KING MONGKUT'S UNIVERSITY OF TECHNOLOGY NORTH BANGKOK

Application of PID controller to control mobile robot position

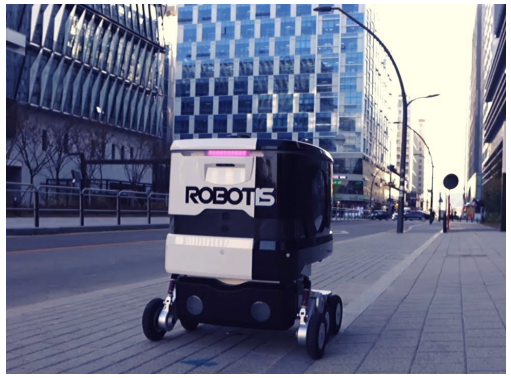
Tanawin Siriwan 6203014610060 sec 1
College of industrial technology
Department of Mechanical engineering Technology

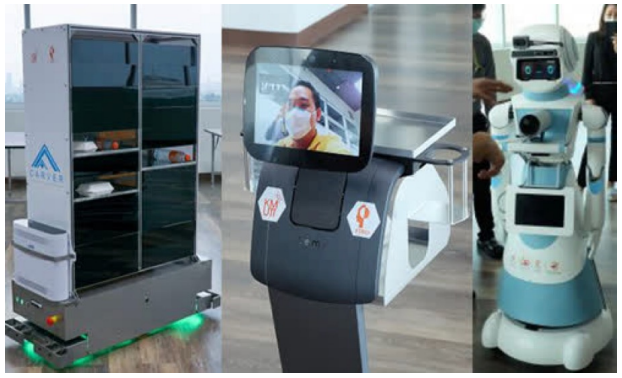
Asst. prof. Dr.Chanin Joochim

Topic

- Introduction
- Opject
- Scope
- Making process
- Priciple
- Test
- Problem
- Summarize

Introduction





Opjective

1

เพื่อศึกษาการใช้ encoder
ในการควบคุมตำแหน่งการเคลื่อนที่ของหุ่นยนต์

2

เพื่อศึกษาการเขียนโปรแกรม Arduino
ในการควบคุมตำแหน่งของหุ่นยนต์

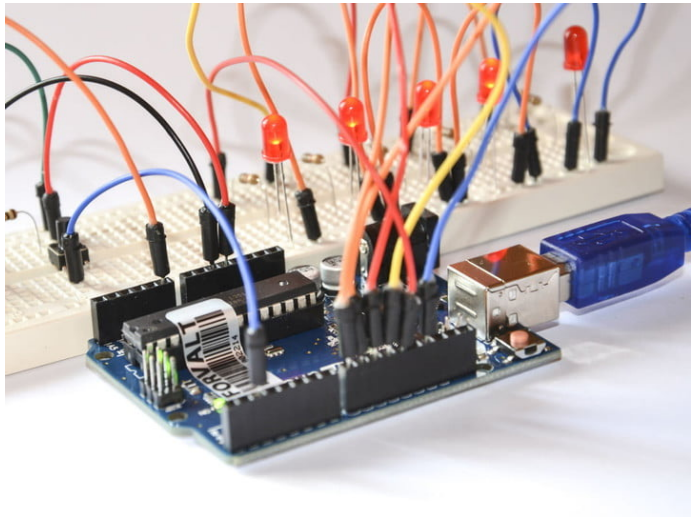
3

เพื่อออกแบบหุ่นยนต์ที่ควบคุม
ตำแหน่งการเคลื่อนที่ด้วย PID

Scope

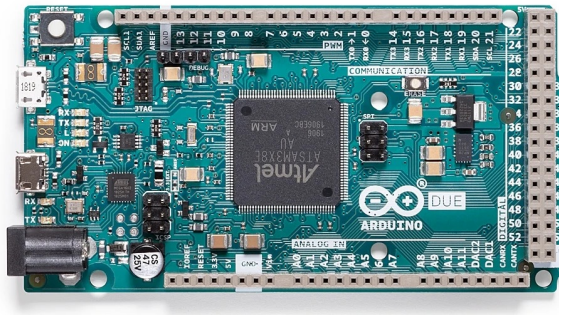
1. ใช้ encoder ที่ติดอยู่กับมอเตอร์ในการอ่านค่าตำแหน่งของหุ่นยนต์
2. หุ่นยนต์สามารถเคลื่อนที่ในรูปแบบสี่เหลี่ยมจัตุรัส
3. หุ่นยนต์สามารถเคลื่อนที่ในรูปแบบสามเหลี่ยมมุมฉาก
4. พื้นที่ในการเคลื่อนที่ในพื้นระนาบเท่านั้น

