ABU Electronic team

Live fast, die young

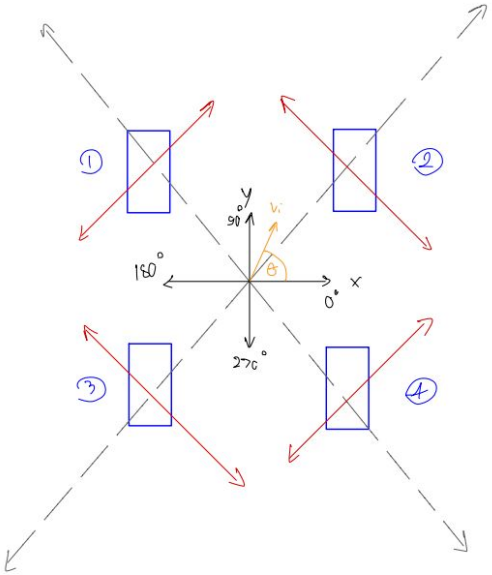
15-19 Jan

Mecanum analytic math model

Mecanum wheel vector analysis

⟷ Force-axis

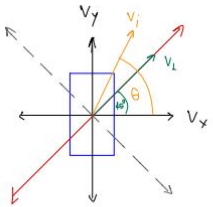
<--> Free-axis



V_i, θ = input

$V_1 = V_4$, $V_2 = V_3$ คำตอบของ 1 คือ ① และ ②

พิจารณา ล้อ ①

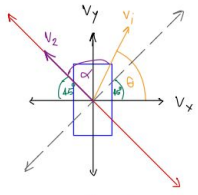


⟷ Force-axis

<--> Free-axis

$$V_1 = V_i\cos(\theta - 45^\circ)$$
$$= V_i\cos(\theta)\cos(45^\circ) + V_i\sin(\theta)\sin(45^\circ)$$
$$= V_x\cos(45^\circ) + V_y\sin(45^\circ)$$
$$\therefore V_1 = \begin{bmatrix}\cos(45^\circ) & \sin(45^\circ)\end{bmatrix}\begin{bmatrix}V_x \\ V_y\end{bmatrix} \quad \#$$; matrix form

พิจารณา ล้อ ②



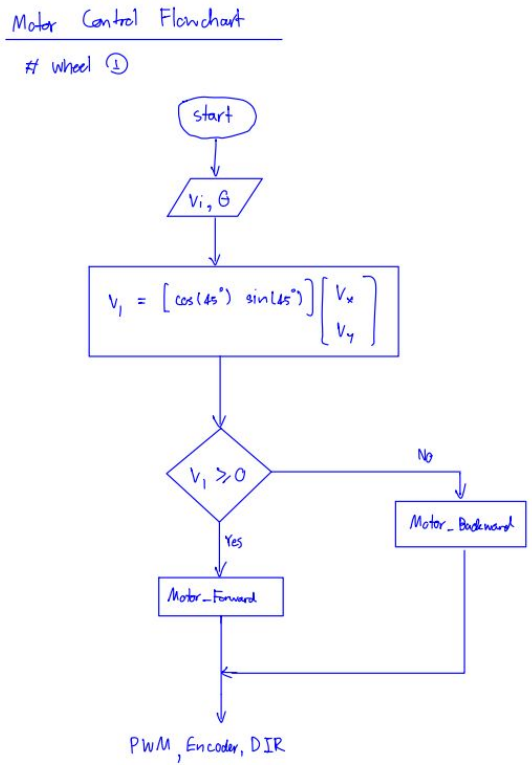
⟷ Force-axis

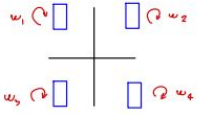
<--> Free-axis

note $\alpha = 180^\circ - 45^\circ - \theta$
$= 135^\circ - \theta$

$$V_2 = V_i\cos(\alpha)$$
$$= V_i\cos(135^\circ - \theta)$$
$$= V_i\cos(135^\circ)\cos\theta + V_i\sin(135^\circ)\sin\theta$$
$$= V_x\cos(135^\circ) + V_y\sin(135^\circ)$$
$$\therefore V_2 = \begin{bmatrix}\cos(135^\circ) & \sin(135^\circ)\end{bmatrix}\begin{bmatrix}V_x \\ V_y\end{bmatrix} \quad \#$$; matrix form

Mecanum analytic math model





$$\begin{bmatrix} V_i \\ \theta \end{bmatrix} \longrightarrow \begin{bmatrix} V_i\cos\theta \\ V_i\sin\theta \end{bmatrix} = \begin{bmatrix} V_x \\ V_y \end{bmatrix}$$

input control

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} \cos 45^\circ & \sin 45^\circ \\ \cos 135^\circ & \sin 135^\circ \\ \cos 135^\circ & \sin 135^\circ \\ \cos 45^\circ & \sin 45^\circ \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix}$$

