KING MONGKUT’S INSTITUTE OF TECHNOLOGY LATKRABANG
SCHOOL OF ENGINEERING
GROUP OF (**ROBOTICS & AI ENGINEERING)**

FINAL PROJECT REPORT:

LINE FOLLOWER ROBOT
AND
MAKEBLOCK ROBOT WITH GRIPPER AND LAUNCHER

NAME: PYAE, PHATTAWIN, THITIPHAN, PHAWARIS
ID NUMBER: 65011497, 65011453, 65011598, 65011454
GROUP NO: 4

Table of content:

Purpose:

1) To use the C programming language and make the car track the black lines to the destination.
2) To better understand the theories of physics through a practical project and learn how we can apply theories in the real world.
3) To improve the skill of programming, 3D drawing and problem solving through critical thinking.
4) To improve the teamwork ability and build good coordination between team members.
5) To learn how to adjust the limited supply in the most efficient way.

Scope:

1) Line Following robot:

   The Line follower robot is designed to have 3 IR sensors or the front of the car to track the line and the other 2 IR sensors are on the back to check if it should stop now or not.

2) Manual Robot:

   The manual Robot is designed to use the gripper to grab the ball and release it to the catapult then throw it over the obstruction into a specific target.

Background Research:

1) Line Following robot:

   1.1 Must have light and balanced weight for 2 DC motors.

   1.2 The center of mass should be closer to the center of force(the motors).

   1.3 Most of the weight should be in the front of the car for the fast speed and stability but also not too front that the robot loses balance.

   1.4 The IR module detects the distance 2 ~ 30cm, detection angle 35 °, the distance can detect potential is adjusted clockwise adjustment potentiometer, detects the distance increases; counter clockwise adjustment potentiometer, reducing detection distance.

2) Manual Robot

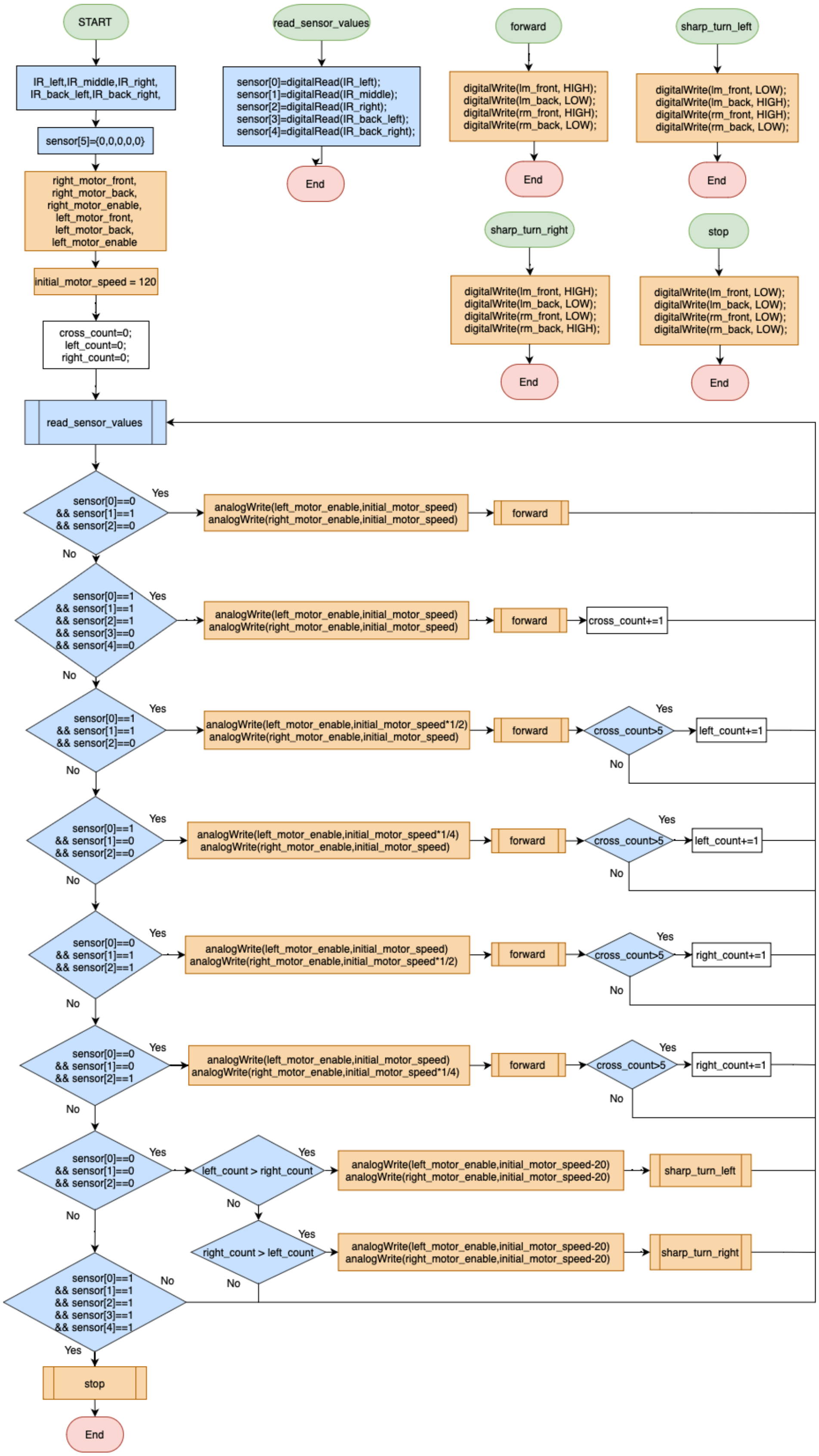
   2.1 The gripper can simply be done with 2 servo motors

   2.2 Motor for the launcher should be high torque and high speed

   2.3 The launching degree should be a little more than 45 degrees for the maximum range

Line Following Robot

1) Flowchart


START
IR_left,IR_middle,IR_right,
IR_back_left,IR_back_right,
sensor[5]={0,0,0,0,0}
right_motor_front,
right_motor_back,
right_motor_enable,
left_motor_front,
left_motor_back,
left_motor_enable
initial_motor_speed = 120
cross_count=0;
left_count=0;
right_count=0;
read_sensor_values
read_sensor_values
sensor[0]=digitalRead(IR_left);
sensor[1]=digitalRead(IR_middle);
sensor[2]=digitalRead(IR_right);
sensor[3]=digitalRead(IR_back_left);
sensor[4]=digitalRead(IR_back_right);
End
forward
digitalWrite(lm_front, HIGH);
digitalWrite(lm_back, LOW);
digitalWrite(rm_front, HIGH);
digitalWrite(rm_back, LOW);
End
sharp_turn_left
digitalWrite(lm_front, LOW);
digitalWrite(lm_back, HIGH);
digitalWrite(rm_front, HIGH);
digitalWrite(rm_back, LOW);
End
sharp_turn_right
digitalWrite(lm_front, HIGH);
digitalWrite(lm_back, LOW);
digitalWrite(rm_front, LOW);
digitalWrite(rm_back, HIGH);
End
stop
digitalWrite(lm_front, LOW);
digitalWrite(lm_back, LOW);
digitalWrite(rm_front, LOW);
digitalWrite(rm_back, LOW);
End
sensor[0]==0
&& sensor[1]==1
&& sensor[2]==0
Yes
No
analogWrite(left_motor_enable,initial_motor_speed)
analogWrite(right_motor_enable,initial_motor_speed)
forward
sensor[0]==1
&& sensor[1]==1
&& sensor[2]==1
&& sensor[3]==0
&& sensor[4]==0
Yes
No
analogWrite(left_motor_enable,initial_motor_speed)
analogWrite(right_motor_enable,initial_motor_speed)
forward
cross_count+=1
sensor[0]==1
&& sensor[1]==1
&& sensor[2]==0
Yes
No
analogWrite(left_motor_enable,initial_motor_speed*1/2)
analogWrite(right_motor_enable,initial_motor_speed)
forward
cross_count>5
Yes
No
left_count+=1
sensor[0]==1
&& sensor[1]==0
&& sensor[2]==0
Yes
No
analogWrite(left_motor_enable,initial_motor_speed*1/4)
analogWrite(right_motor_enable,initial_motor_speed)
forward
cross_count>5
Yes
No
left_count+=1
sensor[0]==0
&& sensor[1]==1
&& sensor[2]==1
Yes
No
analogWrite(left_motor_enable,initial_motor_speed)
analogWrite(right_motor_enable,initial_motor_speed*1/2)
forward
cross_count>5
Yes
No
right_count+=1
sensor[0]==0
&& sensor[1]==0
&& sensor[2]==1
Yes
No
analogWrite(left_motor_enable,initial_motor_speed)
analogWrite(right_motor_enable,initial_motor_speed*1/4)
forward
cross_count>5
Yes
No
right_count+=1
sensor[0]==0
&& sensor[1]==0
&& sensor[2]==0
Yes
No
left_count > right_count
Yes
No
analogWrite(left_motor_enable,initial_motor_speed-20)
analogWrite(right_motor_enable,initial_motor_speed-20)
sharp_turn_left
right_count > left_count
Yes
No
analogWrite(left_motor_enable,initial_motor_speed-20)
analogWrite(right_motor_enable,initial_motor_speed-20)
sharp_turn_right
sensor[0]==1
&& sensor[1]==1
&& sensor[2]==1
&& sensor[3]==1
&& sensor[4]==1
No
Yes
stop
End

