

ラズ☆ロボ・カー紹介

(MFT2017で好評の汎用電動機制御基板によるロボット・カー製作記事)

●目次

- 1) ハード:必要なハードの構成
- 2) ソフト:使用しているデバイス用のpythonプログラム
- 3) システム:ロボット・カーを動かす
NODE-REDプログラム

0.1 はじめに

‘楽しく、早く、安く’ 電子工作を提唱しています。

MFT2017で展示し好評を博した、ラズ☆ロボ・カーの作り方を公開します。

まず、下のURLの映像を見てください。

<https://www.facebook.com/466797943517881/videos/717124401818566/>

https://youtu.be/_opalFqS50s

単体としては左右の距離センサと、左右のDCモータが、それぞれ独立に動いています。そのセンサ情報を、ラズパイ、NODE-RED、MQTTと経由して、相手のロボットに送ることで、この動きを実現しています、このような動きを実現でき、楽しんでいます。

こんなに楽しいロボットカーが、汎用電動機制御基板を使えば、簡単に作れます。

自分は、回路設計が専門なので、ソフト得意の人が、改良してくれて、情報をいただけると助かります。

このように、オープンハードウェアの考えに立ち、情報を公開しますので、みんなで、もっと楽しい電子工作ができると嬉しいです。

0.2 はじめに

単体としては左右の距離センサと、左右のDCモータが、それぞれ独立に動いています。そのセンサ情報を、ラズパイ、NODE-RED、MQTTと経由して、相手のロボットに送ることで、この動きを実現しています、このような動きを実現でき、楽しんでいきます。

こんなに楽しいロボットカーが、汎用電動機制御基板を使えば、簡単に作れます。自分は、回路設計が専門なので、ソフト得意の人が、改良してくれて、情報をいただけると助かります。

このように、オープンハードウェアの考えに立ち、情報を公開しますので、みんなで、もっと楽しい電子工作ができると嬉しいです。

ロボット・カーを動かすには、多くの分野の技術を知る必要があります、その技術、ノウハウを下記の順番で説明します。

- 1) ハード:必要なハードの構成
- 2) ソフト:使用しているデバイス用のpythonプログラム
- 3) システム:ロボット・カーを動かすためのNODE-REDプログラム

1.1 ラズ☆ロボの設計図:ハード

ハード→ソフト→システムの順に説明するよ！！

距離センサ

GP2Y0A21YK0F
10cm～80cm、5V



入力
センサ

アナログ/GPIO入力

3.3V系:6入力
5V系:2入力



MCP3208 12bit
8ch A/Dコン
バータ SPI

サンプリング速
度:50ksps 入
力範囲:0～3.3V



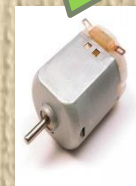
PCA9685
12bitPWM、
16ch(内6ChをT
B6612に使用
)I2C



DCMotor出力
2出力

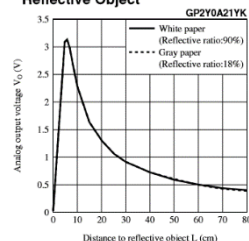


DCMotor



出力
アクチュエータ

Fig.5 Analog Output Voltage vs. Distance to Reflective Object



汎用電動機制御基板



1.2 汎用電動機制御基板で工夫した点

- ・バランスの取れた入出力対応

アナログ入力と、DCモータ制御を一枚の基板で対応しているが見つからなかったので、この基板を作りました。

- ・去年のMFT2016で、ソフトの専門家から、はんだ付けしないでラズパイを使いたいという声を聴きました。

- ・信号の見える化

以前の設計で、DCモータの回転の向きが分かりづらく苦労したので、モータの回転方向

時計回り:clock wise(CW)

反時計回り:Counter clock wise(CCW)

の信号に色を変えてLEDで表示し、実際にモータをつながなくてもデバッグできます。

- ・モータ電源を、外部だけでなく、内部の5V電源も切り替えられるようにしました。

- ・次は、結果的にそうなったという意見もありますが、ラズパイGPIO端子と、ADコンバータの信号線を同じ端子に接続しました、こうすることにより、ADコンバータの強力な入力保護回路が、ラズパイに対しても保護してくれます。

