

「ラズパイによる顔識別システム」

取扱説明書 V1.0

2025/5/16

2025/6/10

2025/6/19

2025/6/27

本プログラムは face_recognition という顔識別ライブラリを使って、Python にて顔を識別し、音声合成による挨拶をするシステムです。

ディープラーニングを用いた顔識別を使用して構築されています。

顔識別のライブラリ(face_recognition)は 2017 年 3 月 に Adam Geitgey 氏 によって最初に公開されました。ただ、データセットが欧米人中心だったので、日本人とくに若い女性や子供に対する精度はイマイチとの評価でした。

しかし、昨年(2024)田口氏による日本人の顔データを含んだデータが公開され日本人に対する精度が向上したことと、Pi5 が発売された現在は実用的なスピードで処理ができることにより、日本での実用的な使い方が安価なラズパイで実現できるようになってきました。

そのデータが

taguchi_face_recognition_resnet_model_v1.dat

です。

本プログラムは、Adam Geitgey 氏のプログラムをベースに新しい田口氏のデータを使えるように改造したものを応用し、登録された人に対して時間帯に応じた挨拶をします。

プログラムとファイルの構成

headshots_OnePerson.py	一人分顔写真を撮り、datasetに保存
headshots_MultiplePeople.py	複数人顔写真を撮り、datasetに保存
train_model.py	datasetの写真トレーニングしてモデルを作る
face_recognition.py	カメラで写真を撮り顔識別をおこないます
config.py	各種設定を保存しています
dataset(ディレクトリ)	顔データを保存するフォルダ
face_dat(ディレクトリ)	処理に必要なファイルを保存するフォルダ
face_cut.py	顔写真を切り出し、データを確認します
kill_video_processes.sh	何かの理由でカメラが使用できなくなった時に解放する

- 一部画面でのプログラム名が違いますが、上記プログラム名が正式ですので、読み替えてください。

使い方

1. headshots_OnePerson.py を使って識別したい人の顔を撮ります。
 - a. 「あなたの名前を登録してください」と出るので、半角英数字で名前を入力してください。漢字は使えません。
 - b. 既に登録された名前があれば、プログラムは終了します。
 - c. スペースキーを押すと写っている画像が保存されます。
 - d. 顔写真は正面、少し左右向き、少し上下向き、必要に応じ少し斜め向きなどの写真を撮ってください。
 - e. 写真データが多いほど基本的には精度が上がります。ただ速度は遅くなります。
 - f. 何枚でも登録できますので、枚数を決めて登録すると良いと思います。
 - g. 正面、少し左右、少し上下の 5 枚程度が良いと思います。
 - h. 一度に一人分のデータしか保存できません。
 - i. 登録が終われば ESC キーでプログラムを終了します。
 - j. 登録したい人数分上記を実行します。
 - k. 写真データは dataset というディレクトリに登録した名前のディレクトリを作り保存されます。
2. headshots_MultiplePeople.py
 - (ア) headshots_OnePerson.py とほぼ同じですが、終わる時に名前入力時に end とタイプします。
 - (イ) 名前を入力すると、顔撮影に戻ります。
 - (ウ) 複数人を撮影する際に便利です。
3. train_model.py を実行
 - a. 順次 dataset 内の写真を処理して、処理に必要なデータを作ります。
4. face_recognition.py を使用して、顔識別システムの動作を確認する。
5. config.py 設定値を保存しています。
 - (ア) 基本的には変更する必要はないですが、あるとすれば、face_size と tolerance_parameter だと思います。
 - (イ) 画像サイズを変えるとモニタに映る画像の大きさや処理する際の画像が変化し、精度や速度に影響が出ます。