2) Coding

In auto robot (Line follower) coding, we use -

1) Flowchart
2) Variables and Data types
3) Conditions and Control flow
4) Expression (if else statement)
5) Array
6) functions and
7) Arduino IDE

```
//Line follower code
//Written by PYAE HTOO KHANT
//RAI_5 Group_4
//10/12/2022

//define sensor pins
int IR_left = 5;
int IR_middle = 4;
int IR_right= 3;
int IR_back_left = 12;
int IR_back_right = 2;

//initiate sensor values
int sensor[5]={0,0,0,0,0};

//define motor pins
int rm_front = 8;
int rm_back = 7;
int rm_enable = 6;
int lm_front = 9;
int lm_back = 10;
int lm_enable = 11;

//Initial speed of motor
int initial_motor_speed = 120;
int left_count=0; //to decide whether to turn left at the final
stage of arena
int right_count=0; //to decide whether to turn right at the final
stage of arena
int cross_count=0; //to count the cross lines

void setup() {
 // put your setup code here, to run once:
 Serial.begin(9600);
 //sensor input
 pinMode(IR_left,INPUT);
 pinMode(IR_middle,INPUT);
 pinMode(IR_right,INPUT);
 pinMode(IR_back_left,INPUT);
 pinMode(IR_back_right,INPUT);

// right wheel
 pinMode(rm_front, OUTPUT);
 pinMode(rm_back, OUTPUT);
 pinMode(rm_enable, OUTPUT);

// left wheel
 pinMode(lm_front, OUTPUT);
 pinMode(lm_back, OUTPUT);
 pinMode(lm_enable, OUTPUT);

}

void loop() {
 // put your main code here, to run repeatedly:
read_sensor_values(); //get sensor values
```

```
 if(sensor[0]==0 && sensor[1]==1 && sensor[2]==0){
   analogWrite(lm_enable,initial_motor_speed);
   analogWrite(rm_enable,initial_motor_speed);
   forward();
   delay(20);// go straight for 0.02 s
 }
 else if(sensor[0]==1 && sensor[1]==1 && sensor[2]==0){
   analogWrite(lm_enable,initial_motor_speed*2/4);
   analogWrite(rm_enable,initial_motor_speed);
   forward();
    delay(20);//tilt to right == turn left by reducing the speed of left motor
   // if (cross_count>=5){
    //    left_count+=1; //After passing 5 cross line in arena, count as left if it goes to left
   // }
 }
 else if(sensor[0]==1 && sensor[1]==0 && sensor[2]==0){
   analogWrite(lm_enable,initial_motor_speed*1/4);
   analogWrite(rm_enable,initial_motor_speed);
   forward();
    delay(20);//more tilt to right == turn more left by reducing the speed of left motor
   if (cross_count>=5){
      left_count+=1; //After passing 5 cross line in arena, count as left if it goes to left
   }
 }
 else if(sensor[0]==0 && sensor[1]==1 && sensor[2]==1){
   analogWrite(lm_enable,initial_motor_speed);
   analogWrite(rm_enable,initial_motor_speed*2/4);
   forward();
```

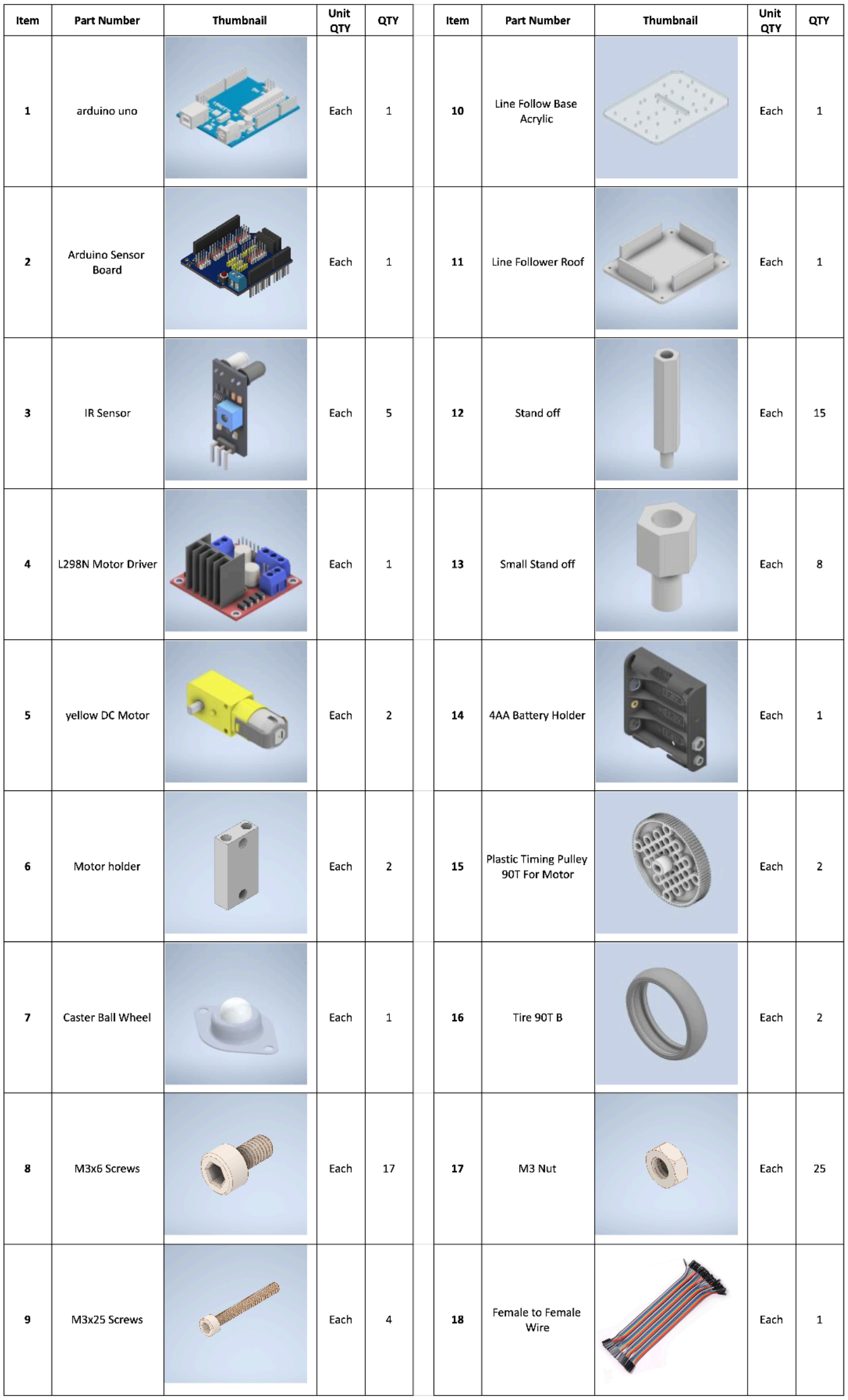
```
    delay(20);//tilt to left == turn right by reducing the speed of right motor
   // if (cross_count>=5){
    //    right_count+=1; //After passing 5 cross line in arena, count as right if it goes to right
   // }
 }
 else if(sensor[0]==0 && sensor[1]==0 && sensor[2]==1){
   analogWrite(lm_enable,initial_motor_speed);
   analogWrite(rm_enable,initial_motor_speed*1/4);
   forward();
    delay(20);//more tilt to left == turn more right by reducing the speed of right motor
   if (cross_count>=5){
     right_count+=1; //After passing 5 cross line in arena, count as right if it goes to right
   }
 }
  else if(sensor[0]==1 && sensor[1]==1 && sensor[2]==1 && sensor[3]==0 && sensor[4]==0){
   analogWrite(lm_enable,initial_motor_speed);
   analogWrite(rm_enable,initial_motor_speed);
   forward();
   delay(20);//pass cross line == just go straight for 0.02 s
   cross_count+=1; //count cross line
 }
  else if(sensor[0]==0 && sensor[1]==0 && sensor[2]==0 && left_count>right_count){ //Check whether to turn left at the last junction of arena
      analogWrite(lm_enable,initial_motor_speed-20); //reduce a little more speed at turning for smooth
      analogWrite(rm_enable,initial_motor_speed-20); //reduce a little more speed at turning for smooth
    sharpturnleft(); //turn left fast
    delay(20);
  }
   else  if(sensor[0]==0  &&  sensor[1]==0  &&  sensor[2]==0  &&
right_count>left_count){  //Check  whether  to  turn  right  at  the
last junction of arena
       analogWrite(lm_enable,initial_motor_speed-20);  //reduce  a
little more speed at turning for smooth
       analogWrite(rm_enable,initial_motor_speed-20);  //reduce  a
little more speed at turning for smooth
    sharpturnright(); //turn right fast
    delay(20);
  }
     else  if(sensor[0]==1  &&  sensor[1]==1  &&  sensor[2]==1  &&
sensor[3]==1 || sensor[4]==1){
    stop();

  }
}
void read_sensor_values() //Read IR sensors values //input
{
 sensor[0]=digitalRead(IR_left);
 sensor[1]=digitalRead(IR_middle);
 sensor[2]=digitalRead(IR_right);
 sensor[3]=digitalRead(IR_back_left);
 sensor[4]=digitalRead(IR_back_right);
}
void forward (){ //motor output
 digitalWrite(lm_front, HIGH);
 digitalWrite(lm_back, LOW);
 digitalWrite(rm_front, HIGH);
 digitalWrite(rm_back, LOW);
}
```

```
void sharpturnleft (){ //motor output
 digitalWrite(lm_front, LOW);
 digitalWrite(lm_back, HIGH);
 digitalWrite(rm_front, HIGH);
 digitalWrite(rm_back, LOW);
}
void sharpturnright(){ //motor output
 digitalWrite(lm_front, HIGH);
 digitalWrite(lm_back, LOW);
 digitalWrite(rm_front, LOW);
 digitalWrite(rm_back, HIGH);
}
void stop(){ //motor output
 digitalWrite(lm_front, LOW);
 digitalWrite(lm_back, LOW);
 digitalWrite(rm_front, LOW);
 digitalWrite(rm_back, LOW);
}
```

3) Construction of robot

3.1 Apparatus

The apparatus we use for our line follower robot are as follows.