Hard ware

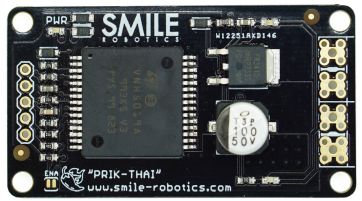


Soft ware

Hard ware

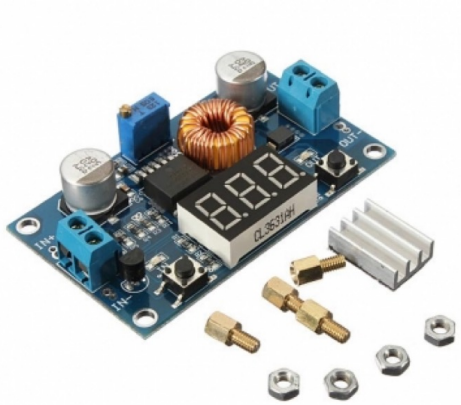


Arduino DUE



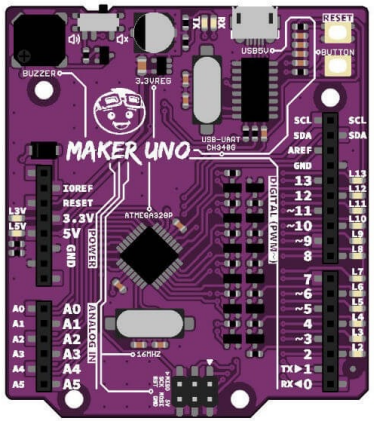
DC motor driver

Vapcell 18650 battery
3600mAh 3.7v

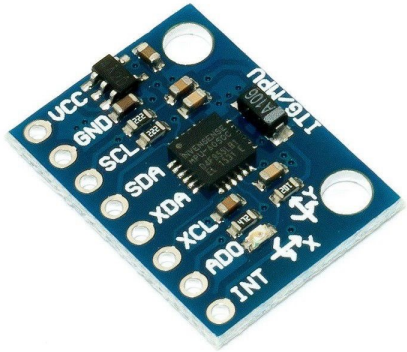


DC to DC stepdown

Hard ware



Maker UNO



GY - 521

LCD i2c display 20 x 4

Hard ware

Toggle switch on - off

DC motor 12V encoder

Ball wheel

wheel

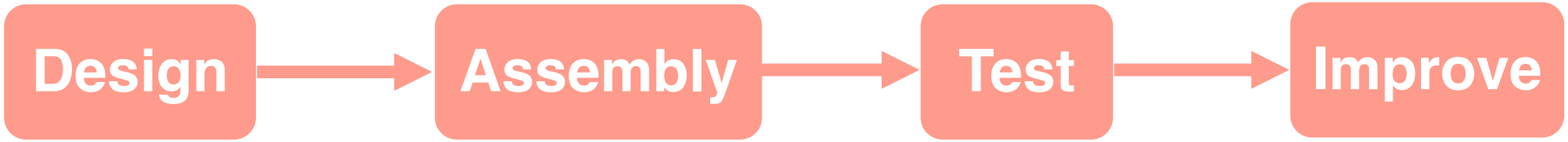
Soft ware


ARDUINO
OPEN-SOURCE
COMMUNITY

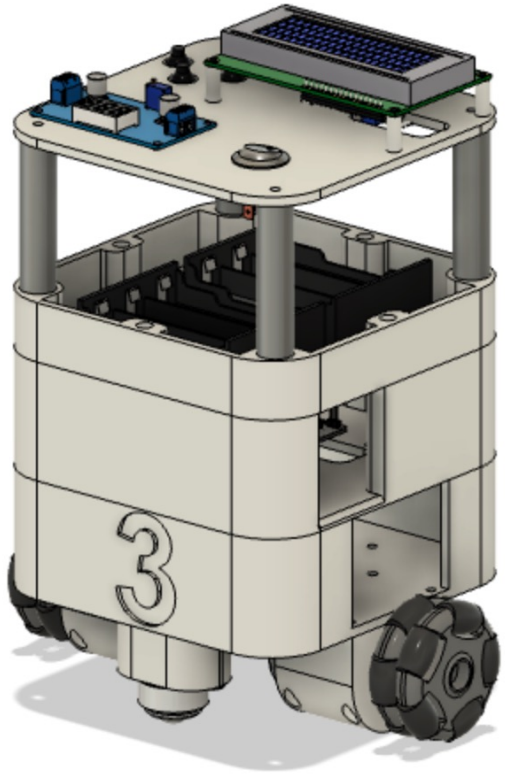

EasyEDA


AUTODESK®
FUSION 360™

Making process



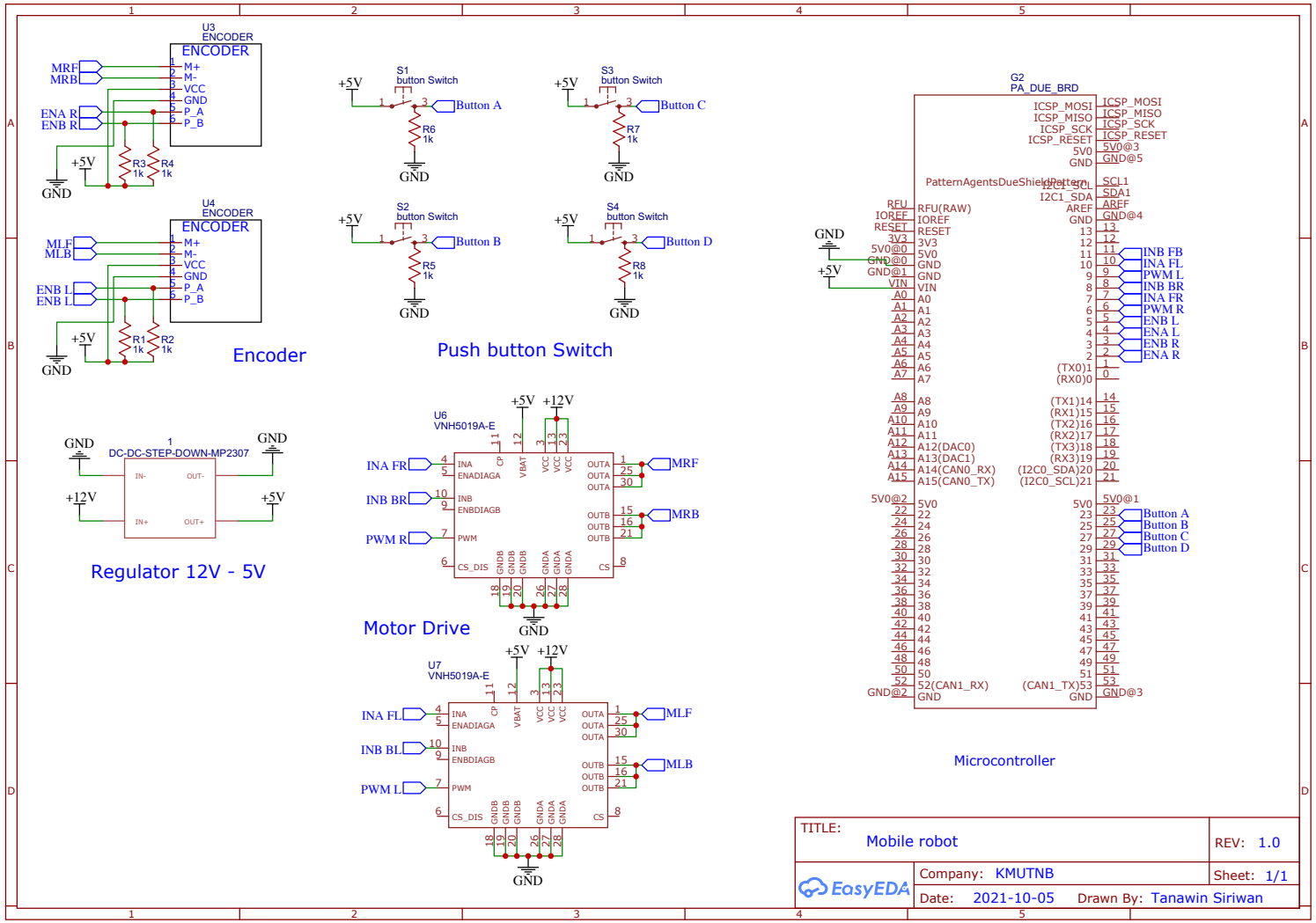
Mechnicsm Design



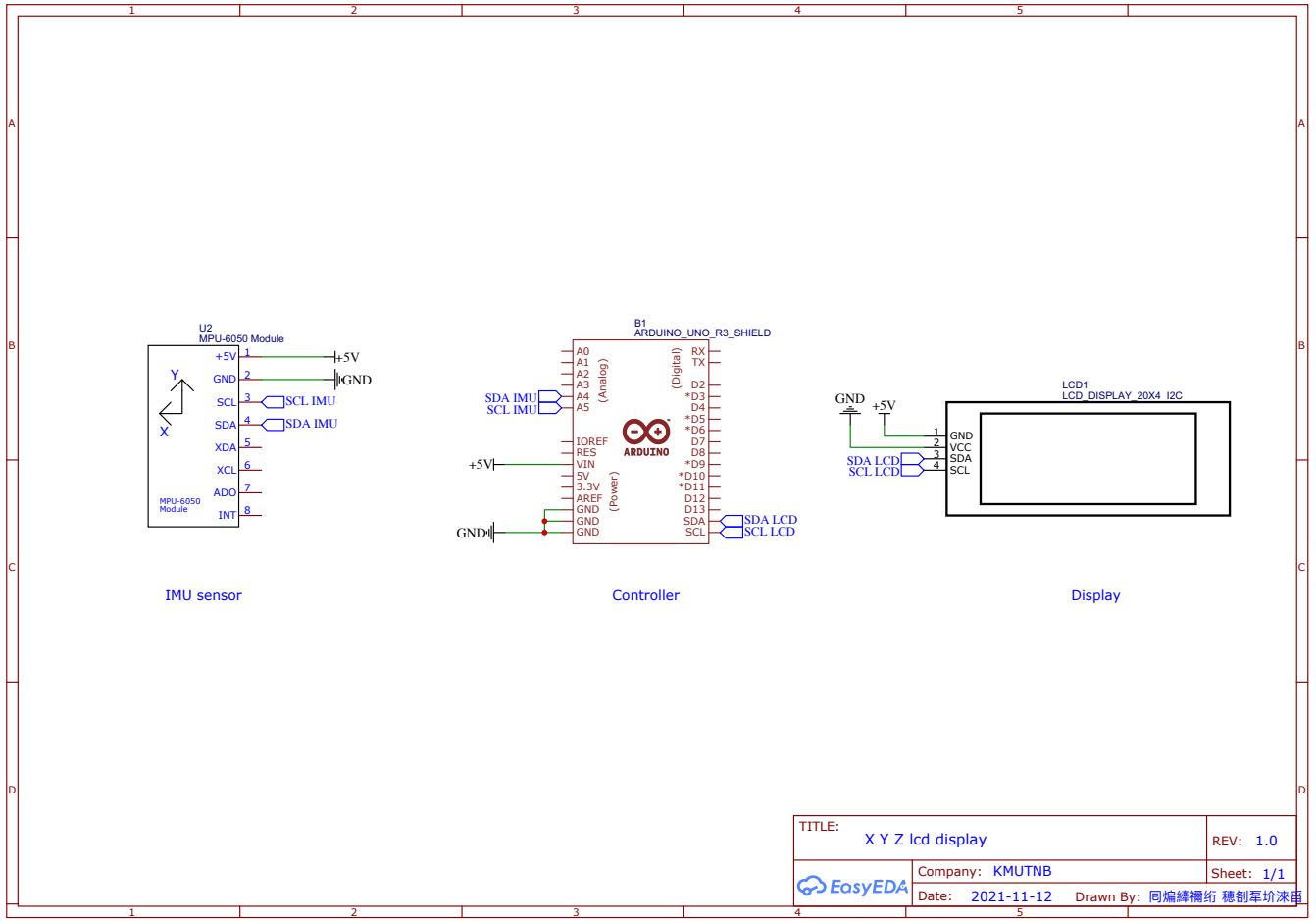
W x L x H
145 x 220 x 330

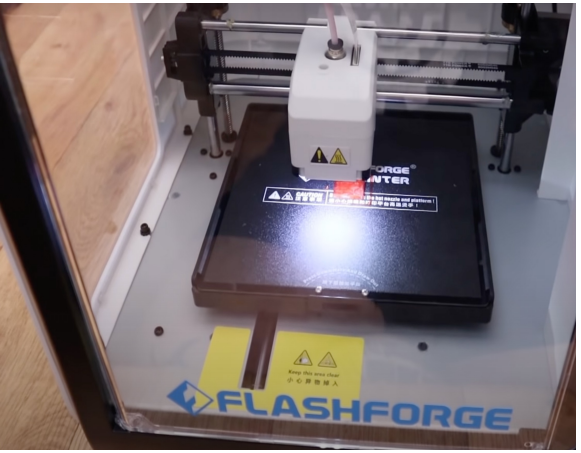
AUTODESK®
FUSION 360™

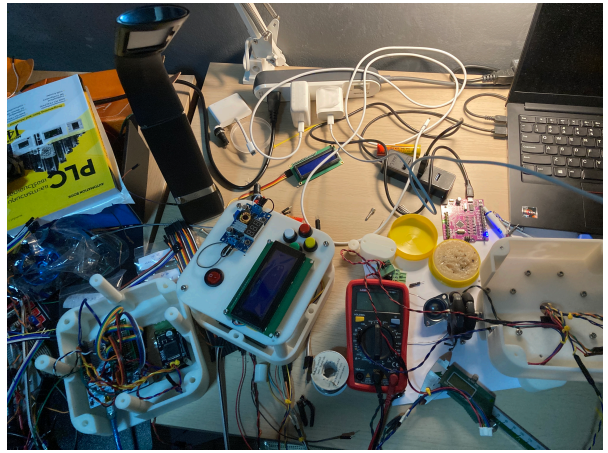
Electronics Design

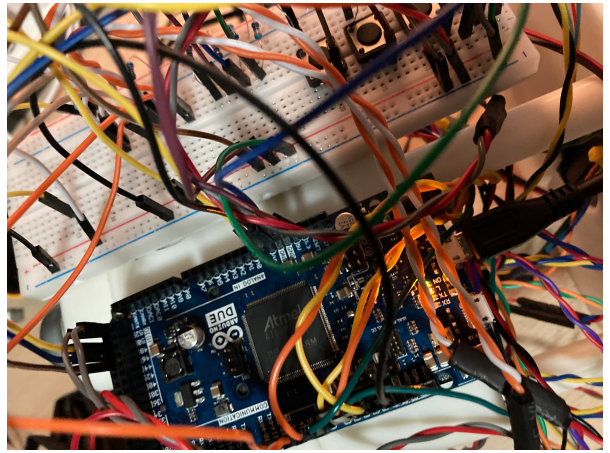


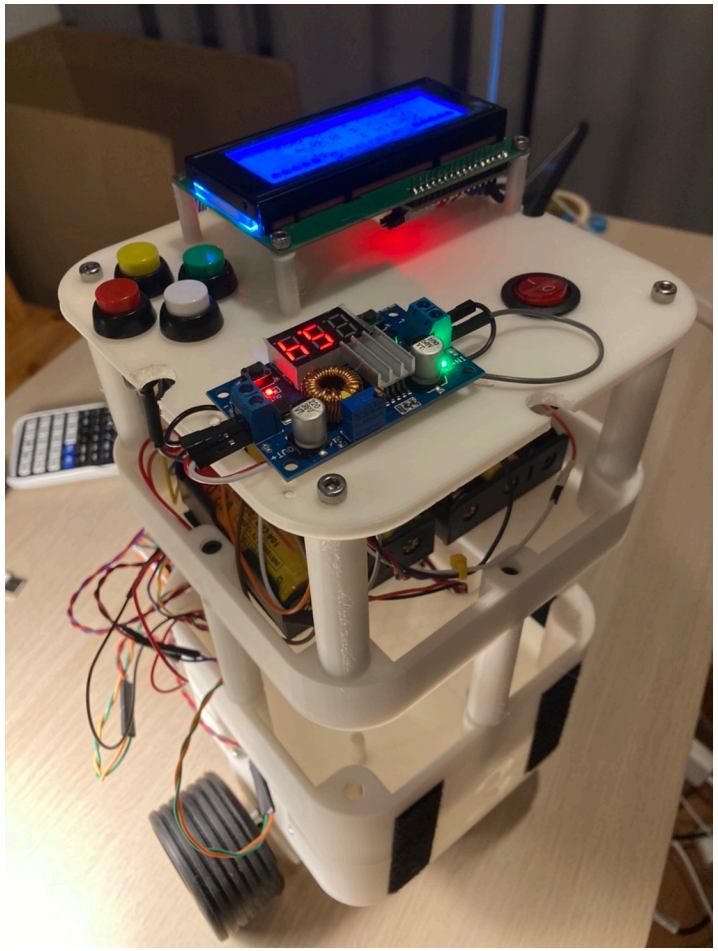
Electronics Design





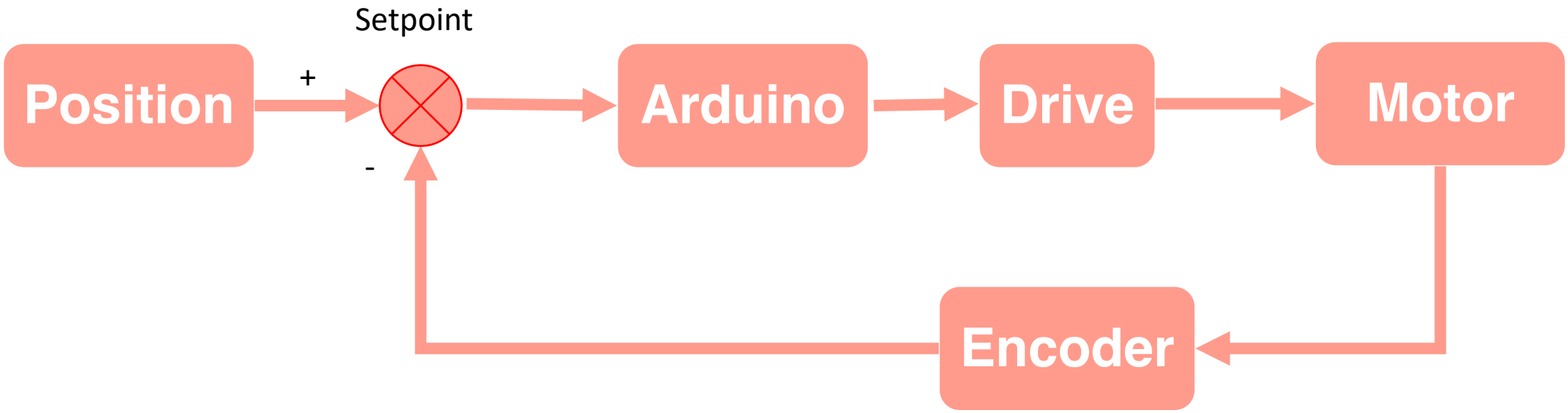






Assembly test improve

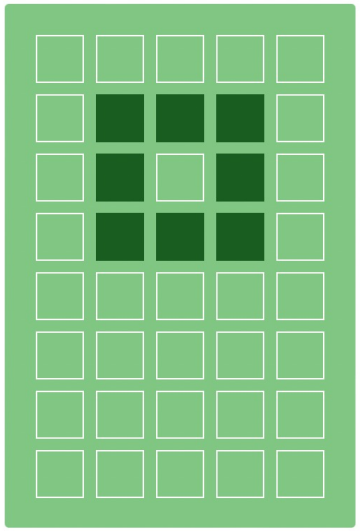
Priciple PID Robot control

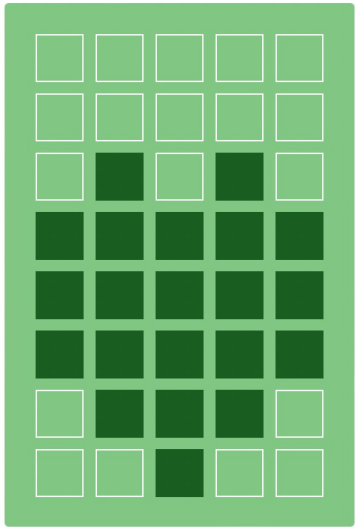


Priciple XYZ axis on LCD display

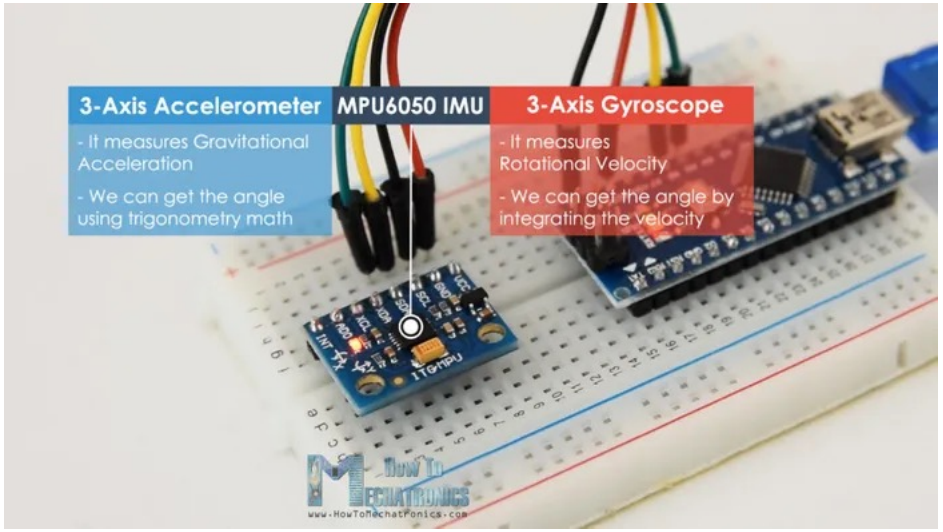
LCD emoji

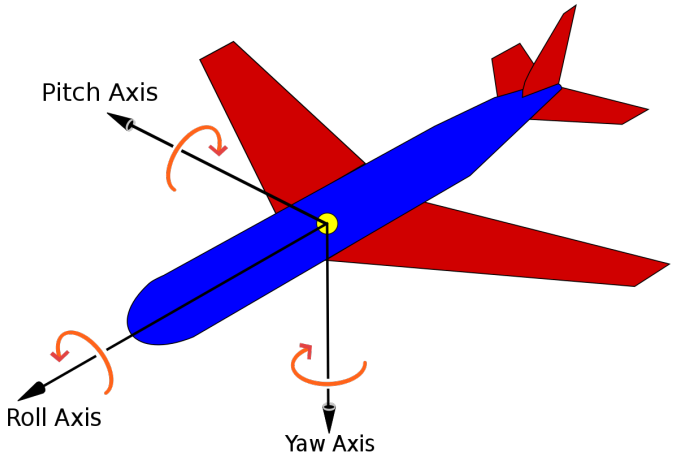
```
21 byte customChar[] = {
22   B00000,
23   B01110,
24   B01010,
25   B01110,
26   B00000,
27   B00000,
28   B00000,
29   B00000
30 };
31
32 byte customChar_Hart[] = {
33   B00000,
34   B00000,
35   B01010,
36   B11111,
37   B11111,
38   B11111,
39   B01110,
40   B00100
41 };
```





GY-521 Module

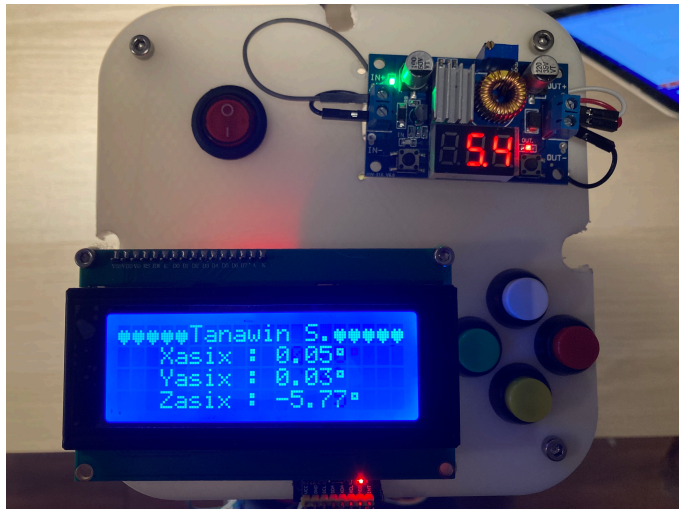




https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/

XYZ axis on display

```
63 void loop() {
64   mpu.update();
65   if ((millis() - timer) > 333) {     // print data every 333ms
66     Serial.print("X : ");
67     Serial.print(mpu.getAngleX());    //get X axis
68     Serial.print("\tY : ");
69     Serial.print(mpu.getAngleY());    //get Y axis
70     Serial.print("\tZ : ");
71     Serial.println(mpu.getAngleZ());  //get Z axis
72     timer = millis();
73
74     lcd.setCursor(0, 0);lcd.write(1);lcd.write(1);lcd.write(1);lcd.write(1);lcd.write(1);lcd.setCursor(5, 0); lcd.print("Tanawin S.");lcd.write(1);lcd.write(1);lcd.write(1);lcd.write(1);lcd.write(1);
75     lcd.setCursor(3, 1); lcd.print("Xasix : "); lcd.print(mpu.getAngleX()); lcd.write(0); lcd.print(" ");
76     lcd.setCursor(3, 2); lcd.print("Yasix : "); lcd.print(mpu.getAngleY()); lcd.write(0); lcd.print(" ");
77     lcd.setCursor(3, 3); lcd.print("Zasix : "); lcd.print(mpu.getAngleZ()); lcd.write(0); lcd.print(" ");
78   }
79 }
```



Mode selecter

```
if ( millis() - last_time_X > period_X) {
  last_time_X = millis();
  if (state_modeREC == 1) {                    /*Read button value condition*/