(ウ) face_size

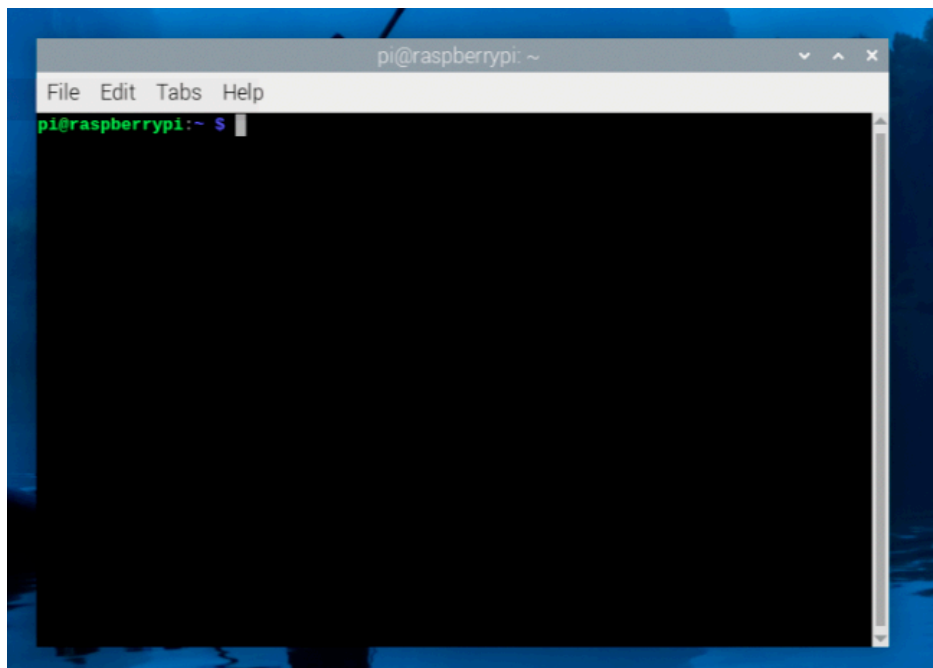
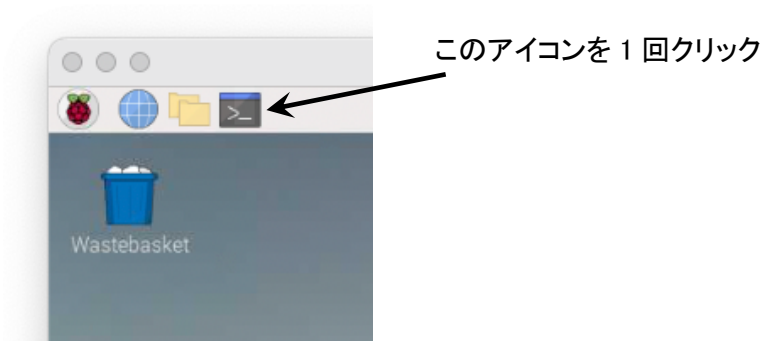
- ① 顔と判別する最小の大きさを設定します。今回の場合は、一度に複数人を識別するとややこしくなるので、100 程度が良いと思います。50 とかにすると離れたところにいる人も識別対象になります。
- ② デフォルトでは、640*480 のカメラ画像に対して、どの程度の大きさ以上の顔を顔と判別するかを決めるパラメータになります。

(エ) tolerance_parameter

- ① 顔を識別する際の精度を設定できます。デフォルトは 0.6 ですが、これを 0.4 とかにすると誤識別は減りますが、識別できずに Unkown になる確率が高くなります。顔データとの差を計算して許容誤差のパラメータになります。