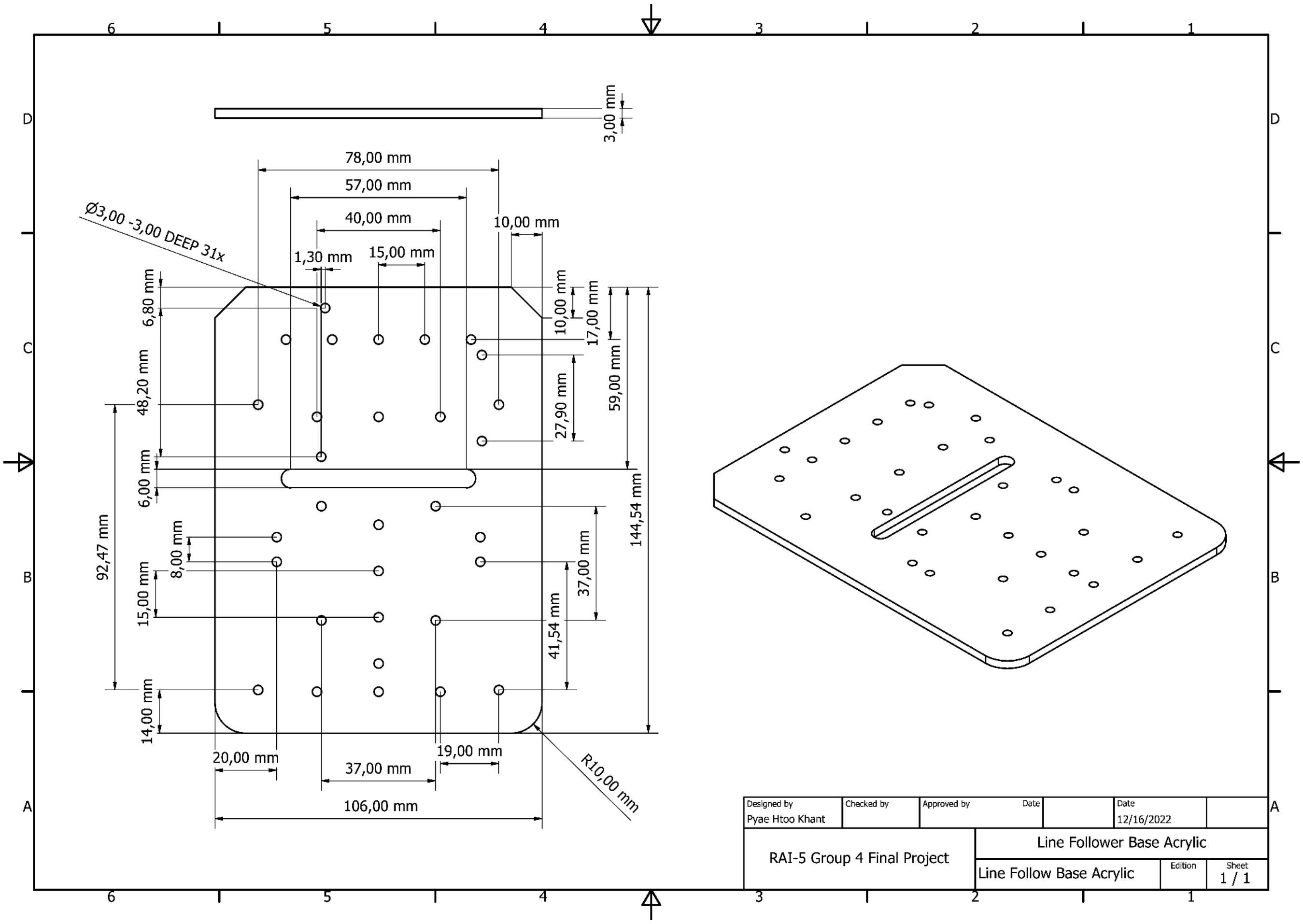
Item	Part Number	Thumbnail	Unit QTY	QTY
1	arduino uno		Each	1
2	Arduino Sensor Board		Each	1
3	IR Sensor		Each	5
4	L298N Motor Driver		Each	1
5	yellow DC Motor		Each	2
6	Motor holder		Each	2
7	Caster Ball Wheel		Each	1
8	M3x6 Screws		Each	17
9	M3x25 Screws		Each	4
10	Line Follow Base Acrylic		Each	1
11	Line Follower Roof		Each	1
12	Stand off		Each	15
13	Small Stand off		Each	8
14	4AA Battery Holder		Each	1
15	Plastic Timing Pulley 90T For Motor		Each	2
16	Tire 90T B		Each	2
17	M3 Nut		Each	25
18	Female to Female Wire		Each	1

We use one press-button switch that is connected to the battery and L298N motor driver.

3.2 Dimension and Drawing

According to the size requirement, we make sure our robot is within the size of 16x16 cm. We measured all the dimensions of electronics components and screw holes to place them on one single acrylic base. The acrylic plate is 3 mm thick and we laser cut out the design on it.

- The dimensions of the acrylic base is as follows.