    stat_A = 1 ;                               /*count = 1*/
  }
  //Serial.print(stat_A);Serial.print(",");
}
if ( millis() - last_time_Y > period_Y) {
  last_time_Y = millis();
  if (state_modeTRI == 1) {                    /*Read button value condition*/
    stat_B = 1;                                /*count = 1*/
  }
  //Serial.print(stat_B);Serial.print(",");
}
if ( millis() - last_time_Z > period_Z) {
  last_time_Z = millis();
  if (state_mode == 1) {                       /*Read button value condition*/
    stat_C = 1;                                /*count = 1*/
  }
}
Serial.print(stat_A); Serial.print(","); Serial.print(stat_B); Serial.println(","); Serial.print(stat_C);
```

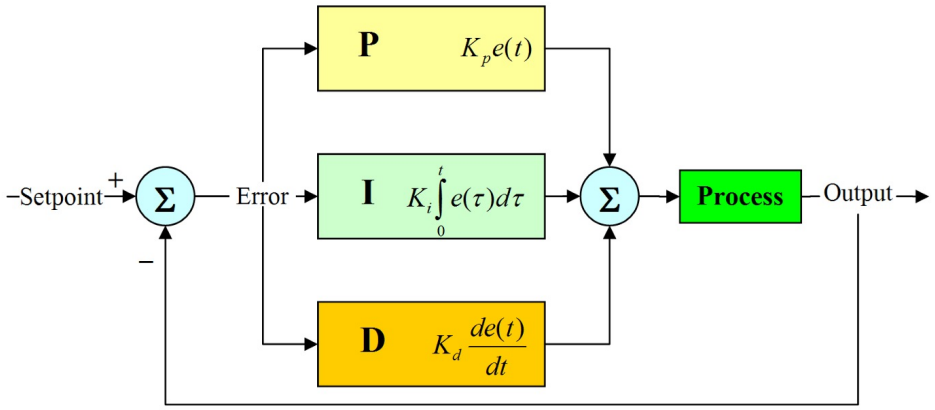
```
void readButtonA() {
  state_modeREC = digitalRead(47);
}

void readButtonB() {
  state_modeTRI = digitalRead(49);
}

void readButtonC() {
  state_mode = digitalRead(51);
}
```

Mode selecter

```
switch (stat_C) {                          /*Condition robot mode*/
  case 1 : Triangle(); break;              /*Robot triangle form*/

}
switch (stat_B) {                          /*Condition robot mode*/
  case 1 : Rectangle(); break;             /*Robot rectangular form*/
}
switch (stat_A) {                          /*Condition robot mode*/
  case 1 : linear(); break;                /*Robot linear form*/
}
```

PID Controller

$$u(t) = K_p e(t) + K_i \int_0^t e\,(\tau) + K_d \frac{d}{dx} e(t)$$



PID block diagram

```
previous_error = setpoint - actual_position
integral = 0
start:
  error = setpoint - actual_position
  integral = integral + (error*dt)
  derivative = (error - previous_error)/dt
  output = (Kp*error) + (Ki*integral) + (Kd*derivative)
  previous_error = error
  wait(dt)
  goto start
```

pseudocode

https://th.wikipedia.org/wiki/ระบบควบคุมพีไอดี

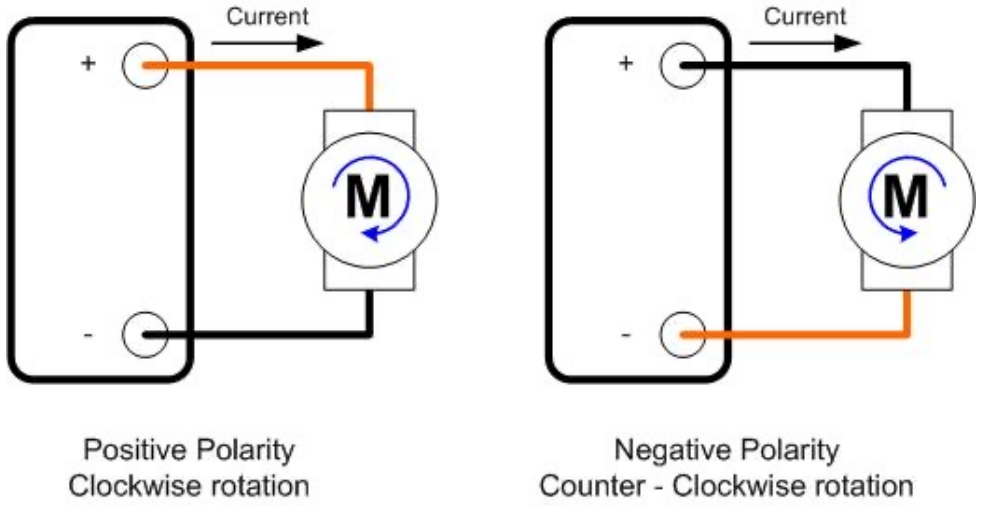
PID tuning

```
/************************************RIGHT wheel PID Tuning*************************************/
//Define Variables we'll be connecting to
double input_R = 0, output_R = 0, setpoint_R = 0;

//Specify the links and initial tuning parameters
double kp_R = 0.5, ki_R = 0.006 , kd_R = 0;
PID myPID_R(&input_R, &output_R, &setpoint_R, kp_R, ki_R, kd_R, DIRECT);
```

```
 //turn the PID RIGHT on
 myPID_R.SetMode(AUTOMATIC);
 myPID_R.SetSampleTime(1);
 myPID_R.SetOutputLimits(-100, 100);
```

```
452 void comput_R() {
453   setpoint_R = A;                  /*modify to fit motor and encoder right*/
454   input_R = encoderPos_R ;         /*data form encoder right*/
455   myPID_R.Compute();               /*calculate output*/
456   pwmOut_R(output_R);              /*drive motor drive module*/
457   Serial.print(setpoint_R);        /*mornitor of motor right*/
458   Serial.print(" ");
459   Serial.println(encoderPos_R);    /*mornitor motor right position*/
```

