Development of embedded system for creating automation robot

Ms. Natthida Patlee	6131501016
Mr. Nuttaphong Samiphak	6131501026
Ms. Wirakarn Kaewkanya	6131501054
Mr. Arunwat Moonbung	6131501066

School of Information Technology, Computer Engineering

Present to

Asst. Prof. Gp.Capt. Dr. Thongchai Yooyativong

Dr. Suppakarn Chansareewittaya

1st Semester / Year 2019

Embedded Systems (1501483)

Electric Drives (1501218

Table of Contents

Introduction of Embedded System and Electric Drive 1
Embedded System 1
Electric Drive 1
Robot information & design 2
Equipment & Components 2
Flowchart 8
Circuit Diagram 9
Sketch Code 10
Anatomy of Robot 14
Discussion 18
Conclusion 18
References 19

Introduction of Embedded System and Electric Drive

For decades, embedded system brought about a revolution in Science. It is also a part of an Internet of Things (IoT) – a technology in which is trend in the future. Embedded system also provides the ability to control electronic devices without requiring human-to-human or human-to-computer interaction, which is commonly found in consumer, industrial, automotive, home appliances, medical, commercial and military applications.

Embedded System

An embedded system is a combination of computer software and hardware which is either fixed in capability or programmable. An embedded system can be either an independent system, or it can be a part of a large system. It is mostly designed for a specific function or functions within a larger system. Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

Electric Drive

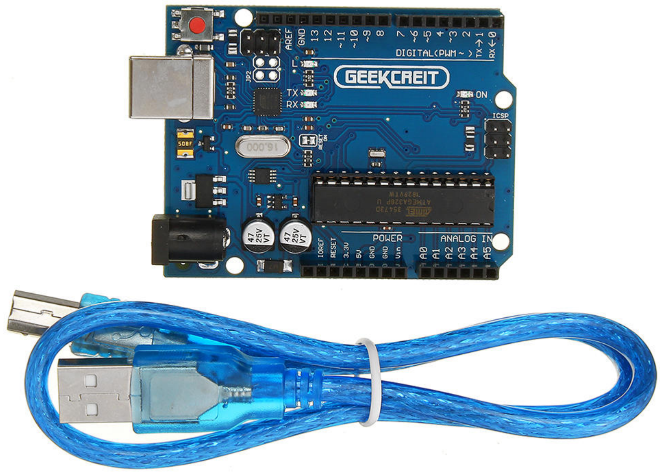
An Electric Drive can be defined as, a system which is used to control the movement of an electrical machine. An electric drive can be built with an electric drive motor as well as a complicated control system to control the motor’s rotation shaft. At present, the controlling of this can be done simply using the software. Thus, the controlling turns into more accurate & this drive concept also offers the ease of utilizing. The types of electrical drives are two such as a standard inverter as well as a servo drive. A standard inverter drive is used to control the torque & speed. A servo drive is used to control the torque as well as speed, and also components of the positioning machine utilized within applications that need difficult motion.

Robot information & design

This robot has been developed for the purpose of studying embedded system and electric drive. We decide to use a Microcontroller along with motor driver and sensor in term to create automation robot which can automatically detect the obstacle and drive back to where it came from.

Equipment & Components

1. Arduino UNO R3



2. Active Buzzer Module with LED

3. Breadboard

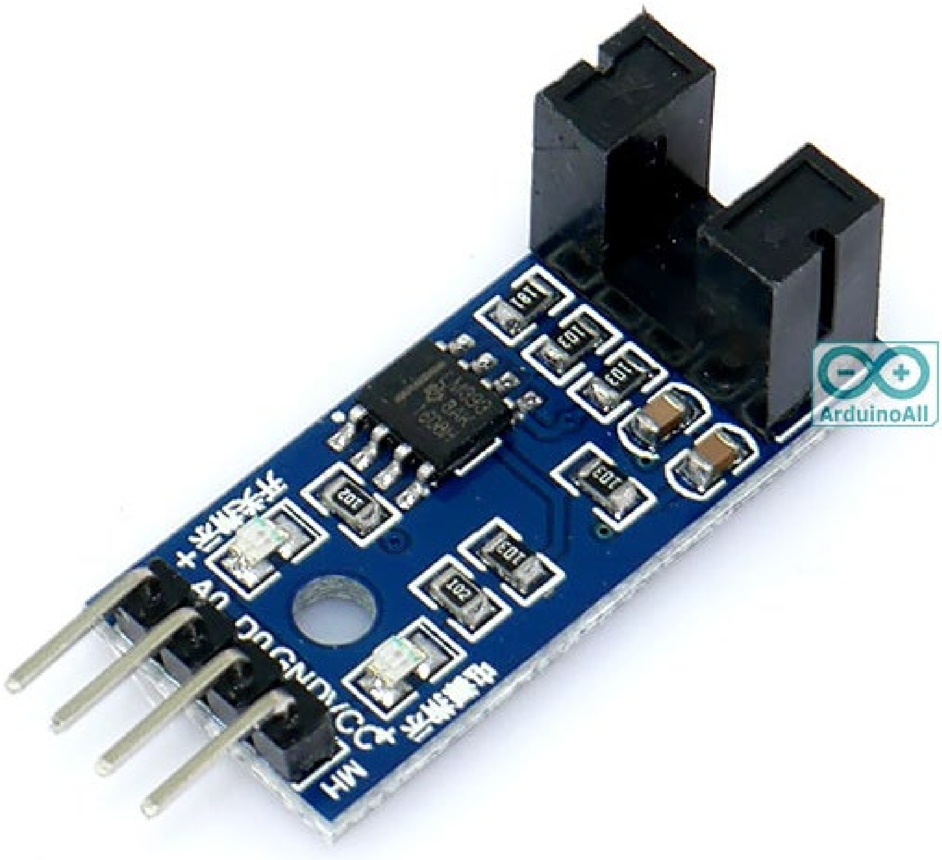
4. Battery Holder Socket with DC Jack & Switch 4*AA

