作成日：2019 年 6 月 18 日

# 画像差分法を用いたピアノの演奏動画を midi ファイルに変換するプログラムについて

プログラム配布先：https://github.com/Saxtomiya/Project

1. プログラム概要

このプログラムでは、ピアノの演奏動画を読み込み、その画像データから演奏した音を検索し、midi データに変換するものである。動画の各フレーム画像から、画像の差分を計算し、一定量の差分が取れた場合にその座標、フレーム位置から弾いた鍵盤を読み取り、出力する。現在の開発段階では、拡張性を意識して開発したものの、添付した Test.mp4 のみで動作するようになっている。

ファイル構成

ExeFile
- ｜－Launcher.exe//実行ファイル
- ｜－Test.mp4//参照する演奏動画
- ｜－Test.mid//演奏をもとに出力した midi ファイル

ScriptFile
- ｜－Launcher.py//実行用ファイル
- ｜－Test.mp4//参照する演奏動画
- ｜－Test.mid//演奏をもとに出力した midi ファイル
- ｜－….py//各種スクリプトファイル

2. 実行方法

・実行ファイルの場合

Launcher.exe ファイルと同じディレクトリ内に Test.mp4 を配置し、Launcher.exe を起動する。出現したコンソールにファイル名(Test.mp4)と入力する。入力が完了するとプログラムが実行され、そのディレクトリに各データを出力した data フォルダと、Output.midi が出力される。

・スクリプトファイルの場合

Script フォルダ内に Test.mp4 を配置し、Launcher.py を実行する。その後ファイル名を入力するとプログラムが実行される。出力結果は実行ファイルと同様である。

実行には、Python3.6 の動作環境と、opencv、numpy、pretty-midi などのモジュールのインストールが必要である。

3. プログラム解説

3－1　プログラム全体の流れ

各プログラムについて、スクリプトファイルをもとに解説する。

このプログラムの処理手順は以下の通りである。

(1) 動画を読み込み、各フレームの画像データに分解、保存する。

(2) フレームの画像から、鍵盤の座標を探索し、切り取る。

(3) 切り取った鍵盤の座標を、射影変換を用いて処理しやすくし、保存する。

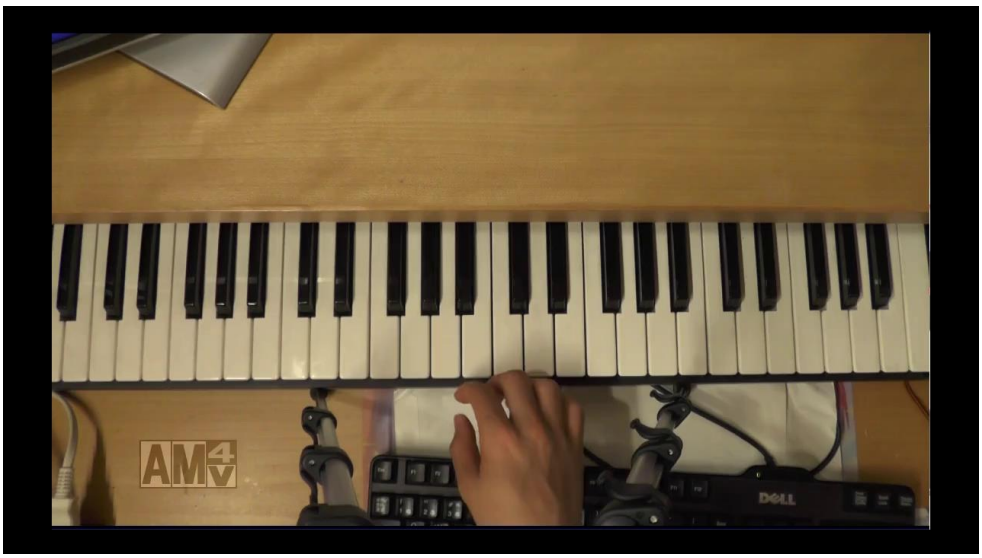
(4) 変換した鍵盤画像をもとに、各鍵盤の x 座標位置を検索、保存する。

(5. 変換した各画像を順番に読み取り、画像差分を検索、大きく差分の出た x 座標から弾かれた鍵盤を取得、保存する。

(6) 保存された鍵盤名、フレーム位置をもとに midi ファイルを作成する。

3－2　s00_make_img.py

このスクリプトでは、(1)の動作を実行する。Launcher から読み込んだパスにある動画を読み込み、各フレームに分解する。分解したフレームは、data/img/01image 内に、フレーム番号.jpg という名前で保存される。



