2024 年度 S セメスター 機械工学総合演習第二　アドバンストプログラミング　レポート

簡単な 1 自由度アバターロボットの製作

03240281　椿　道智[*1]

[*1] 東京大学工学部機械情報工学科 3 年

1. はじめに

カルディかどこかで買ったと思われるお菓子が入っていた変なロボット型の置物（図1）が家の自室においてあった．いつ貰ったのかあまり覚えていないが，ロボットなのに動かないのはおかしい！ので，今回の課題ではこれに魂を吹き込むことにした．

図1　ロボット型の置物

2. 概要

今回モチーフにするロボット型の置物（図1）には，目と口がある．また，首が蝶番で動くようになっている．せっかく人間に似ているので，実世界の自分の動きと連動させたら面白いと考えた．よって，このロボットの目や口を光らせたり，首を動かしたりすることにした．本課題では，総合演習第二のプログラミング演習で学んだ項目 2 つ以上を直接生かすことが期待されている．今回は，「CV プログラミング」と「マイコンプログラミング」の 2 項目を応用し，さらに物体検知アルゴリズム yolo v8 による画像認識を用いて実装した．

3. 機能と実装方法・工夫点

3.1 開発環境

OS: Microsoft Windows 11 Home
CPU: 13th Gen Intel(R) Core(TM) i7-1360P
メモリ: 16GB
ストレージ: 236GB
Python version: Python 3.11.9
IDE: Visual Studio Code 1.90.0

3.2 全体構成（アーキテクチャ）

製作したロボットのプログラムのアーキテクチャを図2に示す．メインの画像認識処理は 3.1 にスペックを示したパソコンで行い，シリアル通信でその結果を Arduino NANO に送信して，Arduino 側の処理を行う．

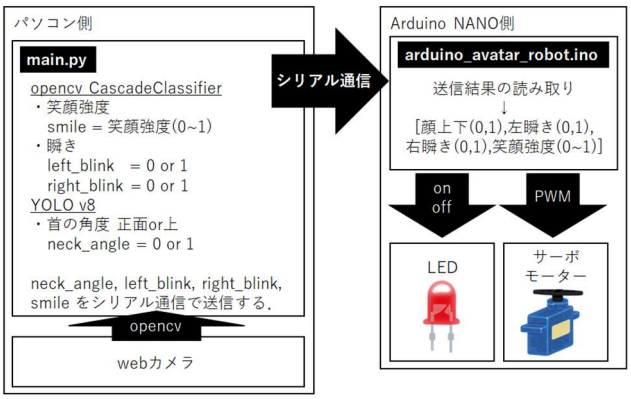

図2　アバターロボットのアーキテクチャ

3.3 パソコン側の処理(main.py)

(1) 使用したライブラリ

- opencv : 4.8.1
- pyserial : 24.1.1
- ultralytics : 8.1.20

(2) 瞬き（目）

opencv の CascadeClassifier「haarcascade_eye_tree_eyeglasses.xml」を用いて，目の開閉を検出し，それに応じて LED を点灯させることにした．目の開閉は，単純に opencv が目として検出するかどうかで判定する．上の分類器が，目を閉じていると検出しないという特性を利用している．

工夫点

① 試行錯誤で，この分類器を用いれば，目を閉じているとき，目を検出しないということに気づき，それをうまく利用して，めんどくさい判定アルゴリズムを回避した点が工夫点である．

Email: tsubaki@g.ecc.u-tokyo.ac.jp [*1]

(3) 笑顔(口)

opencvのCascadeClassifier「haarcascade_smile.xml」を用いて，笑顔の強度を出力し，その強度に応じてLED の点灯個数を替えることにした．Qiita 記事「Python と OpenCV を使った笑顔認識」を参考に，笑顔の程度に応じて「0～1」の連続値を返すようにmain.py に実装した．結果を変数 smile に格納し，シリアル通信で arduino に送信する．

工夫点

① 笑顔検出はQiita記事をほぼそのまま利用したので，工夫点はない(つもり)．

② だが，コピペするだけだと当然動かなかったので，前後のコードとの整合やデバッグ用の描画コードを書いたりして色々追記した．

(4) 首が上を向いているか正面を向いているか

ロボット型置物の頭が蝶番で開くようになっているので，自分の首の動きと同期させてサーボモーターで動かすことにした．「首の角度 python opencv」とググると，「Python と OpenCV+dlib を用いた頭部方向推定」というQiitaの記事がヒットしたが，行列の計算処理が重くなりそうだと考えた．今回は，連続的に首の角度の値を得る必要はなく，上を向いているか正面を向いているかの二値分類ができれば良いと考えた．そこで今回は，汎用な物体検知アルゴリズム YOLO v8 を用いて顔が正面を向いているか上を向いているかを判別させることにした．

