検出には Simaiden らが GitHub で公開している Clothing-Detection[17] を使用した．Clothing-Detection とは, ModaNet データセット [18] および DeepFashion2 データセット [19] で YOLOv3, RetinaNet, FasterRCNN を使用した衣服の検出である．ModaNet は RGB 画像に関連するアノテーションで構成されるストリートファッション画像データセットであり, DeepFashion2 は包括的なファッ



（c）Epoch 数 16 の正解率　（d）Epoch 数 16 の損失関数

図 2　学習結果



（e）手で腕を叩いた音　（f）机を叩いた音

（g）環境無音　（h）環境騒音

図 3　画像変換した音データ一例

ション画像データセットである．YOLOv3, RetinaNet, FasterRCNN はすべて物体検出を目的とした機械学習モデルである．本システムでは, ModaNet を用いて YOLOv3 で学習させた事前学習済みモデルを使用した．手で叩くと音が異なると考えられるアウターやトップスなど, カテゴリに分類されているためである．衣服検出結果の例を図 4 に示す．効果音付与には Pydub[20] を使用した．Pydub とは, 音声ファイルを簡単に処理できる Python モジュールである．入力動画からの音声切り取り, 動画への音声結合は MoivePy を使用した．MoviePy とは, 動画ファイルを簡単に編集できる Python ライブラリである．

図 4　衣服検出結果

3. システム概要

上記手法をもとにボディパーカッションで手を叩いたときの音を判定し，服装に合わせて効果音を付与するシステムを開発した．本システムは，手特徴抽出部，音認識部，効果音付与部の 3 つで構成しており，システムの流れを図 5 で示す．

3.1 手特徴抽出部

一連の流れを図 6 に示す．

3.1.1 画像変換

ユーザから入力された動画を画像に変換し, 動画の再生時間を取得した．この時, 再生時間を変換された画像枚数で割ることで画像一枚あたりの時間を算出しておく．

3.1.2 OpenPose

変換された画像を一枚ずつ OpenPose で読み込み, 取得した関節座標の結果を x 座標と y 座標に分けて csv ファイルに出力した．

3.1.3 判定部

最初に人数の判定を行う．3.1.2 で得られた csv ファイルから首の x 座標の関節座標を抜き取り配列に格納し, 配列の長さを変換された動画枚数で割ることで動画内に映っている人数を算出した．複数人映っている場合は, 一枚前の画像と比較して差が 50 以内である人を同一人物として判断した．そして, それぞれの人ごとに分けた関節座標の csv ファイルを新たに作成した．

次に, 手の動きの判定を行う．csv ファイルから右手と左手の関節座標を x 座標, y 座標に分けて配列に格納し, それぞれの動きをグラフに表した．手の座標のみを格納するのは, 体のどこかを叩くときに必ず使われる部位が手だからである．図 7 に右手を赤左手を青とし, x 座標の縦軸は画像枚数, 横軸は座標値を表し, y 座標の縦軸は座標値, 横軸は画像枚数を表すデータを示す．

このグラフは, 実際に動画の前半で手拍子を行い, 後半で太ももを叩いたときの手の動きを示したものである．このグラフから, 図 8 に示すように, 動きに変化のある黒枠部分で抽出したい動きが行われたと考えられる．従って, グラフの極値を求めることで動きの抽出を行った．叩くときの x 座標に注目した場合, 右手は座標に対して増加→減少となったとき, 左手は座標に対して減少→増加になったときに叩いた動作があるため, 右手は極大値を左手は極小値を求めた．y 座標に注目した場合, 右手左手共に, 座標に対して増加→減少となったときに叩いた動作があるため, 極大値を求めた．また 叩いた瞬間が一瞬ではない場合も考慮して, 同じ座標値が連続で表れたときに値を 1 ずらす操作を行い, 極値と判断されるようにした．その結果を図 9 に示す．そして, ピークが表れた部分の画像から 5 枚前の画像の動画内の時間を開始時間, 5 枚後ろの画像の時間を終了