**1　切り取られるフレーム例**

**data/img/01image/000.jpg**

3－3　s01_img_exchanged.py

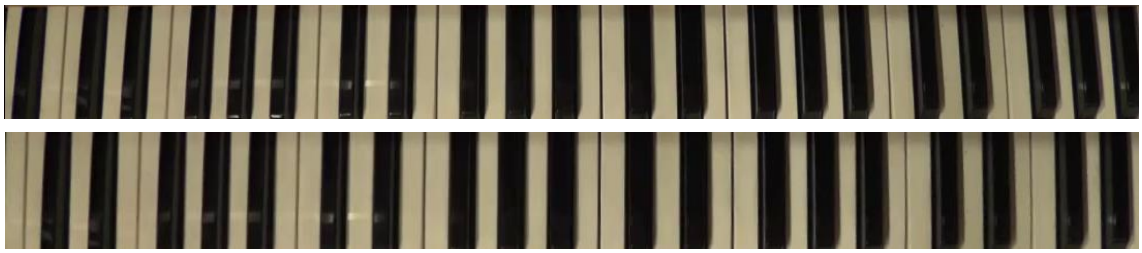
このスクリプトでは、(2)、(3)の動作を実行する。

(2)の動作について

鍵盤の位置は、get_yy 関数で検出する。この関数では、演奏が始まっている　状態の 2 枚のフレームの差分を取り、その y 座標を取得する。カメラを固定しているため、鍵盤以外の座標は変化しないことから、差分で撮れた箇所により鍵盤の上端を判別することができる。これで取得した y 座標から、鍵盤の写っている範囲を切り取る。

**2　切り取りされた画像例**

上記で切り取った画像にエッジ検出を掛ける。得られたエッジの中から、全体が写っている両端の鍵盤を示す直線を探索する。(get_point 関数)今回の動画では A#-B 間のエッジ(左端)と G#-A 間のエッジを検出している。これをもとに射影変換を行い、すべての画像を真っすぐになるように調整する。この動作は make_img 関数で実行している。これにより以下のように変換できる。



**3　変換前の画像(上)と変換後の画像(下)**

この動作で変換した画像を、data/img/02crip/フレーム名.jpg で再び保存する。

3－4　s02_search_keypoint.py

このスクリプトでは、(4)の動作を実行する。鍵盤の画像を探索するにあたって、手や影が映っていない画像を検索する。今回は冒頭 100 フレームの画像をグレースケール化し、合計値が最も高い(影や、白鍵に手が映っていない)画像を検索し、そのフレーム位置を補完する。

検索したフレームの画像を再びグレースケール化し、各 x 座標の合計値を検索する。隣り合った座標ごとにその数値の差を検索し、その値が一定を超えた場合、鍵盤の境目としてその座標を取得する。

以上の手法だと、鍵盤 B-C 間、E-F 間など白鍵が隣り合う座標が探索されない場合があるため、鍵盤の座標間隔が一定以上空いていた場合は、その間隔の中心座標を追加する。これで検索した座標を昇順ソートし、data/keylist.txt に書き出す。

3－5　s03_mov_to_keyname.py

このスクリプトは(5)の動作を実行する。3－3 で作成した画像を読み込み、連続するフレームごとに差分を取る。縦に長く差分が取れたフレームがある場合は 3－4 で作成した txt ファイルを読み込み、その座標がその何番目かを配列として出力する。差分が取れなかった場合は、タイミングを判定するため空の配列を出力する。また、音名を左端から順に格納した配列を用意し、差分の取れた配列の音名を判定する。

以上の動作を行い、フレームごろに改行、同時に複数の音名がある場合はカンマ区切りという形式で/data/score.txt に保存する。以下は score.txt の部分例である。

D#5,
D#5,G#5,

D#5,

3－6　s04_make_midi.py

このスクリプトでは、(6)の動作を行う。3－5 で出力した score.txt をもとに実行する。

Score.txt には画像差分が取れたものすべてが含まれているため、鍵盤を離したフレームも含まれる。そのため、一度差分が取れた音名がもう一度出てきた場合、その音を削除する。

この動作を実行した後、トラックを作成し、各行の要素に音名があった場合はその音をそのフレーム位置の示すタイミングに挿入する操作を繰り返す。全フレーム実行し終えたら、Launcher.py と同じディレクトリに Output.midi として出力する。

４．考察

現段階では、多少の誤差はあるもののある程度の精度の出力を実現しているものの、以下のような課題がある。

・鍵盤がカメラと並行に写っていないと正常に処理できない。
・カメラのブレがある場合、ブレが差分として検出されてしまうため正常な出力ができない。
・一番左の鍵盤がどの音かはスクリプトで判別できない。
・BPM の判別がスクリプトではできないため、固定してしまっている。

これらを改善するために、鍵盤の座標を抽出するアルゴリズムの改善や、カメラブレの補正処理機能の追加、弾かれた鍵盤のフレーム位置から BPM を逆算し、それに合った出力を行う機能を追加する必要がある。