A Mini Project Report

on

Obstacle Avoiding Robot Car

submitted by

Prafulla Pal
(2008410300041)

BACHELOR OF TECHNOLOGY

in

ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRONICS ENGINEERING

RAJKIYA ENGINEERING COLLEGE

CHURK, SONBHADRA, 231 206 (UP), India

December, 2021

DECLARATION

I undersigned hereby declare that the project report **("Obstacle Avoiding Robot Car"),** submitted is a bonafide work done by me under supervision of **(Dr. Himanshu Katiyar)**. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission.

Signature	Signature
Name of the student	Name of the student

DEPARTMENT OF ELECTRONICS ENGINEERING
RAJKIYA ENGINEERING COLLEGE
CHURK, SONBHADRA, 231 206 (UP)
Email: sonbhadrarec@gmail.com, website: http://recsonbhadra.ac.in, Fax: 0544-252002 Tel: 05444-252003

CERTIFICATE

This is to certify that the Report entitled **"Obstacle Avoiding Robot Car"** submitted by **"Prafulla Pal"** is a bonafide record of the mini-project work carried out by him/her under my guidance and supervision. This report in any form has not been submitted to any other university or Institute for any purpose.

Signature

Name of the supervisor

Signature of Head of Department with seal

ACKNOWLEDGEMENT

We feel profound pleasure in bringing out this project report for which we have to go from pillar to post to make it a reality. This project work reflects contributions of many people with whom we had long discussions and without which it would not have been possible. We must first of all, express our heartiest gratitude to respected **Dr. Himanshu Katiyar (Head of Department, Department of Electronics Engineering)** for providing us all guidance to complete the project.

Last, but not least, my parents and friends are also an important inspiration for me. So, with due regards, I express my gratitude to them.

1 TABLE OF CONTENTS

2 TABLE OF FIGURES

3 ABSTRACT

Trajectory planning is one of the most important pivotal point in pick and place tasks done by robotic manipulators. In this work, we have presented a robot, which is compact, autonomous and fully functional.

This robot or a smartcar is built to sense any obstacle in its path, to avoid it and resume its running involving pre-computation of an obstacle free path. Ultrasonic sensors were adapted to implement a real-time obstacle avoidance system for wheeled robots, so that the robot can continually detect surroundings, avoid obstacles, and move toward the target area.

This model has tremendous applications in vacuum cleaners, avoiding concealed paths, parking systems, assembling automobiles and in chemical industries, in scientific exploration, emergency rescue and in other isolated environments.

We use an Arduino UNO with a Motor Shield along with Stepper Motors to make the car, and for sensing we incorporate an Ultrasonic Sensor which accurately and efficiently detects any obstacles in the smart car's path. The Arduino is coded such that the smartcar moves backward when an obstacle arises in front of it with a maximum limit of 50cms in ideal testing conditions.

Throughout the construction of this model, we educated ourselves to the Arduino coding language, the Motor Shield functionality, and comprehensively, with the working of an ultrasonic sensor and its features.

In conclusion, through this project, we aim to construct a model of a smartcar that is beneficial to the quotidian problems of the present generation.

4 INTRODUCTION

4.1 PROBLEMS ADDRESSED

- Need for a robot that can perform trajectory planning effectively
- Need for a system which can detect obstacles and move in a pre-computed path
- Need for the detection of obstacles that appear suddenly
- Need to minimize human risk regarding the upper limit of a human eye
- Need to assist the physically handicapped by incorporating cutting edge technologies in wheelchairs
- Need to address the need for upgrade from inaccurate sensors like CCD cameras, CMOS image sensors, laser light pens, global positioning systems, and so on
- Need for advanced mapping devices during exploration of unknown environments such as interplanetary exploration

4.2 SOLUTIONS OFFERED

- Robot detects obstacles and avoids them in the trajectory
- Robot uses an Ultrasonic Sensor which is capable of detecting obstacles which may appear suddenly, for instance, an animal in front of a wheelchair
- Robot can operate in the environment without much interference
- Robot can detect very minute details, which the human eye may neglect
- Robot can map various topographies and terrains

