UAEU جامعة الإمارات العربية المتحدة United Arab Emirates University

# CSPB476: Robotics and Intelligent Systems

# Mobile Robot Competition

# Spring 2025

Team Name: Falcons

Student Names and ID:

Abdullah Al Amoodi – 202119525

Fadel Al Shamsi – 202102655

Khalifa Al Dhaheri – 202103159

Mohammed Al Nuaimi – 202111646

Obaid Alkaabi – 202102854

Instructor: Dr. Munkhjargal Gochoo

Department of Computer Science and Software Engineering,
College of Information Technology, United Arab Emirates University

## 1. Introduction

There are different types of robots. Because of different aspects such as size and capabilities and that’s why it is hard to define what a robot really is. People have different definitions for the word or term “robot”. Even among robotistic there is a disagreement on what determine a robot and what not. But science fiction has gave us what to expect and what a robot is capable of and gave us an insight of how it may look. So, what is really a robot? Here is a definition of what we think a robot is it not very narrow neither it is board:

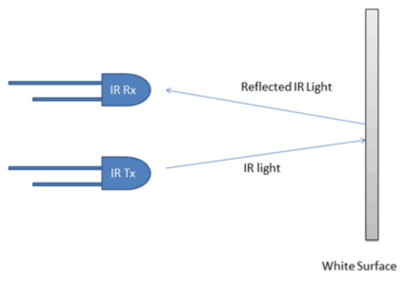
*A robot is an autonomous machine capable of sensing its environment, carrying out computations to make decisions, and performing actions in the real world.*

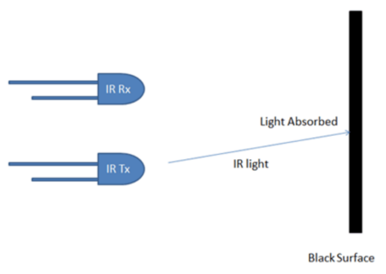
Robots come in many different forms and shapes, some of them are fixed and some are mobile. Mobile robots defined as robots that have a firm foundation allowing them to move freely in their respected environments. One of the most known mobile robots are the **Line Follower Robot** which we will be working on creating.

A mobile robot that can follow and identify a drawn line that is placed on the floor is known as **Line Follower Robot.** The path that the robot should follow which is usually pre-defined may be visible, such as a normal black line on white surface or it even can be invisible such as for an example a magnetic field. But how these mobile robots will follow these types of lines and the answer of that is using sensors to sense nearby obstacles in our case it is beneath our robot. The information and data will be transferred to the CPU using a specific transition bus, as a result the appropriate commands will be chosen by the processor to send them to the driver, and that will lead the line follower robot to follow the path.

## 2. Problem Statement:

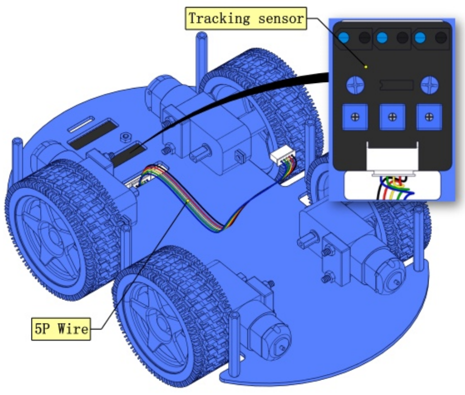
The main objective of this project is to make our own mobile robot (Line follower robot) being able to reach the fastest between the competitors by transversing through the given track that has been given to us by the professor, making sure that our robot being autonomous thought the process, while gaining point each time we reach a milestone, but by doing that challenges rises such as maximizing and maintain speed evetime the robot makes a turn or turns to any corner, and most importantly making sure that our robot stays on track.





## Tasks:

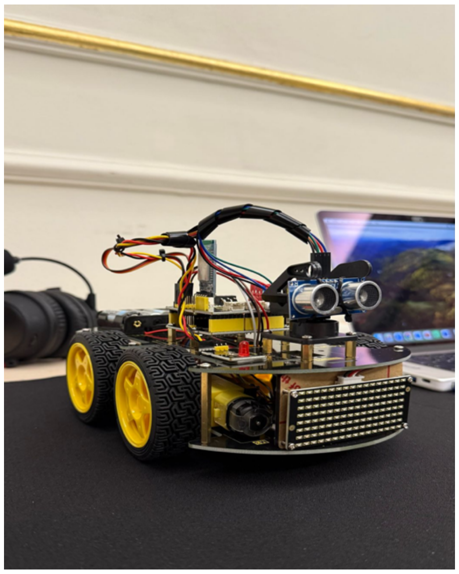
- Designing the robot's mechanical part or body
- Defining the robots' kinematics
- Designing the robot's control system