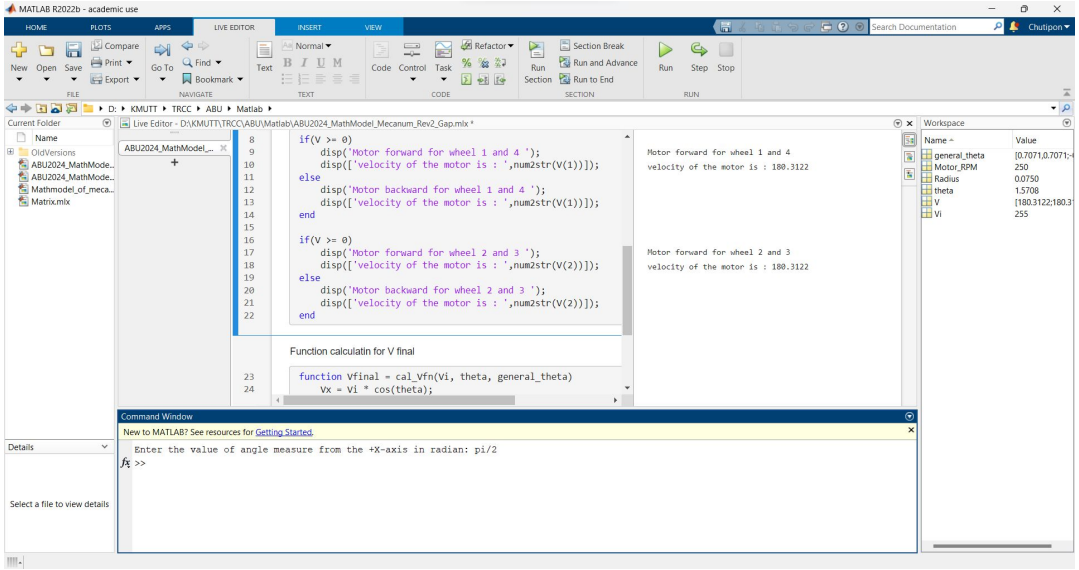
output system

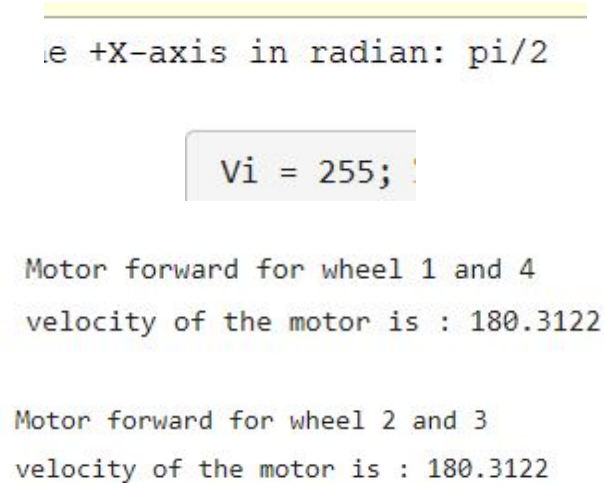
22-24 Jan

Mecanum analytic math model

Mecanum math model write in Matlab

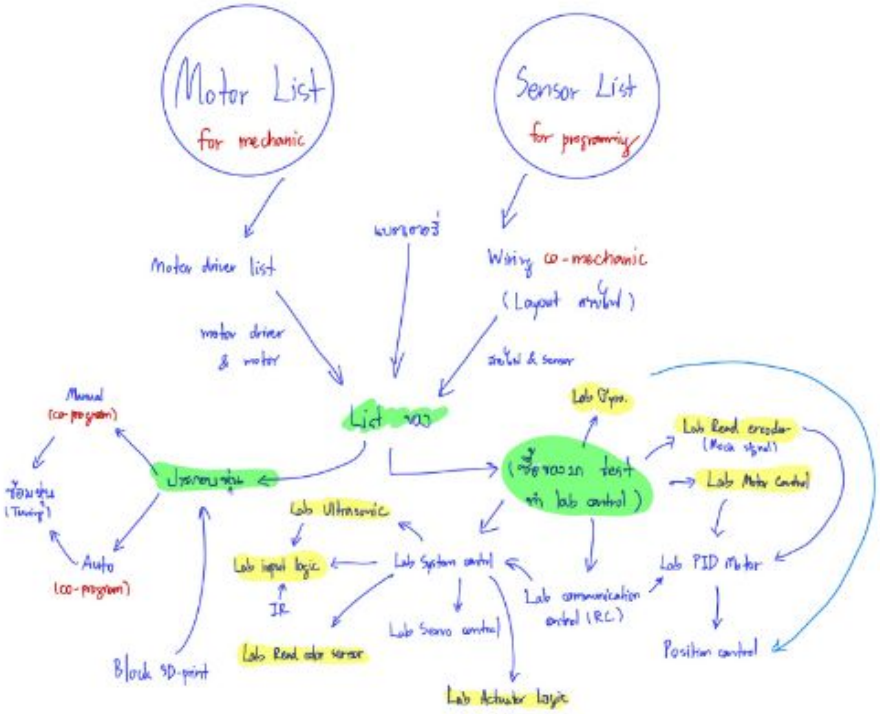
The maximum analog write in arduino is 255, So pretending the maximum speed 255 m/s





List


อรุ่มเจ๊าะ


Motor List
for mechanic
Sensor List
for programmig
Motor driver list
Wiring co-mechanic
(Layout)
motor driver
& motor
Manual
(co-program)
Auto
(co-program)
Lab Ultrasonic
Lab Input logic
IR
Lab System control
Lab Servo control
Lab Read color sensor
Lab communication
control (RC)
Lab PID Motor
Position control
Lab Actuator Layic
Lab Read encoder
(Mesa signal)
Lab Motor Control

25-26 Jan

Test IMU MPU6050 (Gyroscope)

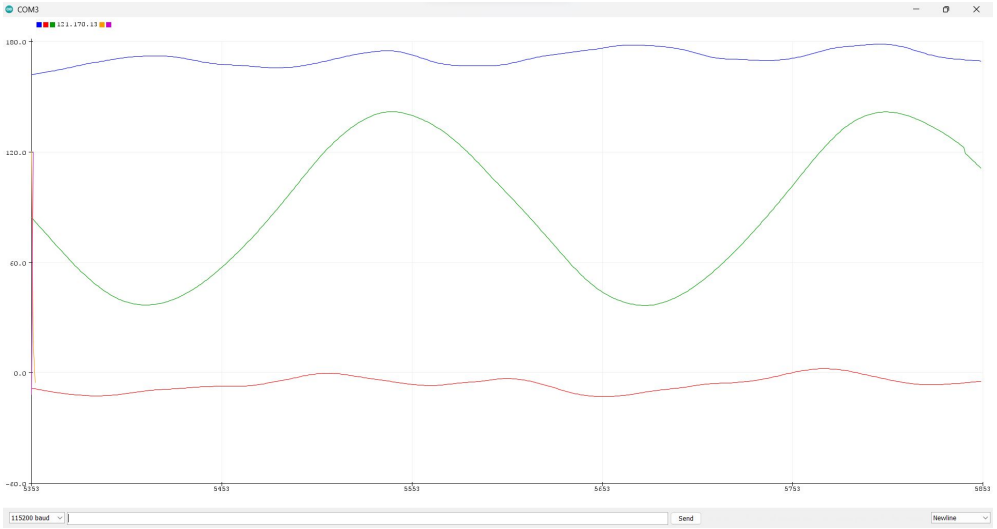
Use quaternion model to get the roll pitch yaw

MahonyAHRS_Arduino | Arduino 1.8.19

File Edit Sketch Tools Help

MahonyAHRS_Arduino

```
  // Normalise quaternion
  norm = sqrt(q0 * q0 + q1 * q1 + q2 * q2 + q3 * q3);
  q0 /= norm;
  q1 /= norm;
  q2 /= norm;
  q3 /= norm;
}

void mpu6050_getRollPitchYaw() {
//  yaw = atan2(2*q1*q2 - 2*q0*q3, 2*q0*q0 + 2*q1*q1 - 1) * 57.29577951;
//  pitch = -asin(2*q1*q3 + 2*q0*q2) * 57.29577951;
//  roll = atan2(2*q2*q3 - 2*q0*q1, 2*q0*q0 + 2*q3*q3 - 1) * 57.29577951;
//  roll = atan2(2*q0*q1 + 2*q2*q3, 1 - 2*q1*q1 - 2*q2*q2) * 57.29577951;
//  pitch = asin(2*q0*q2 - 2*q3*q1) * 57.29577951;
//  yaw = atan2(2*q0*q3 + 2*q1*q2, 1 - 2*q2*q2 - 2*q3*q3) * 57.29577951;
  yaw   = -atan2(2.0f * (q1 * q2 + q0 * q3), q0 * q0 + q1 * q1 - q2 * q2 - q3 * q3) * 57.29577951;
  pitch = asin(2.0f * (q1 * q3 - q0 * q2)) * 57.29577951;
  roll  = atan2(2.0f * (q0 * q1 + q2 * q3), q0 * q0 - q1 * q1 - q2 * q2 + q3 * q3) * 57.29577951;
}

void writeByte(uint8_t address, uint8_t subAddress, uint8_t data)
{
  Wire.beginTransmission(address);  // Initialize the Tx buffer
```