5 COMPONENTS USED

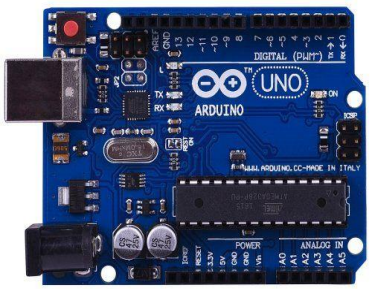


Figure 1 Arduino Uno Controller

Figure 2 DC BO Motor x2

Figure 3 Wheels x2

Figure 4 Castor Wheels

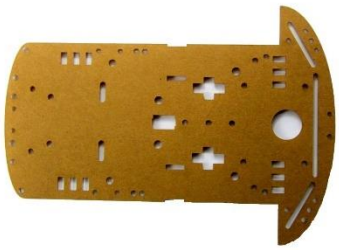


Figure 5 Chassis

Figure 6 9v Battery

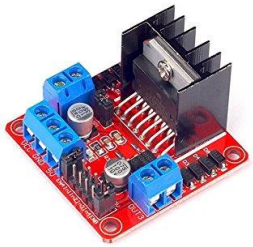


Figure 7 L298D Motor Driver

Figure 8 Ultrasonic sensor

Figure 9 Servo Motor

6 FEATURES OF THE COMPONENTS USED

6.1 ARDUINO UNO MICROCONTROLLER

An Arduino is an open-source microcontroller development board. The most common version of Arduino is the Arduino Uno. It is relatively cheap, plugs straight into a computer's USB port, and it is simple to setup and use when compared to other development boards.

Figure 10 Arduino Uno

6.1.1 Some of the key features of the Arduino Uno include:

- ***Open-source design:*** large community at Arduino.cc/forum/ of people using and troubleshooting it.
- ***Easy USB interface:*** The chip on the board plugs straight into your USB port and registers on your computer as a virtual serial port. This allows us to serially communicate which is an extremely easy protocol.
- ***Convenient power management*** and built-in voltage regulation. 12V can easily be regulated to both 5v and 3.3v.
- ***Easy-to-find*** and cheap, microcontroller.
- Countless number of ***hardware feature*** like timers, PMW pins, external and internal interrupts, and multiple sleep modes.

6.1.2 Some other specifications are

- A 16 MHz clock.
- 32KB of flash memory
- 13 digital pins and 6 analogue pins.
- ICSP connector to re-bootload your chip and for bypassing the USB port and interfacing the Arduino directly as a serial device.
- LED attached to digital pin 13 for and easy debugging of code.
- Reset button to reset the program on the chip.

6.2 ULTRASONIC SENSOR

Figure 11 HC-SR04

The Ultrasonic Sensor sends out a high-frequency sound pulse and then times how long it takes for the echo of the sound to reflect back. The sensor has two openings on its front.

- Tiny speaker to transmit opening ultrasonic waves
- Microphone to receive the ultrasonic waves

The ultrasonic sensor calculates distances by: -

The speed of sound is approximately 341 meters per second in air. The ultrasonic sensor uses this information along with the time difference between sending and receiving the sound pulse to determine the distance to an object. It uses the following mathematical equation:

$$\boldsymbol{Distance = \frac{Time \times Speed\ of\ Sound}{2}}$$

Time = the time between when an ultrasonic wave is transmitted and when it is received.

We divide this number by 2 since the sound wave has to travel to the object and back.

The limitation of an ultrasonic sensor is some objects might not be detected which are shaped or positioned in such a way that the sound wave bounces off the object, but are deflected away from the sensor.

It is also possible for the object to be too small to reflect enough of the sound wave back to the sensor to be detected.

Objects can absorb the sound wave all together (cloth, carpeting, etc.), which means that there is no way for the sensor to detect them accurately.

These are important factors to consider when designing and programming a robot using an ultrasonic sensor.

6.3 L298 Dual H-Bridge Driver

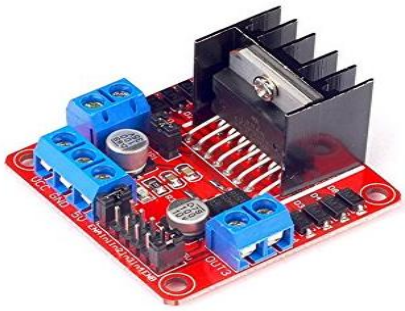


Figure 12 L298N Dual H-Bridge

- L298 is a high voltage and high current motor drive chip which receives TTL logic signals.
- They are mostly used when
    - It is needed to operate different loads like motors and solenoid etc where an H-Bridge is required.
    - High power motor driver is required.
    - Control unit can only provide TTL outputs.
    - Current control and PWM operable single-chip device are needed.
- It has two enable inputs to enable or disable the particular device attached at its output independently.