- ・arduino等、5V系の豊富なI2Cモジュールに対応できるようにI2C 5V系に対応

- ・PWMで3ピンのサーボモータ(例:SG90)を、直接接続できるようにしました。

1.3 必要な部品表

No.	品名	購入先	URL	参考単価	個数	参考価格
1	ラズパイ3	KSY	https://raspberrypi-ksyic.com/	4,830	1	4,830
2	汎用電動機制御基板	ビット・トレード・ワン	http://btoshop.jp/2017/08/25/4562469771304/	3,980	1	3,980
3	タミヤ製ダブルギヤボックス(左右独立4速タイプ)Item No: 70168 モータ2ケ付き	tamiya	http://www.tamiya.com/japan/products/70168double_gear_box/	907	1	907
4	タミヤ製オフロードタイヤセットItem No: 70096	tamiya	http://www.tamiya.com/japan/products/70096off_road_tires/index.htm	388	1	388
5	タミヤ製ボールキャスターItem No: 70144	tamiya	http://www.tamiya.com/japan/products/70144ball_caster/	388	1	388
6	電池ボックス(単3*2) DCモータ用	サトー電気		60	1	60
7	モバイルバッテリー	Anker	https://www.amazon.co.jp/Anker-PowerCore-%E3%83%A2%E3%83%90%E3%82%A4%E3%83%AB%E3%83%90%E3%83%83%E3%83%86%E3%83%AA%E3%83%BC-Android%E5%AF%BE%E5%BF%9C-A1263011/dp/B019GNUT0C/ref=zg_bs_2544551051_1?_encoding=UTF8&psc=1&refRID=EM9HWODD4P29J6AYTNW3	2,399	1	2,399
8	マイクロUSBケーブル	100均		100	1	100
9	距離センサ シャープ測距モジュール GP2Y0A21YK	秋月	http://akizukidenshi.com/catalog/g/gI-02551/	450	2	900
10	L字金具2枚(距離センサ取付用)	100均		100	1	100
11	ラズパイ実装ボードキット(予定) ボード2枚、スペーサ、ビス、ナット セット	ビット・トレード・ワン		500	1	500
					合計	14,552

1.4 組み立て・配線手順

下記の手順で組み立てます

- 1) ダブルギヤボックスを組み立て、左右のタイヤを取り付け
- 2) ラス'パイ実装ボードにダブルギヤボックス取り付け
- 3) ラス'パイ実装ボードに左右の距離センサを取り付け
- 4) ラス'パイ実装ボードにラス'パイ3を取り付け
- 5) ラス'パイ3に、汎用電動機制御基板を取り付け
- 6) ラス'パイ実装ボードに、電池ボックス(単3*2)を取り付け
- 7) ラス'パイ実装ボードに、モバイルバッテリーを取り付け

次に配線をします

- 1) 左右の距離センサと、汎用電動機制御基板をそれぞれ3本の信号線で接続
- 2) ダブルギヤボックスについている左右のモータと、汎用電動機制御基板をそれぞれ2本の信号線で接続
- 3) 電池ボックスと、汎用電動機制御基板を2本の信号線で接続
- 4) モバイルバッテリーから、ラス'パイ3に電源を接続(マイクロUSBケーブル使用)

ここまでで、ハードは完成です、部品がそろえば、数時間でできると思います。

2.1 ソフト:デバイス用のpythonプログラム

必要なソフトの構成を、下記のように考えてみます。

- 1) 入力処理: センサからの情報を論理処理部に伝える。
- 2) 論理処理: 動かしたい内容を論理的にプログラムする。
- 3) 出力処理: 論理処理部からの情報で、アクチュエータ(DCモータ)を制御する。

従来のソフトの作り方は、Python言語で、入力、論理、出力処理を一つのプログラムで作るのが一般的であると思います、自分も今まではそういう作り方でした、ただこのような作り方だと、ネットワーク処理、並列処理が難しくなります。