まず，上を向いた写真を 5 枚，正面を向いた写真を 5 枚，計 10 枚の自分の写真を撮影し(capture.py で自動撮影)，アノテーションツール VoTT で label 付けして，yaml ファイルなどを作成して，「データセット」を自作した．次に，このデータセットを YOLO に学習させてモデルファイル(last.pt)を生成し，そのモデルを用いて正面を向いていれば 0，上を向いていれば 1 のラベル名を返すように main.py に実装した．結果を，neck_angle という変数に格納し，シリアル通信で arduino に送信する．

工夫点

① 顔が正面を向いているか上を向いているかの判定で，特徴点抽出して真面目にアルゴリズムを考えると難しそうだが，YOLOならなんでか知らんけど判定できそう！と考えられたところが工夫点だと思う．(色々沼ってて思いついたときは感動した())

② YOLOのデータセット(アノテーション作業)をささっと製作したところも工夫点って言ってよさそう．

3.4 Arduino NANO 側の処理 (Arduino_avatar_robot.ino)

電子回路演習と同じ手順で，Thinker CAD で作成した回路図を図 3 に示す．Arduino UNO は，Arduino NANO に置き換えて(読み替えて)見てほしい．回路図に示す通り，この avatar ロボットは，学科から配布された，Arduino NANO，サーボモーターと 5 つの赤色 LED，抵抗器から構成されている．ハードウェアという観点で言えば，サーボモーターにはタミヤのユニバーサルアームがついていて，リンクにより頭を開閉している．

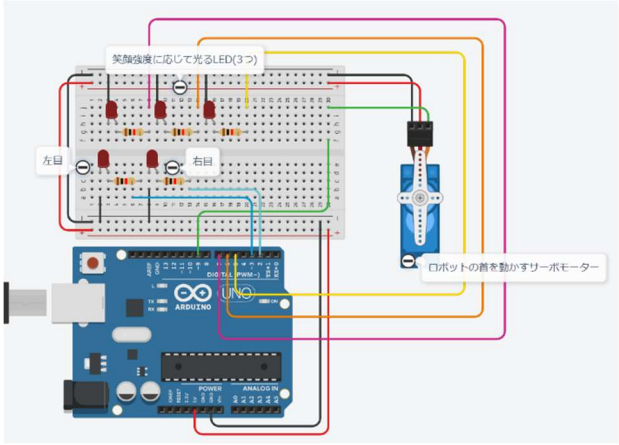

図3　回路図(Thinker CAD で作成)

Arduino 側は，Serial 通信を受信し，受信した値(smile, neck_angle, left_blink, right_blink)に基づいて，サーボモータの PWM 制御と LED の ON OFF 制御を行った．

left_blink が 0 のとき，3 番ピンの LED を光らせ，1 のとき消した．right_blink も 2 番ピンで同様の制御をした．

neck_angle が1(上を向いている)のとき，サーボモータの角度を 170 に，0(正面を向いている)のとき，サーボモータの角度を 10 に，それ以外のバグの時 90 になるように PWM 制御した．

smile はその値の大きさ(0~1)に応じて LED の点灯個数が 0~3 個で変動するように条件分岐で ON OFF 制御した．各条件分岐のパラメタは試行錯誤で調整した．

なお，opencv による検出結果(main.py)をシリアル通信で Arduino に送信し，受信するプログラム関連は，「Python+OpenCV+dlib で居眠りを検出して Arduino でエアコンの設定温度を下げる」(nomolk のブログ)のブログ記事を参考に試行錯誤したものである．

4. プログラムのフローチャート

パソコン側のプログラム(main.py)のフローチャートを図4に示す．また，Arduino 側のプログラム(arduino_avatar_robot.ino)のフローチャートを図5に示す．いずれのフローチャートも drawio で作成した．