5. Battery Holder Socket 3*AA

6. Button

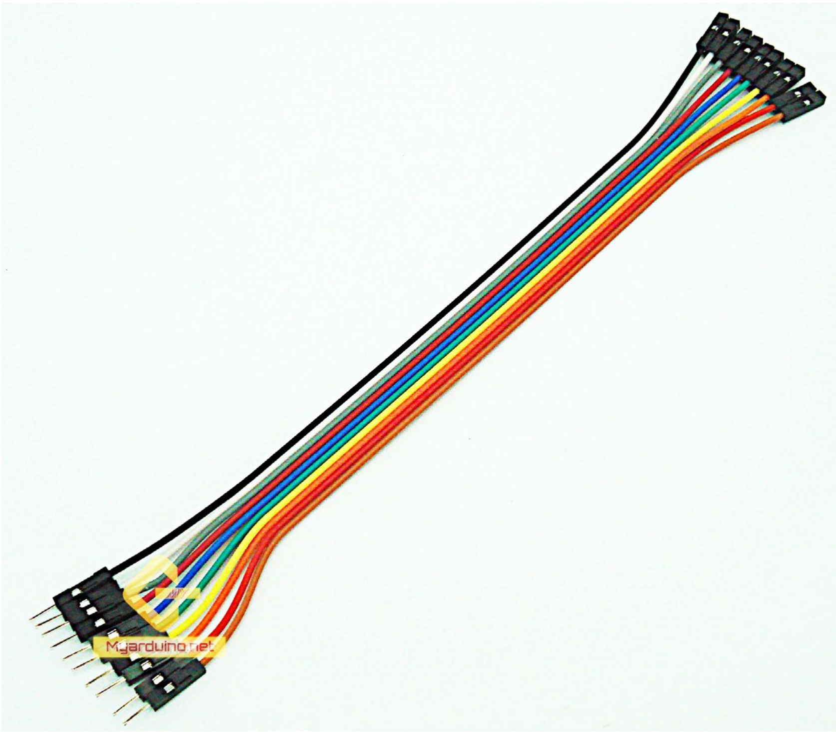
7. DC Gear Motor 3-12V

8. Infrared Sensor module



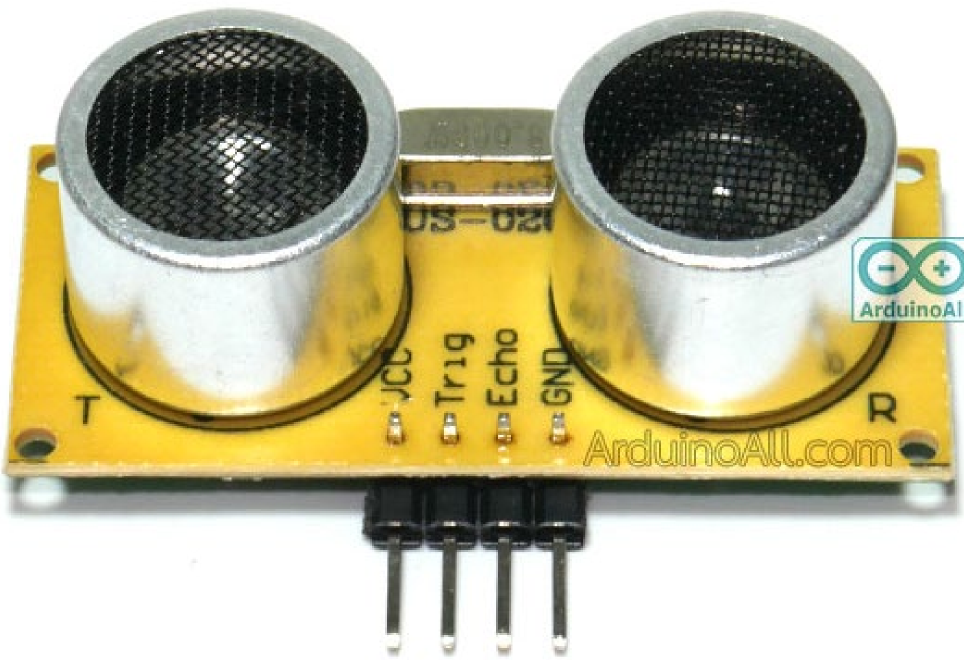
9. Jump Wire (Male to Male)