AcX = -57 | AcY = -799 | AcZ = 10546 | GyX = 111 | GyY = 83 | GyZ = -5

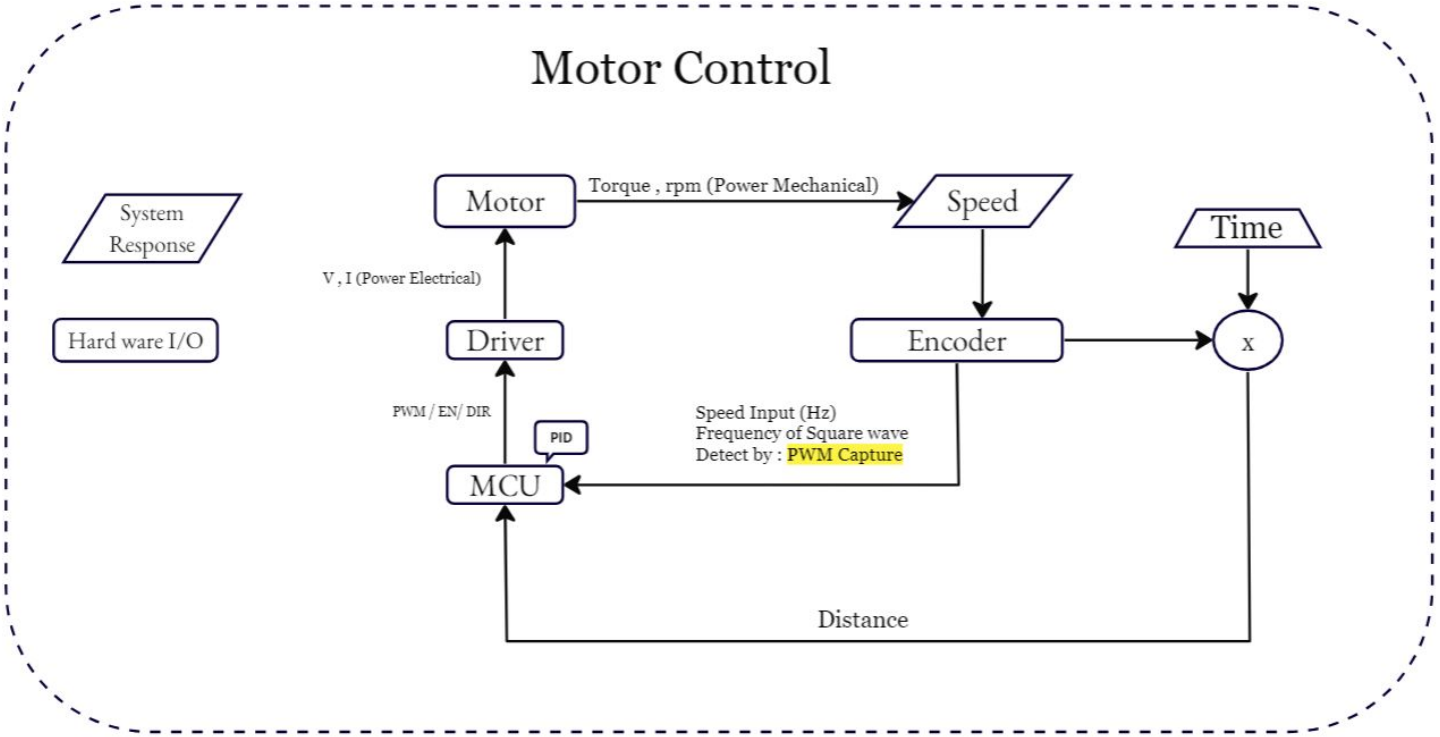
Hardware Concept

01/28/2024

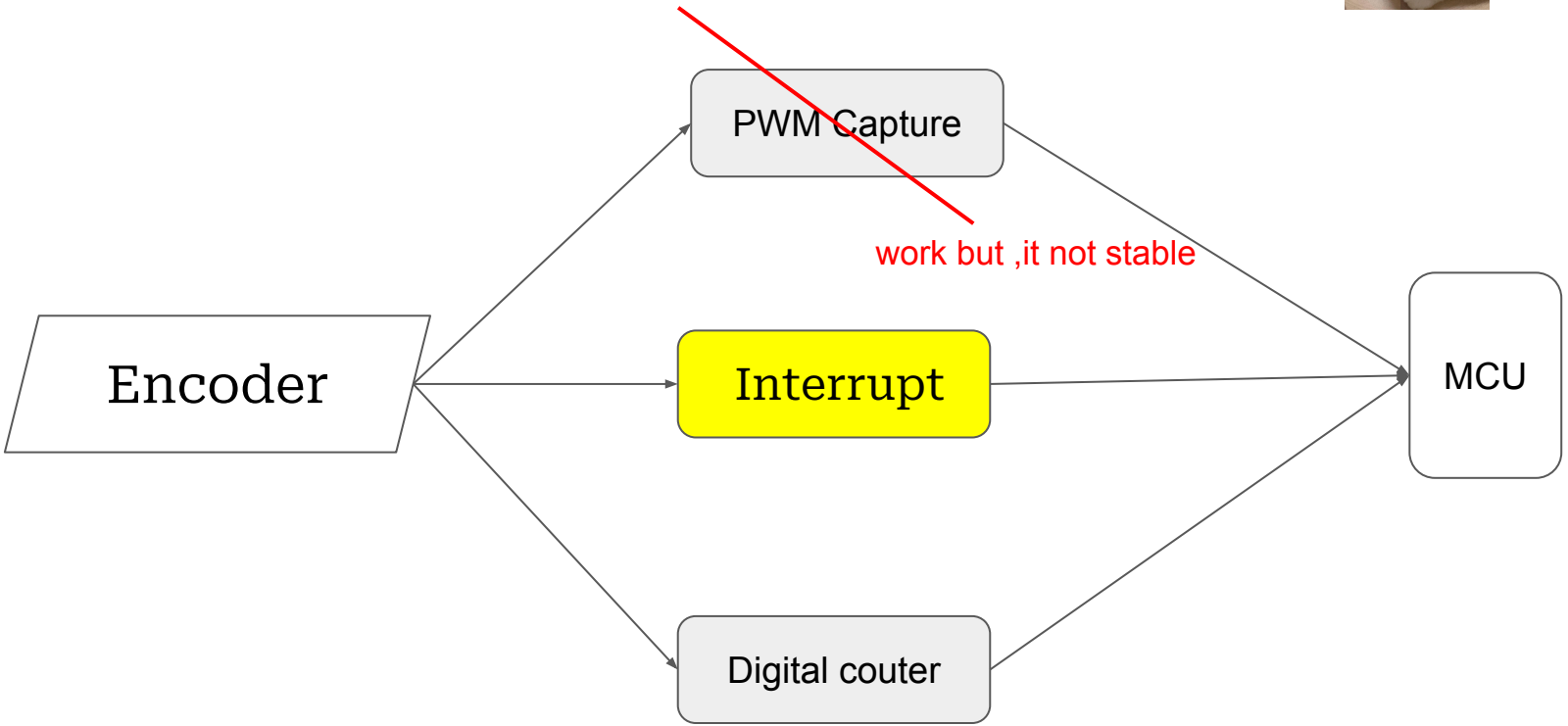
Outline

- Motion Control (Embeded sys.)
  - Speed Control
  - Position Control
- System block diagram
- Embedded board

Motion Control

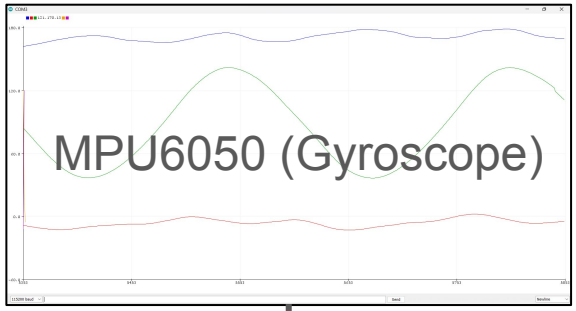


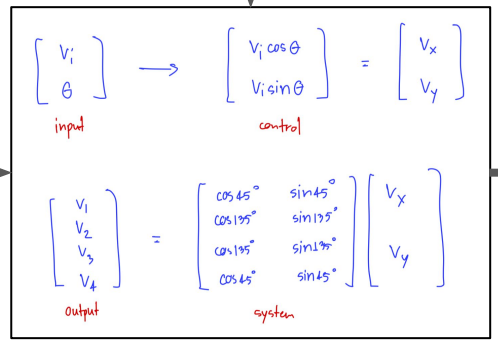
Motion Control



Motion Control

Motion Control





ABU Manual Robot

System block diagram(Manual Robot)

