Checkpoint #3 Report

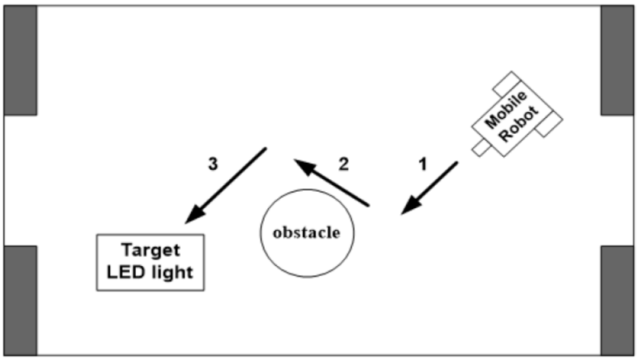
[EECN30169] Mobile Robot 2022

Student ID: 311605004 **Name:** 劉子齊 **Date: 2022.11.15**

1. Introduction:

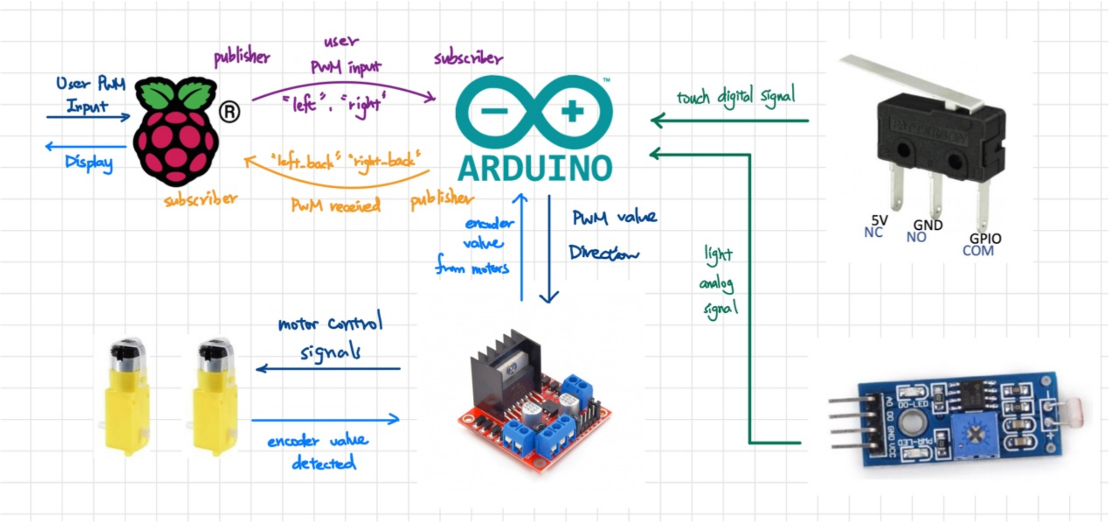
For checkpoint 3, our goal is to implement our mobile robot and make it able to move in the arena freely. Besides, to make our mobile robot able to avoid obstacles and detect the target hockey ball, we have to implement the feature of collision detection and light discerning.

The following figure is the hockey arena for this checkpoint. As we can see in the following figure, our goal is to make our robot to avoid the obstacle in the center of the arena. After avoided from the arena, the next task for our robot is to find the target hockey light by the light sensor. When detected the target hockey, our robot will have to approach the target hockey and catch it. Afterwards, our mobile robot has to approach and catch the target hockey, then the task will be counted as a success.



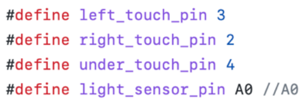
2. Description of Design:

Below is the workflow of my implementation to the communication between the Raspberry Pie, the Arduino, the L298N motor control module, the 2 DC motors, the 3 touch sensors, and the light sensor.

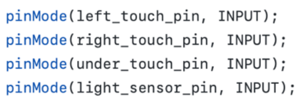


In this project, since the GPIO of our Raspberry Pie got some problem detecting the signals from the sensors, we mainly rely on the Arduino to take care of all the sensors and control our mobile robot.