では、実際に動かしてみましょう。

ターミナルを起動します。



ターミナル画面が表示されます。

英数小文字で ls とタイプし、リターンキーを打つと現在いるディレクトリのファイルやフォルダをリスト表示します。

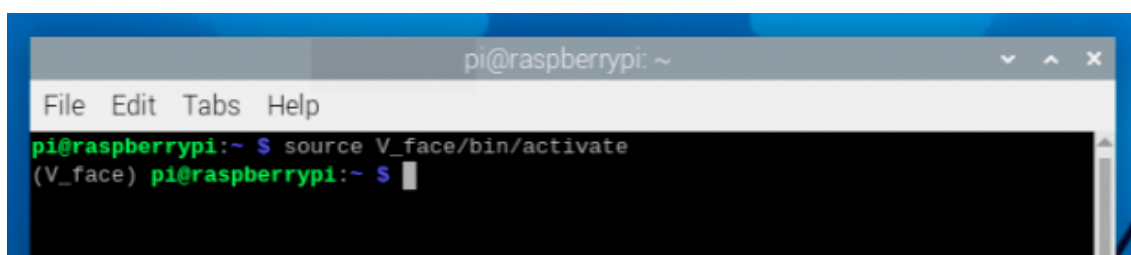
本プログラムは、ラズパイ上に作られた仮想環境で動作します。

仮想環境の名前は V_face です。

プログラムの格納フォルダ(ディレクトリ)は face になります。

まず、仮想環境に入ります。

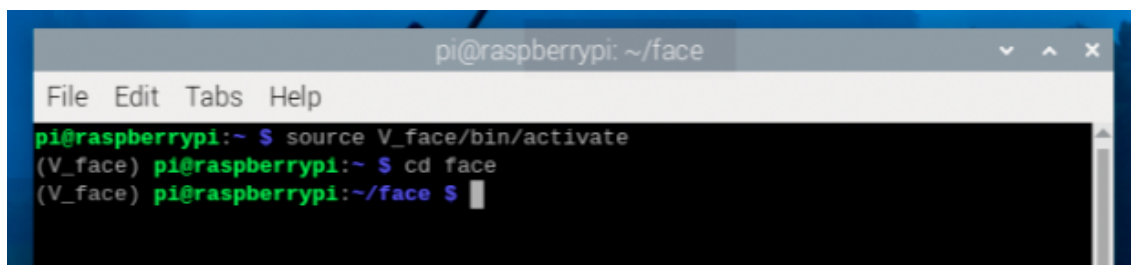
source V_face/bin/activate



```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ source V_face/bin/activate
(V_face) pi@raspberrypi:~ $
```

face ディレクトリに移動

cd face



```
pi@raspberrypi: ~/face
File Edit Tabs Help
pi@raspberrypi:~ $ source V_face/bin/activate
(V_face) pi@raspberrypi:~ $ cd face
(V_face) pi@raspberrypi:~/face $
```