10. Jump Wire (Male to Female)

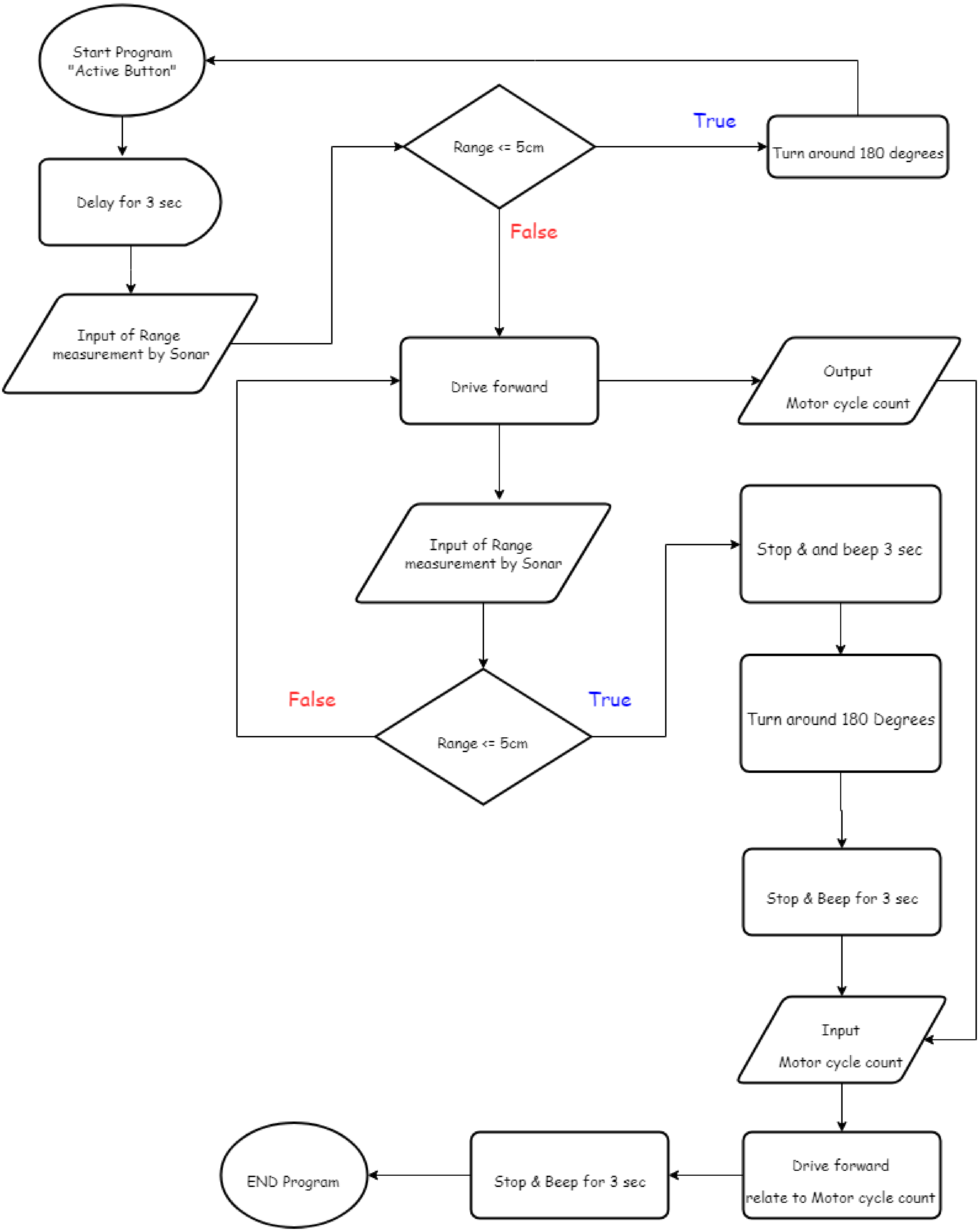


11. Motor Driver L298N

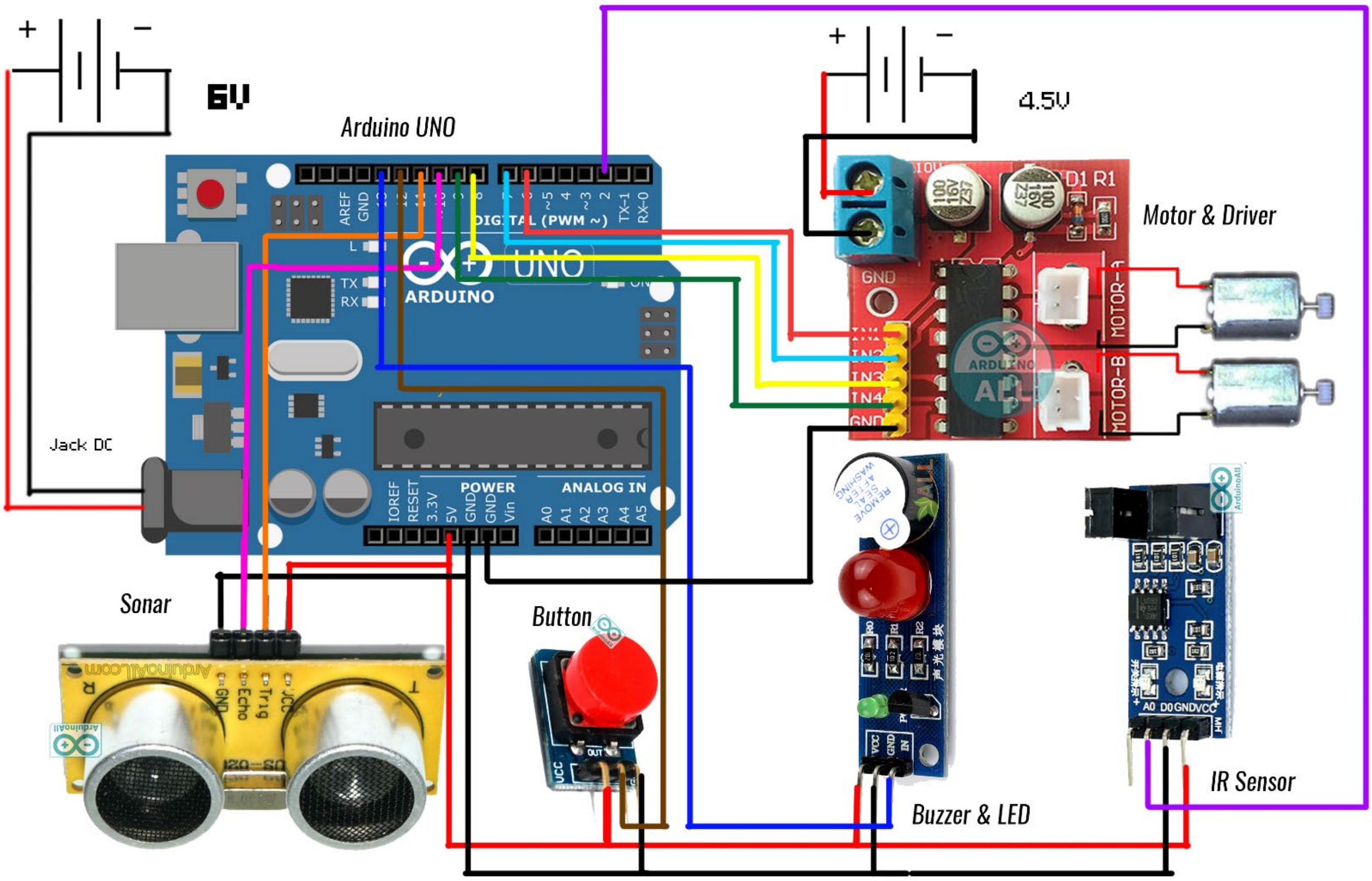
12. Ultrasonic Sensor Module (US-025 or HC-SR04)