Motor Direction



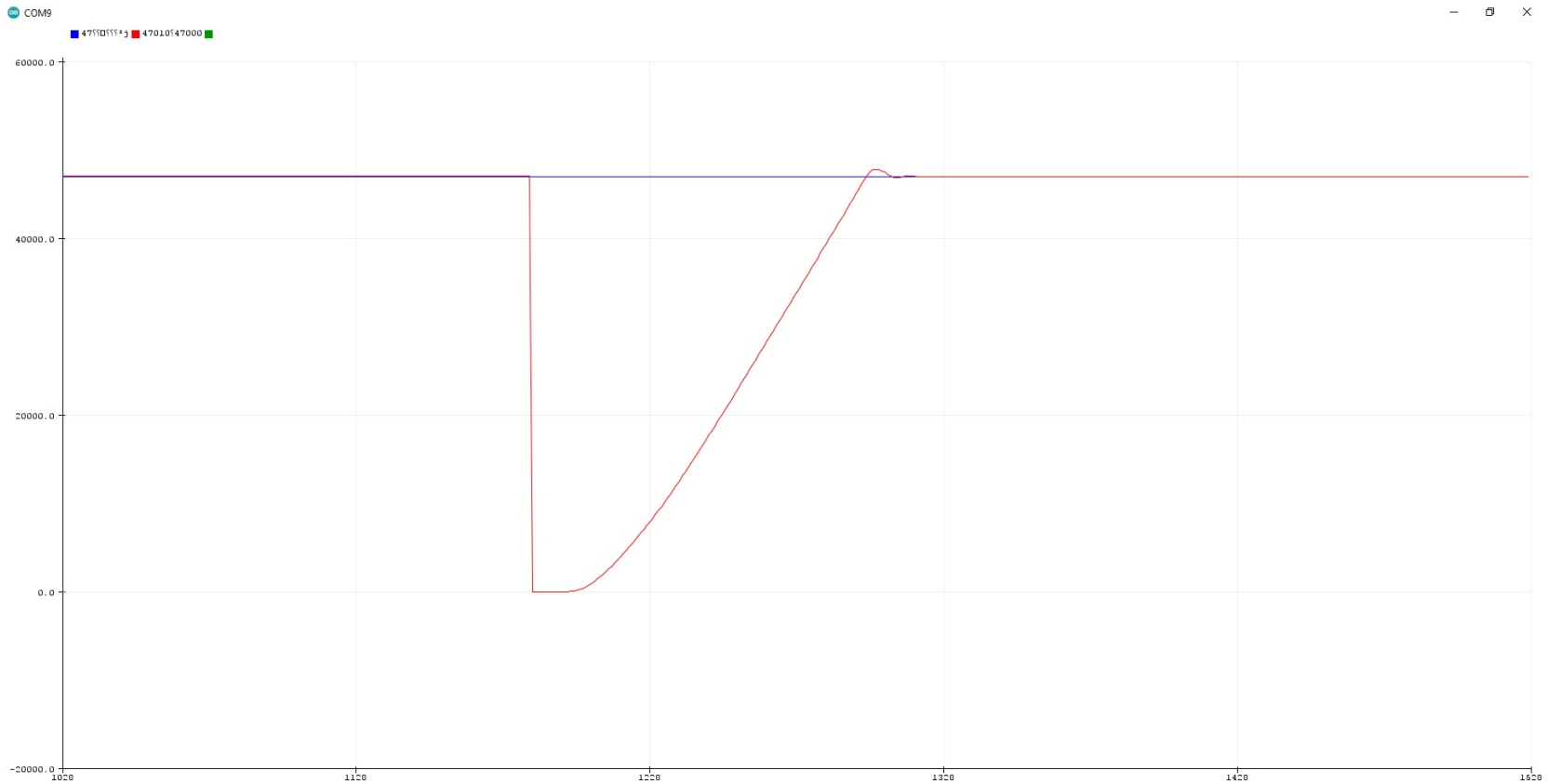

```
/**********************Motor RIGHT direction**********************/
void pwmOut_R(int out) {
  if (out > 0) {
    analogWrite(PWM_R, out);                /*drive motor right CW*/
    digitalWrite(inA_R, HIGH);
    digitalWrite(inB_R, LOW);
  }
  else {
    analogWrite(PWM_R, abs(out));           /*drive motor right CCW*/
    digitalWrite(inA_R, LOW);
    digitalWrite(inB_R, HIGH);
  }
}
```

```
/**********************Motor left direction**********************/
void pwmOut_L(int out) {
  if (out > 0) {
    analogWrite(PWM_L, out);                /*drive motor left CW*/
    digitalWrite(inB_L, HIGH);
    digitalWrite(inA_L, LOW);
  }
  else {
    analogWrite(PWM_L, abs(out));           /*drive motor left CCW*/
    digitalWrite(inB_L, LOW);
    digitalWrite(inA_L, HIGH);
  }
}
```

Graph

Motor driving go to setpoint position

- Setpoint
- Motor encoder



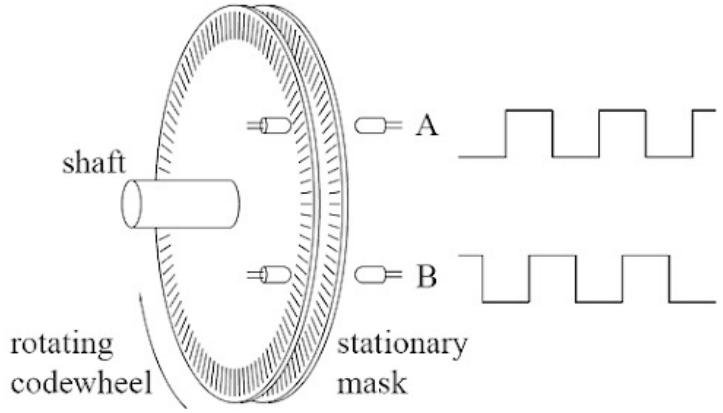
Encoder

```
103  //initialize the variables we're linked to
104  Serial.begin(115200);
105  pinMode(PWM_R, OUTPUT);
106  pinMode(inA_R, OUTPUT);
107  pinMode(inB_R, OUTPUT);
108
109  pinMode(ENCA_R, INPUT);
110  pinMode(ENCB_R, INPUT);
111  attachInterrupt(digitalPinToInterrupt(ENCA_R), readEncoder_R, RISING);
112
113  pinMode(PWM_L, OUTPUT);
114  pinMode(inA_L, OUTPUT);
115  pinMode(inB_L, OUTPUT);
116
117  pinMode(ENCA_L, INPUT);
118  pinMode(ENCB_L, INPUT);
119  attachInterrupt(digitalPinToInterrupt(ENCA_L), readEncoder_L, RISING);
```

```
594 void readEncoder_R() {
595   int b = digitalRead(ENCB_R);
596   if (b > 0) {
597     encoderPos_R++;
598   }
599   else {
600     encoderPos_R--;
601   }
602 }
603
604 /*
605    Encoder motor left
606 */
607 void readEncoder_L() {
608   int b = digitalRead(ENCB_L);
609   if (b > 0) {
610     encoderPos_L++;
611   }
612   else {
613     encoderPos_L--;
614   }
615 }
```

Encoder

12V 0.2A
Speed : 150 RPM
Gear ratio 1:78
360P



$$Pulse\ at\ main\ shaft = 78 * 360 = 28080\ pulse$$
$$Cumcirference = \pi * D = \pi * 58 = 182.12\ mm$$
$$28080\ pulse = 182.12\ mm$$

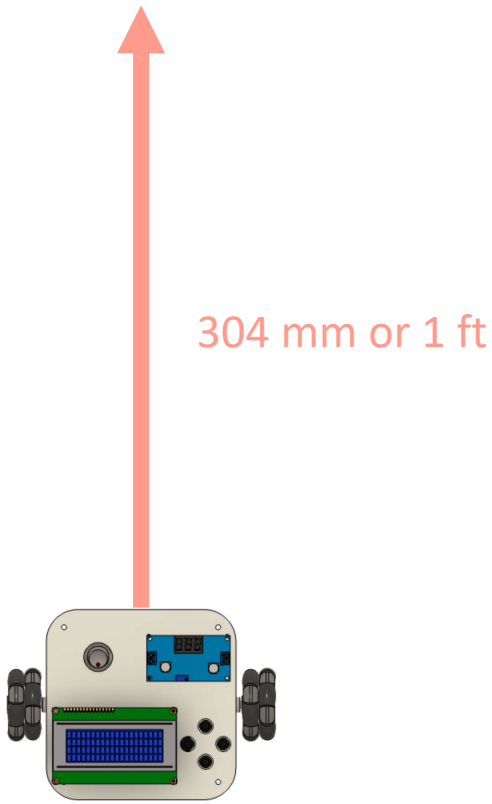
$$find\ pulse\ per\ mm = \frac{182.12}{28080} = 0.00648\ mm$$

$$1\ pulse \approx 0.0064mm$$

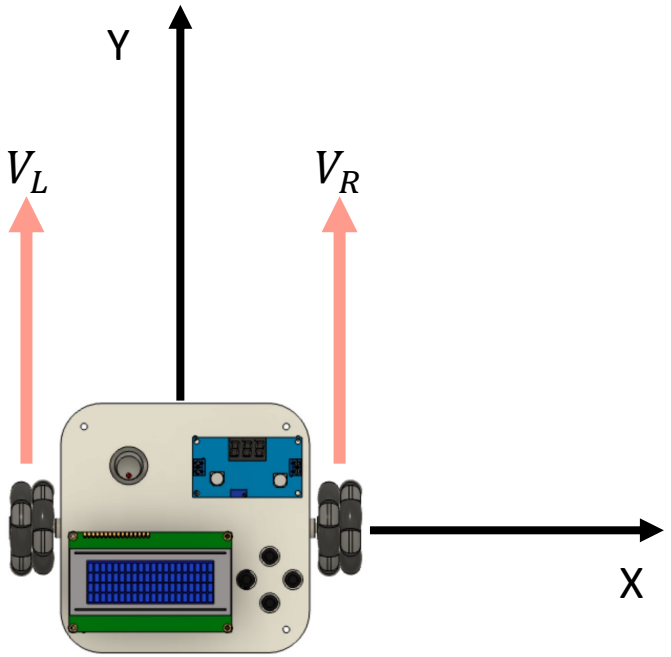
A:
B:

Direct move

$$304\ mm = \frac{304mm}{0.0064} = 47500\ pulse$$

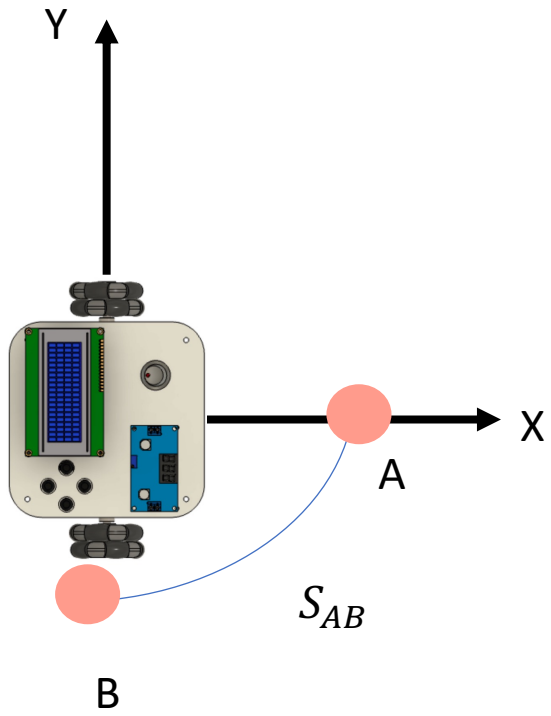



```
502 /*
503    Calculate the pluse of encoder right for command robot go to x = 0 ft, y = 1 ft
504 */
505 float pulse_RW(float pulse) {
506   int feetXX = coorX * 304;
507   float res_pulse = motor_raio * rev_pluse;
508   float tick = dimention_wheel / res_pulse;
509   pulse = feetXX / tick;
510   return pulse;
511 }
512
513 /*
514    Calculate the pluse of encoder left for command robot go to x = 0 ft, y = 1 ft
515 */
516 float pulse_LW(float pulse) {
517   int feetYY = coorY * 304;
518   float res_pulse = motor_raio * rev_pluse;
519   float tick = dimention_wheel / res_pulse;
520   pulse = feetYY / tick;
521   return pulse;
522 }
```




```
/*
   Traget point 1 start robot go to x = 0 ft, y = 1 ft
   Motor right CW direction
   Motor left CW direction
*/
void direct_drive_1() {
  double a, b, c, d;
  A = pulse_RW(a); B = pulse_LW(b);
  C = A; D = B;
  if (encoderPos_R >= A && encoderPos_L >= B) {
    Targetpoint++;
  }
}
```