IBUS

RC

RC

Communication Board

Motion Control Board

Protocal communication : I2C

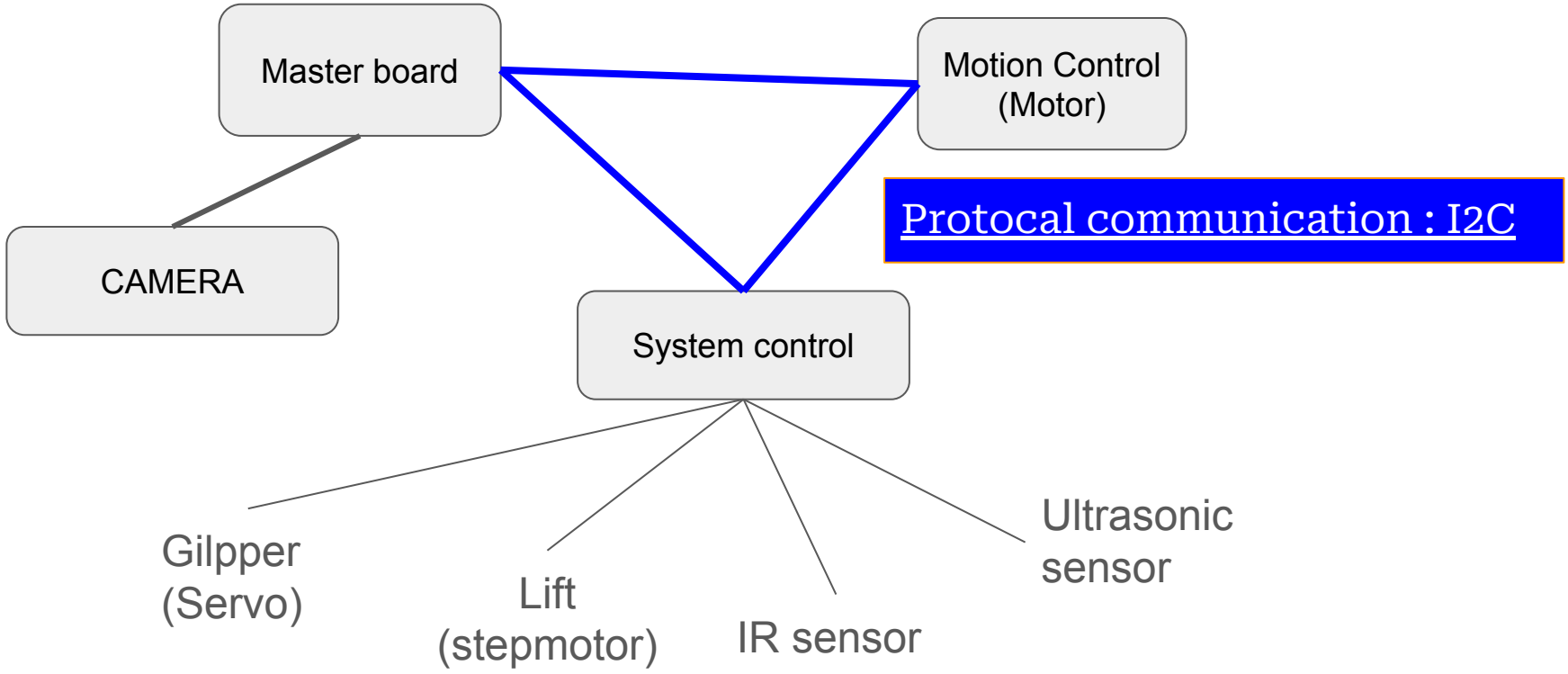
System control Board

Gripper

Ball Lifting

Laser & shooting

System block diagram(Auto Robot)



Embedded board

Arduino IDE
/ Python

Arduino IDE

Python

29 Jan

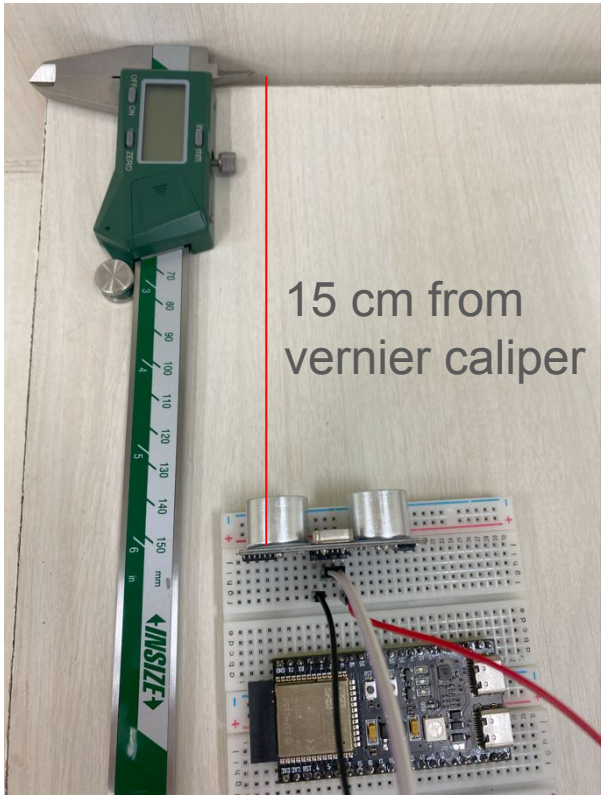
Outline

- Ultrasonic distance detection without delay
- Color reader
- Simple motor control

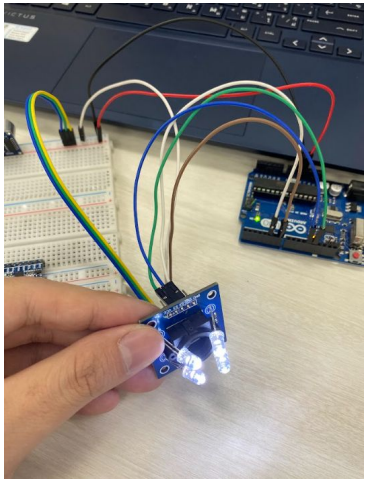
Lab Ultrasonic without delay

```
11:33:11.757 -> 15cm
11:33:12.452 -> 15cm
11:33:13.149 -> 15cm
11:33:13.848 -> 15cm
11:33:14.541 -> 15cm
11:33:15.237 -> 15cm
11:33:15.933 -> 15cm
11:33:16.624 -> 15cm
11:33:17.315 -> 15cm
11:33:18.009 -> 15cm
11:33:18.706 -> 15cm
11:33:19.400 -> 15cm
11:33:20.095 -> 15cm
11:33:20.787 -> 15cm
11:33:21.484 -> 15cm
```


```
unsigned long previousMillis = 0; //microsecond at which the pin was last writen

void setup()
{
  Serial.begin(9600); //begin serial communication
  pinMode(trigPin,OUTPUT); //set pinmodes
  pinMode(echoPin,INPUT);

}

void loop()
{
  unsigned long currentMillis = millis(); //time in milliseconds from which the code was started
  if (currentMillis-previousMillis >= interval) { //check "blink without delay" code
    previousMillis = currentMillis;
    if (trigState == LOW){
      (trigState = HIGH);
    }
    else {
      (trigState = LOW);
    }
  }
  // printing if statement
  if (currentMillis-previousMillis >= interval2) { //check "blink without delay" code
    previousMillis = currentMillis;
    if (printState == LOW){
      (printState = HIGH);
    }
    else {
      (printState = LOW);
    }
  }
  digitalWrite(trigPin,trigState);
  int duration, distance; //variables
```



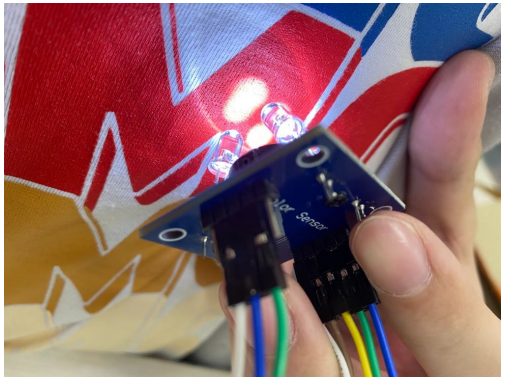
Lab Color Sensor




```
{
  static  uint8_t selChannel;
  static  uint8_t readCount;
  static  sensorData  sd;

  switch(state)
  {
  case 0: // Prompt for the user to start
    Serial.print(F("\n\nReading value for "));
    switch(valType)
    {
    case BLACK_CAL: Serial.print(F("BLACK calibration")); break;
    case WHITE_CAL: Serial.print(F("WHITE calibration")); break;
    case READ_VAL:  Serial.print(F("DATA"));        break;
    default:    Serial.print(F("??"));          break;
    }

    Serial.print(F("\nPress any key to start ..."));
    state++;
    break;
```

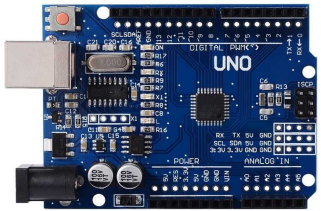
```
22:23:41.585 -> Reading value for DATA
22:23:41.585 -> Press any key to start ...
22:23:56.267 -> RGB is [203,31,37]
```

เข้าเว็ป RGB Color Codes Chart เพื่อแปลงค่าสี



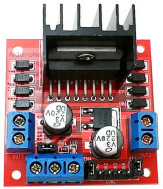
RGB color picker

Lab motor control



Arduino Uno

command



(PWM, DIR, EN)