Flowchart


Start Program
"Active Button"
Delay for 3 sec
Input of Range
measurement by Sonar
Range <= 5cm
True
Turn around 180 degrees
False
Drive forward
Output
Motor cycle count
Input of Range
measurement by Sonar
Stop & and beep 3 sec
False
True
Range <= 5cm
Turn around 180 Degrees
Stop & Beep for 3 sec
Input
Motor cycle count
END Program
Stop & Beep for 3 sec
Drive forward
relate to Motor cycle count

Circuit Diagram



According to Sketch code, all wires must be precisely connected. But, the color of the wire is not necessary to be as same as the circuit diagram above. If Sketch code has been changed, it is important that all connected wires, especially Digital I/O pin are connected properly.

Sketch Code

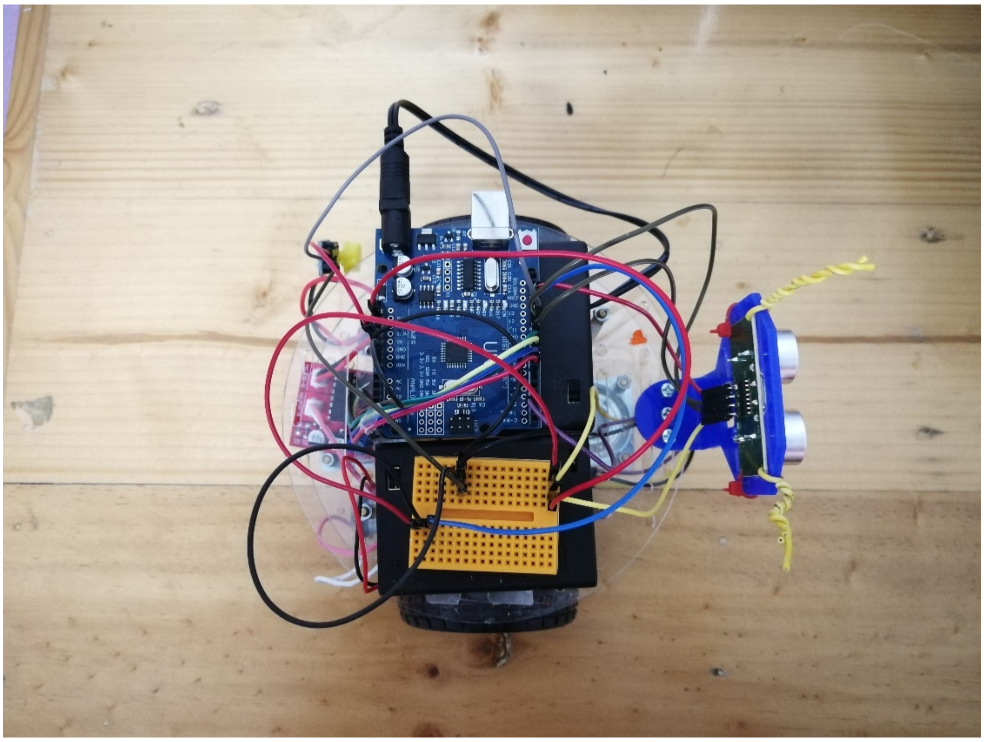
```
//List of Library Include
//License ID : 6131501016 6131501026 6131501054 6131501066
#include <NewPing.h> //Sonar
//------------------------
//List of Pin Number
int pinButton = 12; // Declare Number of PIN that is Connected to Yellow Button at OUTPUT
int pinBuzzer_1 = 13; // Declare Number of PIN that is Connected to Buzzer at IN
int pinIRSensor = 2; // Declare Number of PIN that is Connected to Infrared Sensor D0

#define TRIGGER_PIN  11  // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN     10  // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 400 // Maximum distance we want to ping for (in centimeters).
//Maximum sensor distance is rated at 400-500cm.
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // NewPing setup of pins and maximum distance.

int DRIVE_EN1 = 6; // Declare Number of PIN that is Connected to DC Motor Driver L298N IN1
int DRIVE_EN2 = 7; // Declare Number of PIN that is Connected to DC Motor Driver L298N IN2
int DRIVE_EN3 = 8; // Declare Number of PIN that is Connected to DC Motor Driver L298N IN3
int DRIVE_EN4 = 9; // Declare Number of PIN that is Connected to DC Motor Driver L298N IN4
//------------------------
//List of variable used in Program
//Main Variable
boolean isCounter = false;
//------------------------
//INT Variable for counting

//------------------------

void setup()
{
  Serial.begin(9600); //Connected to Serial Monitor
  //Select Pin MODE
  //INPUT
  pinMode(pinButton, INPUT);
  pinMode(pinIRSensor, INPUT);
  attachInterrupt(0, doCounter, FALLING);
  //OUTPUT
  pinMode(pinBuzzer_1, OUTPUT);
  pinMode(DRIVE_EN1, OUTPUT);
  pinMode(DRIVE_EN2, OUTPUT);
  pinMode(DRIVE_EN3, OUTPUT);
  pinMode(DRIVE_EN4, OUTPUT);

  digitalWrite(pinBuzzer_1, HIGH); // SET DEFAULT TURN OFF THE BUZZER
}
```

```
void loop()
{
  int input = digitalRead(pinButton);
  // INPUT FROM YELLOW BUTTON IF ACTIVE HIGH THEN PROGRAM START (DEFUALT LOW STATUS)
  if (input == 1)
  {
    Serial.print("Initialize the Program | ");
    Serial.print("Status Button = ");
    Serial.println(input);
    delay(100);
    // 3 Seconds before Drive
    int LEDBlinkbegin = 0;
    int Countdown_1 = 3;
    while (LEDBlinkbegin < 3)
    {
      Serial.print("Begin with in ");
      Serial.print(Countdown_1);
      Serial.println(" Seconds ");
      digitalWrite(pinBuzzer_1, LOW);
      delay(1000);
      digitalWrite(pinBuzzer_1, HIGH);
      LEDBlinkbegin = LEDBlinkbegin + 1;
      Countdown_1 --;
      delay(1000);
    }
    // Initialize Sonar
    delay(100);
    // Wait 50ms between pings (about 20 pings/sec). 29ms should be the shortest delay between pings.
    Serial.print("The Distance is : ");
    Serial.print(sonar.ping_cm());
    // Send ping, get distance in cm and print result (0 = outside set distance range)
    Serial.println(" cm");


   // Drive Forward until closer than xx
   int IRCounter = 0;
   while (sonar.ping_cm() >= 20)
   {
     digitalWrite(DRIVE_EN2, HIGH);
     digitalWrite(DRIVE_EN4, HIGH);
     if (isCounter == true)
     {
       isCounter = false;
       Serial.println(IRCounter);
       Serial.print("The Distance is : ");
       Serial.print(sonar.ping_cm());
       // Send ping, get distance in cm and print result (0 = outside set distance range)
       Serial.println(" cm");
       IRCounter++;
     }
   }
   // STOP After closer than xx stop and delay 2 secs
   digitalWrite(DRIVE_EN2, LOW);
   digitalWrite(DRIVE_EN4, LOW);
   digitalWrite(pinBuzzer_1, LOW);
   delay(3000);
   digitalWrite(pinBuzzer_1, HIGH);
```