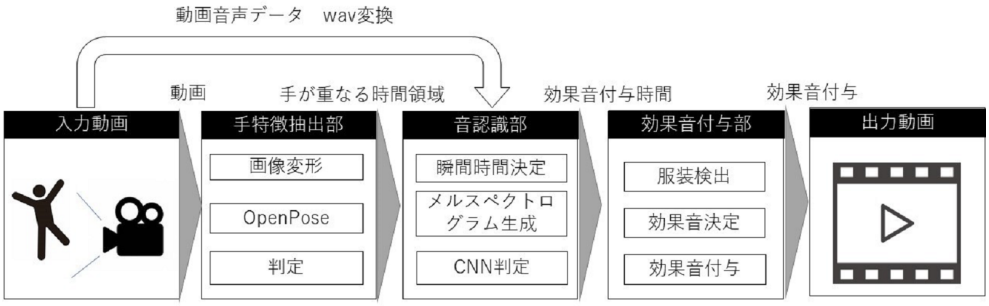


図 5　本システムの一連の流れ

時間として値を音認識部に送る．時間は, 3.1.1 で求めた一枚あたりの時間に画像枚数を掛けることで算出した．

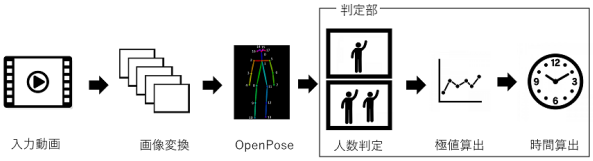
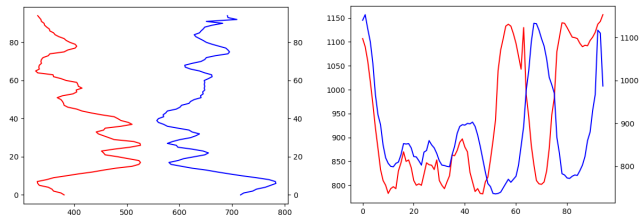
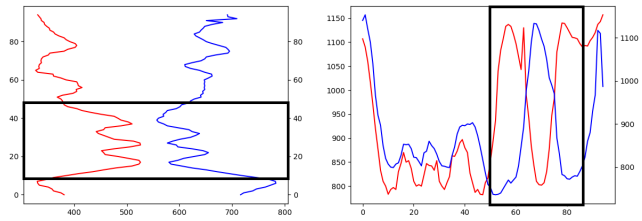


図 6　手特徴抽出部の一連の流れ



(i) x 座標の動き　(j) y 座標の動き

図 7　手の動き

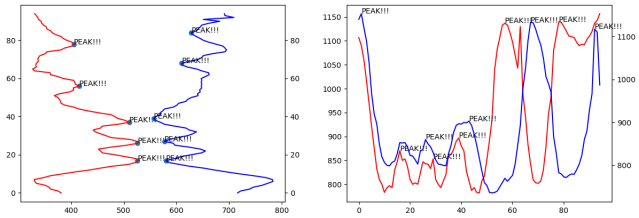


(j) x 座標の動き　(k) y 座標の動き

図 8　手の動きからの推測

3.2　音認識部

音認識の手法について流れ図を図 10 に示す．



(l) x 座標の動き　(m) y 座標の動き

図 9　極値

3.2.1　瞬間時間決定

はじめに動画の音声データを取得するために moviepy で動画の音声を wav 形式で切り取る．この後の音声データを数値化するのに Brian McFee 氏らが開発した librosa の音楽やオーディオ解析が可能な python パッケージを使用する際に，wav でしか対応していないため音を切り取りして wav 変換を行った．3.1 から出力される動画の開始時間と終了時間の範囲内 (以降は時間領域と定義する) から手で叩く瞬間を決定した．予めメルスペクトログラムで数値化していた動画のデータから, 時間領域内で手で叩く瞬間はその音の周波数データが加算されて周りより値が大きくなると考え，最大値を手で叩く瞬間とした．

3.2.2　メルスペクトログラム生成

3.2.1 で決定した時間を使用して前後 0.1 秒の範囲でメルスペクトログラムをグラフ化した．生成したグラフの名前を瞬間時間とし，画像を出力する．

3.2.3　CNN 判定

訓練データであらかじめ学習してある CNN のモデルデータを使用し，3.2.2 で出力した画像の真偽を判定した．真と判定された場合は，画像の名前の時間を抜き取り配列に保存しておく．全ての画像を確認した後に, 保存してある時間を全て次のシステムに送る．

