顔認識および顔追跡を搭載した小型ロボットの制御に関する研究

A study on control of small robot equipped with facial recognition and face tracking

葛 伝奇 [1)]
指導教員　林 誠治 [1)]

1）拓殖大学工学部　機械・電子システム工学専攻　林研究室

キーワード：Raspberry Pi，Zumo Robot，顔認識，OpenCV

1. はじめに

昨年および今年，コロナウイルスの影響で自宅自粛する回数がどんどん増えてきた．その際，お年寄りや子供が単独自粛するため，心理面などの原因により寂しさおよび不安感が生じることとなっている．本研究では，人間がロボットと接することで人間の寂しさや不安感を少しでも低減できないか？という最終目標を目指す中で，まずはロボットとのコミュニケーションに関わる重要な要素と考えられる (1) 顔認識（対象者が誰なのかを判断し，親しみを持つ）(2) ロボットとの物理的な位置関係（ロボットを怖がらず適度で自然な距離を確保するため）に着目し，人物認識型リアルタイム制御ロボットのプロトタイプを製作し評価することを本研究の目的とする．

2. 本研究で使用する機器および画像処理プログラム

2.1 ハードウェア

本研究では図 1 に示す Raspberry Pi On Zumo という簡易な移動機構を持つ小型ロボットを使用する．この小型ロボットは主に以下の 3 つの部分で構成される．

- Raspberry Pi 3B+: ARM プロセッサを搭載したシングルボードコンピュータであり，イギリスのラズベリーパイ財団によって開発されている．
- Arduino Leonardo: マイクロコントローラ Arduino シリーズの一種である．UART シリアル通信と SPI 通信をサポートしており，USB ケーブル経由でのシリアル通信も可能である．
- Zumo Shield v1.2: Zumo Shield は Arduino 用の小型シールドであり，3 軸加速度センサーおよび 3 軸磁場センサーなどを搭載している．

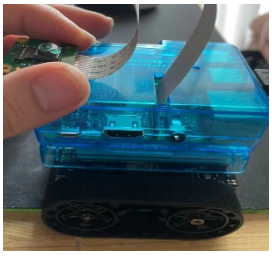

図 1: Raspberry On Zumo

2.2 画像処理ライブラリーOpenCV

OpenCV とは，インテルが開発したオープンソースであり，コンピュータビジョン向けの画像処理・画像解析および機械学習などの機能を持つ C/C++，Java，Python，MATLAB 用ライブラリーである．本研究では，画像解析および機械学習などの相性も比較的に良いとされる Python を主に使用するため，Python 向けラッパーの OpenCV-Python を導入している．これにより Python だけで OpenCV を容易に活用することができる．

3. 本研究で開発するソフトウェア

本研究における PC および Raspberry Pi でのプログラム開発環境を表 1 に示す．

表 1 プログラム開発環境

	PC	Raspberry Pi
OS	Window 10	Raspberry Pi OS
Camera	Web Camera	Pi camera V2

3.1 顔認識プログラム

ロボットと利用者の顔との距離の変化に応じてロボットを移動する必要があるため，利用者の顔領域の抽出を行う．同時にカメラに写っている顔が誰なのかを判断する顔認識を行う．これらは次の 3 つのプロセスからなる．

1. face dataset: ロボットに搭載されているカメラモジュールから複数の利用者の顔写真を登録する
2. face training: face dataset に基づいて顔特徴を分析し，モデルを学習する
3. face recognition: 作成されたモデルを使用して，リアルタイム入力画像の顔を認識する

3.2 データセットの用意

face dataset を生成する際，顔写真に対応したユーザー名を登録する必要があり，登録の順番によりユーザー名に紐づけた数字を割り当てることにした．haarcascade という OpenCV にある画像処理ライブラリーを使用する．本研究では，利用者の顔の正面だけを認識させたいため，目の特徴を取得するための haarcascade eye，顔の前面全体の特徴を取るための

haarcascade frontalface alt, 笑顔の表情を認識するための haarcascade smile という 3 つのデータ分類器を使用した. カメラモジュールにより得られた画像の中に顔領域が検出できれば, 事前に設定された画像の枚数まで自動的に顔領域だけを切り出し, グレースケールに変換した後, dataset というフォルダに保存する. face dataset ステップのフローチャートを図 2 に示し,