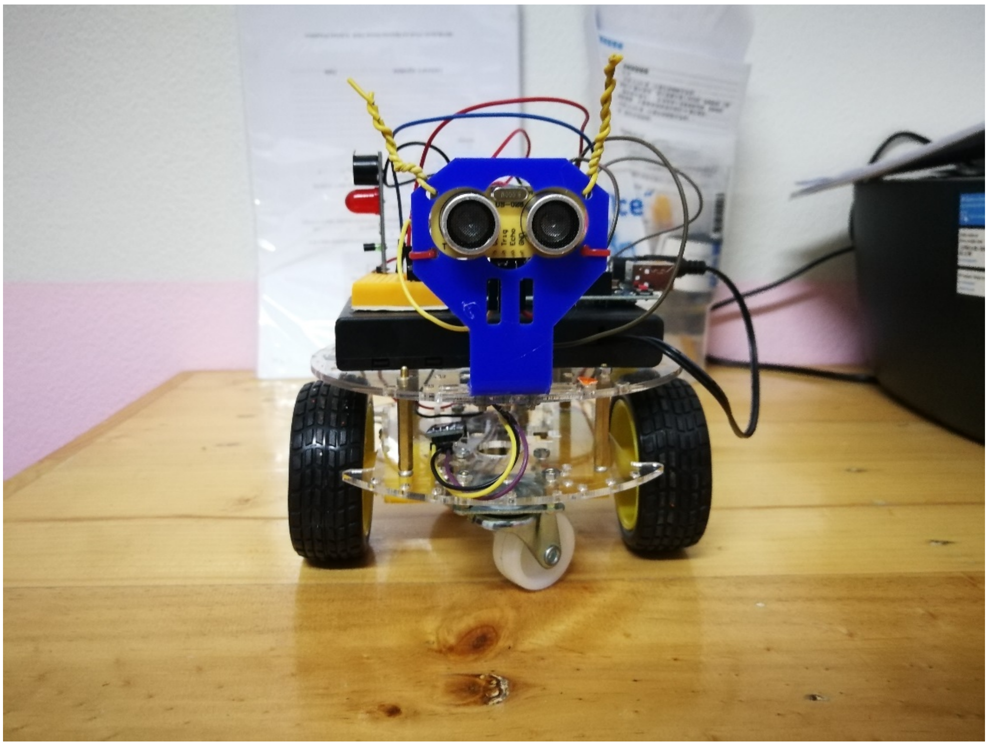
```
// TURN 180 Degrees
Serial.println("");
Serial.println("TURN 180 Degrees STARTED");
int IRCounterTurn = 0;
while (IRCounterTurn < 16  ) // TURN 180 Degree Adjustable follow Motor speed
{
  digitalWrite(DRIVE_EN2, HIGH);
  digitalWrite(DRIVE_EN3, HIGH);
  if (isCounter == true)
  {
    isCounter = false;
    Serial.println(IRCounterTurn);
    Serial.print("The Distance is : ");
    Serial.print(sonar.ping_cm());
    // Send ping, get distance in cm and print result (0 = outside set distance range)
    Serial.println(" cm");
    IRCounterTurn++;
  }
}
Serial.println("TURN 180 Degrees FINISHED");

// AFTER TURN 180 STOP AND DELAY FOR 2 SEC
digitalWrite(DRIVE_EN2, LOW);
digitalWrite(DRIVE_EN3, LOW);
digitalWrite(pinBuzzer_1, LOW);
delay(3000);
digitalWrite(pinBuzzer_1, HIGH);

// Drive back to where it came from
int IRCounterBack = 0;
while (IRCounterBack < IRCounter)
{
  int IRSensorBack = digitalRead(pinIRSensor);
  digitalWrite(DRIVE_EN2, HIGH);
  digitalWrite(DRIVE_EN4, HIGH);
  if (isCounter == true)
  {
    isCounter = false;
    Serial.println(IRCounterBack);
    Serial.print("The Distance is : ");
    Serial.print(sonar.ping_cm());
    // Send ping, get distance in cm and print result (0 = outside set distance range)
    Serial.println(" cm");
    IRCounterBack++;
  }
}

digitalWrite(DRIVE_EN2, LOW);
digitalWrite(DRIVE_EN4, LOW);
//AFTER DRIVE TO START POSITION STOP AND BLINK
```

```
    Serial.println("");
    Serial.print("Checking IR Count from Last IRCounter = ");
    Serial.println(IRCounter);
    Serial.print("Checking IR Count from Last IRCounterBack = ");
    Serial.println(IRCounterBack);
    Serial.println("Note that IRCounter must equal to IRCounterBack");
    Serial.println("");

    //DO FINAL BLINK
    int LEDBlinkEnd = 0;
    while (LEDBlinkEnd < 6)
    {
      digitalWrite(pinBuzzer_1, LOW);
      delay(100);
      digitalWrite(pinBuzzer_1, HIGH);
      LEDBlinkEnd = LEDBlinkEnd + 1;
      delay(100);
    }

  Serial.println(" ");
  Serial.println("The Program has finished Process Please Push Yellow the button to Start [End Program]");
}
else
{
  Serial.println("Program Robot has not been started yet | Status button = 0");
  delay(1000);
}

}

void doCounter()
{
  isCounter = true;
}
```