Robot turn around



$$Body = 220\,mm$$

$$r = \frac{Body}{2} = \frac{220}{2} = 110mm$$

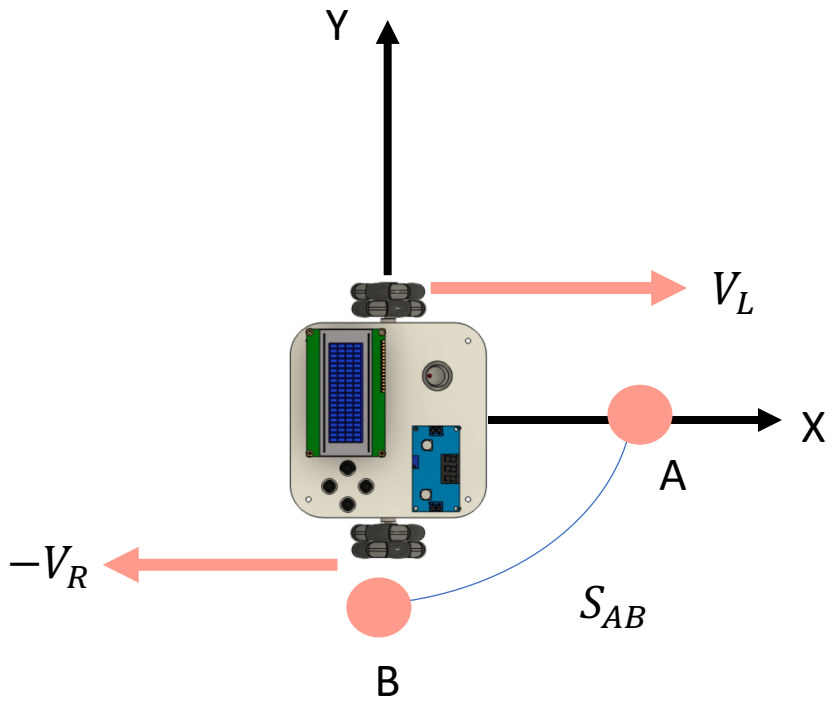
$$\theta = 90°$$

$$S_{AB} = 2\pi \cdot r \cdot \frac{\theta}{360}$$

$$S_{AB} = 2\pi \cdot 110 \cdot \frac{90}{360} = 172.78\,mm$$

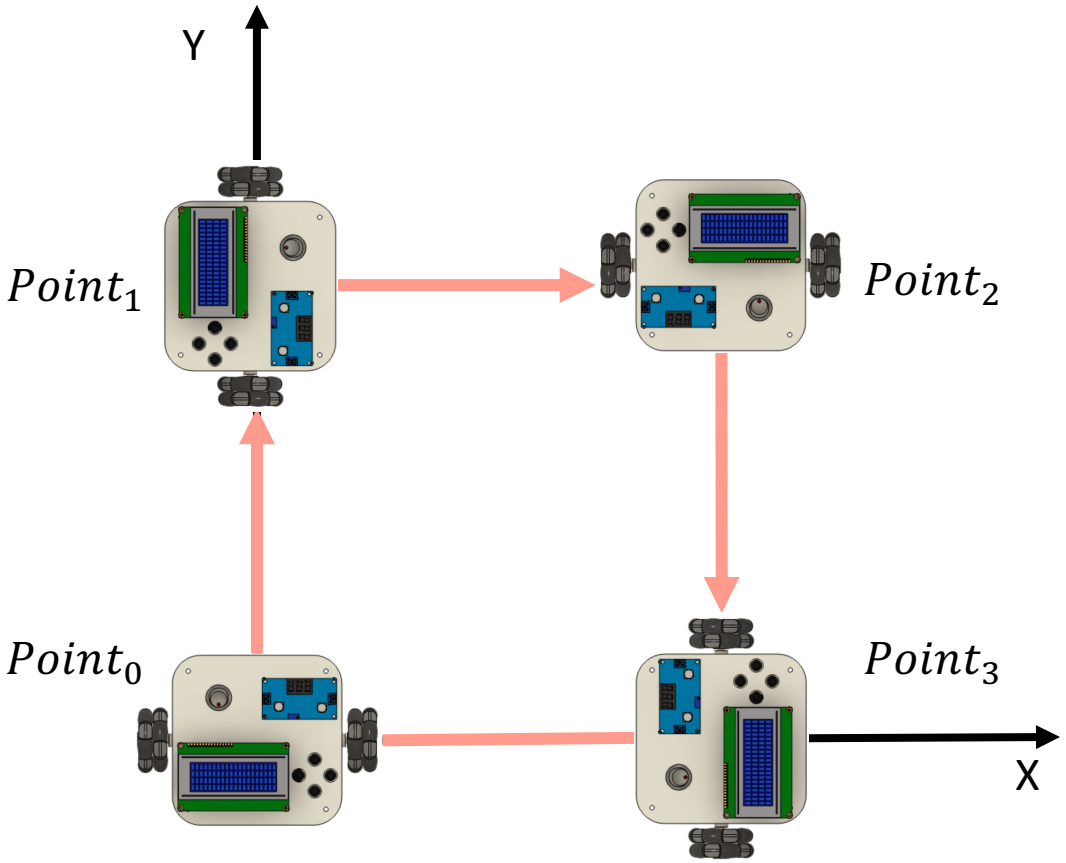
$$pulse = \frac{S_{AB}}{0.0064} = \frac{172.78}{0.0064} = 26996\,pulse$$

```
524 /*
525    Calculate the pluse of encoder for command robot turn around 90 degree
526 */
527 float pulse_r90(float pulse) {
528   float distance = 2 * 3.14 * (body / 2) * 0.13;
529   float res_pulse = motor_raio * rev_pluse;
530   float tick = dimention_wheel / res_pulse;
531   pulse = distance / tick;
532   return pulse;
533 }
```



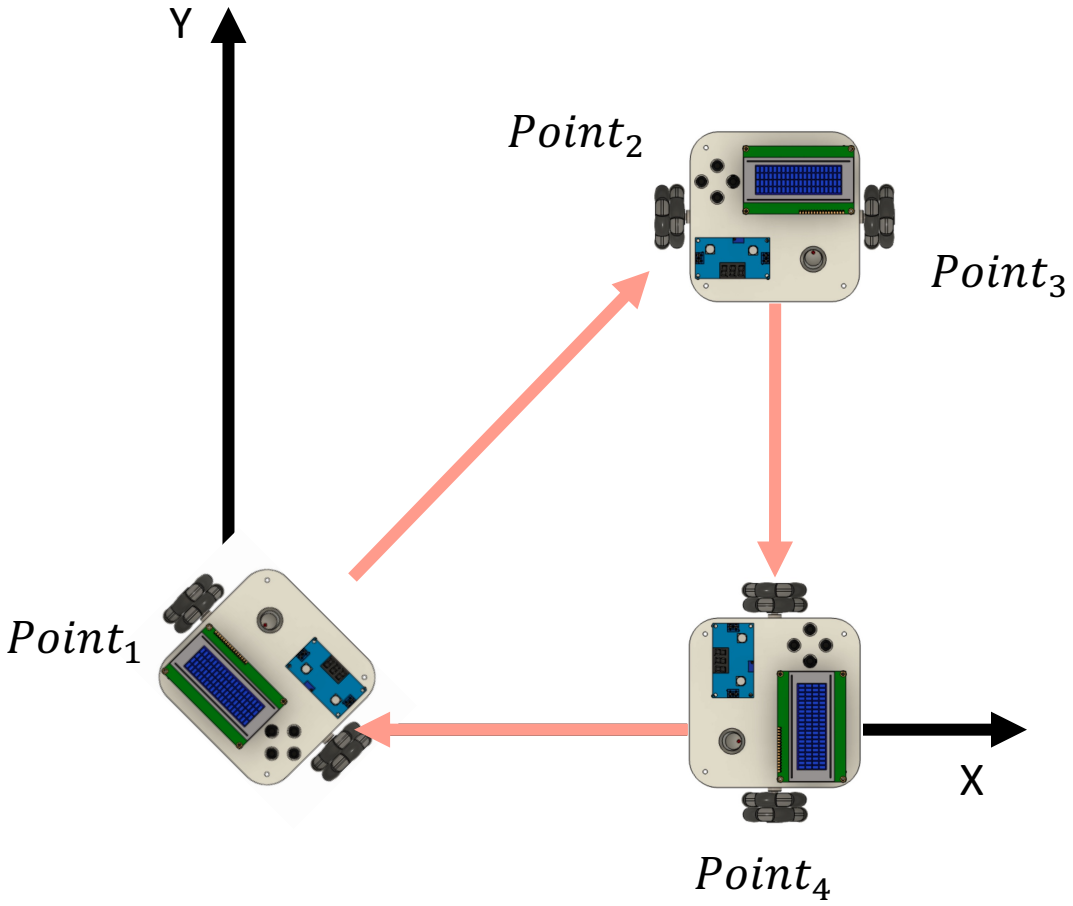

```
/*
   Traget point 5 robot turn arount 90 degree
   Motor right CCW direction
   Motor left CW direction
*/

void drive_Tri_90() {
  double a, b;
  Set_tri_R[4] = A = Set_tri_R[3] - pulse_r90(a) - 4500; Set_tri_L[4] = B = Set_tri_L[3] + pulse_r90(b) + 4000;
  if (encoderPos_R <= A && encoderPos_L >= B) {
    Targetpoint++;
  }
}
```