今回、ネットワーク処理、並列処理をできるだけ容易に行うため、論理処理部をNODE-REDにし、

入力、出力処理部を別々に、pythonプログラムで作ることにしました。

2.2 ソフト:デバイス用のpythonプログラム

入力処理部のソフト仕様

- ADコンバータから、値(ch0からch6)を読む
左右の距離センサは、ch6,7に接続しているので、基準距離(例:30cm)を0とする
基準距離より近ければ -0.5から0の値
基準距離より遠ければ、 0から+0.5の値
- その値を、JSON形式で、複数の値を同時にMQTTプロトコルで、NODE-REDに送る
JSON形式にしたのは、node-redで扱いやすく、複数の値を同時に送れ、値の種類を自由に定義できるからです。
MQTTにしたのは、IoT用のプロトコルで、NODE-REDで標準でノードがあり、扱いやすいからです。

出力処理部のソフト仕様

- node-redからMQTTプロトコルで、複数の値を同時にJSON形式で受ける。
- DCモータを動かすための、値(ch6からch7)を読む
読んだ値(-0.5~0~+0.5)により、DCモータを前進、後進させる

2.3 ラズ☆ロボの設計図:ソフト(デバイスドライバ相当)

オープンソースのPythonライブラリを少し改造しています

①センサ入力用Pythonソフト
センサの情報をNODE-REDに伝える
入力: 12bitアナログ入力
出力: MQTT、JSON形式

②DCMotor出力用Pythonソフト
③サーボ出力用Pythonソフト
NODE-REDの情報でモータを制御する
入力: MQTT、JSON形式
出力: DCMotor 2ch、orサーボ8ch

```
MCP3208-MQTTout-pub-ut-pub-DCmotor.py (3.4.2)
File Edit Format Run Options Windows Help

#-*- coding: utf-8 -*-
# v1.0 2017/7/15 jh1cdv
# v1.1 2017/7/28 jh1cdv
# Input: MCP3208からSPI通信で12ビットのデジタル値を取得。0から7の8チャンネル使用
# Output: -0.5 ~ 0 ~ +0.5
# mqtt topic=/in-ch0,1
#

import RPi.GPIO as GPIO
from time import sleep

import context # Ensures paho is in PYTHONPATH
import paho.mqtt.client as mqtt

import json

def on_connect(mqttc, obj, flags, rc):
    print("rc: " + str(rc))

def on_message(mqttc, obj, msg):
    print(msg.topic + " " + str(msg.qos) + " " + str(msg.payload))

# Scale: 0 to 80 cm
# Distance to reflective object L (cm)
```



```
Thonny - /home/pi/raz-robo/pwm/pwm-pca9685-mqttin-sub.py @ 249:1
File Edit View Run Tools Help

pwm-pca9685-mqttin-sub.py
# Simple demo of the PCA9685 PWM servo/LED controller library.
# This will move channel 0 from min to max position repeatedly.
# Author: Tony DiCola
# License: Public Domain
#
# name: pirobo
# Author: jh1cdv
# date: 2017/07/02
# input sensor1,2
# process input to output -0.5 +0.5
# output: rotationserbo

import RPi.GPIO as GPIO
from time import sleep
```

M出力: PCA9685
により制御

下記3つのライブラリをインストールしています

- Adafruit_Python_PCA9685
- Adafruit-Motor-HAT-Python-Library
- paho.mqtt.python



2.4 ソフト:デバイス用のpythonプログラム

まず、ハードがソフトから見れるかどうか確認するため、sshターミナルソフト(例: Mobaxterm)で、下記コマンドを実行し、I2Cデバイスを確認しておきます。

```
pi@raspberrypi:~ $ sudo i2cdetect -y 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
60: 60  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70: 70  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
pi@raspberrypi:~ $
```

Pythonプログラムのファイルのリストを下記に示します

No.	名称	内容
1	MCP3208-MQTTout-pub-DCmotor.py	アナログ入力をMQTTでNODE-REDに渡す
2	pwm-pca9685-mqttin-sub-dcmotor.py	NODE-REDから、MQTTで情報を受け取り、DCモータを制御する