Designed by	Checked by	Approved by	Date	Date
Pyae Htoo Khant				12/16/2022

RAI-5 Group 4 Final Project

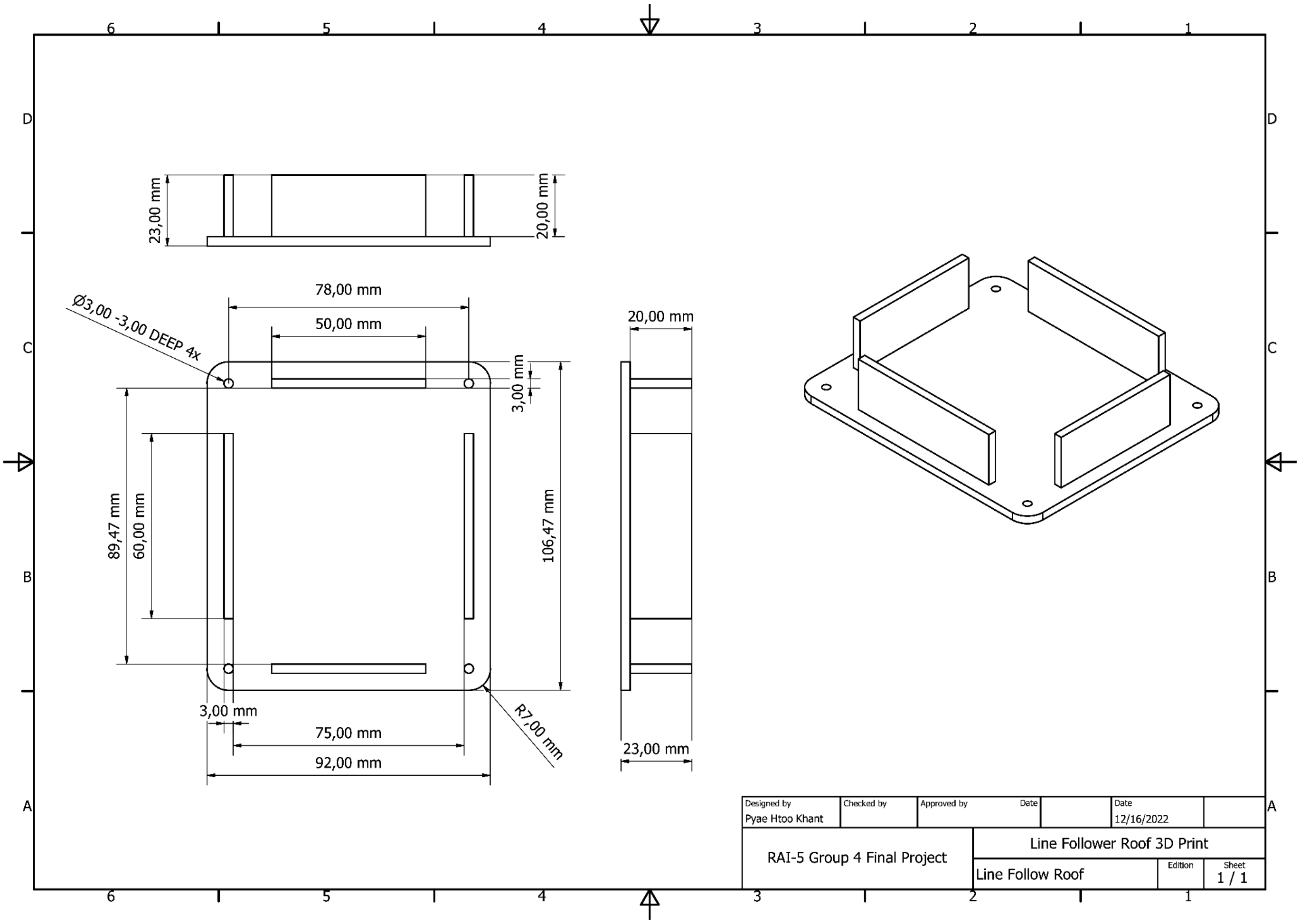
Line Follower Base Acrylic

Line Follow Base Acrylic

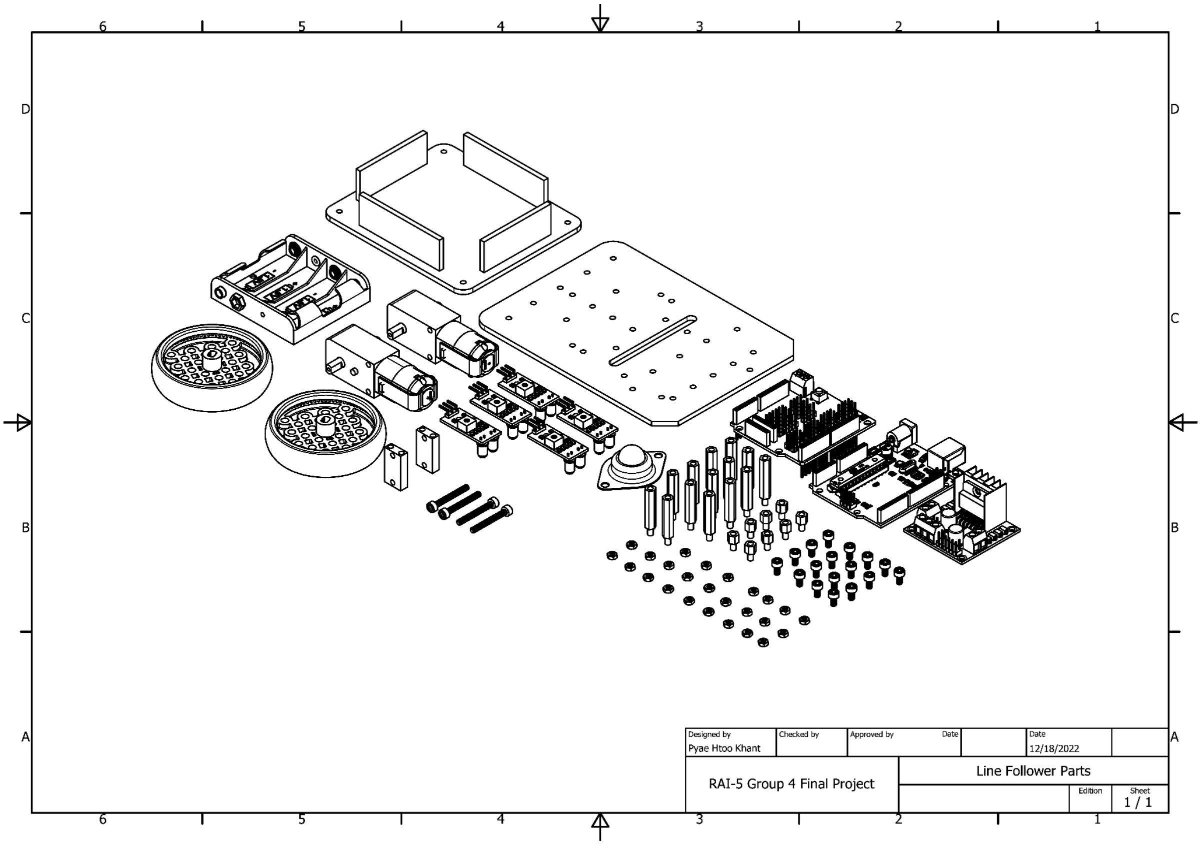
Edition

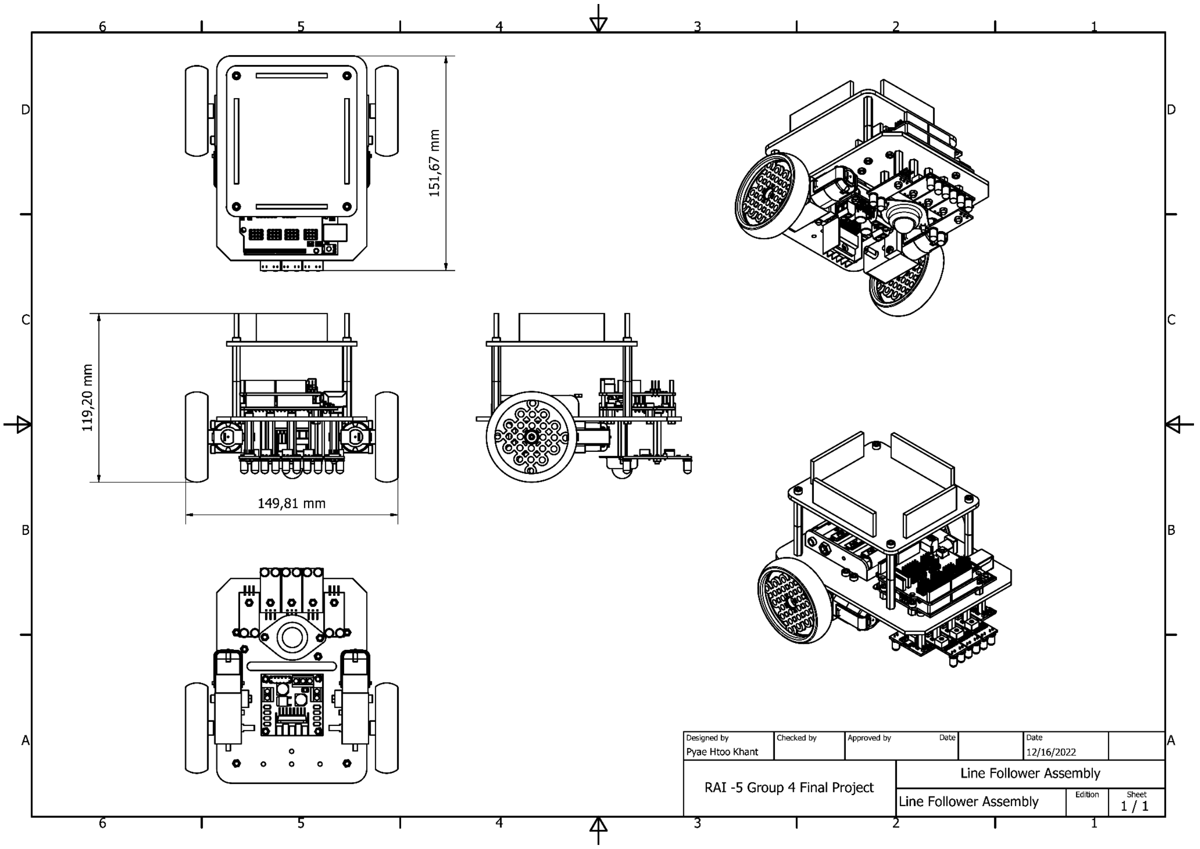
Sheet 1 / 1

- To carry the items (7 cm cube) by the robot, we 3D printed the bracket shaped roof and mounted it with the standoffs.



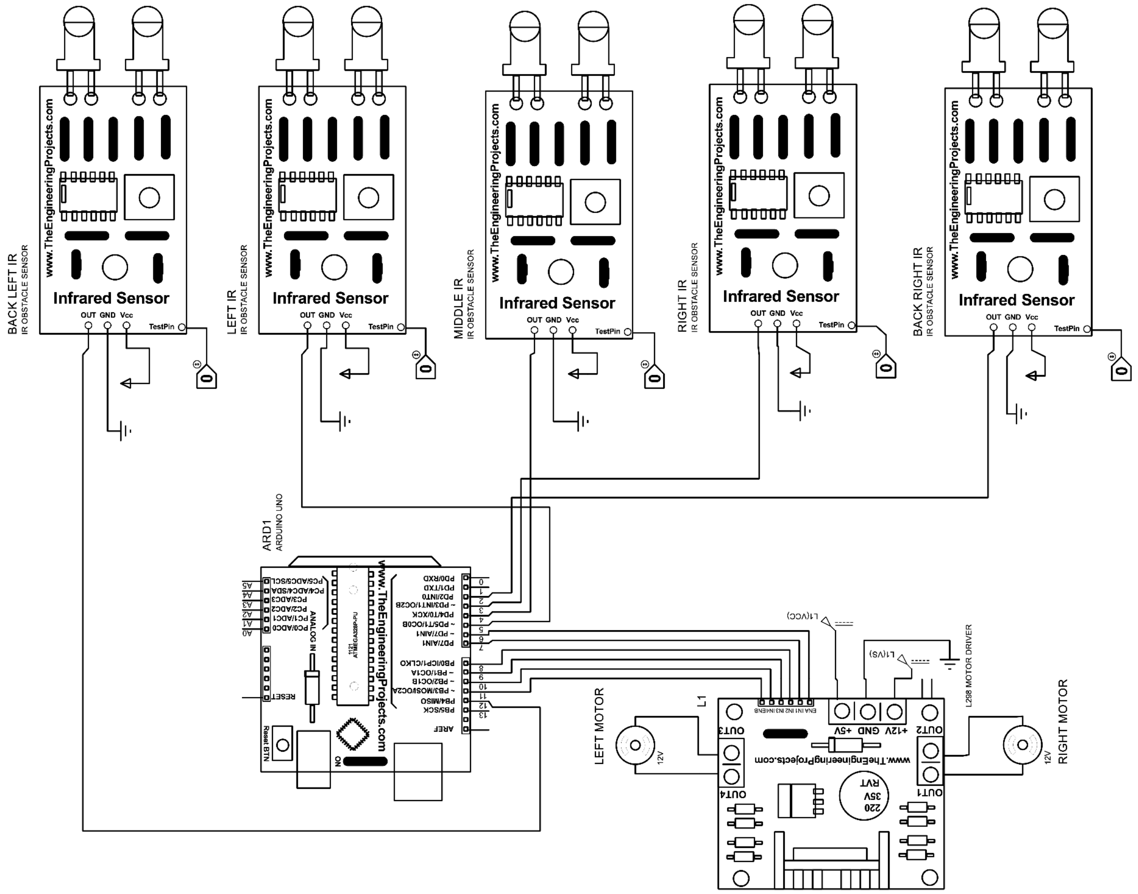
- This is the final design of the line follower robot.



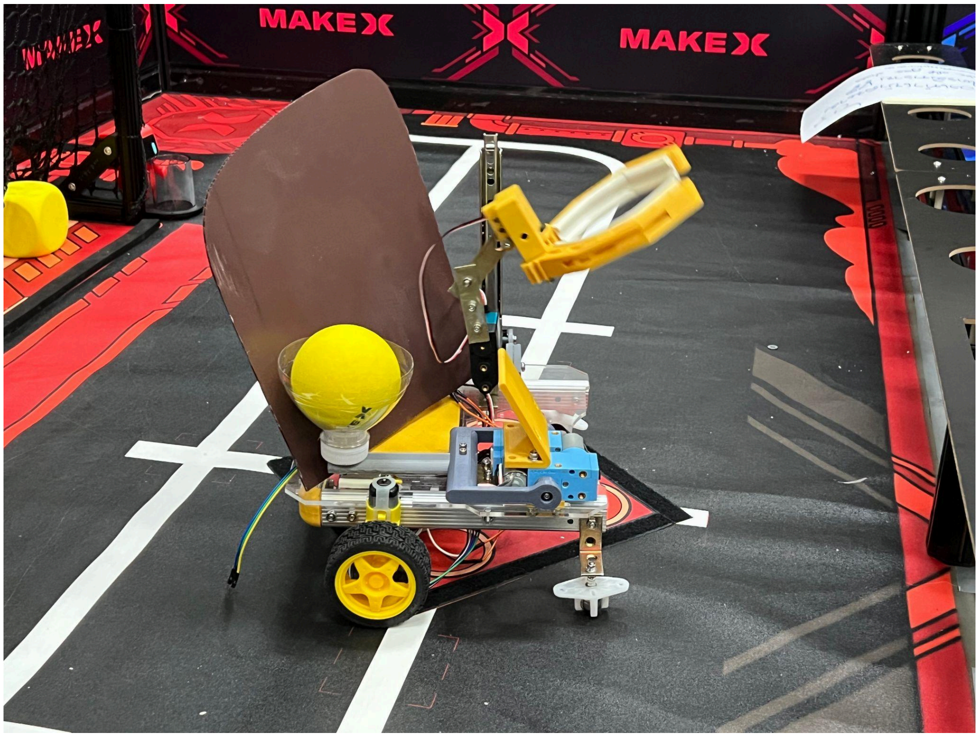


3) Electronic system

We use Arduino Uno, Sensor Shield, L298N motor driver, 5 IR sensor modules and 3 motors. The connection of the electronic parts are as follows.