Rectangle move



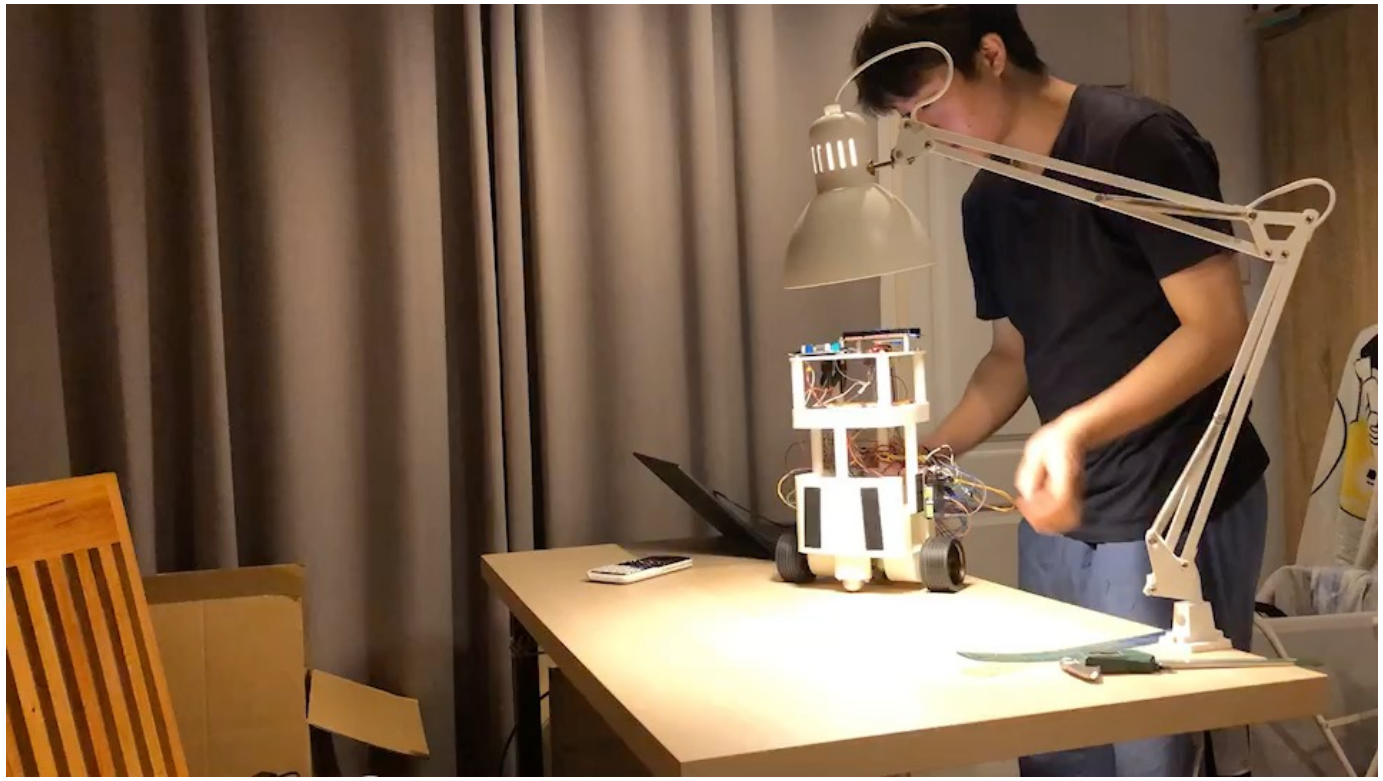

```
202 void Rectangle() {
203   comput_R();
204   comput_L();
205
206   Serial.println(Targetpoint);
207   switch (Targetpoint) {
208     case 1 : direct_drive_1(); break;
209     case 2 : A90_drive_1(); break;
210     case 3 : direct_drive_2(); break;
211     case 4 : A90_drive_2(); break;
212     case 5 : direct_drive_3(); break;
213     case 6 : A90_drive_3(); break;
214     case 7 : direct_drive_4(); break;
215     case 8 : A90_drive_4(); break;
216   }
217 }
```

Triangle move

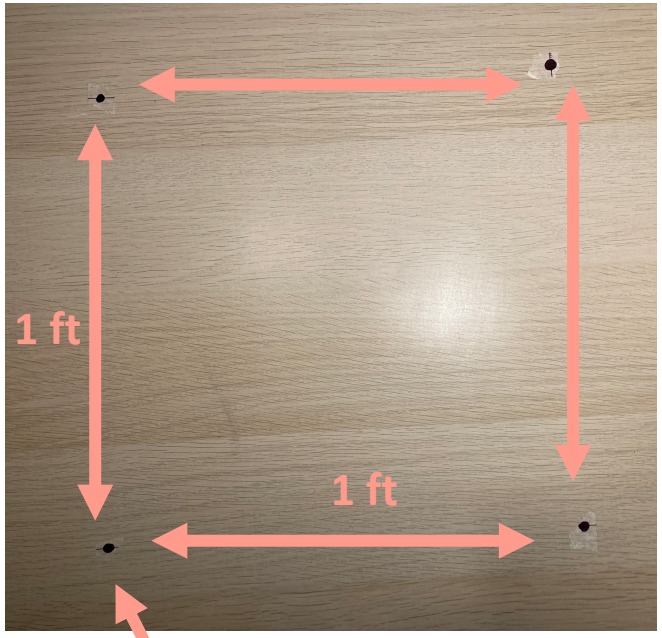


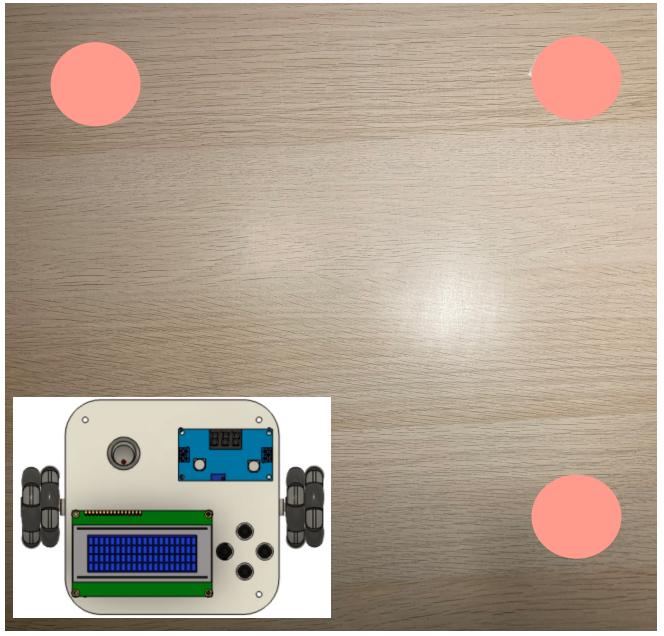

```
182 void Triangle() {
183   comput_R();
184   comput_L();
185
186   Serial.println(Targetpoint);
187   switch (Targetpoint) {
188     case 1 : drive_Tri_45(); break;
189     case 2 : driff_drive(); break;
190     case 3 : return_drive(); break;
191     case 4 : back_drive(); break;
192     case 5 : drive_Tri_90(); break;
193     case 6 : to_home(); break;
194     case 7 : drive_Tri_90_2(); break;
195   }
196 }
```

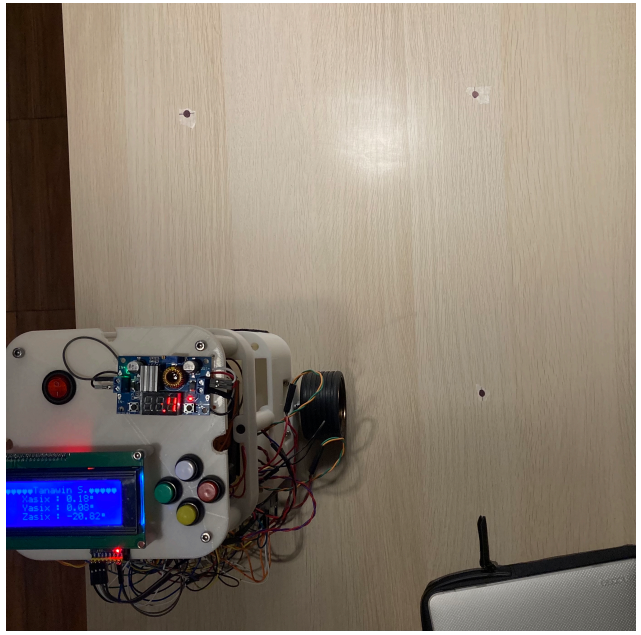
Calibration



Test field

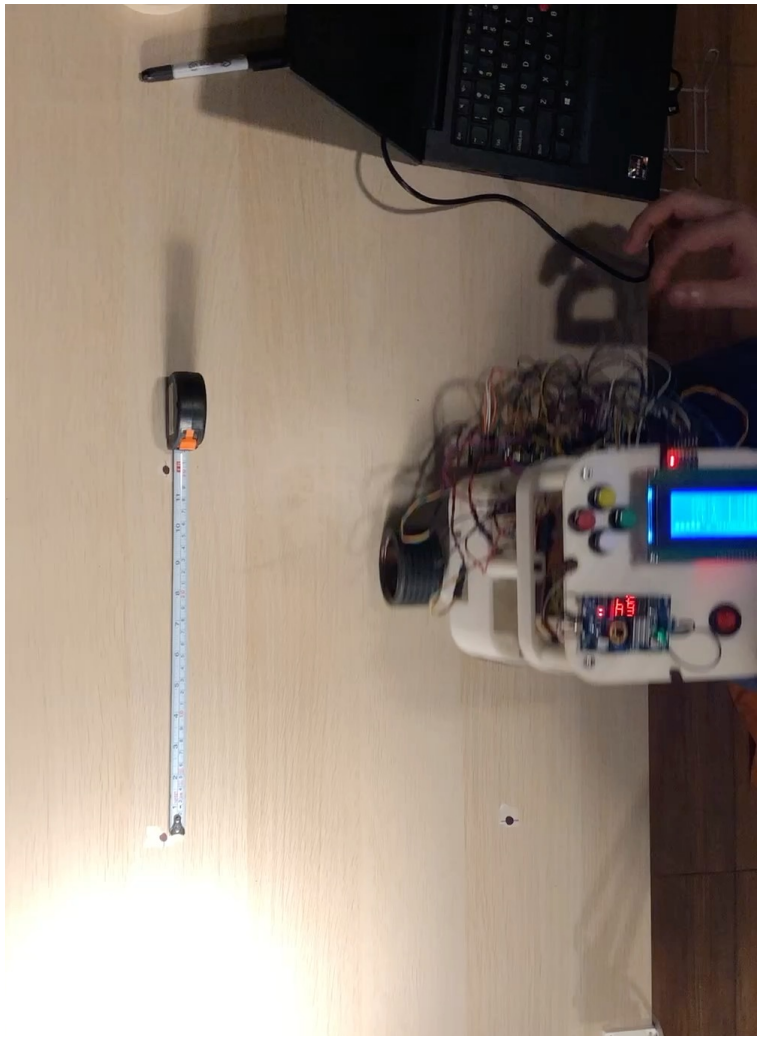


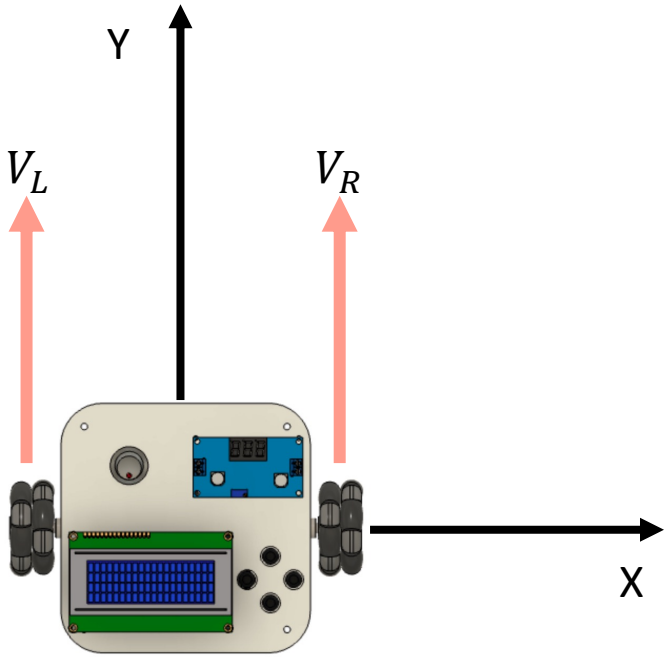




1ft x 1ft

Direct Drive test





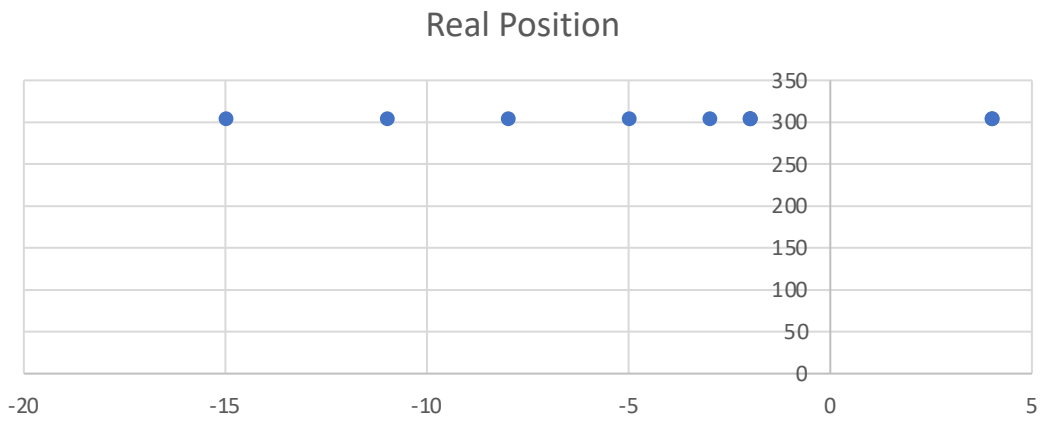
	X	Y
1	-8	304
2	-5	304
3	-15	304
4	-2	304
5	-3	304
6	-2	304
7	4	304
8	-11	304
9	-2	304
10	4	304