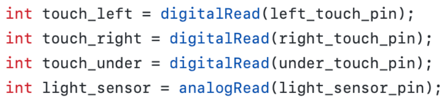
To make our mobile able to detect the return signal of the touch sensors and the light sensor, we set the input pins as the following figure. We set the touch sensors as digital read, since we only need to know whether the touch sensors were pressed or not. However, we set the light sensor as analog read, since it will be a lot easier for us to detect the target hockey correctly by the exact light strength value returned from the light sensor.



```
#define left_touch_pin 3
#define right_touch_pin 2
#define under_touch_pin 4
#define light_sensor_pin A0 //A0
```



```
pinMode(left_touch_pin, INPUT);
pinMode(right_touch_pin, INPUT);
pinMode(under_touch_pin, INPUT);
pinMode(light_sensor_pin, INPUT);
```

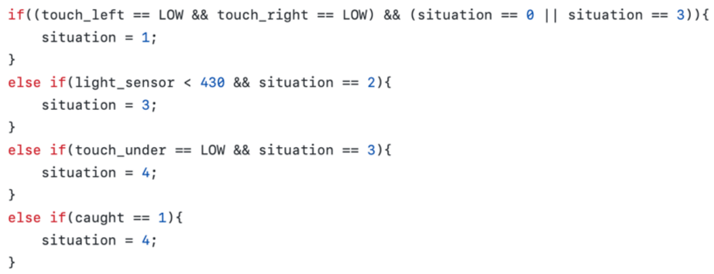


```
int touch_left = digitalRead(left_touch_pin);
int touch_right = digitalRead(right_touch_pin);
int touch_under = digitalRead(under_touch_pin);
int light_sensor = analogRead(light_sensor_pin);
```

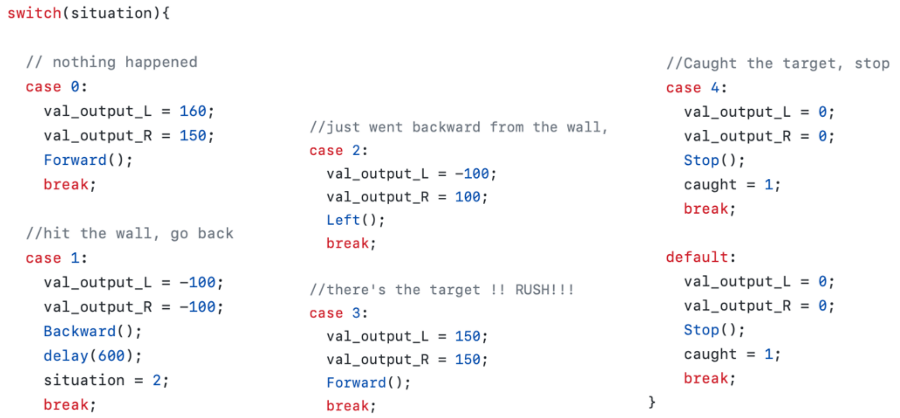
With all the return values from the sensors, we now move onto the motion control. The implementation of the motion control is shown in the following figures. Basically, I separated the whole process into 5 situations, which started from 0 as the initial situation.

The first situation is situation 0, which the mobile robot has only 1 task, which is to go straight until it runs into the wall. As the 2 touch sensors attached at the front of the mobile robot detect the wall, the robot will enter situation 1, which the robot will go backwards for 600 milliseconds. After moving backwards, the robot will enter situation 2, which it will start to turn left. During situation 2, when the light sensor attached at the bottom of the robot detects the target hockey, which is obviously brighter than the surroundings, the robot will stop turning and enter situation 3.

As the robot enters situation 3, it means the robot had found the target hockey. At this moment, there is only one mission for the mobile robot, which is rush towards the target hockey and catch it. If the robot catches the target hocket, which will determine on the touch sensor attached at the bottom of the robot, the robot will enter situation 4 and stop right at the position it caught the target, which indicates the accomplishment of the task.

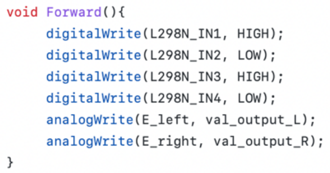


```
if((touch_left == LOW && touch_right == LOW) && (situation == 0 || situation == 3)){
    situation = 1;
}
else if(light_sensor < 430 && situation == 2){
    situation = 3;
}
else if(touch_under == LOW && situation == 3){
    situation = 4;
}
else if(caught == 1){
    situation = 4;
}
```