3.3　効果音付与部

効果音付与までの流れを図 11 に示す．

3.3.1　衣服検出

本システムでは, 入力動画の途中で登場する人物が着替えることや人数が増えることは無いものとし, 入力動画の

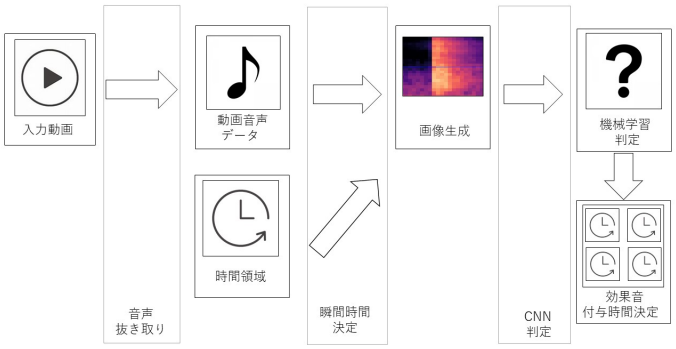


図 10　音認識フローチャート

中央のフレーム画像を抜き出し, その画像で衣服の検出をする. 検出されるものはバッグ, ベルト, ブーツ, 履物, アウター, ドレス, サングラス, パンツ, トップス, ショーツ, スカート, 帽子, スカーフとネクタイの 13 種類である.

3.3.2　効果音決定

衣服検出で検出された衣服の種類の組み合わせで効果音を決定する. 13 種類の組み合わせでは用意する効果音の種類がとても多くなるため, 動画中で叩かないと考えられるバッグ, ベルト, ブーツ, 履物, サングラス, ショーツを除いたアウター, ドレス, パンツ, トップス, スカート, 帽子, スカーフとネクタイの 7 種類を使用した. 組み合わせはアウターのみ, トップスとパンツ, アウターと帽子など 18 種類の効果音を設定した.

3.3.3　効果音付与

入力動画の音声と 3.3.2 で決定された効果音を pydub の AudioSegment で読み込み NumPy 配列に変換する. 3.2 の音認識部から送られてきた時間を元に入力動画の音声に効果音を足す. そして, AudioSegment で NumPy 配列を音声にする. 最後にこの音声と入力動画を moviepy で結合させる.

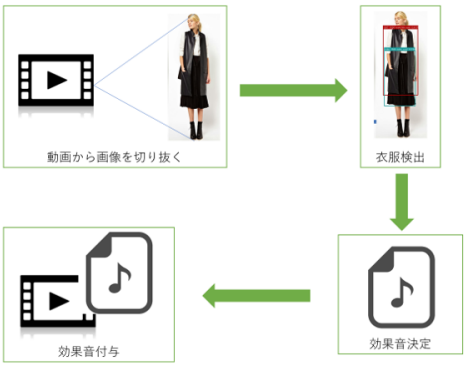


図 11　効果音付与フローチャート

3.4　出力結果

出力結果の一部を図 12 に示す. 本システムを通した結果, 効果音が付与されたのは赤枠の部分であった. 実際に手拍子をしており, 効果音を付与するべき部分として正しかったことが確認できた.

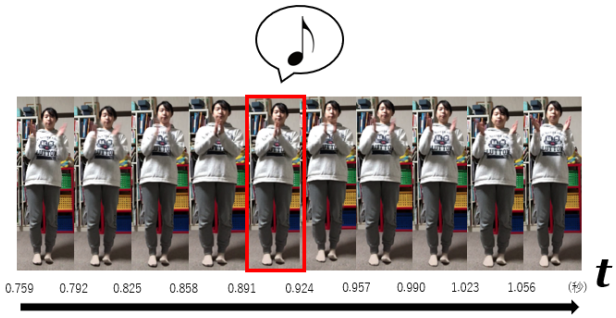


図 12　本システムの出力例（t:時間経過）

3.5　開発環境

今回使用した言語は python3.6 で，表 1 のスペックの GPU サーバーで実行する.