Average : X = -4 , Y = 304
SD. : X = 6.03, Y = 0

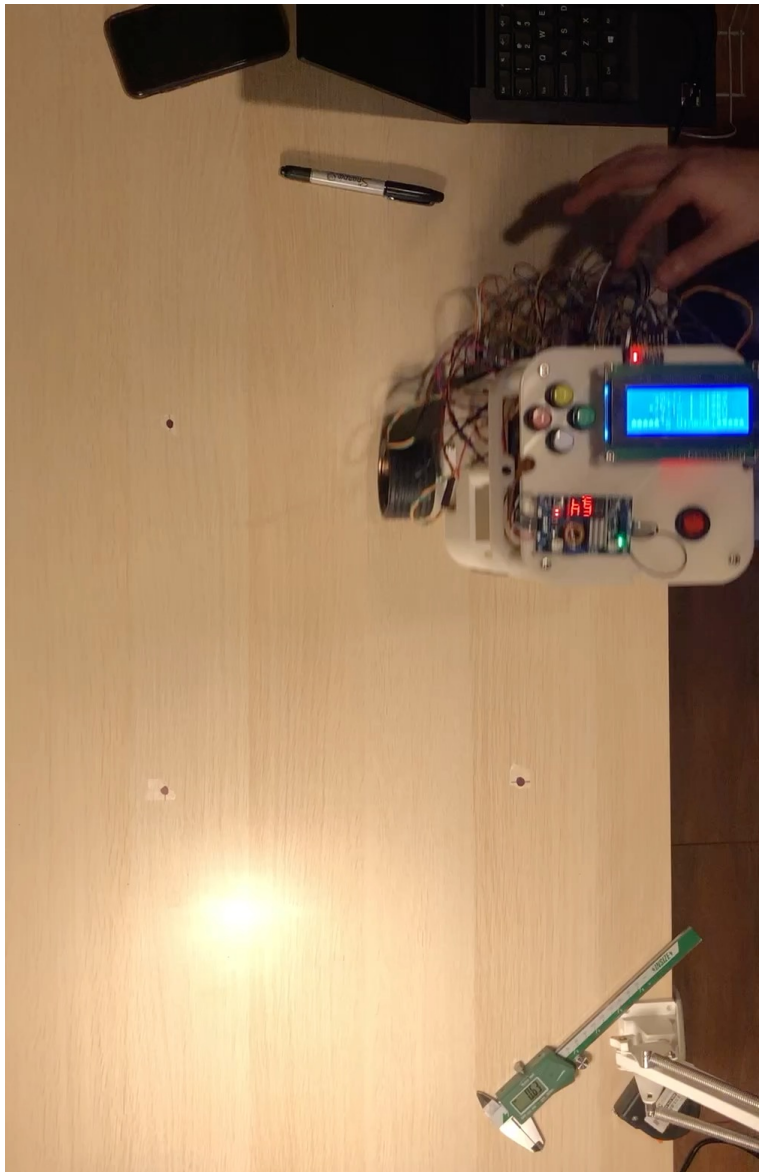
$$Position = \sqrt{\left(X_{measure} - X_{origin}\right)^2 + \left(Y_{measure} - Y_{origin}\right)^2}$$

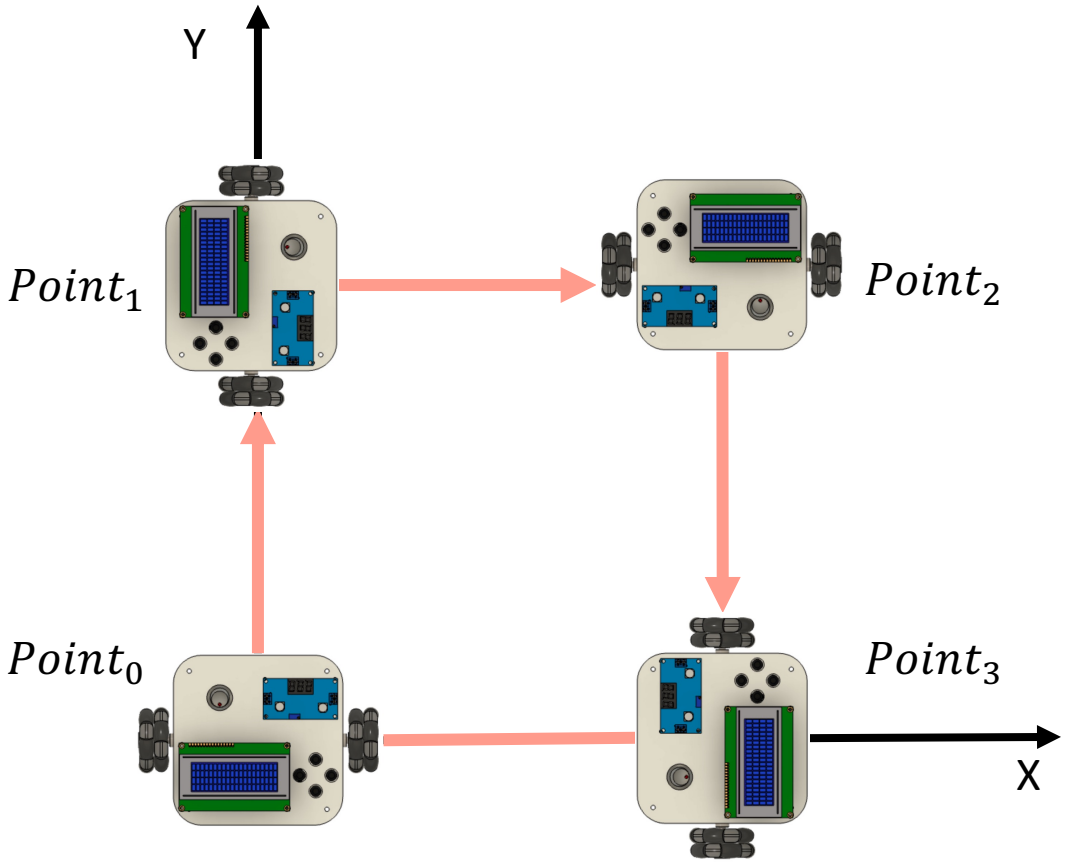
$$Position = \sqrt{(-4-0)^2 + (304-0)^2} = 304.02\ mm$$

$$\theta = \tan^{-1}(y/x) = \tan^{-1}\left(-\frac{304}{4}\right) = -89.24°$$



Rectangle drive test





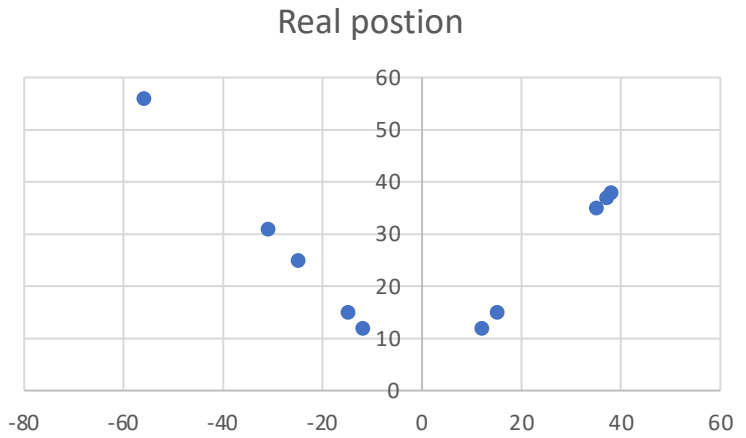
	X	Y
1	15	15
2	12	12
3	-31	31
4	-56	56
5	-25	25
6	-15	15
7	38	38
8	37	37
9	35	35
10	-12	12

Average : X = -0.2 , Y = 27.6
SD. : X = 26.48, Y = 14.45

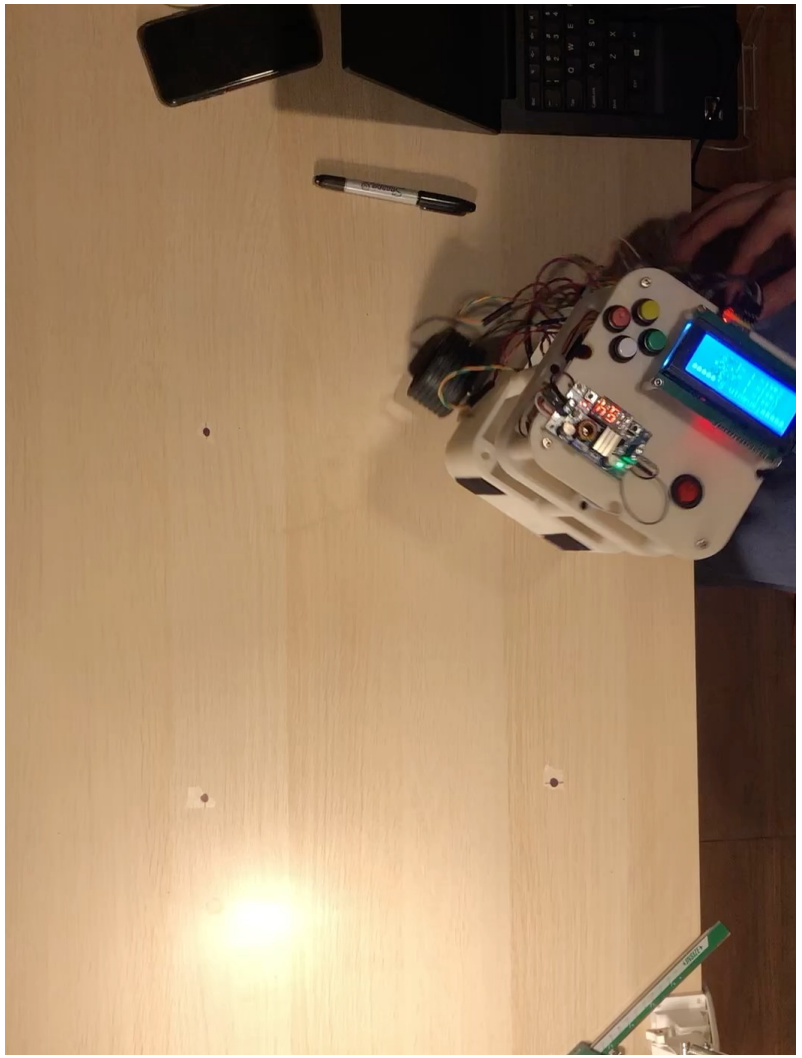
$$Position = \sqrt{\left(X_{measure} - X_{origin}\right)^2 + \left(Y_{measure} - Y_{origin}\right)^2}$$