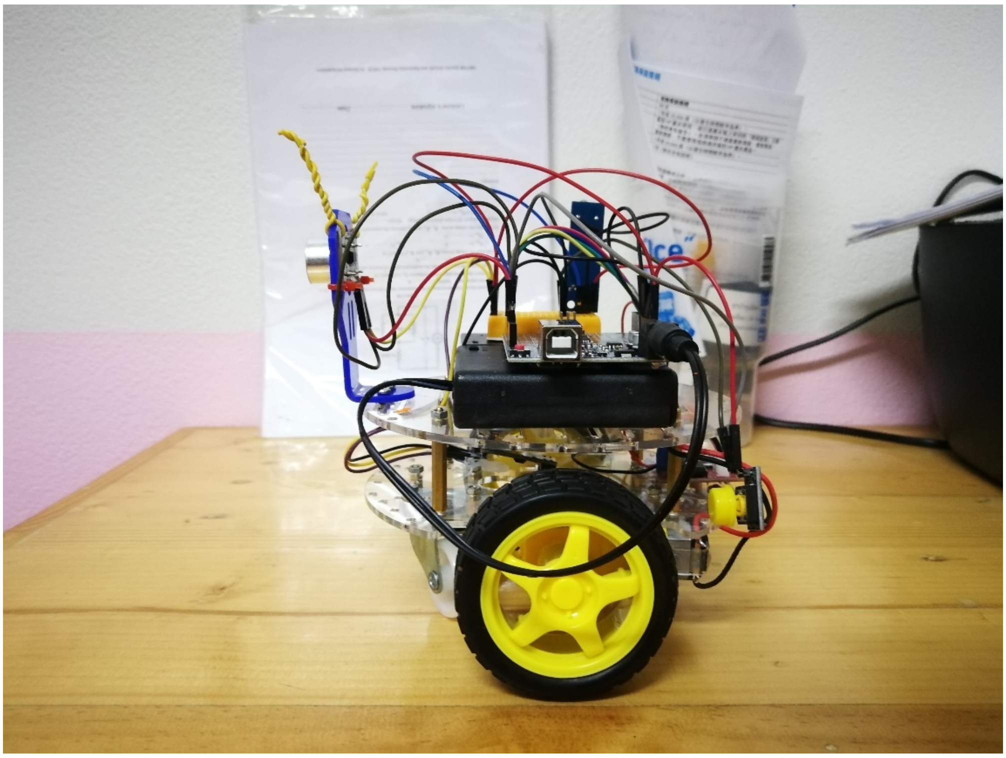
Anatomy of Robot



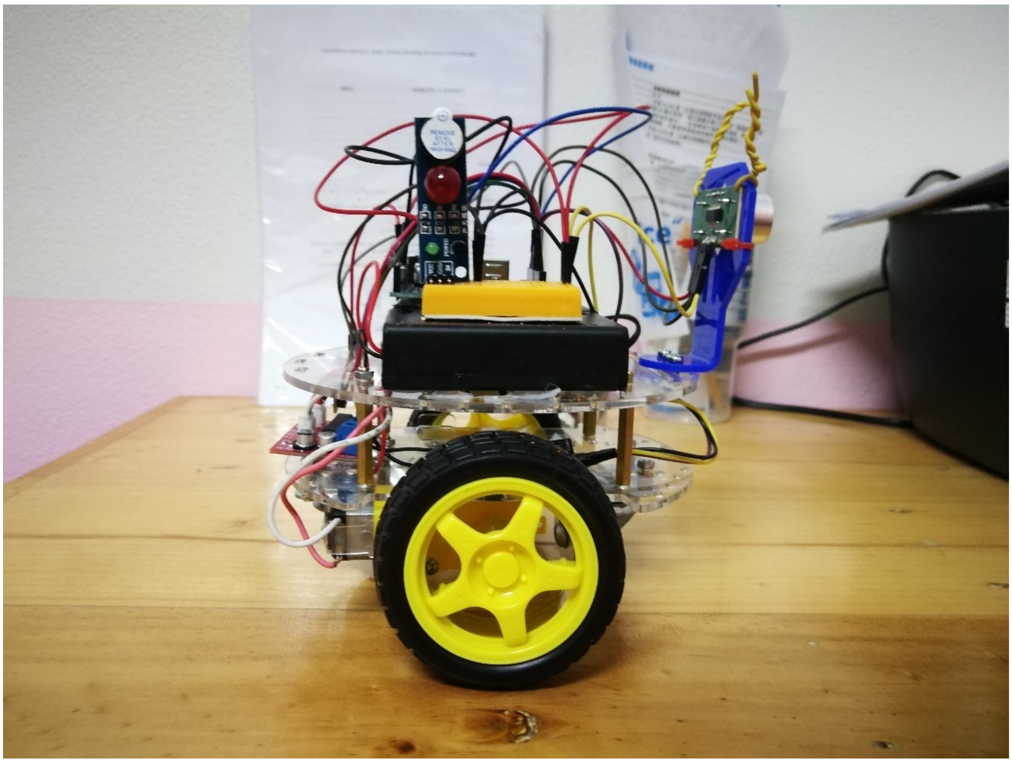
(Top view of Robot without Case)