表 1　GPU サーバースペック

パーツ	スペック
OS	Ubuntu Server 18.04.4 LTS
CPU	E5-2699 v4(2.2GHz, 22 コア) x (合計 44 コア)
GPU	Tesla P100 x4(64GB メモリ)
メモリ	512GB
ライブラリ	CUDA 10.2 with cuDNN 7.6.5(/usr/local/cuda)

4.　関連研究

4.1　動作と音

高橋らは, 筋電センサを用いることで動作に合わせて音を出している [21]. しかし, この手法はデバイスを必要とするため 誰でも簡単に使うことはできない. そこで我々は, デバイスを必要とせず動画を撮影するだけで誰でも簡単に動作に合わせて音を出せるシステムを開発した.

4.2　動きの抽出

稲田らは, 動きのピークを算出することでダンスの一致度を評価している [22]. 我々も, 体を叩くときには手の動きのピークが関係していると考えシステムを実装した.

4.3　音認識

ものは，叩かれたり引っかかれたりすると独自の音を出し，これらの音はものの材料特性と何かしらのアクションが組み合わさって生成される．Andrew Owens らは，視覚シーン内の物理的相互作用を研究する方法として，物体が打たれたときにどのような音がなるか予測する方法を提案している．ドラムスティックでオブジェクトを叩いたり引っ

掻いたりする人々のサイレントビデオから音を合成するアルゴリズムを開発している [23]．この時に Andrew Owens らは，動画の過去時系列を考慮するために RNN でデータを学習しているが，我々が提案するシステムは OpenPose である程度の時間を決めるため時系列を考慮しなくて良いと考え CNN で学習を行った．

5. 評価

5.1 評価内容

大学生で動画編集の経験がある 5 人と経験がない 6 人 (合計 11 人) に対して「叩いたタイミングに効果音が付与されていたか」の観点から，本システムの目的である「動画編集の経験がなくとも質の高い動画を作れたか」の評価実験を行った．

5.2 実験

ユーザにボディパーカッションの動画を撮ってもらい，実際に本システムで効果音を付与した．動画を取る際の条件で，できるだけ生活音がない場所で取ってもらう．

5.3 アンケート

いくつか設問をたて，設問に対して 5 段階評価を用意し，1「悪い」〜5「良い」として回答してもらった．またコメントについては自由記述で書いてもらった．

5.4 精度評価

生成した動画に対して，手で叩いた回数，動画内で効果音がなった回数，手が重なった瞬間に音を出した回数，成功率を表でまとめた．

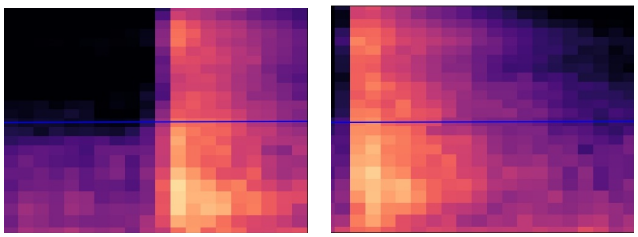
6. 結果・考察

6.1 結果

6.1.1 アンケート

初めに表 2 に出力した動画の結果に対しての評価をしてもらった．Q1 から 7 割程度の人たちは，1 部分の音の出力に失敗していたが効果音の付与に成功した．しかし 1 部の人たちは音が一回も出力されなかった．全く音が出なかった人たちの共通点は叩いた音が小さい，環境音が混ざっている，叩いた音が低音で認識できていないの 3 点が考えられる．叩いた音の小ささと環境音が混ざっていることに関しては，音を全く出力できなかった人が場所を変えて再度動画を撮ってもらったがその時は全て音に対して効果音を付与することができた．また低音に関しては今回集めた音データのほとんどが上手く叩けたときの音のみで学習させており，叩きそこねて乾いた音を学習させていなかった．実際，音が出なかったときのほとんどが衣服を叩いたがうまく叩けておらず鈍い音が出た場合だった．また評価を 3, 4 と答えた人の中で「音が遅れたり速く聞こえる」と意見する人がいた．これは叩いた瞬間の時間を出力できていないと考えられる．実際に音が速く出てしまうと答えた人のメルスペクトログラム画像を図 13 に示す．本来は瞬間時間を中心として画像を生成しているが，この図 (o) は適切な画像より早い時間に音が出ていることが分かる．このことから何らかの音が手で叩いた瞬間時間の大きさより大きいと判断して付与しているため, 音が送れて出ていると考えられる．