L298n motor driver

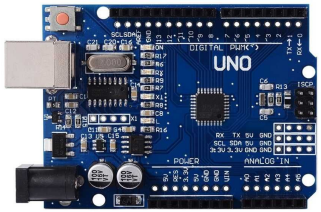
x2

speed & direction control

12V DC motor

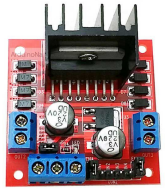
x4

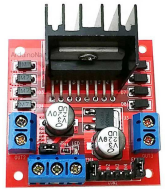
Lab motor control



Arduino Uno

command





L298n motor driver

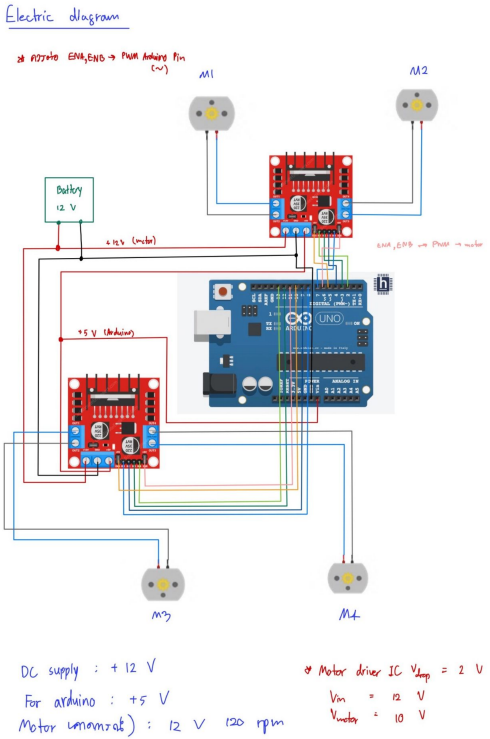
speed & direction control

Front wheel

Rear wheel

12V DC motor

Motor control wiring diagram

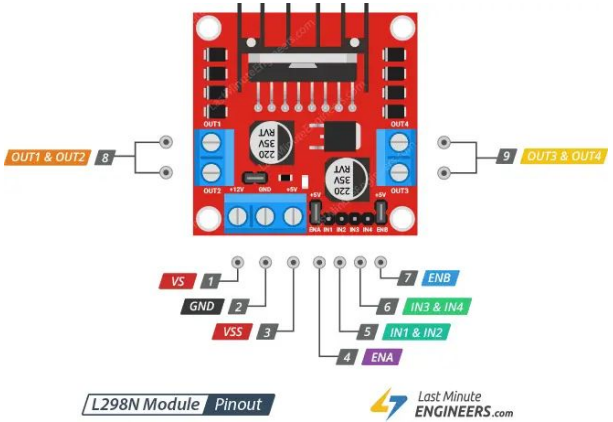



```
//Pin Definition
in1  motor_1_pin_1 = 2      motor_3_pin_1 = 8
in2  motor_1_pin_2 = 3      motor_3_pin_2 = 9
ENA  motor_1_speed_pin = 5  motor_3_speed_pin = 10

in3  motor_2_pin_1 = 7      motor_4_pin_1 = 13
in4  motor_2_pin_2 = 4      motor_4_pin_2 = 12
ENB  motor_2_speed_pin = 6  motor_4_speed_pin = 11
```

To control 1 motor (Direction + speed)

2 input pins + 2 output pins + 1 EN (A or B) pin

Speed and direction Control



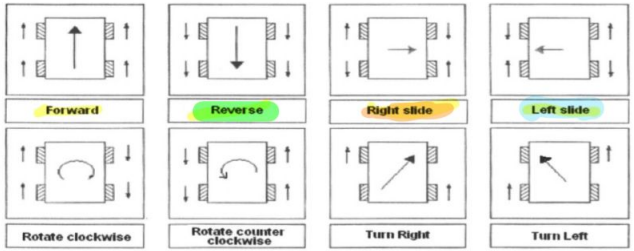


Fig 8: Required actuation for general movement. Source from [4]

Direction control Pins

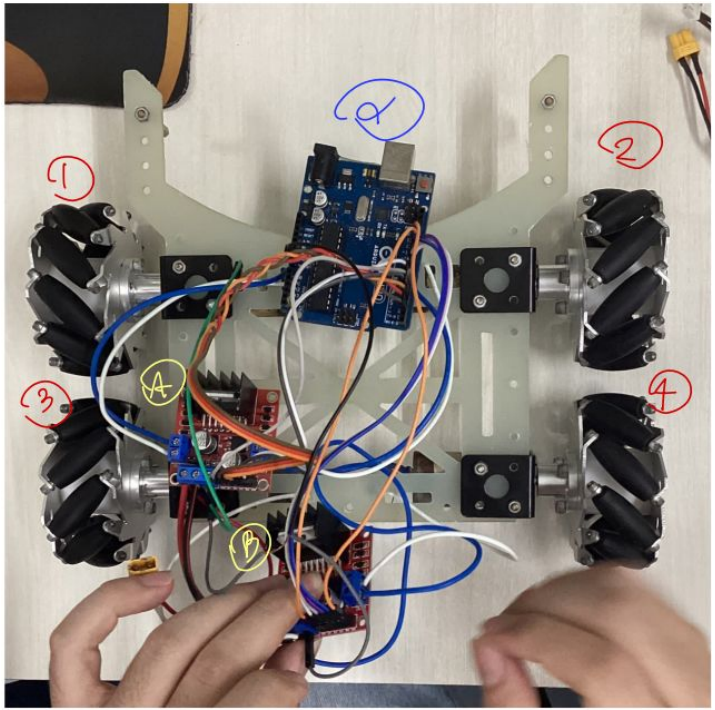
Input1	Input2	Spinning Direction
Low(0)	Low(0)	Motor OFF
High(1)	Low(0)	Forward
Low(0)	High(1)	Backward
High(1)	High(1)	Motor OFF