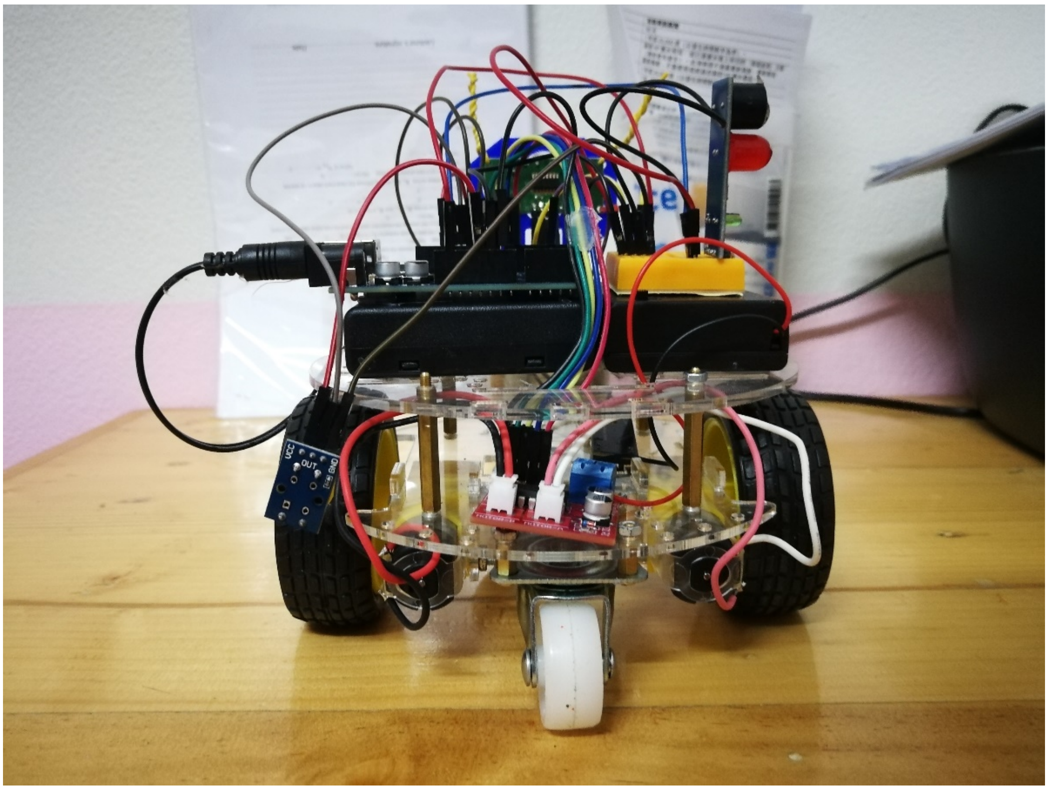
(Front view of Robot without Case)



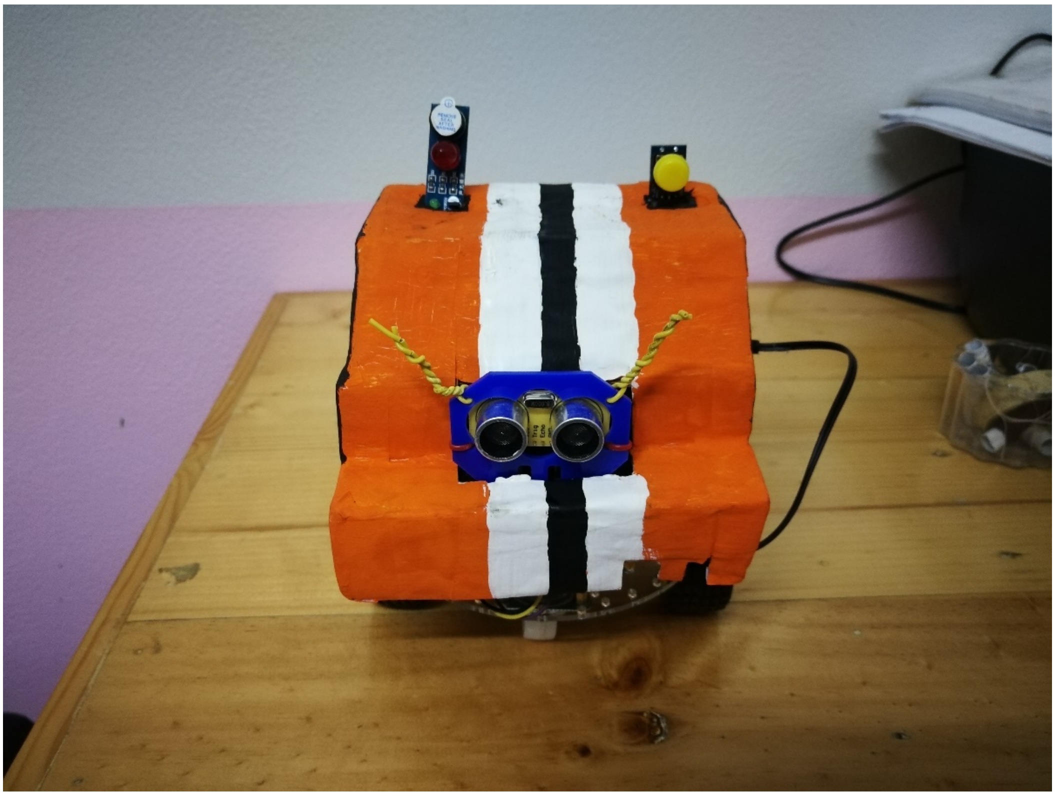
(Left side view of Robot without Case)