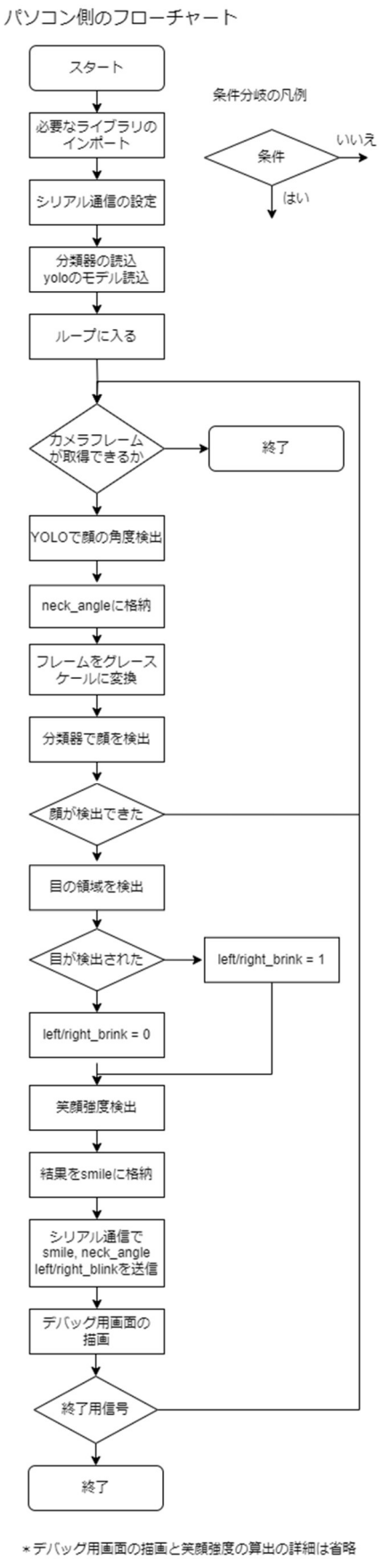

図4　パソコン側のプログラムのフローチャート

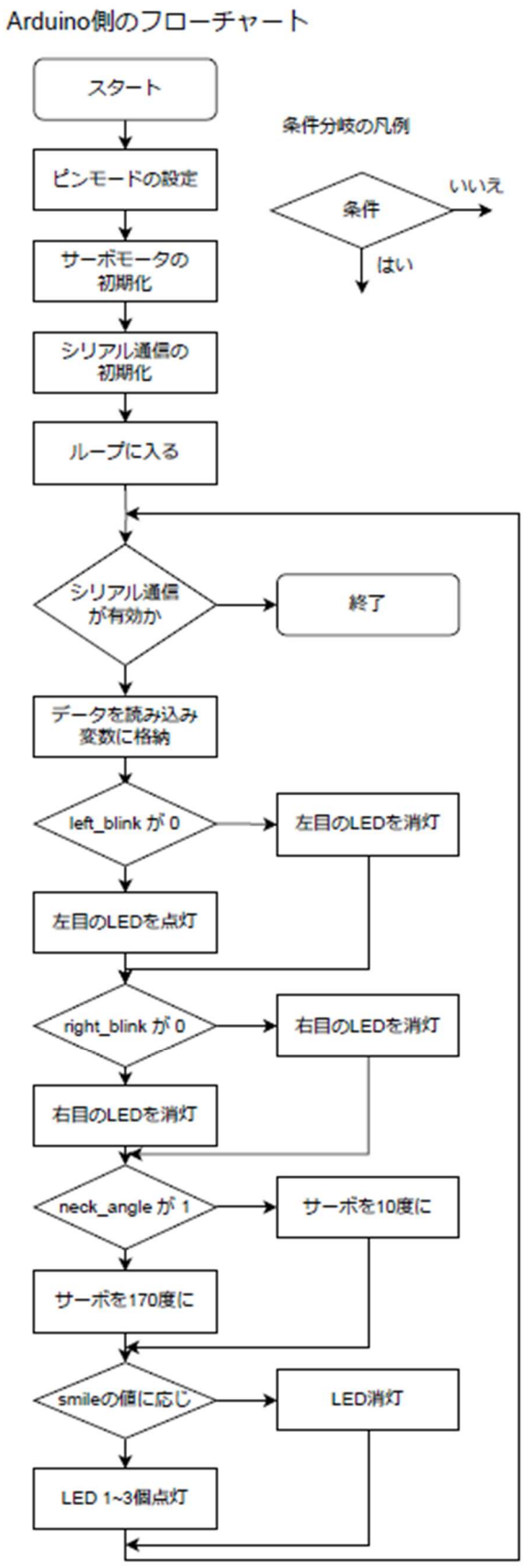

図5　Arduino 側のプログラムのフローチャート

5. ロボットの写真・プログラムの実行時の写真

図6に改造が完成したロボットのし写真を，図7にロボットの様子を，図8に opencv の描画ツールを用いてで Python で作成したデバッグ用の画面を示す．

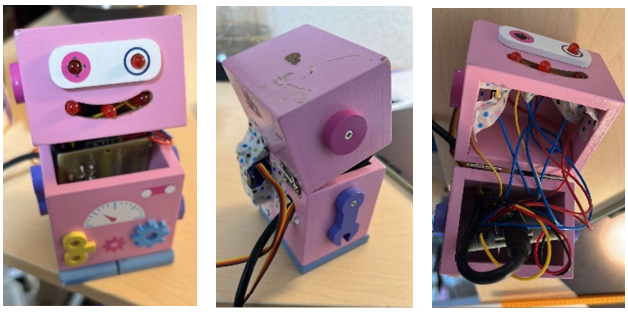

図6　ロボットの写真

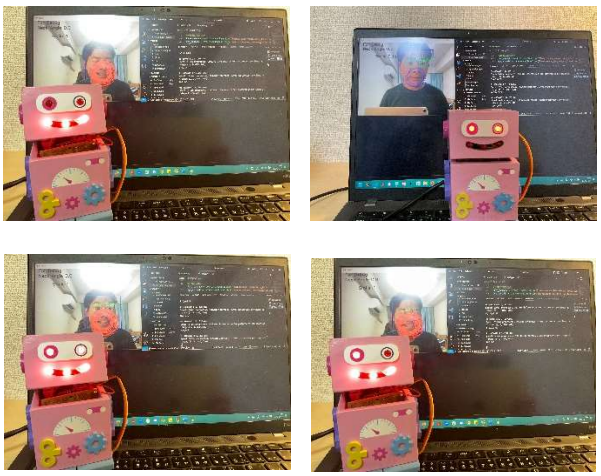

図7　実行時の写真

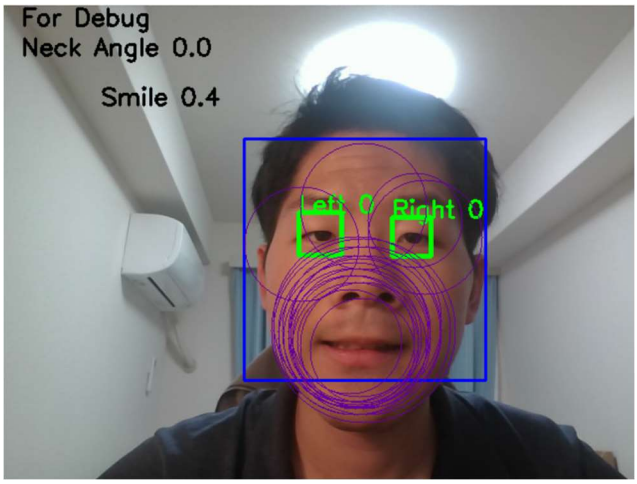

図8　デバッグ用画面

6. 発表の用いたスライド

発表に使用したスライドを図9に示す．

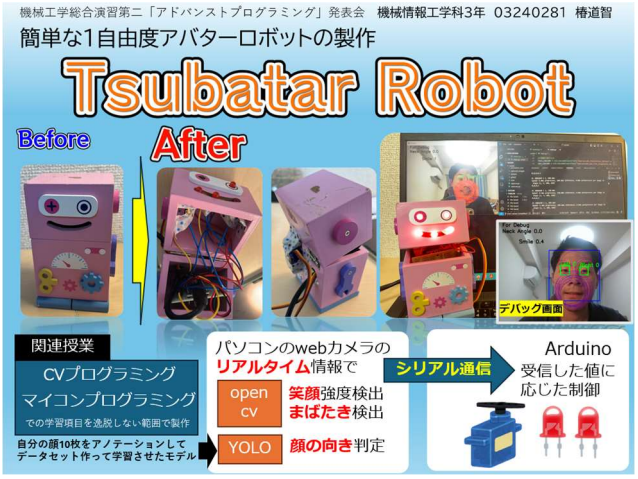

図9　発表に用いたスライド（PowerPoint で作成）

7. 改善点（今後(本当は)やるべきこと）

ソフトウェア周りは 1 日で作ったので，YOLO の学習データセットに写真が 10 枚しかなく，Validation もしっかりしていないので，画像認識の精度が良くない．例えば，眼鏡をはずしてコンタクトの状態でデータセットを作ったため，眼鏡を付けた状態だと少し精度が良くない気がする．物体検知アルゴリズムで，状態推定を試みるならば，本来，眼鏡をかけたり外したり，白黒反転させたり，部屋を暗くしたり明るくしたりと，色々なシチュエーションの画像でデータセットを作るべきなので，モデルの改善が必要である．（Validation や Test もすべき）ただ，今回は時間が無いので，「動けばいい」を目標にしていたので目標は達成．