Q2 については上手く効果音を付与できた人ほど評価が高くなったと考える．しかしほとんどの人から「音の変化がない」という意見を貰った．本システムは複数の音を奏でる予定であったがそこまでの実装ができず, 1 つの音を付与するだけのシステムを開発した．効果音を複数付与したり音の大きさを変えるのは今後のシステム向上の参考にする．



（n）成功例　（o）失敗例 (速く音が出る場合)

図 13 速く音が出てしまう人の音データ

6.1.2 精度

今回の評価で生成した動画すべてに対しての結果を表 3 で示す．

6.2 考察

音が付与されなかった原因として, 以下のことが考えられる．

本システムでは叩いた瞬間が一瞬ではないことを考慮して値をずらす操作を行ったが, 図 14 の画像 (p) に示すように, 叩いた瞬間の時間が長く変化が小さいと図 14 の画像 (q) に示すように値をずらしても上手く反映されなかった．(右手を赤左手を青とし, 縦軸を画像枚数横軸を座標値とする) そのため, 手の動きだけではなく, 他の部位との関係も含め判定をする必要があった．

また叩いたときにしっかり動きの判定がされていた場合でも, 学習した音に対して叩いた音が小さい，環境音が混ざる，音が低音の 3 つのパターンのときに音が鳴ったと判断されなかった．このことから叩いた音の大きさ・高低音・環境音が混ざっている場合も考慮して学習しなければならないと考えられる．

7. 今後の展望

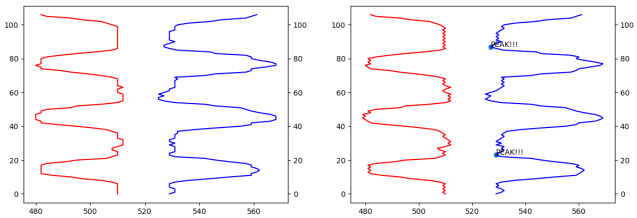
本システムは叩いたときに様々な音を出すことを想定し

表 2 アンケート回答

番号	質問	1	2	3	4	5	平均
Q1	手で叩いたタイミングに音が付与されていたか	1	2	3	3	2	3.27
Q2	実際に使用してみてどうだったか	1	2	3	3	2	3.27

表 3 各動画の正答率

被験者	手で叩いた回数	タイミングよく音を出した回数	成功率
A	7 回	7 回	100.0%
B	7 回	1 回	14.2%
C	12 回	1 回	8.3%
D	8 回	3 回	37.5%
E	25 回	22 回	88.0%
F	40 回	30 回	75.0%
G	15 回	12 回	80.0%
H	41 回	36 回	87.8%
I	24 回	22 回	91.6%
J	15 回	0 回	0.0%
K	70 回	42 回	60.0%



（p）値をずらさなかった場合　（q）値をずらした場合

図 14 x 座標の動き

ていたが，条件が複雑になってしまい実装まで至らなかった．今後の方針として，ただ手の座標の動きを追うだけでなくどの部位を叩いたかも考慮したシステムにすることで，部位ごとに違う効果音を奏でられるようにし，音の大きさを判断して付与する効果音の大きさも自動で変更できるように改善していく．またユーザが使用しやすいように web アプリケーションを制作し，簡単に誰でも使用できる環境を構築していく．

8. まとめ

我々はボディパーカッションの動画から手の特徴を抽出し，自動で効果音を付与するシステムを提案し，評価によって効果音が適切に付与できているか実用性を示した．しかし，課題として効果音を 1 つしか付与できない点と，想定していない音が機械学習で学習できていない問題があるため，時間領域の取得以外に叩いた部位の判定も行い複数の音を奏でられるようにし，学習の精度を上げることによって音が出ないケースを減らすように改善する．

参考文献

[1] Google:月間 6,500 万ユーザーを超えた YouTune、2020 年の国内利用実態——テレビでの利用も 2 倍に
入手先 ⟨https://www.thinkwithgoogle.com/intl/ja-jp/marketing-strategies/video/youtube-recap2020-2/⟩.

[2] UUUM:UUUM ネットワークのチャンネル加入総数が 10,000 チャンネルを突破!
入手先 ⟨https://www.uuum.co.jp/2020/04/03/48293⟩.