適切な、フォルダに、上記2ファイルをコピーしておく
自分は、WINSCPを使いコピーしています。

2.5 ソフト:デバイス用のpythonプログラム

Pythonプログラムには、下記ライブラリ等のインストールが必要です

No.	名称	内容	URL
1	Adafruit_Python_PCA9685	PWM出力のライブラリ	https://github.com/adafruit/Adafruit_Python_PCA9685
2	Adafruit-Motor-HAT-Python-Library	DCモータ出力のライブラリ	https://github.com/adafruit/Adafruit-Motor-HAT-Python-Library
3	paho.mqtt.python	MQTTのpythonライブラリ	https://github.com/eclipse/paho.mqtt.python
4	mosquitto	MQTTブローカ	http://mosquitto.org/2013/01/mosquitto-debian-repository/

ライブラリのインストールは、下記のようにしました。

- 1) まずGITのサイトにブラウザで見えます。
- 2) サイトの説明にある下記コマンドをsshで実行します

2.6 ソフト:デバイス用のpythonプログラム

実行結果を張っておきます。

```
sudo pip install adafruit-pca9685
pi@raspberrypi:~ $ sudo pip install adafruit-pca9685
Installing collected packages: adafruit-pca9685
Successfully installed adafruit-pca9685-1.0.1
pi@raspberrypi:~ $
```

**ほかのライブラリも同様にインストールします。
MQTTブローカは、下記のようにinstallします。**

```
Sudo apt-get install mosquitto
```

次に、pythonのプログラムを実行してみます。

```
pi@raspberrypi:~ $ cd raz-robo/spi-mcp3208-ad-conv/
pi@raspberrypi:~/raz-robo/spi-mcp3208-ad-conv $ sudo python3 MCP3208-MQTTout-
pub-DCmotor.py
print文を入れてありますので、適切な値が表示されれば成功です。
```

2.7 ソフト:デバイス用のpythonプログラム

苦勞した点

・出力用pythonプログラムで、jsonデータを読み込むとき、node-redで想定外の文字が余計に付加され、うなく読めませんでした、pythonの文字列置換機能で、予定な文字を削除して、何とか読めました。

工夫した点

・自立制御の一見複雑そうな動きを、左右全く独立で動かすことにより、簡単に動かすことができました、これは、以前 子供の科学という雑誌で、左右の太陽電池と左右のモータをクロスでつないで、光の方向に走る模型のカーというアイデアを見ました、今回そのアイデアを応用しています。

自分でも、こんなに容易に、左右・前後に人？(白い大きなもの?)についていくロボットカーが動いて、感激しています。

3.1 システム: NODE-REDプログラム

NODE-REDは、最初IBMが開発し、現在はオープンソースになっている、WEB用のJAVASCRIPT言語による、開発環境です。

ただ、今回の例では、なんとJAVASCRIPTのプログラムを1行も書かなくて、機能を実現できてしまいました。

ラスパイに標準でついてきます、少し頑張れば、Windows10でも動かせます、自分が、どこのハードの開発環境(NODE-RED)で、動かしているのが分からなくなり、それぞれのハードの得意、性能を考え、ハード分散の開発が容易にできる優れものです、是非動かしてもらえると楽しいですよ？

ラズ☆ロボの設計図:システム(システムソフト相当)