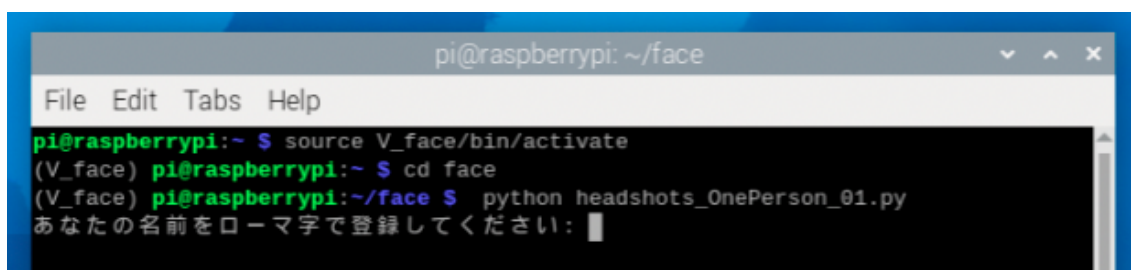
この状態でプログラムを起動します。

python headshots_OnePerson.py

や

python headshots_MultiplePeople .py

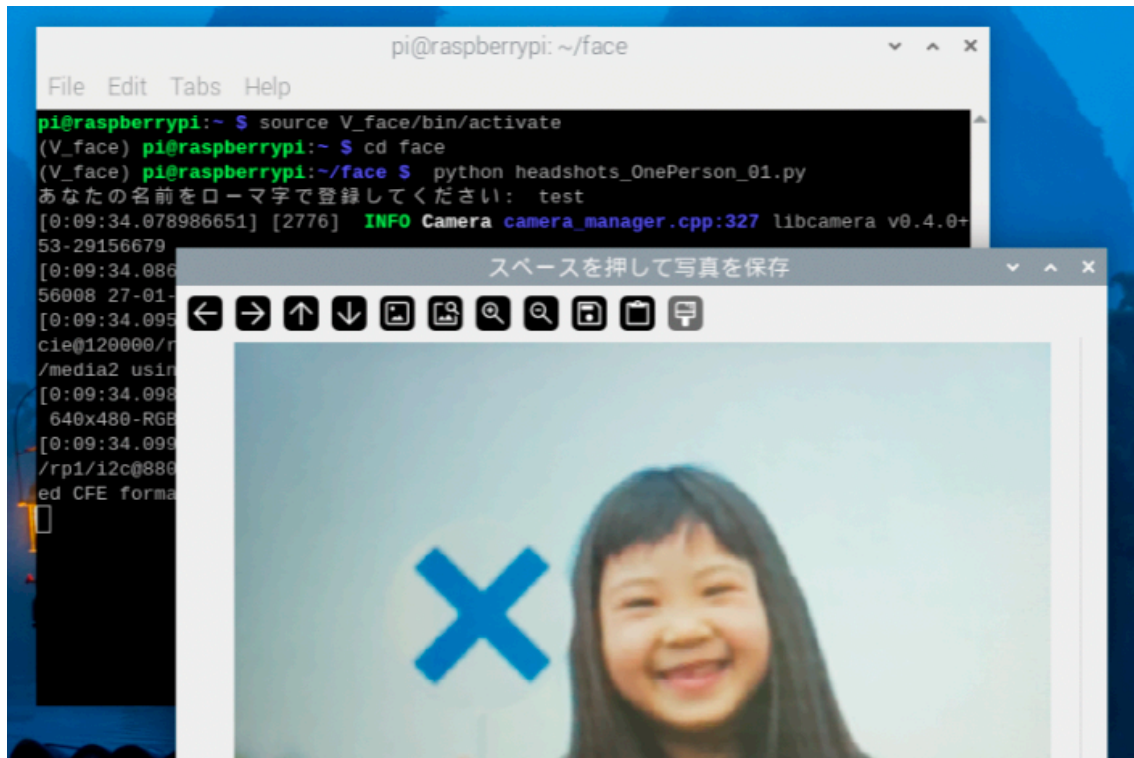
として、プログラムを起動し、顔写真を撮影していきます。



```
pi@raspberrypi: ~/face
File Edit Tabs Help
pi@raspberrypi:~ $ source V_face/bin/activate
(V_face) pi@raspberrypi:~ $ cd face
(V_face) pi@raspberrypi:~/face $ python headshots_OnePerson_01.py
あなたの名前をローマ字で登録してください: 
```

