

# checkpoint2 report

---

## 硬體

---



## 使用步驟

---

### SSH遠端登入Rpi

```
$ ssh ubuntu@xxx.xxx.xx.xx
```

### shell A (start ROS master)

```
$ roscore
```

### shell B (communicate with Arduino)

```
$ rosruncpp rosserial_python serial_node.py /dev/ttyACM0
```

### shell C (user input)

```
$ cd ~/checkpoint2/catkin_ws
```

```
$ source devel/setup.bash
```

```
$ rosruncpp pkg checkpoint2
```

依序輸入三個數字 (方向,右輪轉速,左輪轉速)

前進(1,200,201)

右轉(1,150,200)

左轉(1,200,150)

後退(0,200,200)

停止(2,0,0)

## Rpi code

---

```

#include "ros/ros.h"
#include "std_msgs/Int32.h"
#include <iostream>

using namespace std;

int main(int argc ,char **argv){
    ros::init(argc,argv,"checkpoint_2_pub");
    ros::NodeHandle nh;
    ros::Publisher number_publisher=nh.advertise<std_msgs::Int32>("array",1);
    ros::Rate loop_rate(10);

    ros::Duration(2).sleep();
    std_msgs::Int32 msg;

    int n=0;
    while(ros::ok() ){
        int i;
        if(n==1){cout<<"input a num:(r)";}
        if(n==2){cout<<"input a num:(l)";}
        if(n==0){cout<<"input a num:(MODE)";}
        n++;
        cin>>i;
        msg.data=i;
        if(n==3){n=0;}
        if(i== 256){break;}

        number_publisher.publish(msg);
        cout<<"PUB";
        ros::spinOnce();
        loop_rate.sleep();
    }
}

```

## Arduino code

```

#include <PID_v1.h>
#include <ros.h>
#include <math.h>
#include <std_msgs/Int32.h>
// #include <std_msgs/Float32MultiArray.h>
const byte encoder0pinA = 2;//A pin -> the interrupt pin 0
const byte encoder0pinB = 12;//B pin -> the digital pin 3
int in1 =8; //The enabling of L298PDC motor driver board connection to the digital interface port 5
int in2 =9; //The enabling of L298PDC motor driver board connection to the digital interface port 4
int ena =5;
const byte encoder1pinA = 3;//A pin -> the interrupt pin 0
const byte encoder1pinB = 13;//B pin -> the digital pin 3
int in3 =10; //The enabling of L298PDC motor driver board connection to the digital interface port 5
int in4 =11; //The enabling of L298PDC motor driver board connection to the digital interface port 4
int enb =6;
byte encoder0PinALast;
byte encoder1PinALast;
double durationright,abs_durationright;//the number of the pulses
double durationleft,abs_durationleft;
boolean Directionright;//the rotation direction
boolean Directionleft;
boolean resultright;
boolean resultleft;
ros::NodeHandle nh;
//int count=1;