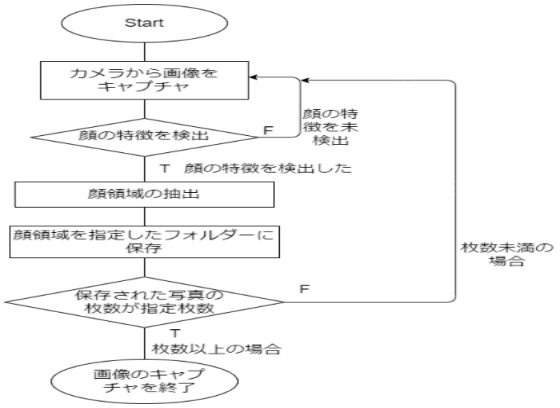

図 2 face dataset ステップのフローチャート

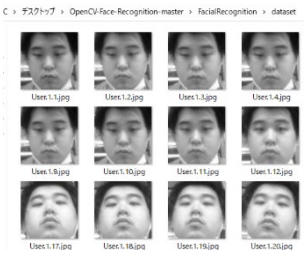

図 3 dataset フォルダの画像の一部

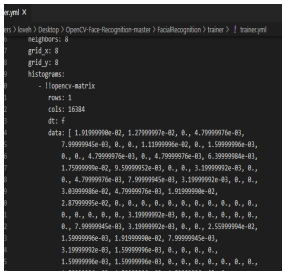

図 4 trainer.yml の作成

dataset フォルダ中の画像の一部を図 3 に示す.

3.3 顔特徴モデルの生成

face training ステップでは, face dataset で用意された dataset というフォルダ中のすべての画像を用いて haarcascade の 3 つの分類器(目, 顔全体および表情の特徴) を使用することで学習しモデルを生成する. 生成された顔特徴に対応するモデル(trainer.yml) の一例を図 4 に示す.

4. 実行結果

4.1 顔認識プログラムの実行

カメラモジュールで撮影された画像の顔領域に対して, trainer.yml モデルを用いて登録されている顔写真特徴の一致度を計算し, 最も一致度が高かったユーザー名をリアルタイムで表示する. もし, 顔データが未登録の場合や極端に一致度が低い場合は「Unknow」と表示している. face recognition ステップのフローチャートを図 5 に示し, その実行結果の一例を図 6 に示す.

4.2 顔追跡プログラムの実行

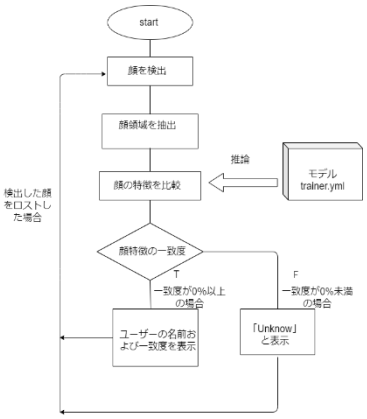

図 5 face recognition ステップのフローチャート

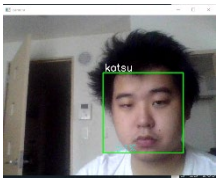

図 6 認識の実行結果例

顔追跡プログラムでは, 入力画像から顔領域を検出した場合, リアルタイムで顔領域の矩形の対角線の大きさから, ユーザーとロボットの間の距離を推定する. 顔領域の矩形の対角線の大きさとロボットとユーザーの距離が比例する形で, つまり顔領域のサイズが大きくなるとロボットが遠ざかるように, ロボットの移動機構を司る Zumo Shield に前進/後退および停止などの制御命令を発行する. その際, Raspberry Pi と Arduino は USB ケーブルで接続しシリアル通信を行う. ロボットの移動は Arduino 専用の Zumo ライブラリーを利用する. また, 撮影の際, 検出した顔領域がカメラ領域の中央にくるように, Zumo Shield の左右回転命令も同時に用いてロボット自体を回転させたいと考えている.

5.まとめ

現在の研究段階で, 顔認識した場合の一致度は 60～75%である. 認識の一致度を向上させる方策として, 顔の位置を微妙に上下左右に動かすことで少し異なる角度から得た顔写真を顔画像データベースに追加することが考えられる. また, 顔認識を実行する際に, 取り直しすることなしに画像処理だけで背景の明るさを調整することによって, ある程度一致度を高めることが可能なのではないかと考える. さらに, 顔追跡プログラムを早急に完成させ, ロボットと人間の距離の在り方について実際に被験者に体験してもらいながら評価を行いたい.

参考文献

[1] 北山直洋, “Python で始める OpenCV4 プログラミング”, 株式会社カットシステム

[2] Raspberry Pi on Zumo, Physical Computing Lab, https:// www.physical-computing.jp/product/1302 (参照 2020-07-10)