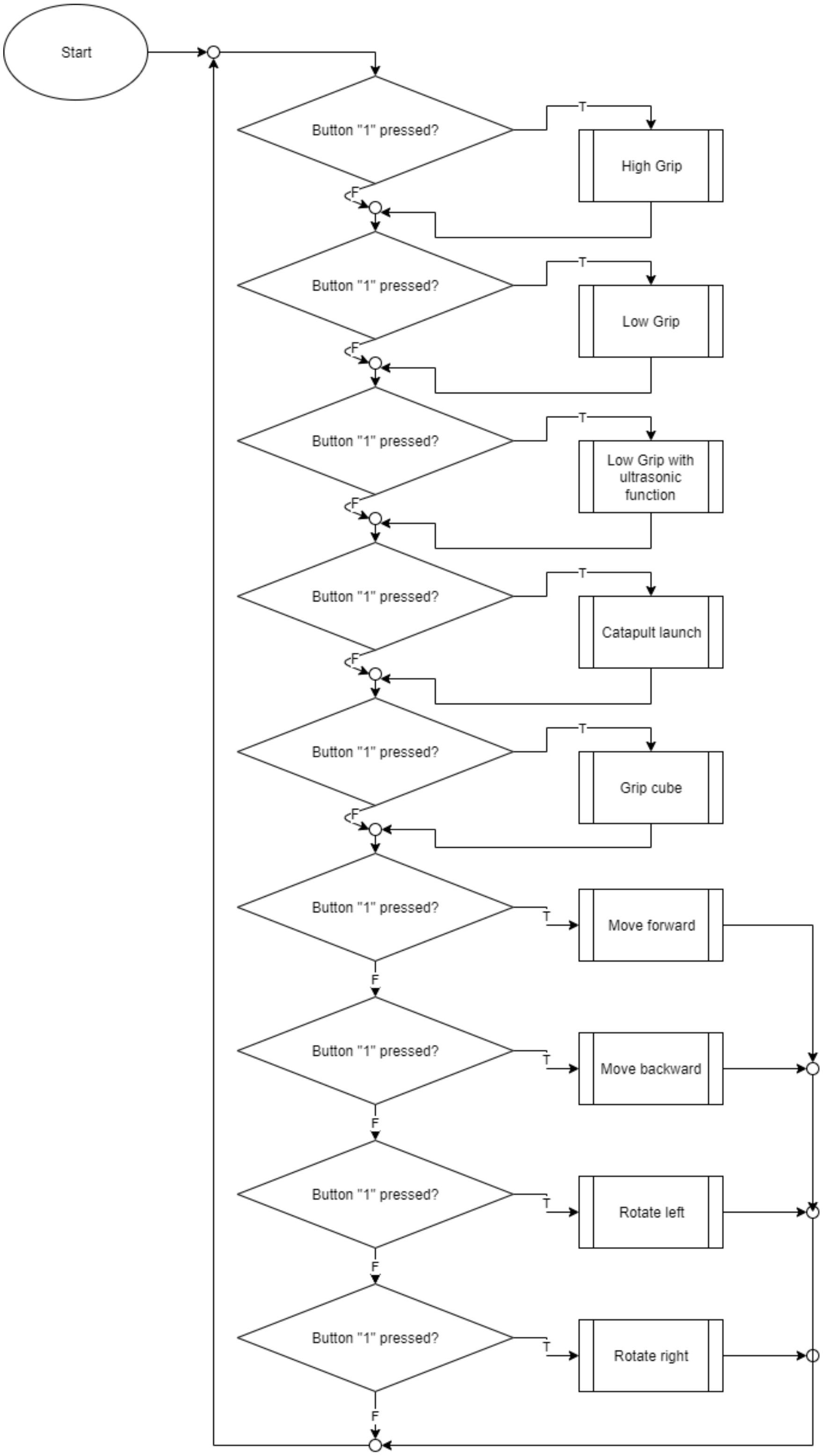


Manual robot

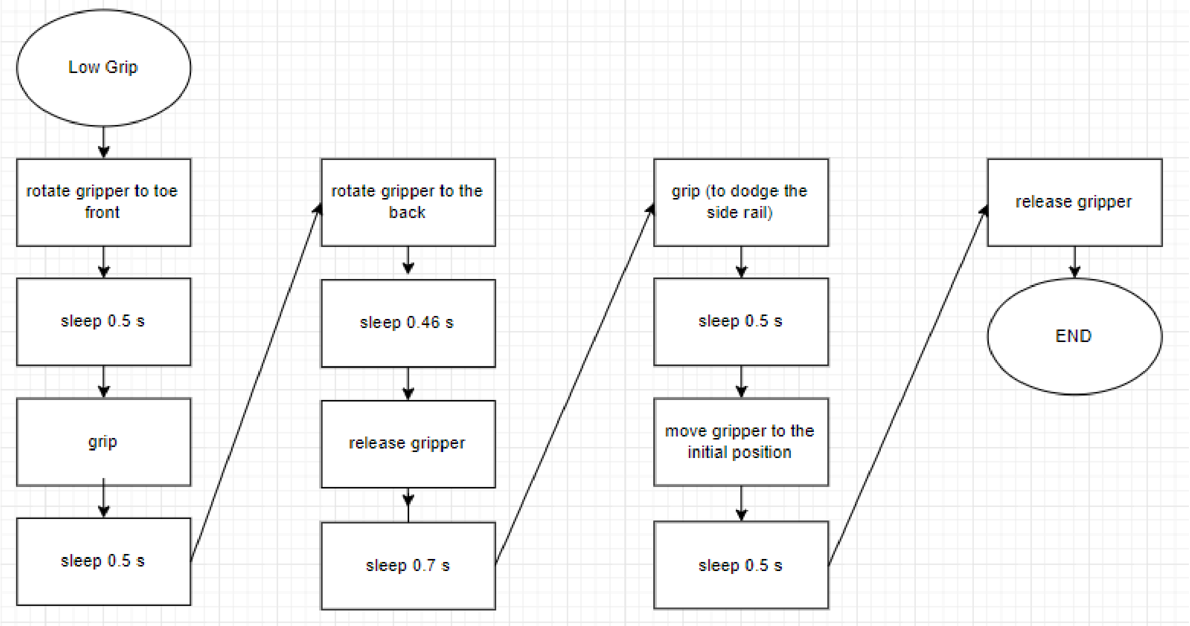


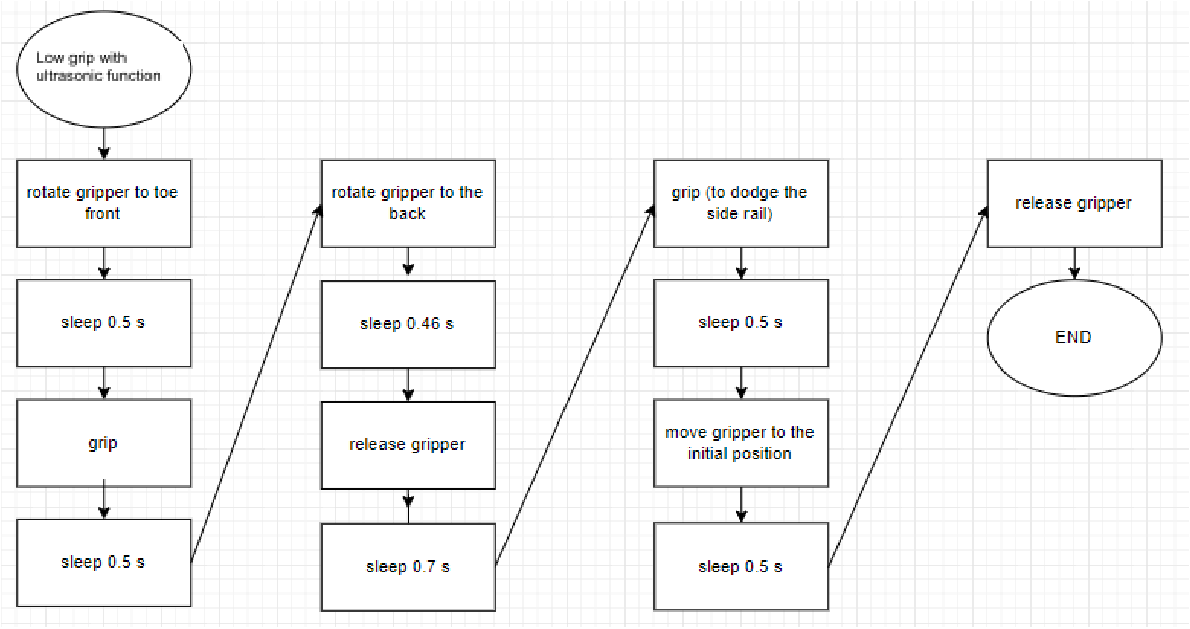
1) Flowchart

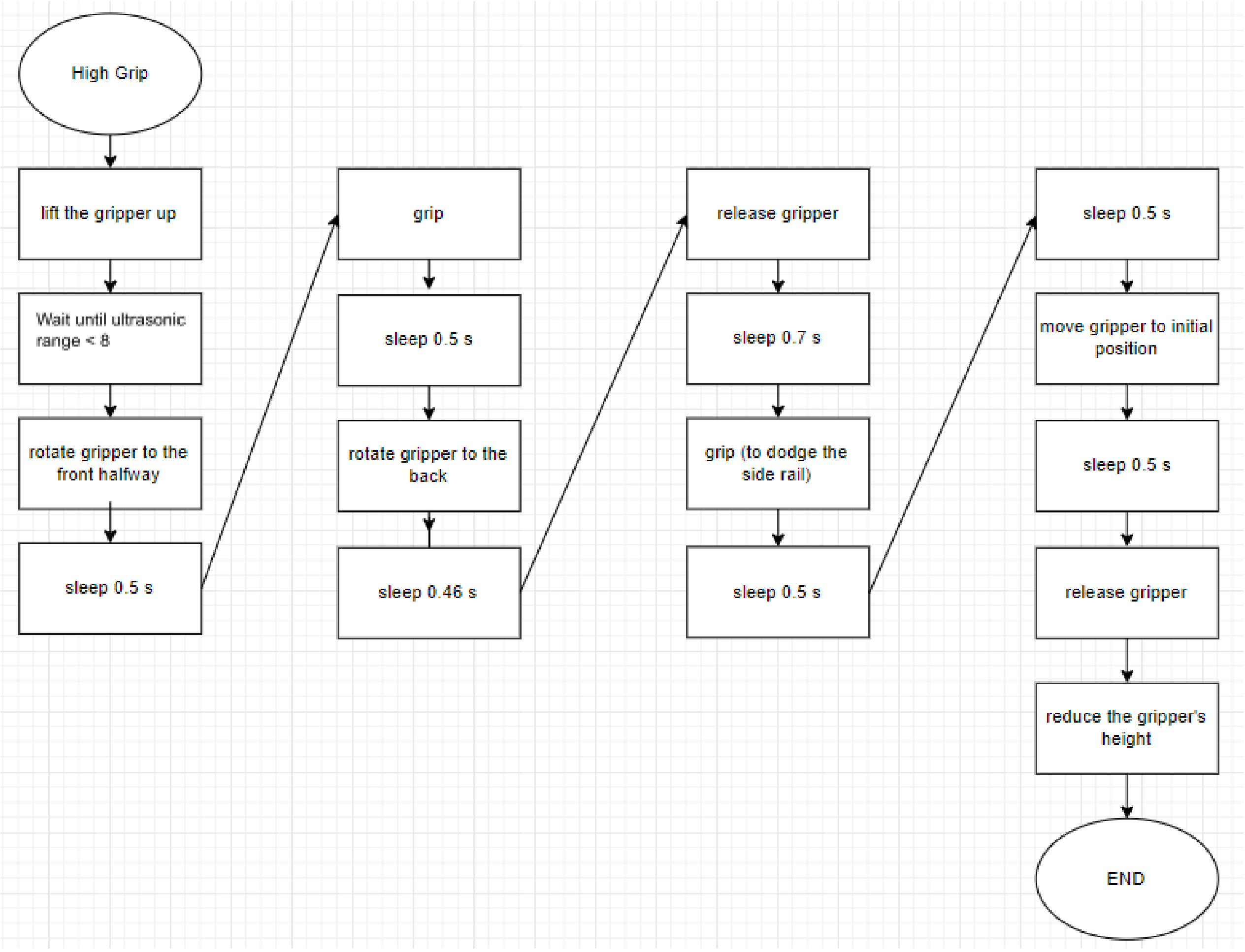
Main function

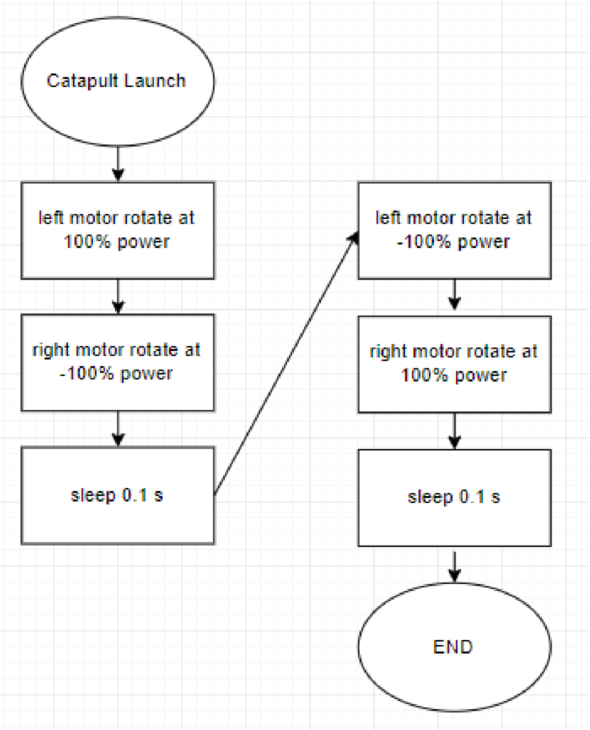


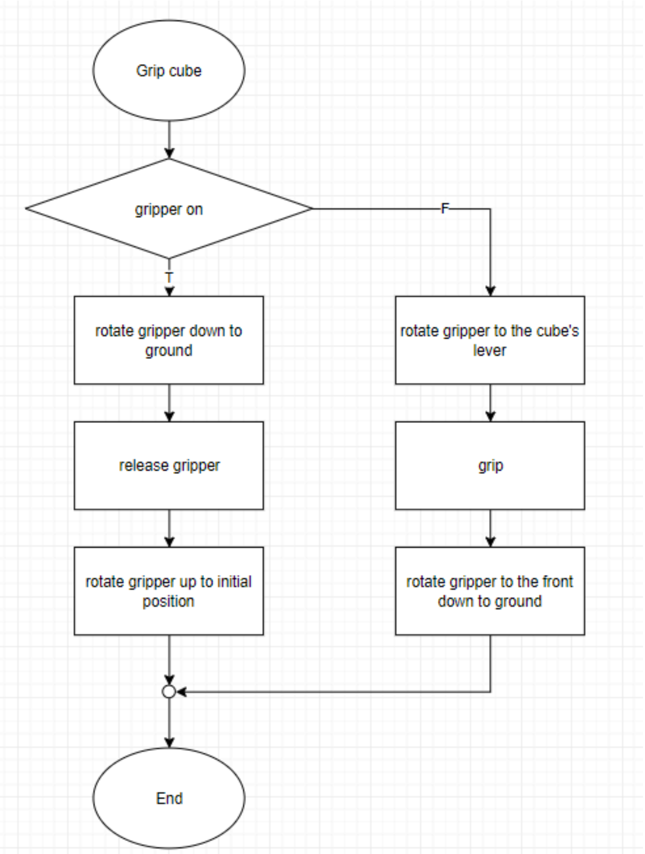
Other functions:

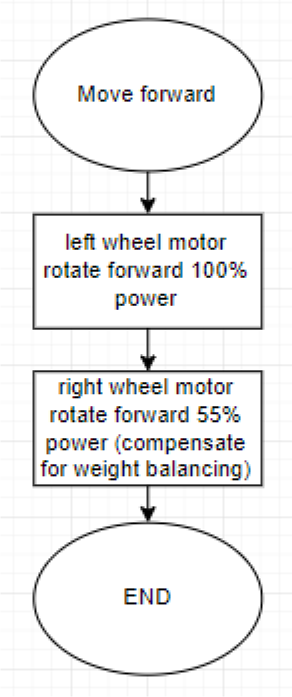

Low Grip
rotate gripper to toe front
sleep 0.5 s
grip
sleep 0.5 s
rotate gripper to the back
sleep 0.46 s
release gripper
sleep 0.7 s
grip (to dodge the side rail)
sleep 0.5 s
move gripper to the initial position
sleep 0.5 s
release gripper
END

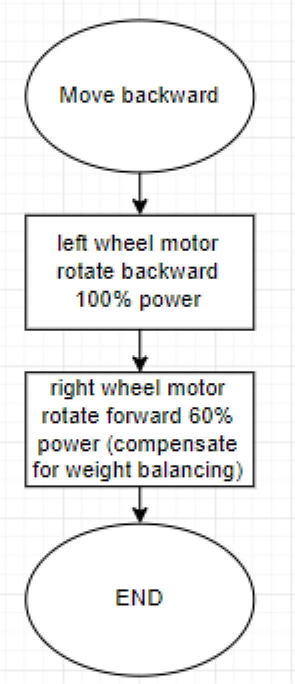

Low grip with ultrasonic function
rotate gripper to toe front
sleep 0.5 s
grip
sleep 0.5 s
rotate gripper to the back
sleep 0.46 s
release gripper
sleep 0.7 s
grip (to dodge the side rail)
sleep 0.5 s