設計面での改善点としては，Serial 通信のタイムラグのせいなのか，PC の性能の問題なのか，応答性に Latency を感じるのでそれを改善したい．また，口の LED がそろって光らないといった現象が起ころことがあり，配線不良か電流が定格を超えて必要になっていることが考えられるので，そのあたりを改善すべきである．

8. プログラム・データセットのフルバージョン

アップロード可能なファイルのデータ容量が限られているので，UTOL には基盤のコードである main.py と arduino_avatar_robot.ino の 2 点しか提出していないが，この 2 点のファイルだけでは実行できない．実行に必要な xml ファイル（opencv の分類器）や pt ファイル（YOLO 用の重みファイル），YOLO の学習データセット，前処理用のコードなどを github の以下のリポジトリで公開した．

https://github.com/Michi-Tsubaki/avatar_robot

採点などで実行される際には，必要に応じて参照してほしい．

9. 反省

はじめは，「振動アクチュエータ駆動のロボットを制御して，笑顔を撮影しに行くロボット」を製作しようと考えていたが，ESP 32 - cam のマイコンの wifi 設定がうまくいっていないのか，安かったので壊れているのか，camera が開けず，色々試行錯誤していたら時間が溶

けてしまった．そのままテスト期間に突入したので，結局諦めて，適当なロボットを提出してしまうあたり，スケジュール管理能力が無いなと反省している．急いで作ったので，当然秋月に行く時間もなく，家にあるもので済ませたので，目も口も赤色のLEDになってしまった．気持ち悪い見た目になって反省している．

結局使わなかった ESP 32 CAM は，夏休みの時間があるときにでも供養する．（時間があれば）

10. 計算機演習の学習内容についての意見