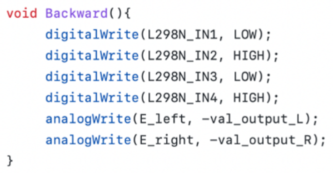


```
switch(situation){

  // nothing happened
  case 0:
    val_output_L = 160;
    val_output_R = 150;
    Forward();
    break;

  //hit the wall, go back
  case 1:
    val_output_L = -100;
    val_output_R = -100;
    Backward();
    delay(600);
    situation = 2;
    break;

  //just went backward from the wall,
  case 2:
    val_output_L = -100;
    val_output_R = 100;
    Left();
    break;

  //there's the target !! RUSH!!!
  case 3:
    val_output_L = 150;
    val_output_R = 150;
    Forward();
    break;

  //Caught the target, stop
  case 4:
    val_output_L = 0;
    val_output_R = 0;
    Stop();
    caught = 1;
    break;

  default:
    val_output_L = 0;
    val_output_R = 0;
    Stop();
    caught = 1;
    break;
}
```

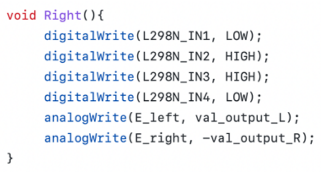
For the motion control of the mobile robot, I applied the achievements of the previous checkpoint. The motion signal is send form the Arduino to the L298N motor control module according to different cases, which I mainly made use of the 3 cases defined below.



```
void Forward(){
    digitalWrite(L298N_IN1, HIGH);
    digitalWrite(L298N_IN2, LOW);
    digitalWrite(L298N_IN3, HIGH);
    digitalWrite(L298N_IN4, LOW);
    analogWrite(E_left, val_output_L);
    analogWrite(E_right, val_output_R);
}
```



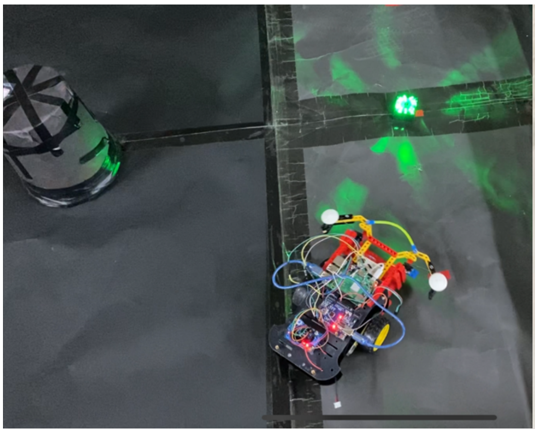
```
void Backward(){
    digitalWrite(L298N_IN1, LOW);
    digitalWrite(L298N_IN2, HIGH);
    digitalWrite(L298N_IN3, LOW);
    digitalWrite(L298N_IN4, HIGH);
    analogWrite(E_left, -val_output_L);
    analogWrite(E_right, -val_output_R);
}
```

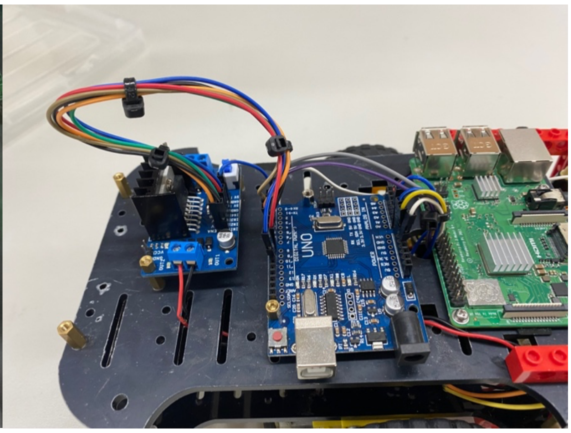


```
void Right(){
    digitalWrite(L298N_IN1, LOW);
    digitalWrite(L298N_IN2, HIGH);
    digitalWrite(L298N_IN3, HIGH);
    digitalWrite(L298N_IN4, LOW);
    analogWrite(E_left, val_output_L);
    analogWrite(E_right, -val_output_R);
}
```

Lastly, during the implementation, we found we would need to know the exact value which the light sensor is receiving to adjust the threshold value we applied to determine whether we are facing the target hockey or not. Hence, we applied the subscriber and the publisher we learned in the previous checkpoint to send the light value detected from the Arduino to the Raspberry Pie.

3. Result





The figures above of the mobile robot we designed, and how the arena and the target hockey look like. There is a point that is worth to mention, which we spent plenty of time arranging the wires of our mobile robot. As shown in the right figure above, we made our robot to look extremely clean by using cable ties on the wires, which was emblazed by the professor.

4. Discussion

The main problem we met in this checkpoint is we got some problem detecting the Raspberry Pie’s GPIO signal. However, after discussing with the teaching assistants, we decided to abort reading sensors’ signal from the Raspberry Pie, but directly detect them from the Arduino, and this solved our problem eventually.