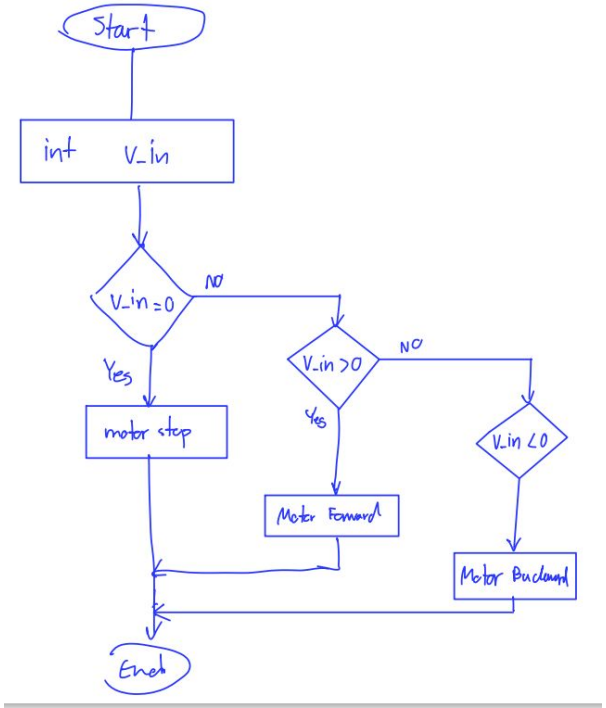
(In3, In4) = same

Speed Control Pins

ENA, ENB = PWM
range integer 0-255

Setup





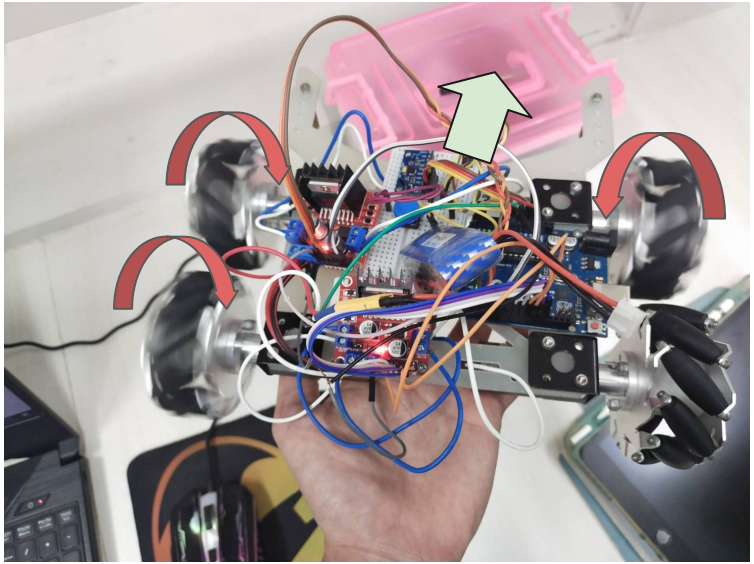
Simple condition flowchart

Coding condition


```
39 void loop() {
40     motorControl(100);
41 }
42
43 void motorControl(int v_in)
44 {
45   if (v_in == 0)
46   {
47     //Stop
48     // Front drive motor
49     digitalWrite(motor_1_pin_1, LOW);
50     digitalWrite(motor_1_pin_2, LOW);
51     analogWrite(motor_1_speed_pin, 0);
52
53     digitalWrite(motor_2_pin_1, LOW);
54     digitalWrite(motor_2_pin_2, LOW);
55     analogWrite(motor_2_speed_pin, 0);
56
57     // Rear drive motor
58     digitalWrite(motor_3_pin_1, LOW);
59     digitalWrite(motor_3_pin_2, LOW);
60     analogWrite(motor_3_speed_pin, 0);
61
62     digitalWrite(motor_4_pin_1, LOW);
63     digitalWrite(motor_4_pin_2, LOW);
64     analogWrite(motor_4_speed_pin, 0);
65   }
66   else if (v_in >= 0)
67   {
68     //Forward
69     // Front drive motor
```

v_in = 0 → Motor stop
v_in >= 0 → Motor Forward
v_in <= 0 → Motor Backward

Result



Test v_in = 100
(Move forward with a speed of 100**)**

Result :
Motor 1, 2, 3 move forward **OK**
Motor 4 is not working (disconnected)

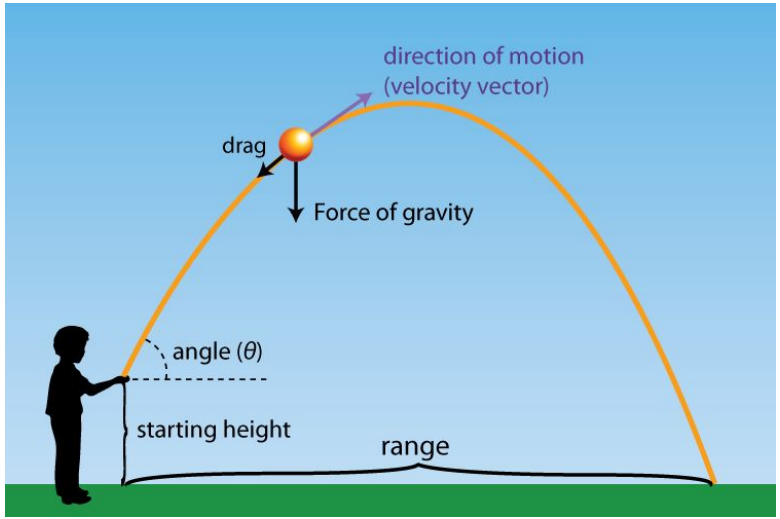
**wire to motor-4 pole is poorly connect
soldering does’nt fix
need to change to a new motor

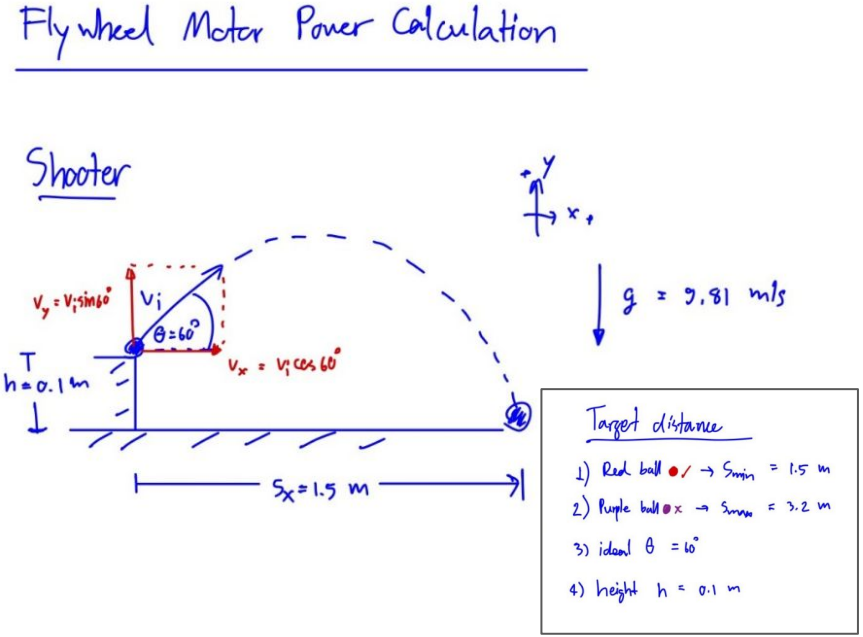
30 Jan

Outline

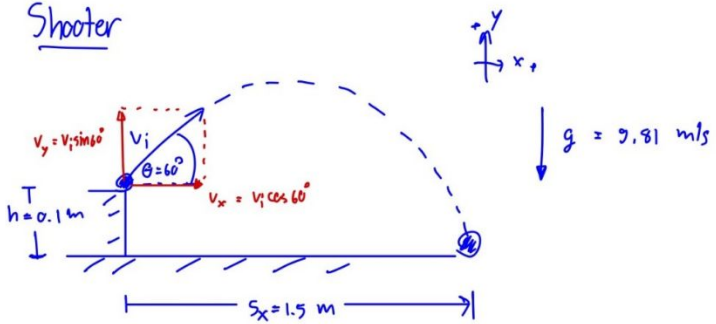
- Motor Power of Flywhee
  - Ball shooting
  - Math model
  - Matlab calculation

Projectile


direction of motion
(velocity vector)
drag
Force of gravity
angle (θ)
starting height
range


Flywheel Motor Power Calculation
Shooter
y
x
g = 9.81 m/s
$v_y = v_i \sin 60^\circ$
v_i
$\theta = 60^\circ$
$v_x = v_i \cos 60^\circ$
T
h = 0.1 m
$S_x = 1.5$ m
Target distance
1) Red ball ● / → S_{min} = 1.5 m
2) Purple ball ● x → S_{max} = 3.2 m
3) ideal θ = 60°
4) height h = 0.1 m

Math model



Target distance

1) Red ball ● / → S_{min} = 1.5 m
2) Purple ball ● x → S_{max} = 3.2 m
3) ideal θ = 60°
4) height h = 0.1 m

$$S_x = v_i \cos\theta \; t$$

$$-h = v_i \sin\theta \cdot t - \frac{1}{2} g t^2$$

$$v_i = \frac{S_x}{\cos\theta} \sqrt{\frac{g}{2(S_x \tan\theta + h)}}$$

S_{min} = 1.5 m → $v_{i,min}$ = 4.04 m/s

S_{max} = 3.2 m → $v_{i,max}$ = 5.97 m/s

Use v_ball = 6 m/s for
motor power calculation

Shooting angle in matlab

Shooting Flywheel

```
Sx = input('Please give me the horizontal distance in meters: ')
Shooting_angle_degrees = input('Please give me the value of angle
H = input('Please give me the height from ground in meters w: ')
```

```
g = 9.8100
```

Convert theta to rad

```
Shooting_angle_radians = deg2rad(Shooting_angle_degrees)
```

```
Shooting_angle_radians = 1.0472
```

Formulas for calculation

```
sq_term = sqrt(g/(2*((Sx*tan(Shooting_angle_radians))+H)))

V_initial = (Sx*sq_term)/cos(Shooting_angle_radians)
```

```
sq_term = 1.3483

V_initial = 4.0450
```

Display

```
disp(['The initial speed is : ',num2str(V_initial),' m/s'])
```

```
The initial speed is : 4.045 m/s
```