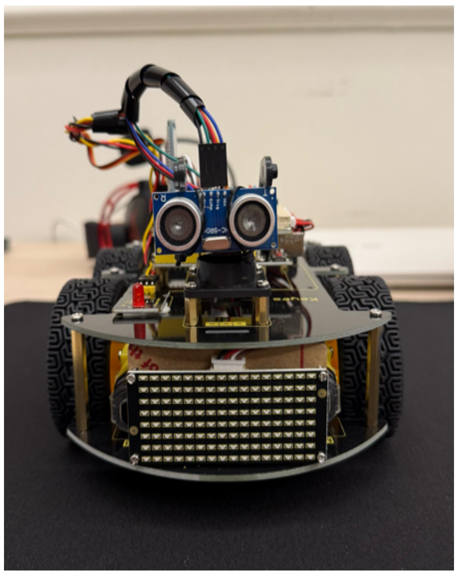


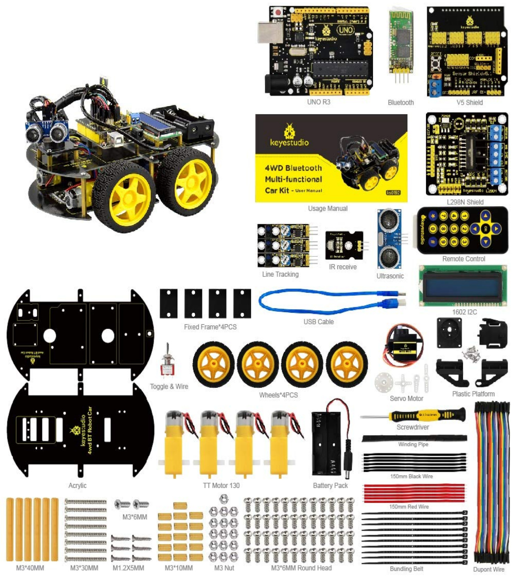
## 3. Our Robot: (System Design)

Our robot is 4WD Keyestudio Bluetooth Multi-Functional Car, is a platform built with a microcontroller. It is built around the ATmega-328 chip. The device is for education. It lets people learn and experiment with Arduino. The robot handles several things, such as line tracking, obstacle avoidance, infrared remote control, Bluetooth remote control as well as distance measurement. The path that the robot should follow which is usually pre-defined may be visible, such as a normal black line on white surface, the robot contains 3 IR sensors at the bottom organized to read the given values, and 4 motors two at the right and two at the left which are all connected and operating according to given sensors values.

This set has different programs that hold someone's attention. It lets people add more external circuit modules to grow its ability. It makes learning Arduino more practical plus direct. People put less time on theory and more time on making working systems. The robot and its important parts are in the following pictures:

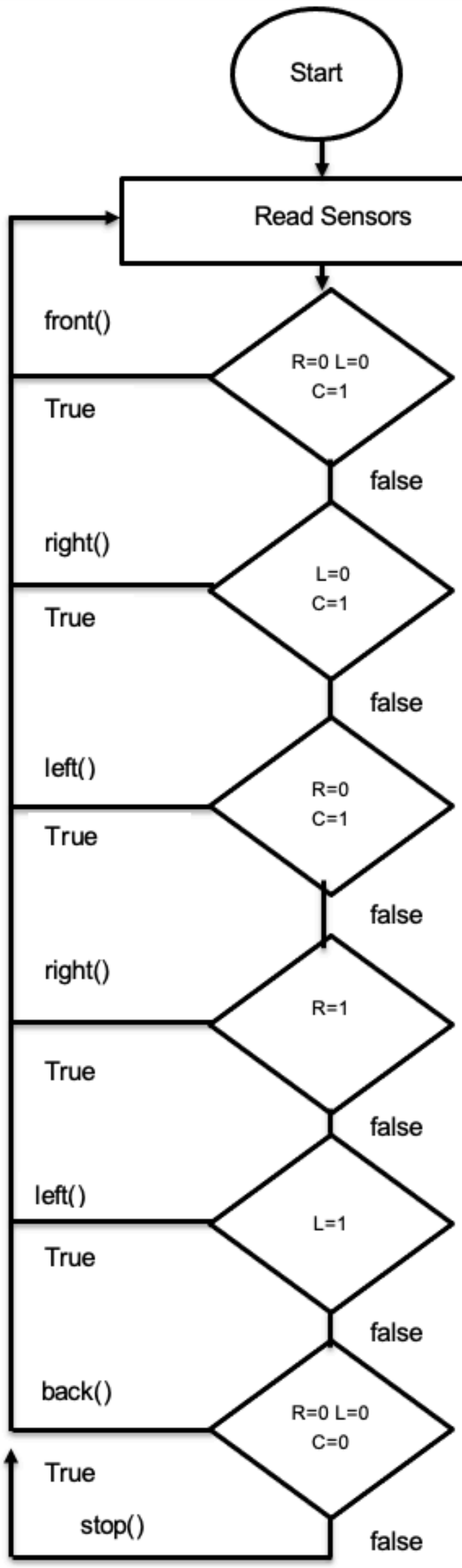






**YouTube link** to a demonstration video of a robot completing the steps. ( https://youtu.be/O5Nb6DHOR-o?si=MI4pFXVdRubqZ3yZ )