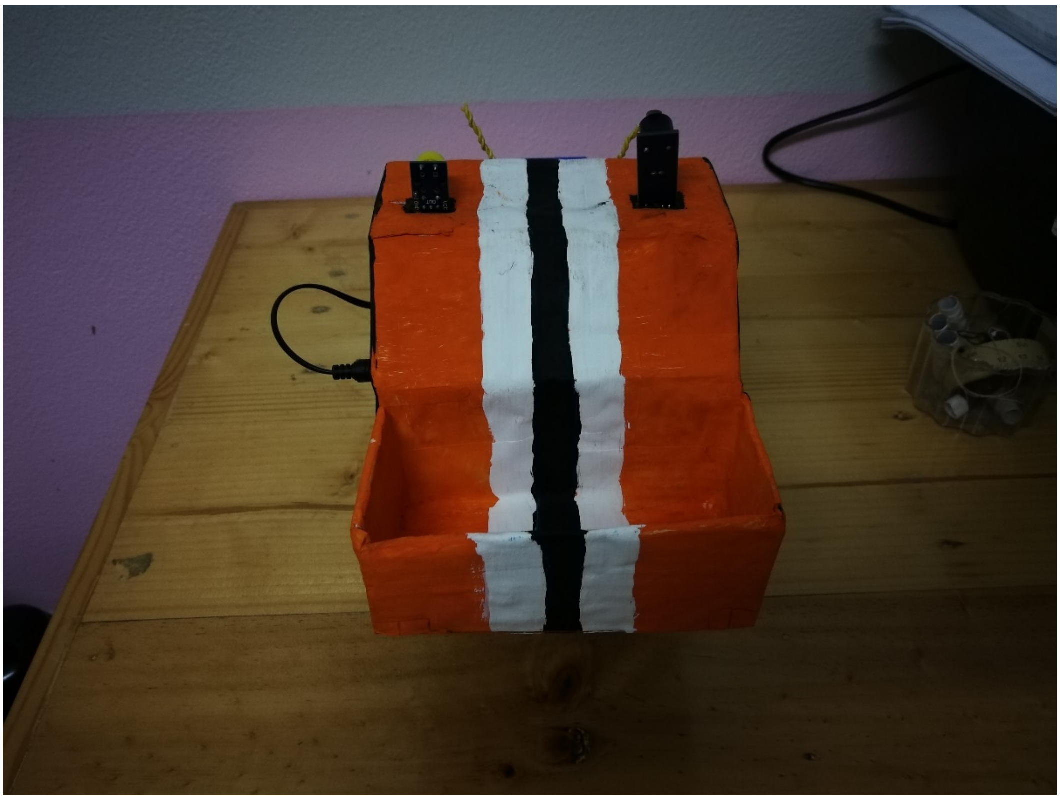
(Right side view of Robot without Case)



(Behind view of Robot without Case)



(Front view of Robot with Case)



(Behind view of Robot with Case)

Discussion

Although the robot can drive forward, turn around and drive back to the beginning point, there is a problem during the process of developing this robot. The robot is supposed to straightly drive forward, but it turns out the DC motor in both sides are not drive at the same speed which cause robot slantingly drives. In term to fix this problem, we have changed DC motor which result in less slantingly drives, but still not straightly drive forward. Not only the DC motor that have a problem, another factor is also wheels of the robot that are not the same quality. Our team tried to change a couple of wheels and found that the robot drives more stable but still not completely straight.

Conclusion

The combination of Embedded system and Electric Drive allows us to develop a mouse robot which can automatically drive forward, detect the obstacle and drive back to the begin position. However, the mouse robot still drives not completely straight, even though, we have already changed and fixed. We can only make it drive close to straight as much as possible. This is only one little step of applying Embedded System. Our team look further in the future to do the research for upgrading this robot so that it can be more functional and implemented in the daily life.

References

Electrical4U. (2019, May 21). Electric Drive : Types, Block Diagram, Classification and Its Applications. Retrieved December 1, 2019, from https://www.elprocus.com/electric-drive-types-block-diagram-classification/.

Embedded Systems Tutorial: History, Types, Advantages, EXAMPLES. (n.d.). Retrieved December 1, 2019, from https://www.guru99.com/embedded-systems-tutorial.html.

Sriharsha. (2018, December 30). What is an Electrical Drive? Retrieved December 1, 2019, from https://www.electrical4u.com/electrical-drives/.