move gripper to the initial position
sleep 0.5 s
release gripper
END

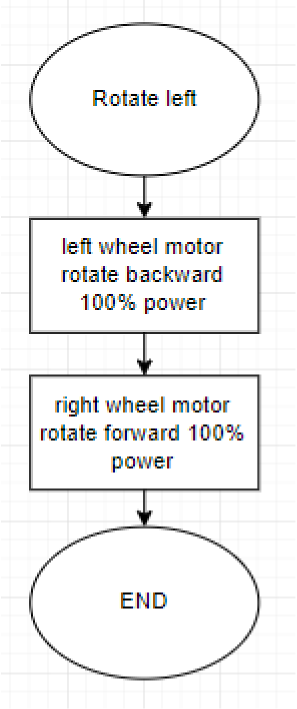

High Grip
lift the gripper up
Wait until ultrasonic range < 8
rotate gripper to the front halfway
sleep 0.5 s
grip
sleep 0.5 s
rotate gripper to the back
sleep 0.46 s
release gripper
sleep 0.7 s
grip (to dodge the side rail)
sleep 0.5 s
sleep 0.5 s
move gripper to initial position
sleep 0.5 s
release gripper
reduce the gripper's height
END

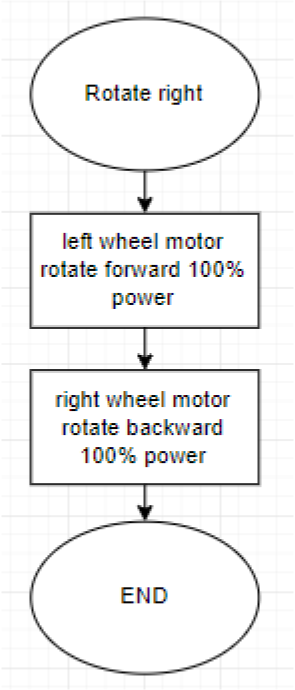

Catapult Launch
left motor rotate at 100% power
right motor rotate at -100% power
sleep 0.1 s
left motor rotate at -100% power
right motor rotate at 100% power
sleep 0.1 s
END


Grip cube
gripper on
F
T
rotate gripper down to ground
release gripper
rotate gripper up to initial position
rotate gripper to the cube's lever
grip
rotate gripper to the front down to ground
End