Flow Chart



## 4. Design Analysis

The function that we uses to analys our line following robot:

1. front():
   This function sets both left and right motors to turn with PWM of 60 and move forward when sensor read that it's on the line.
2. right():
   This function increases left wheels PWM and reverses their direction, in which enabling the sharper turns by allowing the wheels to accelerate in the opposite direction.
3. left():
   This function increases right wheels PWM and reverses their direction, in which enabling the sharper turns by allowing the wheels to accelerate in the opposite direction.
4. back():
   If the robot goes out off the track, this function makes it reverse so it returns to the path and realign.
5. stop():
   when robot reached the end of the track, it helps it to stop
6. tracking():
   This function combines all the others. In which it uses them including to the built-in sensors trace the line and follow it.
   Example how sensors are used: if the left and middle sensor detect the line, the robot will turn left to stay centered. Based on the given code it can handle all possible sensor combinations.

## 5. Conclusion

As a conclusion, our team has successfully been able to create and implement a Line Following robot that was able to follow the specific line that have been given while maintain a good amount of speed and precision, also being able to avoid obstacles in the way. We may have faced many challenges during our journey of creating a mobile robot but thanks to our team effort and skills we were able to overcome these challenges and even learn more from them. We truly believe that our Line Following Robot was a great success hoping that it will inspire others to visit this vast world of robotics and even one day creating their own robot, either it was mobile or not.

In this project our team has truly learned and gained experience of the robotics field, that will hopefully help us and inspire us to do more related robotics projects in the near future, but robotics wasn't the only thing we learned during this project we also improved our team work and way of communication, trusting each other to do their own part and solving different problems. At the end we are proud of what we accomplished and very happy that we have been part of this project.

## 6. References

1. **Keyestudio Wiki for V2.0 Kit for Arduino**
   Keyestudio. (n.d.). *V2.0 Kit for Arduino – Keyestudio Wiki*. Retrieved April 18, 2025, from https://wiki.keyestudio.com
2. Silicon Labs. (n.d.). *USB to UART Bridge VCP drivers*. Silicon Laboratories. Retrieved April 18, 2025, from https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers
3. **Arduino Functions Reference**
   Arduino. (n.d.). *Function | Arduino Reference*. ArduinoGetStarted. Retrieved April 18, 2025, from https://www.arduinogetstarted.com
4. **Atmega328P Microprocessor Datasheet**
   Microchip Technology Inc. (2016). *Atmel-7810: Automotive Microcontrollers – ATmega328P Datasheet* [PDF]. Retrieved from https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