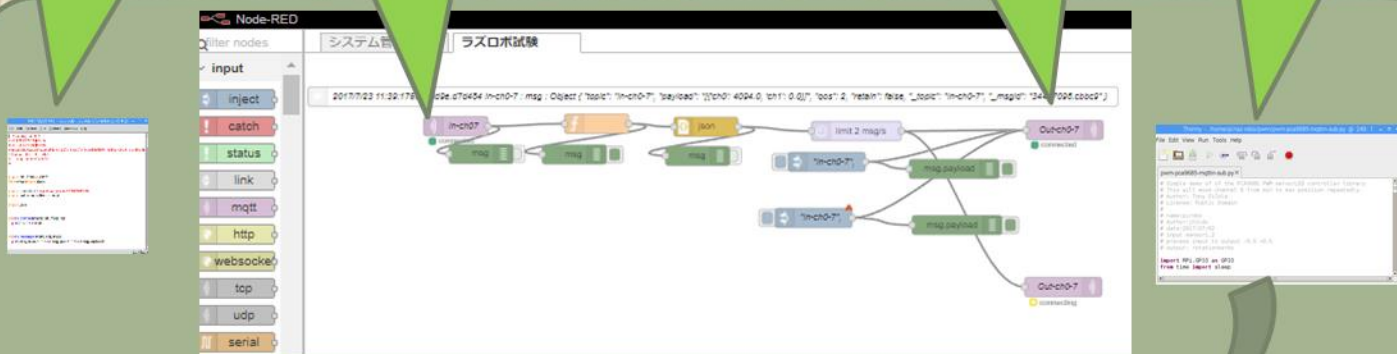
NODE-REDで、情報の行き先を指定しています。

①センサ
②Pythonソフト
③Node-Red
の順に情報が流れる

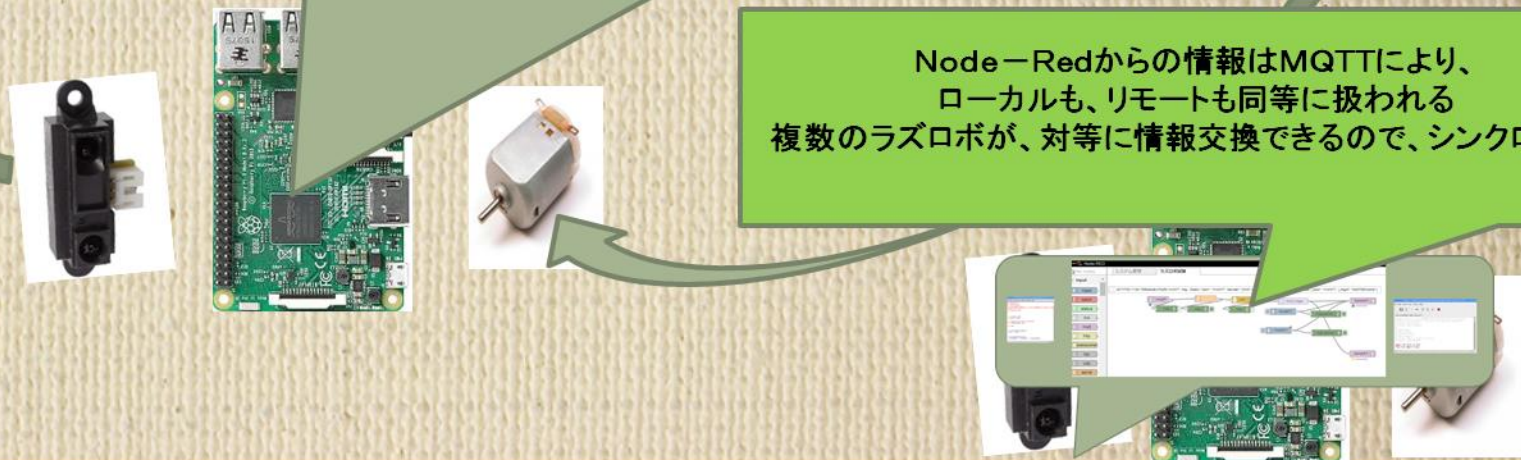
センサの情報を取得
センサ入力用Pythonソフト
からのMQTT入力
JSON形式