[3] カンタン動画入門:【2021】動画編集ソフトおすすめランキング無料有料 38 選 Windows/Mac
入手先 ⟨https://douga-tec.com/?p=10393⟩.

[4] Telorain
入手先 ⟨https://telorain.com/⟩.

[5] モブスタ:動画編集の作業手順, ワークフローを詳しく解説
入手先 ⟨https://mobstyley.net/video/video-editing/video-editing-workflow/⟩.

[6] something fun!:動画編集のやり方を初心者でもわかりやすく解説!おすすめソフトも紹介
入手先 ⟨https://www.somethingfun.co.jp/video_tips/how-to-edit-movie⟩.

[7] dougabiz.com:動画編集の仕事内容とは？必要なスキルやプロが使う動画編集ソフト紹介
入手先 ⟨https://dougabiz.com/column/video-production/1963/⟩.

[8] 大串健吾:音楽と感情
入手先 ⟨https://www.jstage.jst.go.jp/article/sobim/30/1/30_1_3/_pdf/-char/ja⟩.

[9] 山崎晃男:音楽と感情についての心理的研究
入手先 ⟨https://osaka-shoin.repo.nii.ac.jp/?action=pages_view_main&active_action=repository_view_main_item_detail&item_id=4154&item_no=1&page_id=3&block_id=24⟩.

[10] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh:Realtime Multi-Person 2DPose Estimation using Part Affinity Fields.
入手先 ⟨https://arxiv.org/abs/1611.08050⟩.

[11] GitHub:ZheC/Realtime_Multi-Person_Pose_Estimation
入手先 ⟨https://github.com/ZheC/Realtime_Multi-Person_Pose_Estimation⟩.

[12] GitHub:michalfaber/keras_Realtime_Multi-Person_Pose_Estimation
入手先 ⟨https://github.com/michalfaber/keras_Realtime_Multi-Person_Pose_Estimationl⟩.

[13] GitHub:DeNA/Chainer_Realtime_Multi-Person_Pose_Estimation
入手先 ⟨https://github.com/DeNA/Chainer_Realtime_

Multi-Person_Pose_Estimation⟩.
[14] 情報処理学会論文誌ジャーナル編集委員会:渡辺裕, スポーツ情報処理の研究開発動向:映像情報メディア年報 2018 シリーズ（最終回）
入手先 ⟨https://www.ite.or.jp/contents/annual-rep/1811nenpo.pdf⟩.
[15] 小田昌宏:keras によるディープラーニング
入手先 ⟨https://www.jstage.jst.go.jp/article/mit/36/2/36_47/_pdf/-char/ja⟩.
[16] The latest in Machine Learning — Papers With Code :Image Classification on ImageNet
https://www.overleaf.com/project/601027d10284ff438e6f3cee
入手先 ⟨https://paperswithcode.com/sota/image-classification-on-imagenet⟩.
[17] GitHub:simaiden/Clothing-Detection
入手先 ⟨https://github.com/simaiden/Clothing-Detection⟩.
[18] GitHub:eBay/modanet
入手先 ⟨https://github.com/eBay/modanet⟩.
[19] GitHub:switchablenorms/DeepFashion2
入手先 ⟨https://github.com/switchablenorms/DeepFashion2⟩.
[20] GitHub:jiaaro/pydub
入手先 ⟨https://github.com/jiaaro/pydub⟩.
[21] 高橋大斗, 川野邊誠:筋電センサを用いた動作楽器の開発と幼児向けリトミックでのコミュニケーション能力強化法の提案「エンタテインメントコンピューティングシンポジウム（EC2017）」
入手先 ⟨https://ipsj.ixsq.nii.ac.jp/ej/index.php?active_action=repository_view_main_item_detail&page_id=13&block_id=8&item_id=183549&item_no=1⟩.
[22] 稲田健太郎, 石川孝明, 渡辺裕:OpenPose を用いた複数人のダンスの一致度評価「情報処理学会第 80 回全国大会」
入手先 ⟨https://www.ipsj.or.jp/event/taikai/80/ipsj_web2018/html/author/wa.html⟩.
[23] arXiv Cornell University:Visually Indicated Sounds
入手先 ⟨https://arxiv.org/abs/1512.08512⟩.

© 2021 Information Processing Society of Japan