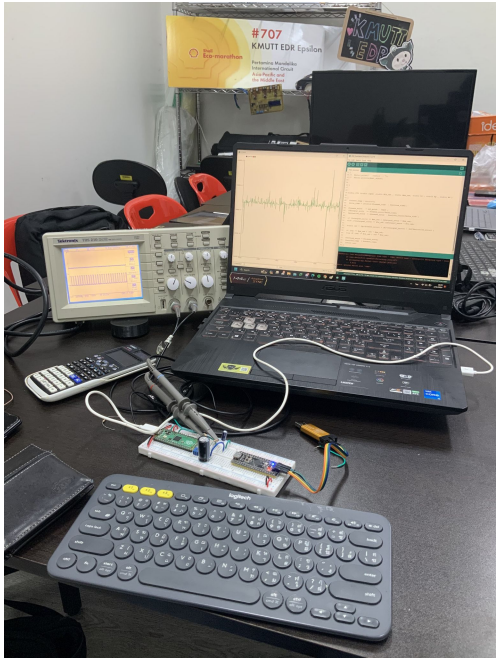
Include the fix value initial condition

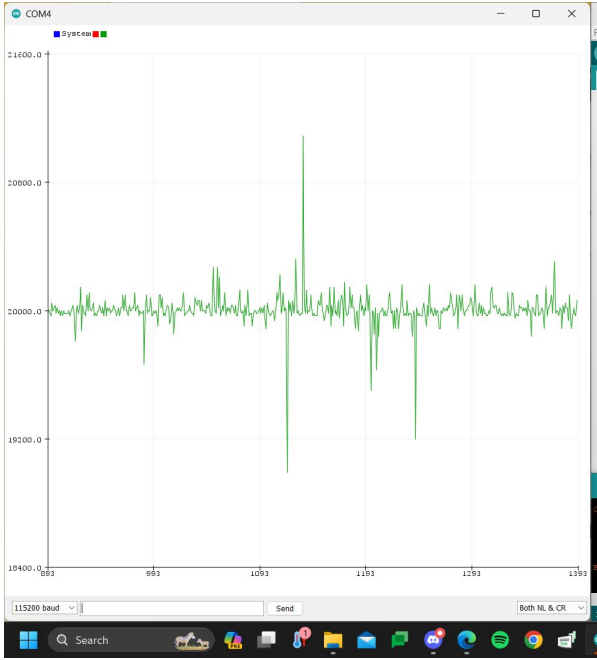
Shooting Flywheel

```
Sx = 1.5 %input('Please give me the horizontal distance in meters: ')
Shooting_angle_degrees = 60 %input('Please give me the value of angle
H = 0.1 %input('Please give me the height from ground in meters w: ')
```

Shooting angle in matlab

Linear control (PID)



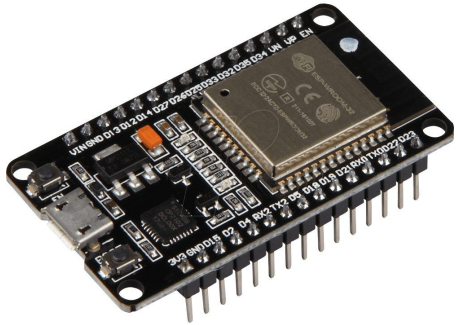


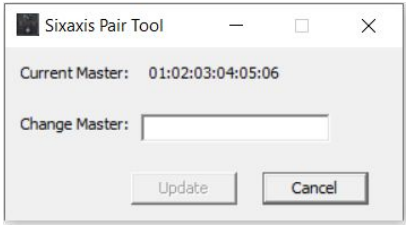
7 Feb

Outline

- PS3 controller Bluetooth-connect with ESP32

PS3 Bluetooth-connect with ESP32 (WROOM-32)





Get/Change MAC Address from PS3 controller

PS3 Bluetooth-connect with ESP32 (WROOM-32)

Ps3Demo | Arduino 1.8.19

File Edit Sketch Tools Help

Ps3Demo §

```
#include <Ps3Controller.h>

int player = 0;
int battery = 0;

void notify()
{
    //--- Digital cross/square/triangle/circle button events ---
    if( Ps3.event.button_down.cross )
        Serial.println("Started pressing the cross button");
    if( Ps3.event.button_up.cross )
        Serial.println("Released the cross button");

    if( Ps3.event.button_down.square )
        Serial.println("Started pressing the square button");
    if( Ps3.event.button_up.square )
        Serial.println("Released the square button");

    if( Ps3.event.button_down.triangle )
        Serial.println("Started pressing the triangle button");
    if( Ps3.event.button_up.triangle )
        Serial.println("Released the triangle button");

    if( Ps3.event.button_down.circle )
        Serial.println("Started pressing the circle button");
    if( Ps3.event.button_up.circle )
        Serial.println("Released the circle button");

    //---------------- Digital D-pad button events --------------
```

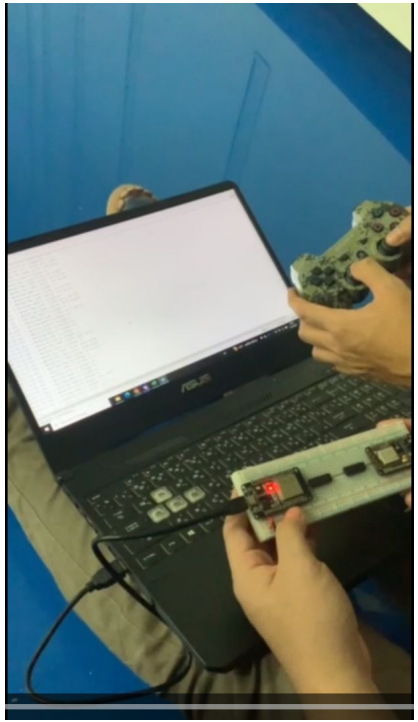
Done uploading.

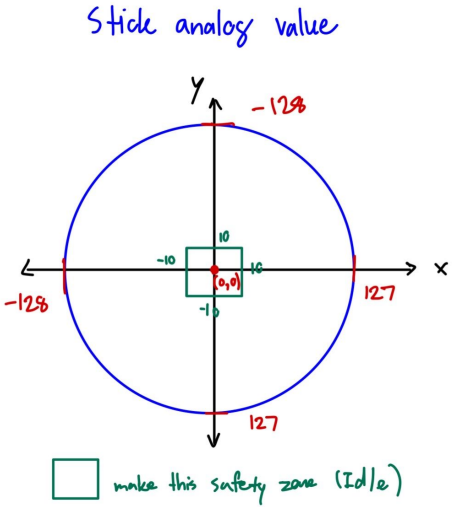
```
Leaving...
Hard resetting via RTS pin...
```

```
197 void setup()
198 {
199     Serial.begin(115200);
200
201     Ps3.attach(notify);
202     Ps3.attachOnConnect(onConnect);
203     Ps3.begin("01:02:03:04:05:06");
204
205     Serial.println("Ready.");
206 }
207
```

Change MAC Address of ESP32
to the same as PS3

PS3 Bluetooth-connect with ESP32 (WROOM-32)





Stick → Analog value : -128 to 127

Buttons → Analog value : 0 to 255 → Make it Digital to be useful

12 Feb

Outline

- Get digital value from PS3 buttons (Toggle)
- Get analog X-Y values from PS3 left joystick
- Get used to “ledc…” (ex. ledcWrite) to analogWrite in ESp32 by controlling LED brightness by PWM controller
- Control LED brightness by PS3 left joystick

PS3 All toggle button (ESP32-WROVER-IE)

PS3_all_toggle_button | Arduino 1.8.19

File Edit Sketch Tools Help

PS3_all_toggle_button

```
#include <Ps3Controller.h>

// Variables to hold button states
bool crossButtonState = false;
bool squareButtonState = false;
bool triangleButtonState = false;
bool circleButtonState = false;
bool upButtonState = false;
bool rightButtonState = false;
bool downButtonState = false;
bool leftButtonState = false;
bool L1ButtonState = false;
bool R1ButtonState = false;
bool L2ButtonState = false;
bool R2ButtonState = false;
```

Toggle by If code (Ex. Cross button)

```
// Callback Function
void notify() {

  //--- Digital cross/square/triangle/circle toggle button events ---
  // Cross button
  if (Ps3.event.button_down.cross) {
    Serial.print("Cross presssed = ");
    crossButtonState = !crossButtonState;
    Serial.println(crossButtonState);
  }
```