とりあえず、test と入力すると

撮影モードになり



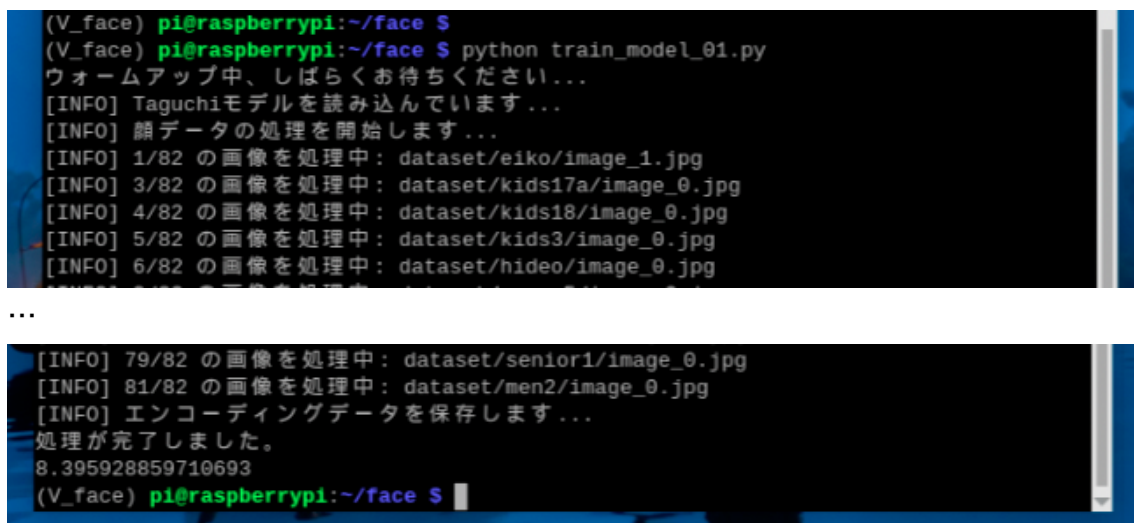
スペースキーを押せばシャッターを切ります。

撮影が終われば、esc キーを押せばプログラムは終了します。

人数分撮影が終われば

python train_model.py

として、顔データを処理します。



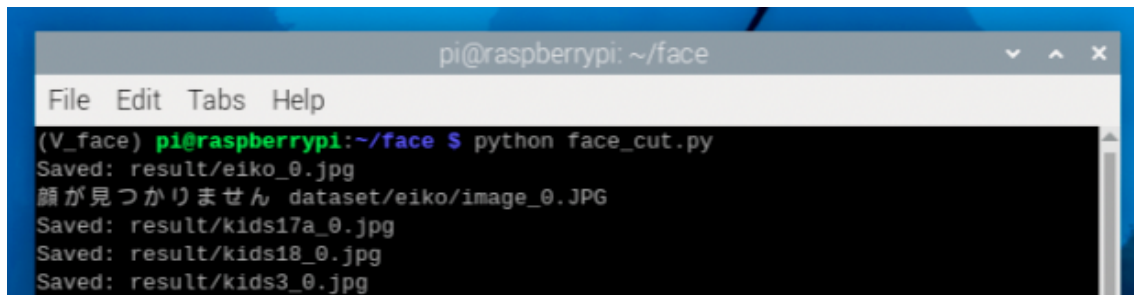
写真の処理が終われば、エンコードデータを保存して、処理時間を表示して終了します。

face_cut.py

顔データが正常に判別できるか確認するためのツールとして用意しました。
実行すると顔の部分を切り出して、result というディレクトリに保存します。

python face_cut.py

と実行すると



```
pi@raspberrypi: ~/face
File Edit Tabs Help
(V_face) pi@raspberrypi:~/face $ python face_cut.py
Saved: result/eiko_0.jpg
顔が見つかりません dataset/eiko/image_0.JPG
Saved: result/kids17a_0.jpg
Saved: result/kids18_0.jpg
Saved: result/kids3_0.jpg
```