## 7. Appendix

## First Code:

```
#define ML_Ctrl 4     //define direction control pin of B motor
#define ML_PWM 5   //define PWM control pin of B motor
#define MR_Ctrl 2    //define direction control pin of A motor
#define MR_PWM 9   //define PWM control pin of A motor
const int sensor_l = 8;//define the pin of left line tracking sensor
const int sensor_c = 7;//define the pin of middle line tracking sensor
const int sensor_r = 11;//define the pin of right line tracking sensor
int l_val,c_val,r_val;//define these variables
void setup() {
  Serial.begin(9600);//start serial monitor and set baud rate to 9600
  pinMode(ML_Ctrl, OUTPUT);//set direction control pin of B motor
  pinMode(ML_PWM, OUTPUT);//set PWM control pin of B motor to OUTPUT
  pinMode(MR_Ctrl, OUTPUT);//set direction control pin of A motor to OUTPUT
  pinMode(MR_PWM, OUTPUT);//set PWM control pin of A motor to OUTPUT
  pinMode(sensor_l,INPUT);//set the pins of left line tracking sensor to INPUT
  pinMode(sensor_c,INPUT);//set the pins of middle line tracking sensor to INPUT
  pinMode(sensor_r,INPUT);//set the pins of right line tracking sensor to INPUT
}
void loop()
{
  tracking(); //run main program
}

void tracking()
{
  l_val = digitalRead(sensor_l);//read the value of left line tracking sensor
  c_val = digitalRead(sensor_c);//read the value of middle line tracking sensor
  r_val = digitalRead(sensor_r);//read the value of right line tracking sensor
  if(c_val == 1)//if the state of middle one is 1, which means detecting black line
  {
    front();//car goes forward
  }
  else
  {
    if((l_val == 1)&&(r_val == 0))//if only left line tracking sensor detects black
trace
    {
      left();//car turns left
    }
else if((l_val == 0)&&(r_val == 1))//if only right line tracking sensor detects black
trace
    {
      right();//car turns right
    }
    else// if left and right line tracking sensors detect black trace or they don’t
read
    {
      Stop();//car stops
    }
  }
}
void front()//define the status of going forward
{
  digitalWrite(ML_Ctrl,HIGH);//set direction control pin of B motor to HIGH
  analogWrite(ML_PWM,60);//set PWM control speed of B motor to 70
  digitalWrite(MR_Ctrl,HIGH);//set direction control pin of A motor to HIGH
  analogWrite(MR_PWM,60);//set PWM control speed of A motor to 70
}
void back()//define the state of going back
{
  digitalWrite(ML_Ctrl,LOW);//set direction control pin of B motor to LOW
  analogWrite(ML_PWM,60);//set PWM control speed of B motor to 200
  digitalWrite(MR_Ctrl,LOW);//set direction control pin of A motor to LOW
  analogWrite(MR_PWM,60);//set PWM control speed of A motor to 200
}
void left()//car turns left
{
  digitalWrite(ML_Ctrl,LOW);//set direction control pin of B motor to LOW
  analogWrite(ML_PWM,60);//set PWM control speed of B motor to 200
  digitalWrite(MR_Ctrl,HIGH);//set direction control pin of A motor to HIGH level
  analogWrite(MR_PWM,60);//set PWM control speed of A motor to 200
}
void right()//define the right-turning state
{
  digitalWrite(ML_Ctrl,HIGH);//set direction control pin of B motor to HIGH level
  analogWrite(ML_PWM,60);//set PWM control speed of B motor to 200
  digitalWrite(MR_Ctrl,LOW);//set direction control pin of A motor to LOW
  analogWrite(MR_PWM,60);//set PWM control speed of A motor to 200
}
void Stop()//define the state of stop
{
  analogWrite(ML_PWM,0);//set PWM control speed of B motor to 0
  analogWrite(MR_PWM,0);//set PWM control speed of A motor to 0
}//*********************************************************
```

## Second Code:

#define Rpwm_pin 10//enable B

```
#define Lpwm_pin 5//enable A
 int M1_Speed = 80; // speed of motor 1
 int M2_Speed = 80; // speed of motor 2
 int LeftSpeed = 250;  // Left  Speed
 int RightSpeed = 250; // Right Speed
#define IN1 9
#define IN2 8
#define IN3 7
#define IN4 6

 void setup() {//input output setup
  pinMode(IN1,OUTPUT);//set motors as outputs
  pinMode(IN2,OUTPUT);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);

  pinMode(Rpwm_pin,OUTPUT);//set enables as outputs
  pinMode(Lpwm_pin,OUTPUT);

  pinMode(A0, INPUT); // initialize Left sensor setup as input
  pinMode(A1, INPUT); // initialize Right sensor setup as input

}

void loop() {

  int LS = digitalRead(A0);//reading the value of the left IR sensor
  int RS = digitalRead(A1);//reading the value of the right IR sensor

if(RS==0 && LS==0) {// if both sensors detect black, the car moves forward
    forward();
}

  else if(RS==0 && LS==1) {// if only the right sensor detects black, the car moves left
     left();
 }

  else if(RS==1 && LS==0) {// if only the left sensor detects black, the car moves right

    right();
}

  else if(RS==1 && LS==1) {// if both sensors detect white, the car stops
    Stop();
 }
}
```

```
// motion controling functions
void forward()
{
digitalWrite(IN1, HIGH);
digitalWrite(IN2, LOW);
digitalWrite(IN3, HIGH);
digitalWrite(IN4, LOW);
analogWrite(Rpwm_pin, M1_Speed);
analogWrite(Lpwm_pin, M2_Speed);
}
void backward()
{
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
digitalWrite(IN3, LOW);
digitalWrite(IN4, HIGH);
analogWrite(Rpwm_pin, M1_Speed);
analogWrite(Lpwm_pin, M2_Speed);
}
void Stop()
{
digitalWrite(IN1, LOW);
digitalWrite(IN2, LOW);
digitalWrite(IN3, LOW);
digitalWrite(IN4, LOW);
}
void right()
{
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
digitalWrite(IN3, HIGH);
digitalWrite(IN4, LOW);
analogWrite(Rpwm_pin, LeftSpeed);
analogWrite(Lpwm_pin, RightSpeed);
}
void left()
{
digitalWrite(IN1, HIGH);
digitalWrite(IN2, LOW);
digitalWrite(IN3, LOW);
digitalWrite(IN4, HIGH);
analogWrite(Rpwm_pin, LeftSpeed);
analogWrite(Lpwm_pin, RightSpeed);
}
```