モータを制御
DCMotor出力用Pythonソ
フトへのMQTT出力
JSON形式

①Node-Red
②Pythonソフト
③センサ
の順に情報が流れる



Node-Redからの情報はMQTTにより、
ローカルも、リモートも同等に扱われる
複数のラズロボが、対等に情報交換できるので、シンクロできる

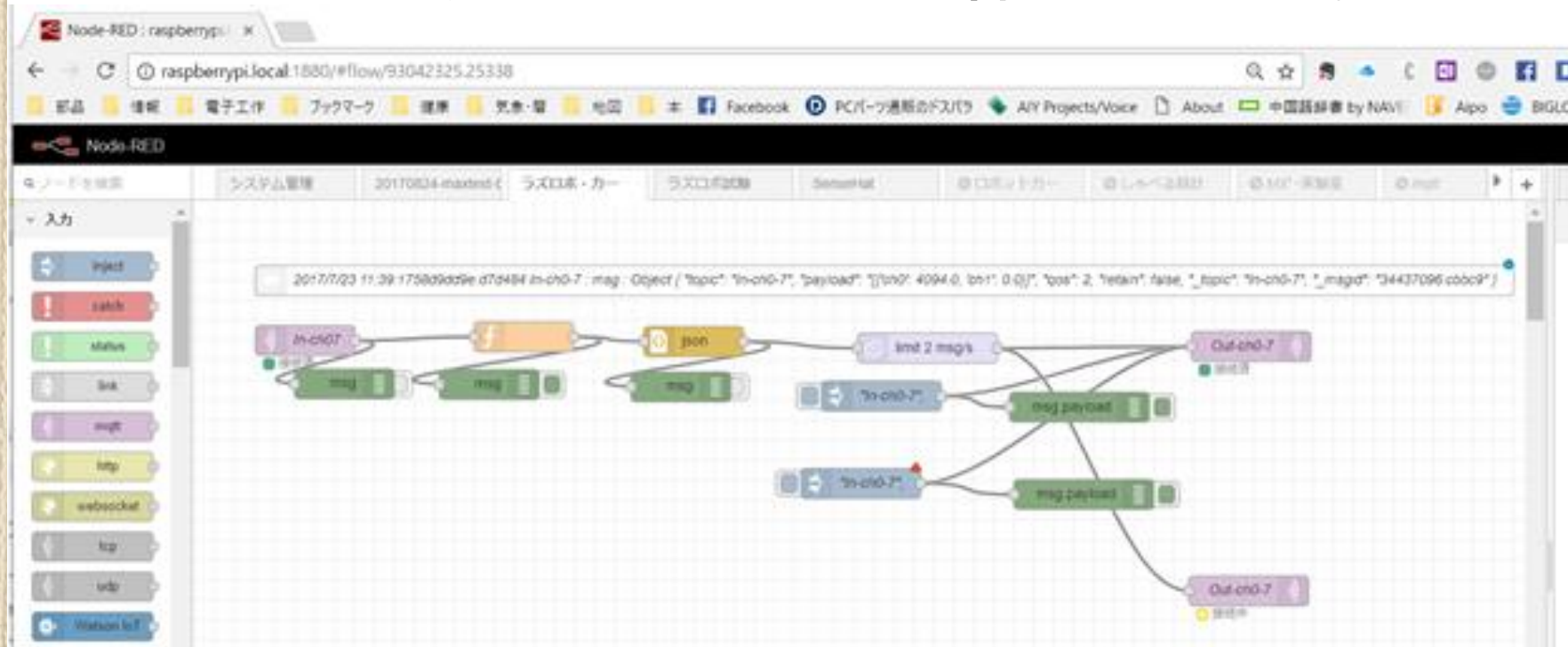


3.3 システム: NODE-REDプログラム

ラズパイのnode-redを起動します。

ブラウザから <http://raspberrypi.local:1880> (raspberrypi.localは、ラズパイの名前設定で設定した名前になります、デフォルトだと、raspberrypi.localです。)node-redに接続します。

node-redのソースを、node-redにコピーすると下記のようになります。



設定するのは、5種のノードです、ほかのノードはデバッグ用ですので、動かすだけなら必要ありません。

3.4 システム: NODE-REDプログラム

設定するのは、下記5種のノードです。

mqtt in ノードを編集

削除 中止 完了

プロパティ

サーバ localhost:1883

トピック in-ch07

QoS 2

名前 in-ch07

delay ノードを編集

削除 中止 完了

プロパティ

動作 メッセージの流量制限

全てのメッセージ

流量 2 メッセージ / 1 秒

☒ 仲介メッセージを削除

名前 名前

function ノードを編集

削除 中止 完了

プロパティ

名前 名前

コード

```
return msg;
```

出力数 1

コードの記述方法はノードの「情報」を参照してください。

json ノードを編集

削除 中止 完了

プロパティ

名前 名前

オブジェクトからJSONへ変換

☐ JSON文字列フォーマット

mqtt out ノードを編集

削除 中止 完了

プロパティ

サーバ localhost:1883

トピック Out-ch0-7

QoS 保持

名前 Out-ch0-7

注釈: トピックやQoSをメッセージのプロパティを用いて設定する場合は、無記入にしてください。

node-redの設定が出来たら、pythonのプログラムを起動します

sudo python3 MCP3208-MQTTout-pub-DCmotor.py &

sudo python3 pwm-pca9685-mqttin-sub-dcmotor.py

node-redのデバッグwindowに、センサの値が表示されれば成功です。

3.5 システム: NODE-REDプログラム

苦勞した点

- ・赤外線距離センサの動作環境に苦勞しました。

当初、自分が前を歩くと、ついてくるロボットカーを考えていたのですが、うまく扱えばついてくるようなときもありますが、安定して動かすため(赤外線の反射の影響と推定)には、実験の結果下記の環境でした。