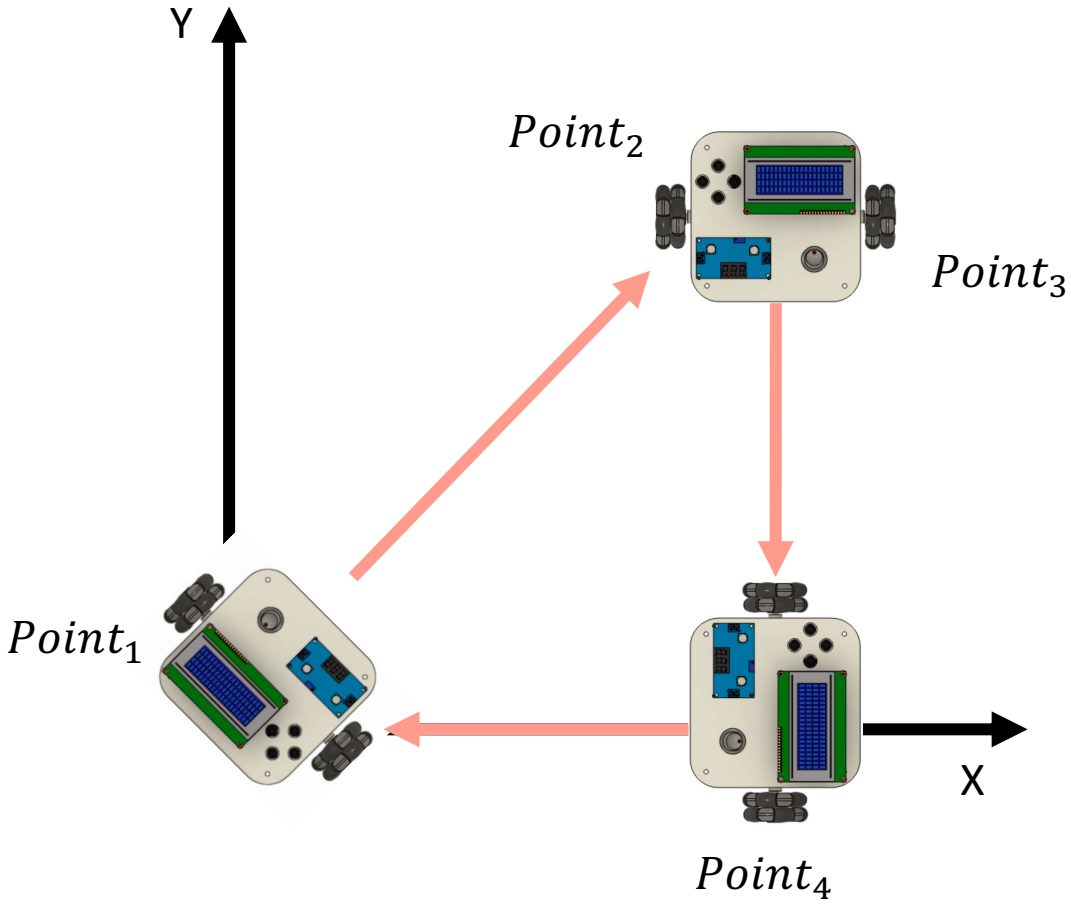
$$Position = \sqrt{(-0.2 - 0)^2 + (27.6 - 0)^2} = 27.6\,mm$$

$$\theta = \tan^{-1}(y/x) = \tan^{-1}\left(-\frac{27.6}{0.2}\right) = -89.58°$$



Triangle drive test





	X	Y
1	-25	25
2	-30	30
3	48	48
4	31	-31
5	42	-42
6	21	21
7	27	27
8	21	-21
9	11	11
10	37	-37

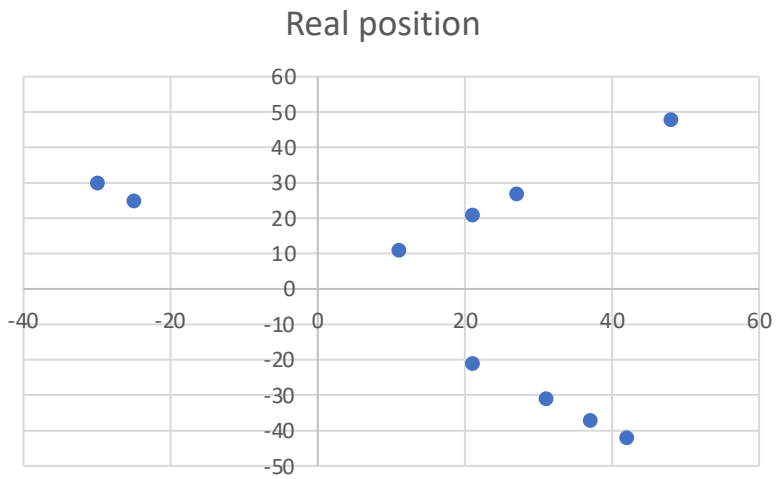
Average : X = 18 , Y = 3.1

SD. : X = 26.47, Y = 32.58

$$Position = \sqrt{\left(X_{measure} - X_{origin}\right)^2 + \left(Y_{measure} - Y_{origin}\right)^2}$$

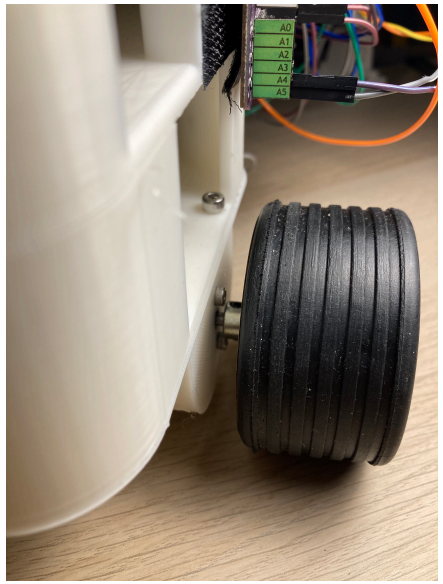
$$Position = \sqrt{(18-0)^2 + (3.1-0)^2} = 18.26\,mm$$

$$\theta = \tan^{-1}(y/x) = \tan^{-1}\left(\frac{3.1}{18}\right) = 9.77°$$

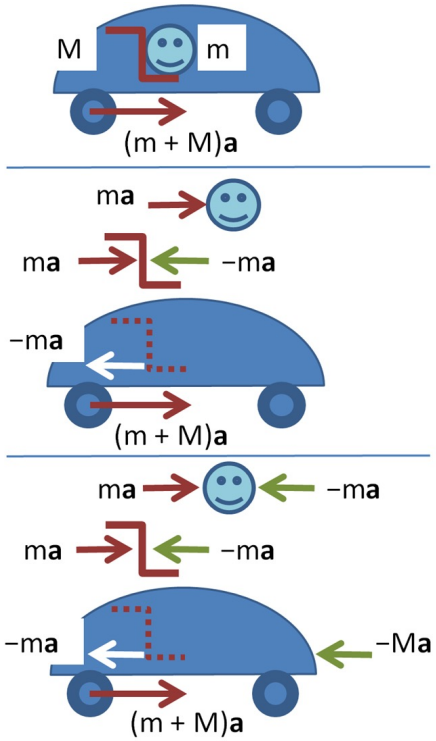


Summarize

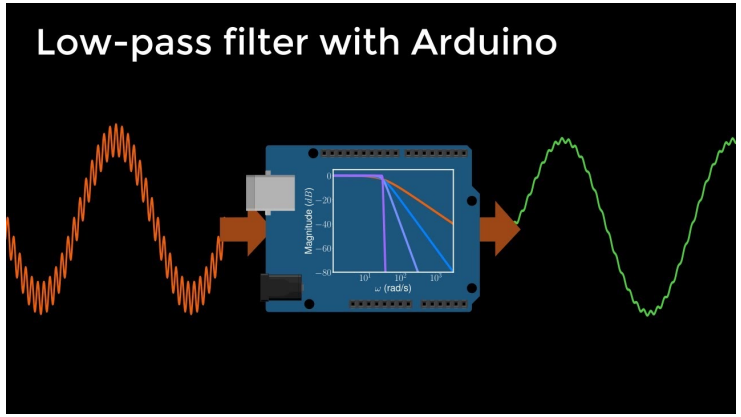
Direct drive
X = -4 mm
Y = 304 mm



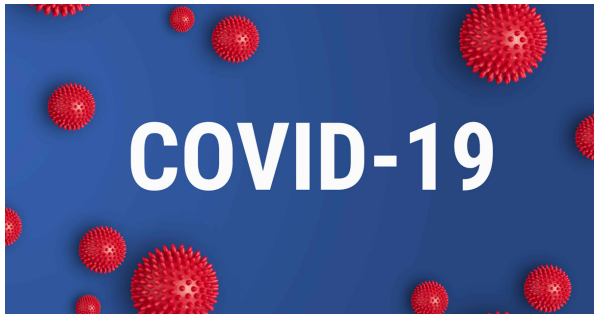
Rectangle drive
X = 0.2 mm
Y = -27.6 mm

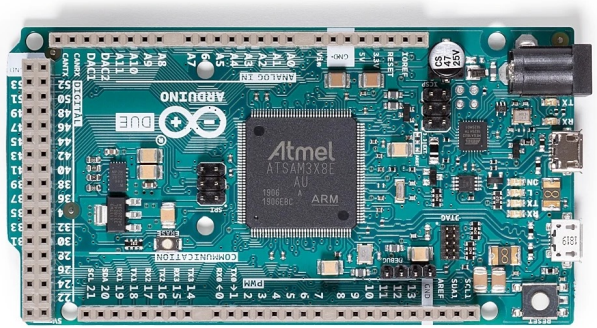


Triangle drive
X = 18 mm
Y = -3.1 mm

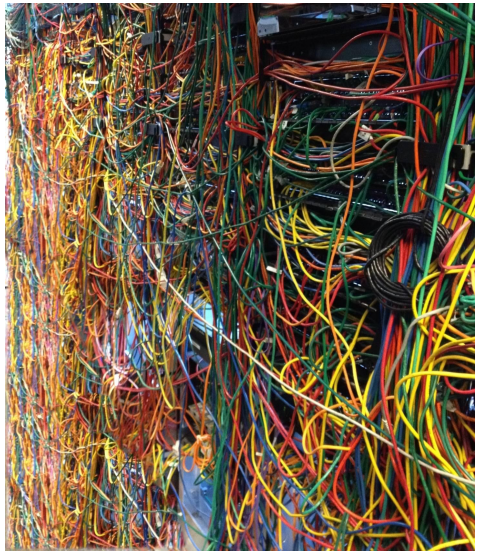


Problem

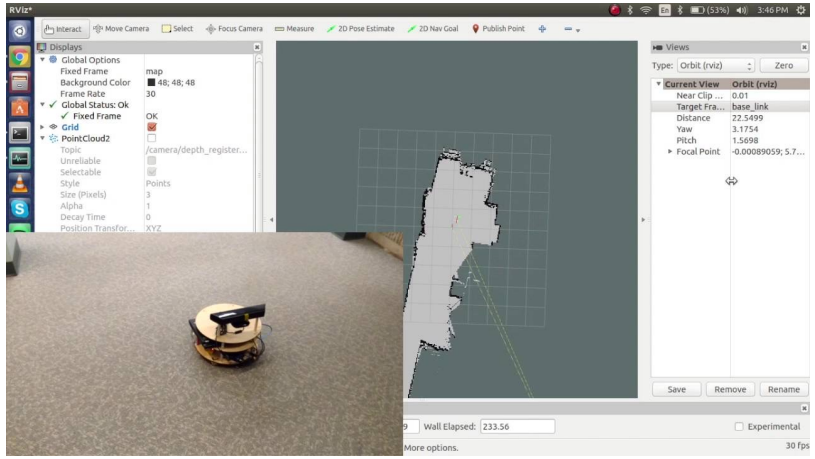


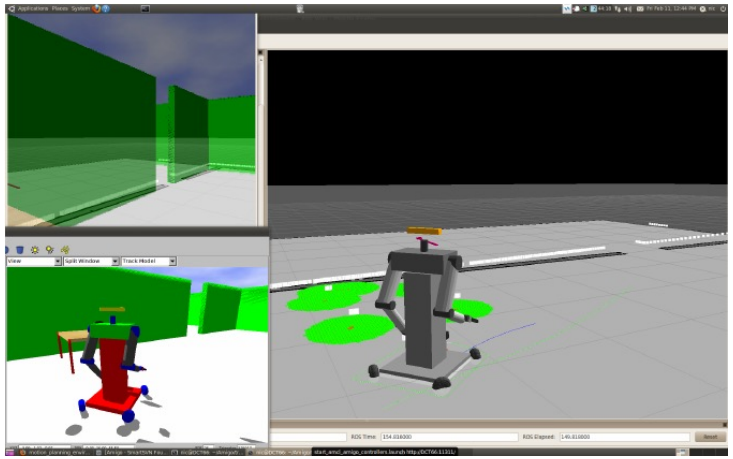


Test field



Applied


RViz*
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
map
Background Color
48; 48; 48
Frame Rate
30
Global Status: Ok
Fixed Frame
OK
Grid
PointCloud2
Topic
/camera/depth_register...
Unreliable
Selectable
Style
Points
Size (Pixels)
3
Alpha
1
Decay Time
0
Position Transfor...
XYZ
Views
Type: Orbit (rviz)
Zero
Current View
Orbit (rviz)
Near Clip ...
0.01
Target Fra...
base_link
Distance
22.5499
Yaw
3.1754
Pitch
1.5698
Focal Point
-0.00089059; 5.7...
Save
Remove
Rename
Wall Elapsed:
233.56
Experimental
More options.
30 fps



Question

Thx