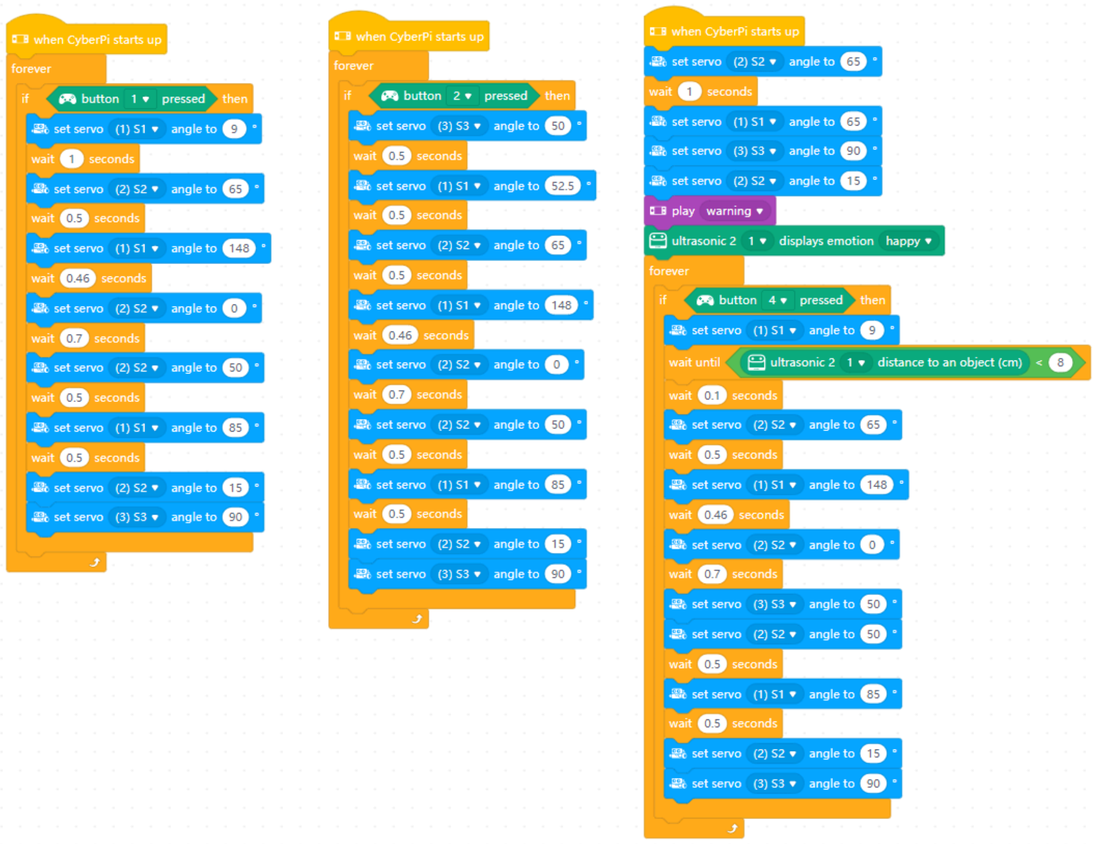

Move forward
left wheel motor rotate forward 100% power
right wheel motor rotate forward 55% power (compensate for weight balancing)
END


Move backward
left wheel motor rotate backward 100% power
right wheel motor rotate forward 60% power (compensate for weight balancing)
END


Rotate left
left wheel motor rotate backward 100% power
right wheel motor rotate forward 100% power
END


Rotate right
left wheel motor rotate forward 100% power
right wheel motor rotate backward 100% power
END

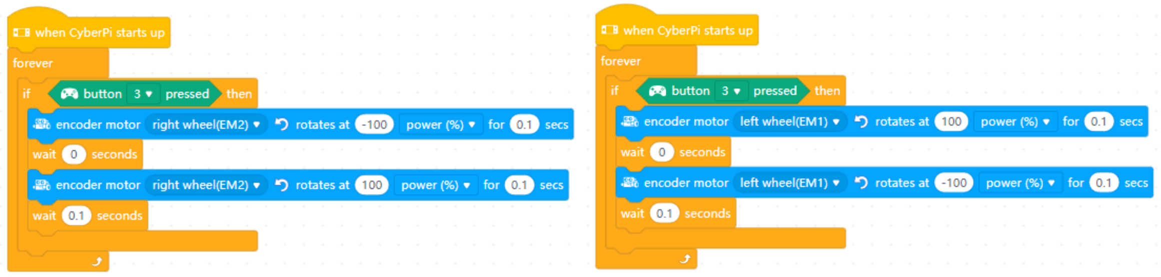
1) Coding



```
when CyberPi starts up
forever
    if button 1 pressed then
        set servo (1) S1 angle to 9 °
        wait 1 seconds
        set servo (2) S2 angle to 65 °
        wait 0.5 seconds
        set servo (1) S1 angle to 148 °
        wait 0.46 seconds
        set servo (2) S2 angle to 0 °
        wait 0.7 seconds
        set servo (2) S2 angle to 50 °
        wait 0.5 seconds
        set servo (1) S1 angle to 85 °
        wait 0.5 seconds
        set servo (2) S2 angle to 15 °
        set servo (3) S3 angle to 90 °
```

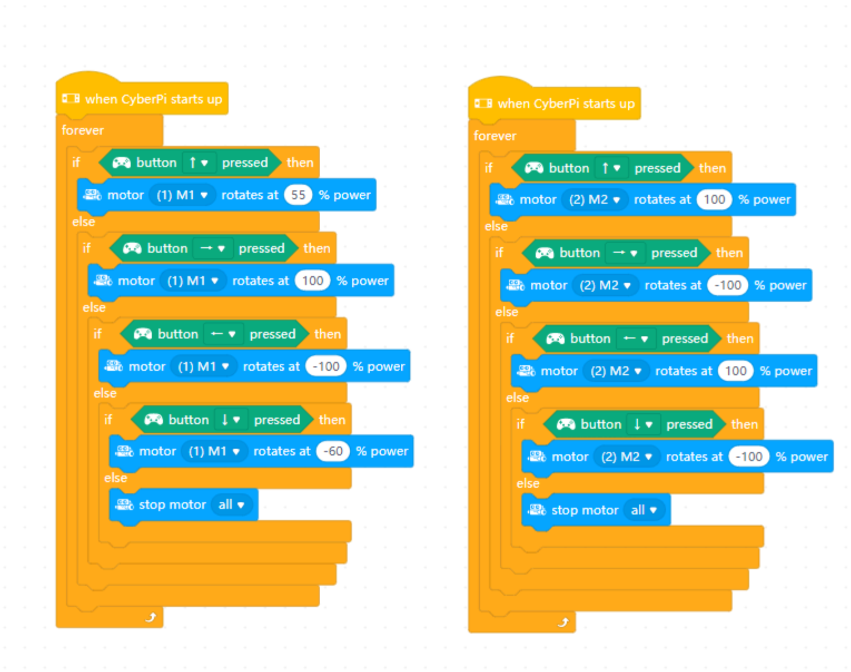
```
when CyberPi starts up
forever
    if button 2 pressed then
        set servo (3) S3 angle to 50 °
        wait 0.5 seconds
        set servo (1) S1 angle to 52.5 °
        wait 0.5 seconds
        set servo (2) S2 angle to 65 °
        wait 0.5 seconds
        set servo (1) S1 angle to 148 °
        wait 0.46 seconds
        set servo (2) S2 angle to 0 °
        wait 0.7 seconds
        set servo (2) S2 angle to 50 °
        wait 0.5 seconds
        set servo (1) S1 angle to 85 °
        wait 0.5 seconds
        set servo (2) S2 angle to 15 °
        set servo (3) S3 angle to 90 °
```

```
when CyberPi starts up
set servo (2) S2 angle to 65 °
wait 1 seconds
set servo (1) S1 angle to 65 °
set servo (3) S3 angle to 90 °
set servo (2) S2 angle to 15 °
play warning
ultrasonic 2 1 displays emotion happy
forever
    if button 4 pressed then
        set servo (1) S1 angle to 9 °
        wait until ultrasonic 2 1 distance to an object (cm) < 8
        wait 0.1 seconds
        set servo (2) S2 angle to 65 °
        wait 0.5 seconds
        set servo (1) S1 angle to 148 °
        wait 0.46 seconds
        set servo (2) S2 angle to 0 °
        wait 0.7 seconds
        set servo (3) S3 angle to 50 °
        set servo (2) S2 angle to 50 °
        wait 0.5 seconds
        set servo (1) S1 angle to 85 °
        wait 0.5 seconds
        set servo (2) S2 angle to 15 °
        set servo (3) S3 angle to 90 °
```