下(床、机上)を黒い布で覆う

検出する反射物を、A3白程度の大きさにする

- ・当初、データの見える化で、node-redのdashboardを使って、webブラウザでデータをみえるようにしていたのですが、毎秒1回以上更新すると、負荷が重くなるせいか、うまく動かなくなり、いまはwebブラウザで見ないように設定しています。

又、pythonプログラムでは、0.1秒ごとにデータを送り、delayノードで 処理量を毎秒2メッセージに調整しています。

ここまでの、単体動作の説明です。

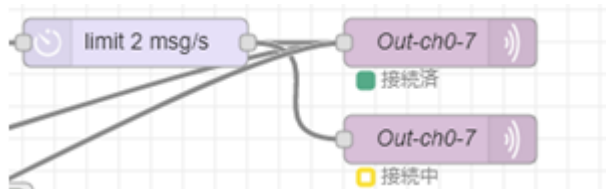
デモ映像では、2台のラスパイが、シンクロ動作ができています。

どうやってシンクロ動作ができるのか説明します。

3.6 システム: NODE-REDプログラム

ここまでが、単体動作の説明です。
デモ映像では、2台のラズパイが、シンクロ動作ができています。
どうやってシンクロ動作ができるのか説明します。

追加するのは、たった一つのノードだけです。



mqtt out ノードを編集

▼ プロパティ

📍 サーバ

📄 トピック

🔒 QoS

🏷️ 名前

注釈: トピックやQoSをメッセージのプロパティを用いて設定する場合は、無記入にしてください。

中身もサーバのアドレスが違っただけです。
新たにつなぐ相手側のラズパイは、自分に接続されているセンサと同じように、別のラズパイの信号を読み込んで、動作するので、2台のラズパイが、あたかもシンクロしているように動くように見えるという考えです。
実際やってみると、拍子抜けするぐらいに簡単に動いてしまい、感激しました。

3.7 システム: NODE-REDプログラム

pythonプログラムと、NODE-REDプログラムをダウンロードできるようにしますので動かしてもらえると嬉しいです。

当面は、下記からダウンロード願います。

<https://sites.google.com/site/kaihoudenou/home/20170805-mft2017-20170803.zip?attredirects=0&d=1>

BTOさんから、更新情報含めて公開予定です。

これから作りたいもの

- ・DCモータとサーボモータを同時に制御して、リモコンカメラ付き ラズ☆ロボ・カー
 - ・センサを増やして、机から落ちない ラズ☆ロボ・カー
 - ・道路をトレースして走る ラズ☆ロボ・カー
 - ・自分の代わりに、色々なところに行ってくれる ラズ☆ロボ・カー
 - ・自分の代わりに、誰かのところに行って、会話できる ラズ☆ロボ・カー
- 夢は広がります、みんなで楽しみましょう！！

4. まとめ

- ・早く

ホームサーバ: NODE-RED、MQTT 素早く開発、ライブラリが豊富

- ・安く

ホームサーバ部品: ラズパイ ZERO、2, 3

- ・楽しく

node-red+MQTTで動かしているので、電子ブロック感覚で、
機能を変更して遊べます。

電子工作の世界では、ハードも、ソフトも 多種多様になり、どんどん良い製品が出てきています、しかも、最近は、オープン・ソース・ソフトウェア、オープン・ハードウェアと呼ばれ、無料で使わせてもらえる恵まれた環境になっていると思います、自分はハードが専門なのに、こんなソフトまで使わせてもらって、楽しんでいきます、皆さん、特にソフトの専門家の方に、色々作ってもらい、情報を公開してもらえると嬉しいです。