COM3

Send

```
15:57:47.121 -> Ready.
15:57:48.053 -> Connected.
15:57:48.709 -> Square pressed = 1
15:57:49.131 -> Triangle presssed = 1
15:57:49.365 -> Left pressed = 1
15:57:49.692 -> Left pressed = 0
15:57:50.067 -> Down pressed = 1
15:57:50.396 -> Left pressed = 1
15:57:50.724 -> Left pressed = 0
15:57:51.147 -> Left pressed = 1
15:57:51.381 -> Left pressed = 0
15:57:51.661 -> Down pressed = 0
15:57:51.941 -> Down pressed = 1
15:57:53.156 -> Down pressed = 0
```

Autoscroll Show timestamp Newline 115200 baud Clear output

get X-Y value from left stick (ESP32-WROVER-IE)

PS3_getXYValue | Arduino 1.8.19

File Edit Sketch Tools Help

PS3_getXYValue

```
#include <Ps3Controller.h>
#include <analogWrite.h>

int rightX = 0;
int rightY = 0;
int portTransferX = 12;
int portTransferY = 14;

// Callback Function
void notify() {
    // Get Joystick value
    rightX = (Ps3.data.analog.stick.rx);
    rightY = (Ps3.data.analog.stick.ry);

    // Print to Serial Monitor
    Serial.print("X value = ");
    Serial.print(rightX);
    Serial.print(" - Y value = ");
    Serial.println(rightY);
}

// On Connection function
void onConnect() {
  // Print to Serial Monitor
  Serial.println("Connected.");
}


void setup() {
```

COM3

Send

```
16:23:18.223 -> X value = 127 - Y value = 3
16:23:18.270 -> X value = 127 - Y value = 3
16:23:18.270 -> X value = 127 - Y value = 3
16:23:18.270 -> X value = 127 - Y value = 3
16:23:18.270 -> X value = 127 - Y value = 3
16:23:18.317 -> X value = 26 - Y value = 3
16:23:18.317 -> X value = 11 - Y value = 3
16:23:18.317 -> X value = 1 - Y value = 2
16:23:18.317 -> X value = -1 - Y value = 2
16:23:18.363 -> X value = -1 - Y value = 2
16:23:18.363 -> X value = -1 - Y value = 2
16:23:18.363 -> X value = -1 - Y value = 2
16:23:18.363 -> X value = -1 - Y value = 2
```

Autoscroll Show timestamp | Newline | 115200 baud | Clear output

Done uploading.

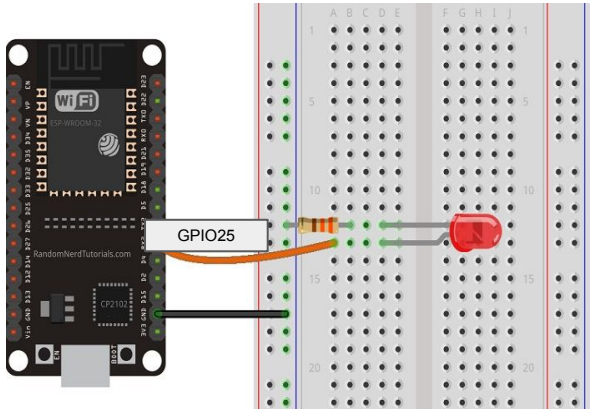
```
Leaving...
Hard resetting via RTS pin...
```

19 | ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3

Type here to search | EN | 30°C มีเมฆส่วนใหญ่ | 16:23 12/2/2567

Idle point : x = - 3 to -1 , y = -2 to 1

ESP32 LED PWM Controller



Essential variables

1. ledcChannel (Choose from 0 to 15)
2. PWM Frequency (0-5000 Hz)
3. resolution (0-16 bits, if 8 bits → 0-255)

Essential functions

1. ledcSetup (ledcChannel, freq, resolution)
2. ledcAttachPin(GPIO, ledcChannel)
3. ledcWrite(ledcChannel, analogValue)

ledcWrite in ESP32 = analogWrite in Arduino

ESP32_LED_PWM_Controller | Arduino 1.8.19

File Edit Sketch Tools Help

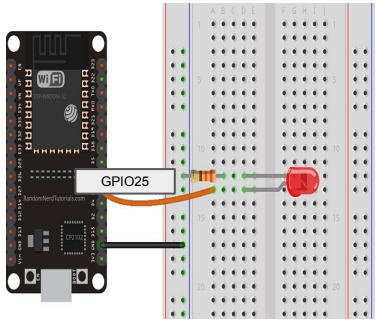
ESP32_LED_PWM_Controller

```
const int ledPin = 25;    //GPIO25

//setting PWM properties
const int ledChannel = 0;     // PWM Channel. There are 16 CH from 0 to 15
const int freq = 5000;        //PWM Signal frequency
const int resolution = 8;     // 8 bits --> value 0-255

void setup() {
  // configure LED PWM funtionalities
  ledcSetup(ledChannel, freq, resolution);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(ledPin, ledChannel);
}

void loop() {
  // increase the LED brightness
  for(int dutyCycle =0; dutyCycle <= 255; dutyCycle++){
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
  }

  // decrease the LED brightness
  for(int dutyCycle =255; dutyCycle >= 0; dutyCycle--){
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
  }
```

Done Saving.

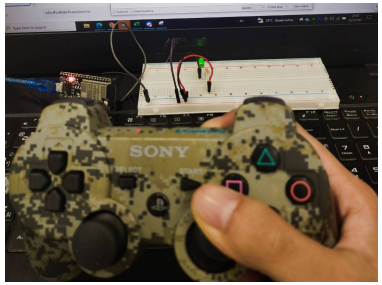
Can cyclicly control the brightness of LED

XY value from PS3 to control LED by PWM




```
3  const int ledPin = 25;     //GPIO25
4  int brightness = 0;
5
6  //setting PWM properties
7  const int ledChannel = 0;      // PWM Channel. There are 16 CH from 0 to 15
8  const int freq = 5000;         //PWM Signal frequency
9  const int resolution = 8;      // 8 bits --> value 0-255
```

Same PWM setup as previous lab



Control the brightness by the “radius distance” from the left stick center

get XY value

```
15 void notify() {
16     // Get Joystick value
17     rightX = (Ps3.data.analog.stick.rx);
18     rightY = (Ps3.data.analog.stick.ry);
19
20     // Print to Serial Monitor
21     Serial.print("X value = ");
22     Serial.print(rightX);
23     Serial.print(" - Y value = ");
24     Serial.print(rightY);
```

Maths

```
26     int squareRightX = pow(rightX, 2);
27     int squareRightY = pow(rightY, 2);
28     int sumSquare = squareRightX + squareRightY;
29     brightness = sqrt(sumSquare);
30     Serial.print(" - Brightness = ");
31     Serial.println(brightness);
32     if(brightness >= 16){
33     ledcWrite(ledChannel, brightness);
34     }
35     else{
36     ledcWrite(ledChannel, LOW);
37     }
38 }
```

Result

COM3

```
21:45:17.453 -> X value = -1 - Y value = -3 - Brightness = 3
21:45:17.500 -> X value = -1 - Y value = -3 - Brightness = 3
21:45:17.500 -> X value = -1 - Y value = -3 - Brightness = 3
21:45:17.500 -> X value = -1 - Y value = -3 - Brightness = 3
21:45:17.547 -> X value = -1 - Y value = -3 - Brightness = 3
21:45:17.547 -> X value = -1 - Y value = -3 - Brightness = 3
21:45:17.547 -> X value = 21 - Y value = -32 - Brightness = 38
21:45:17.547 -> X value = 21 - Y value = -32 - Brightness = 38
21:45:17.593 -> X value = 21 - Y value = -32 - Brightness = 38
21:45:17.593 -> X value = 28 - Y value = -56 - Brightness = 62
21:45:17.593 -> X value = 31 - Y value = -90 - Brightness = 95
21:45:17.593 -> X value = 39 - Y value = -128 - Brightness = 133
21:45:17.593 -> X value = 42 - Y value = -128 - Brightness = 134
```