顔を判別できたものは、Saved となりますが、

顔が判別できなかった写真は

「顔が見つかりません」とメッセージが出ます。

上記の例では、eiko ディレクトリにある image_0.JPG という写真に顔が見つからなかったことを表しています。

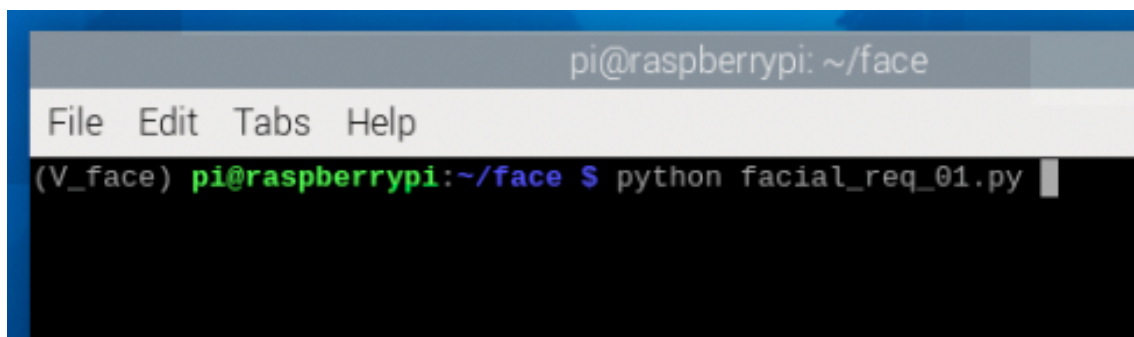
削除して、撮影をやり直してください。

顔識別のテストは

python face_recognition.py

とします。

とりあえず、テストデータが入った状態で、以下を試してください。



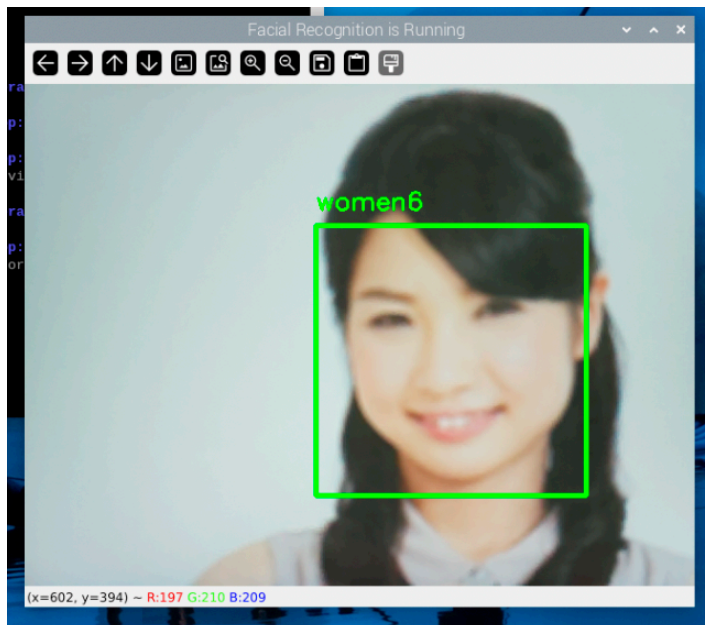
```
pi@raspberrypi: ~/face
File Edit Tabs Help
(V_face) pi@raspberrypi:~/face $ python facial_req_01.py
```

実行すると

```
pi@raspberrypi: ~/face
File Edit Tabs Help
(V_face) pi@raspberrypi:~/face $ python facial_req_01.py
tolerance_parameter= 0.6
ウォームアップ中ちょっと待って

[INFO] loading encodings + face detector...
[0:02:32.724509797] [2144] INFO Camera camera_manager.cpp:353-29156679
[0:02:32.731923832] [2172] INFO RPI pisp.cpp:720 libpisp ve56008 27-01-2025 (21:50:51)
[0:02:32.741273416] [2172] INFO RPI pisp.cpp:1179 Registercie@120000/rp1/i2c@880000/ov5647@36 to CFE device /dev/media3/media0 using PiSP variant BCM2712_C0
[0:02:32.744252278] [2144] INFO Camera camera.cpp:1202 conf640x480-RGB888 (1) 640x480-GBRG_PISP_COMP1
[0:02:32.744353259] [2172] INFO RPI pisp.cpp:1484 Sensor: /rp1/i2c@880000/ov5647@36 - Selected sensor format: 640x480-Selected CFE format: 640x480-PC1g
women6
ok
ok
ok
ok
ok
```

自分の顔を写すと登録されていないので、Unkown と表示されると思います。
添付の写真をカメラにかざしてください。



注)別の写真の場合があります。

カメラ画像が別ウィンドウで表示され、顔を識別します。

上記の場合、女性は women6 として登録されているので、「women6」を表示し、以後識別名が同じなら「ok」と表示するようにプログラミングされています。

2 度以上識別されれば「ok」と表示されるので、この部分のプログラムを改造することで、「women6」を識別した場合の処理を行うことが可能となります。

例えば挨拶システムであれば、

「ok」と表示されるところで、合成音声で「women6 さん、おはようございます。」と発話させるプログラムを実行すれば、簡単な挨拶システムとすることが可能であると考えます。

プログラムを終了する際は、カメラウィンドウを選択している状態で、esc キーを押してください。

<<おまけ>>

なんらかの理由でカメラがフリーズする場合があります。その際プログラムを終了した上で、

`sh kill_video_processes.sh`

として、カメラを掴んでいるプロセスを止めてください。

もしくは、ラズパイを再起動してください。

© 川端孝宣 2025 All rights reserved.