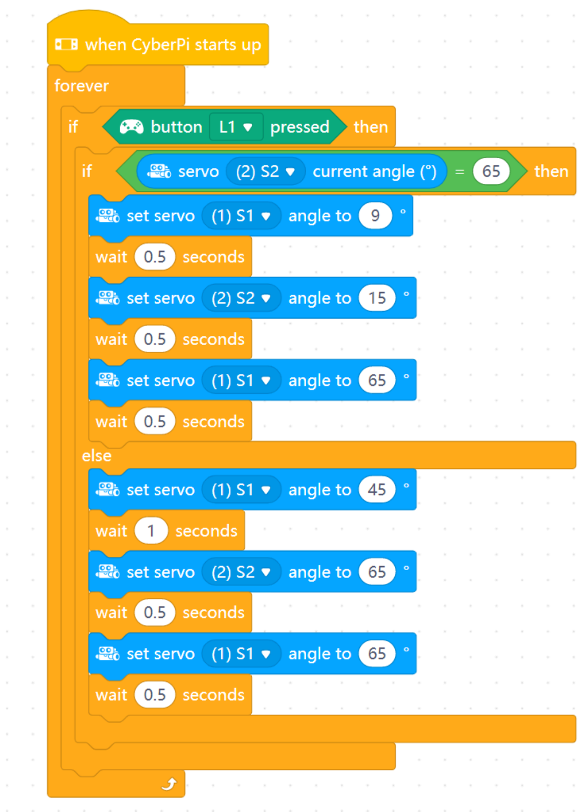
```
when CyberPi starts up
forever
    if button 3 pressed then
        encoder motor right wheel(EM2) rotates at -100 power (%) for 0.1 secs
        wait 0 seconds
        encoder motor right wheel(EM2) rotates at 100 power (%) for 0.1 secs
        wait 0.1 seconds
```

```
when CyberPi starts up
forever
    if button 3 pressed then
        encoder motor left wheel(EM1) rotates at 100 power (%) for 0.1 secs
        wait 0 seconds
        encoder motor left wheel(EM1) rotates at -100 power (%) for 0.1 secs
        wait 0.1 seconds
```



```
when CyberPi starts up
forever
    if button ↑ pressed then
        motor (1) M1 rotates at 55 % power
    else
        if button → pressed then
            motor (1) M1 rotates at 100 % power
        else
            if button ← pressed then
                motor (1) M1 rotates at -100 % power
            else
                if button ↓ pressed then
                    motor (1) M1 rotates at -60 % power
                else
                    stop motor all
```

```
when CyberPi starts up
forever
    if button ↑ pressed then
        motor (2) M2 rotates at 100 % power
    else
        if button → pressed then
            motor (2) M2 rotates at -100 % power
        else
            if button ← pressed then
                motor (2) M2 rotates at 100 % power
            else
                if button ↓ pressed then
                    motor (2) M2 rotates at -100 % power
                else
                    stop motor all
```



```
when CyberPi starts up
forever
    if button L1 pressed then
        if servo (2) S2 current angle (°) = 65 then
            set servo (1) S1 angle to 9 °
            wait 0.5 seconds
            set servo (2) S2 angle to 15 °
            wait 0.5 seconds
            set servo (1) S1 angle to 65 °
            wait 0.5 seconds
        else
            set servo (1) S1 angle to 45 °
            wait 1 seconds
            set servo (2) S2 angle to 65 °
            wait 0.5 seconds
            set servo (1) S1 angle to 65 °
            wait 0.5 seconds
```

```
import gamepad, mbot2, event, time, cyberpi, mbuild

@event.start
def on_start():
   while True:
       if gamepad.is_key_pressed('N1'):
           mbot2.servo_set(9,"S1")
           time.sleep(1)
           mbot2.servo_set(65,"S2")
           time.sleep(0.5)
           mbot2.servo_set(148,"S1")
           time.sleep(0.46)
           mbot2.servo_set(0,"S2")
           time.sleep(0.7)
           mbot2.servo_set(50,"S2")
           time.sleep(0.5)
           mbot2.servo_set(85,"S1")
           time.sleep(0.5)
           mbot2.servo_set(15,"S2")
           mbot2.servo_set(90,"S3")

@event.start
def on_start1():
   while True:
       if gamepad.is_key_pressed('N2'):
           mbot2.servo_set(50,"S3")
           time.sleep(0.5)
           mbot2.servo_set(52.5,"S1")
           time.sleep(0.5)
           mbot2.servo_set(65,"S2")
           time.sleep(0.5)
           mbot2.servo_set(148,"S1")
           time.sleep(0.46)
           mbot2.servo_set(0,"S2")
           time.sleep(0.7)
           mbot2.servo_set(50,"S2")
           time.sleep(0.5)
           mbot2.servo_set(85,"S1")
           time.sleep(0.5)
           mbot2.servo_set(15,"S2")
           mbot2.servo_set(90,"S3")
```

```
@event.start
def on_start2():
   mbot2.servo_set(65,"S2")
   time.sleep(1)
   mbot2.servo_set(65,"S1")
   mbot2.servo_set(90,"S3")
   mbot2.servo_set(15,"S2")
   cyberpi.audio.play('warning')
   mbuild.ultrasonic2.play("happy", 1)
   while True:
       if gamepad.is_key_pressed('N4'):
           mbot2.servo_set(9,"S1")
           while not mbuild.ultrasonic2.get(1) < 8:
               pass

           time.sleep(0.1)
           mbot2.servo_set(65,"S2")
           time.sleep(0.5)
           mbot2.servo_set(148,"S1")
           time.sleep(0.46)
           mbot2.servo_set(0,"S2")
           time.sleep(0.7)
           mbot2.servo_set(50,"S3")
           mbot2.servo_set(50,"S2")
           time.sleep(0.5)
           mbot2.servo_set(85,"S1")
           time.sleep(0.5)
           mbot2.servo_set(15,"S2")
           mbot2.servo_set(90,"S3")

@event.start
def on_start3():
   while True:
       if gamepad.is_key_pressed('N3'):
           mbot2.EM_set_power(-100, "EM2")
           time.sleep(0.1)
           mbot2.EM_stop("EM2")
           time.sleep(0)
           mbot2.EM_set_power(100, "EM2")
           time.sleep(0.1)
           mbot2.EM_stop("EM2")
           time.sleep(0.1)
```

```
@event.start
def on_start4():
   while True:
       if gamepad.is_key_pressed('N3'):
           mbot2.EM_set_power(100, "EM1")
           time.sleep(0.1)
           mbot2.EM_stop("EM1")
           time.sleep(0)
           mbot2.EM_set_power(-100, "EM1")
           time.sleep(0.1)
           mbot2.EM_stop("EM1")
           time.sleep(0.1)

@event.start
def on_start5():
   while True:
       if gamepad.is_key_pressed('Up'):
           mbot2.motor_set(55,"M1")

       else:
           if gamepad.is_key_pressed('Right'):
               mbot2.motor_set(100,"M1")

           else:
               if gamepad.is_key_pressed('Left'):
                   mbot2.motor_set(-100,"M1")

               else:
                   if gamepad.is_key_pressed('Down'):
                       mbot2.motor_set(-60,"M1")

                   else:
                       mbot2.motor_stop("all")

@event.start
def on_start6():
   while True:
       if gamepad.is_key_pressed('Up'):
           mbot2.motor_set(100,"M2")

       else:
           if gamepad.is_key_pressed('Right'):
               mbot2.motor_set(-100,"M2")

           else:
               if gamepad.is_key_pressed('Left'):
                   mbot2.motor_set(100,"M2")

               else:
                   if gamepad.is_key_pressed('Down'):
                       mbot2.motor_set(-100,"M2")

                   else:
                       mbot2.motor_stop("all")

@event.start
def on_start7():
   while True:
       if gamepad.is_key_pressed('R1'):
           if gamepad.is_key_pressed('Up'):
               mbot2.motor_set(10,"M1")

           else:
               if gamepad.is_key_pressed('Left'):
                   mbot2.motor_set(10,"M1")

               else:
                   if gamepad.is_key_pressed('Right'):
                       mbot2.motor_set(-10,"M1")

                   else:
                       if gamepad.is_key_pressed('Down'):
                           mbot2.motor_set(-10,"M1")

                       else:
                           mbot2.motor_stop("all")

@event.start
def on_start8():
   while True:
       if gamepad.is_key_pressed('R1'):
           if gamepad.is_key_pressed('Up'):
               mbot2.motor_set(2,"M2")
```

```
        else:
            if gamepad.is_key_pressed('Left'):
                mbot2.motor_set(-2,"M2")

            else:
                if gamepad.is_key_pressed('Right'):
                    mbot2.motor_set(2,"M2")

                else:
                    if gamepad.is_key_pressed('Down'):
                        mbot2.motor_set(-2,"M2")

                    else:
                        mbot2.motor_stop("all")
@event.start
def on_start3():
   while True:
       if gamepad.is_key_pressed('L1'):
           if mbot2.servo_get("S2") == 65:
               mbot2.servo_set(9,"S1")
               time.sleep(0.5)
               mbot2.servo_set(15,"S2")
               time.sleep(0.5)
               mbot2.servo_set(65,"S1")
               time.sleep(0.5)

           else:
               mbot2.servo_set(45,"S1")
               time.sleep(1)
               mbot2.servo_set(65,"S2")
               time.sleep(0.5)
               mbot2.servo_set(65,"S1")
               time.sleep(0.5)
```

Discussion

1) Line follower Robot

   1.1 The weight balancing of a line follower should be towards the front a little bit so that the level of the infrared sensors stay the same, meaning that the received signal will be more stable, resulting in a more stable movement and

   1.2 response from the robot. Also we cannot balance the weight too much towards the front because the rear wheels will lose traction and therefore the robot loses control.

   1.3 By putting the infrared sensor of the line follower robot closer to each other, we will have more conditions to work with as there will be scenarios that 2 sensors detect black. Because of that, it will be easier to tune the robot to be more stable.

   1.4 Using only one caster wheel for the line follower robot will reduce the friction between the robot and the ground and also the robot can be tuned stable more easily compared to using two caster wheels.

2) Manual Robot

   2.1 The gripper should have more force and be made with a material that has more coefficient of friction so it can lift something heavier and more stable.

   2.2 Add more joints to the gripper's arm to make it move more freely. And with the gripper has more force and coefficient of friction it won't fall down but it mustn't move too far because the center of mass will change and maybe it will flop down.

   2.3 By moving the board behind the gripper and the catapult a little bit further, the gripper will have more space to move to the back to have more angle and the ball can fall down to the catapult easier.3

Future works

1) Try to make 2 cars more beautiful, precise and neat with another material.
2) Increase the distance the catapult can throw.
3) Make the gripper hold more weight.
4) Make the Line follower robot not only follow black lines but also every colour lines.

References

Auto Robot

- https://www.instructables.com/Line-Follower-Robot-With-Arduino-Really-Fast-and-R/ Manual Robot

Manual Robot

- ercost60. (2012, February 21). Servo Catapult. YouTube. https://www.youtube.com/watch?v=wBN9sTBDDLE
- MakeX Robotics Competition Commitee (Ed.). (2021, September). 2022 Season MakeX Explorer Eco Pioneer Rules Guide V1.01. https://www.makeblock.in.th/wp-content/uploads/2022/03/2022-season-makex-explorer-eco-pioneer-rules-guide-v1.01.pdf

Member Participation

1) 65011497 - Pyae Htoo Khant

   For Line follower robot :(10%)
   - Research and discussion
   - Ideate the design
   - Draw the prototype design
   - Draw the final design
   - Assembly drawing of the robot
   - Draw the circuit diagram
   - Draw flowchart
   - Implement the robot
   - Code the program

   For Manual robot :(7%)
   - Ideate the concept of catapult launcher with DC motor
   - Design the catapult launcher arm
   - Calculate the velocity of catapult launcher and range
   - Draw the back slope design for the transmission of the ball to the catapult
   - Hardware maintenance

   Presentation(4%)
   - Document the presentation.
   - Record the testing video

   Report(4%)
   - Document the report

   Total Contribution: 25%

   Pyee

   (Pyae Htoo Khant)

2) 65011453 - Phattawin Kummarraphat

   For Line follower robot :(7%)

   - Research and discussion
   - Co-design the robot.
   - Code the prototype robot.

   For Manual robot :(10%)

   - Design the prototype of the manual robot.
   - Draw the gripper design.
   - Design the prototype of launcher
   - Hardware maintenance
   - Code the manual robot.

   Presentation(4%)

   - Recheck the presentation slides and scripts before presenting.

   Report(4%)

   - Document the report
   - Film a robot video.
   - Recheck the report’s format before submitting.

Total Contribution: 25%

Ph. K.

(Phattawin Kummaraphat)

3) 65011598 - Thitiphan Chenrukmatupoom

For Line follower robot :(7%)

- Research and discussion
- Co-design the robot.
- Code the prototype robot.

For Manual robot :(10%)

- Design the catapult base and chassis.
- Draw the back slope design for the transmission of the ball to the catapult
- Hardware maintenance

Presentation(4%)

- Recheck the presentation slides and scripts before presenting.

Report(4%)

- Document the report
- Film a robot video.
- Recheck the report’s format before submitting.

Total Contribution: 25%

(Thitiphan Chenrukmatupoom)

4) 65011454 - Phawaris Tangsripairoje

- Document the report. (10%)
- Presentation. (10%)
- Suggested ideas for building both robots. (5%)

Total Contribution: 25%

(Phawaris Tangsripaoroje)