OpenCV，OpenGL・並列計算の演習は教材が網羅的かつ体系的だったので，大変充実していた．数値計算プログラミングに関しては，授業の意図があまりよくわからなかった．数値計算をしっかりと学ぶという意図であれば，MATLAB ではなく，cpp などで行列計算の実装や差分の取り方(中心差分と前進差分でどうちがうか)，連立方程式の解き方などを学びたかったし，MATLAB に慣れようということなら，古典制御で連成振動問題を解くような課題ではなく，現代制御での行列計算の注意みたいなことが学べるような内容が良かった．「プログラミングの復習」（バトルシップ）の授業も個人的には良くないと思っていて，コンパイルの仕方とかの事務手続き的な内容しか授業で説明がなく，アルゴリズム（dfs,bfs,a-star,…）とかの授業をせずに，適当にゲーム課題を出して「遊びながら考えて」というのはあまり効果的な学びになっていない気がした．少なくとも，探査方法について説明があっていい気がした．

ただ，atcoder みたいなことをひたすらやっている電気系の演習と違い，ロボット（実世界）のプログラミングに生きる内容が充実していて，全体として網羅的な構成の演習だったので良かったです．

11. 参考にした記事など

- nomolk のブログ「Python+OpenCV+dlib で居眠りを検出して Arduino でエアコンの設定温度を下げる」https://nomolk.hatenablog.com/entry/2018/07/28/005600
- Qiita「Python と OpenCV を使った笑顔認識」（@fujino-fpu(秀則 藤野)） https://qiita.com/fujino-fpu/items/99ce5950f4554fbc17d
- Qiita「OpenCV Haar-Cascade による顔検出」（@tnoce in AliEaters (Alibaba Cloud Developers meetup)）https://qiita.com/tnoce/items/c819c85a85c16d246be8