```

```

//int count=1;
double val_outputright;//Power supplied to the motor PWM value.
double val_outputleft;
double Setpointright=0;
double Setpointleft=0;
double Kp=0.6, Ki=5, Kd=0;

int MODE;
int count=0;

PID rightPID(&abs_durationright, &val_outputright, &Setpointright, Kp, Ki, Kd, DIRECT);
PID leftPID(&abs_durationleft, &val_outputleft, &Setpointleft, Kp, Ki, Kd, DIRECT);

void num(const std_msgs::Int32& msg){
    count++;
    if(count==1){
        MODE=msg.data;
    }
    if(count==2){
        Setpointleft=msg.data;
    }
    if(count==3){
        Setpointright=msg.data;
        count=0;
    }
}
ros::Subscriber<std_msgs::Int32> sub("array",&num);

void setup()
{
    Serial.begin(57600);//Initialize the serial port
    pinMode(in1, OUTPUT);    //L298P Control port settings DC motor driver board for the output mode
    pinMode(in2, OUTPUT);
    pinMode(ena, OUTPUT);
    pinMode(in3, OUTPUT);    //L298P Control port settings DC motor driver board for the output mode
    pinMode(in4, OUTPUT);
    pinMode(enb, OUTPUT);
    pinMode(encoder0pinA, INPUT);
    pinMode(encoder0pinB, INPUT);
    pinMode(encoder1pinA, INPUT);
    pinMode(encoder1pinB, INPUT);
    nh.initNode();
    nh.subscribe(sub);
    rightPID.SetMode(AUTOMATIC);//PID is set to automatic mode
    rightPID.SetSampleTime(100);//Set PID sampling frequency is 100ms
    leftPID.SetMode(AUTOMATIC);//PID is set to automatic mode
    leftPID.SetSampleTime(100);//S
    EncoderInit();//Initialize the module

    Serial.print("INIT DONE");
}

void loop()
{
    if(MODE==1){advance();}//Motor Forward
    if(MODE==0){back();}
    if(MODE==2){Stop();}
    abs_durationright=abs(durationright);
    resultright=rightPID.Compute();//PID conversion is complete and returns 1

```

```

    if(resultright)
    {
        durationright = 0; //Count clear, wait for the next count
    }

    abs_durationleft=abs(durationleft);
    resultleft=leftPID.Compute();//PID conversion is complete and returns 1
    if(resultleft)
    {
        durationleft = 0; //Count clear, wait for the next count
    }

    nh.spinOnce();
}

```

```

void EncoderInit()
{
    Directionright = true;//default -> Forward
    pinMode(encoder0pinB,INPUT);
    attachInterrupt(0, wheelSpeedright, CHANGE);
    Directionleft = true;//default -> Forward
    pinMode(encoder1pinB,INPUT);
    attachInterrupt(1, wheelSpeedleft, CHANGE);
}

void wheelSpeedright()
{
    int Lstate = digitalRead(encoder0pinA);
    if((encoder0PinALast == LOW) && Lstate==HIGH)
    {
        int val = digitalRead(encoder0pinB);
        if(val == LOW && Directionright)
        {
            Directionright = false; //Reverse
        }
        else if(val == HIGH && !Directionright)
        {
            Directionright = true; //Forward
        }
    }
    encoder0PinALast = Lstate;
    if(!Directionright) durationright++;
    else durationright--;
}

void wheelSpeedleft()
{
    int Rstate = digitalRead(encoder1pinA);
    if((encoder1PinALast == LOW) && Rstate==HIGH)
    {
        int valR = digitalRead(encoder1pinB);
        if(valR == LOW && Directionleft)
        {
            Directionleft = false; //Reverse
        }
        else if(valR == HIGH && !Directionleft)
        {
            Directionleft = true; //Forward
        }
    }
    encoder1PinALast = Rstate;
    if(!Directionleft) durationleft++;
    else durationleft--;
}

```

```
void advance()//Motor Forward
{
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
    analogWrite(ena,val_outputright);
    digitalWrite(in3,HIGH);
    digitalWrite(in4,LOW);
    analogWrite(enb,val_outputleft);
}

void back()//Motor reverse
{
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
    analogWrite(ena,val_outputright);
    digitalWrite(in3,LOW);
    digitalWrite(in4,HIGH);
    analogWrite(enb,val_outputleft);
}

void Stop()//Motor stops
{
    digitalWrite(ena, LOW);
    digitalWrite(enb, LOW);
}
```

## 問題討論

---

### ROS通訊

一開始我們使用陣列(std\_msgs::Float32MultiArray)傳送資料，但是Arduino與ROS通訊過程中，經常出現"Lost sync with device, restarting..."錯誤。後來將ROS Topic改用std\_msgs::Int32的訊息格式問題才消失。

### 馬達編碼器讀取

一開始因為Arduino中斷腳位設定錯誤，一直讀不到其中一邊馬達的轉速。