Thus, H-Bridge is basically used to control the rotating direction in DC motors.

6.4 DC BO Motor

Figure 13 DC Motor

Bo motor (Battery Operated) lightweight DC geared motor which gives good torque and rpm at lower voltages. You can get bo motor with varying rated speed. This motor can run at approximately 200 rpm when driven by a single Li-Ion cell. Great for battery operated lightweight robots. Here we are using 2 BO Motors.

An Electric DC motor is a device which converts electric energy into mechanical energy. The working of DC motor is based on the principle that when a current carrying conductor is placed in a magnetic field, it experiences a mechanical force.

The direction of mechanical force is given by Fleming's Left-hand Rule and its magnitude is given by $F = IBL$ Newton.

6.5 How Does L298N Motor Driver work with DC Motor?

Let's take a look at how the L298N H-Bridge motor driver works with DC motor.

In order to simplify things, I have drawn one set of the H-bridge driver with 4 switches.

Step 1: Let's apply a positive voltage to the top of the H-bridge and apply a negative voltage to the bottom.

Step 2: Now, when we close these two switches.

The positive is applied to the left side of the motor and the negative is applied to the right side. In this case, the motor will rotate **clockwise.**

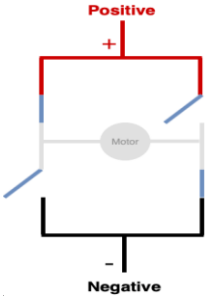


Figure 14 Connection in Step 2

Step 3: Vice versa, if you close the other two switches and leave the other two open.

Now, the positive is applied to the right side of the motor and the negative is applied to the other side. In this case, the motor will rotate **anti-clockwise.**

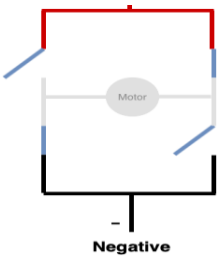


Figure 15 Connection in Step 3

7 PROCEDURE

7.1 GETTING THE HARDWARE READY

- Switch it on and off and then look in to making LED on the left of Arduino with the letter L next to it blink on and off for 2 seconds at a time.
- The Arduino Uno requires a male USB A to male USB B cable. Plug the USB cable in to the Arduino and your computer, the LED will start blinking. It is the default program stored on the chip.
- The USB cable powers the device. Arduinos can also run standalone by using a power supply in the bottom left of the board.
- Once programming is done it does not require to be constantly connected to a machine we can opt to power it separately. This is entirely dependent on the use case and circumstances we want to use the devices in.

7.2 GETTING THE SOFTWARE READY

- Download and install the Arduino IDE (integrated development environment). The Arduino Programmer is based on the Processing IDE and uses a variation of the C and C++ programming languages.
- Before doing anything in the Arduino programmer, set the board-type and serial port.
- To set the board, go to the following:
    - Tools – Boards
    - Select Arduino Uno
    - To set the serial port, go to the following:
        - Tools – Serial Port
        - Select the serial port that looks like:
        - /dev/tty.usbmodem [random numbers]

7.3 WRITING THE CODE

To write an Arduino sketch or a code, some basic knowledge about C and Embedded C should be known.

Some important things to look out for in an Arduino code are: -

- An Arduino program is called a sketch.

- All lines of code in an Arduino sketch are processed from top to bottom.
- Arduino sketches are typically broken into five parts-
    1. **Header** Usually, the sketch starts with a header that explains what the sketch is doing, and the author.
    2. **Global Variables** Next, global variables are defined. Often this is where constant names are given to the different Arduino pins.
    3. **Setup Routine** After the initial variables are set, the Arduino begins the setup routine. In the setup function, we set initial conditions of variables when necessary, and run any preliminary code that we only want to run once. This is where serial communication is initiated, which is required for running the serial monitor.
    4. **Loop** From the setup function, we go to the loop routine. This is the main routine of the sketch. This is not only where your main code goes, but it will be executed over and over, so long as the sketch continues to run.
    5. **Functions** Below the loop routine, there are often other functions listed. These functions are user-defined and only activated when called in the setup and loop routine.

All of that said, the only two parts of the sketch which are mandatory are the Setup and Loop routines.

- Code must be written in the Arduino Language, which is roughly based on C.
- Almost all statements written in the Arduino language must end with a semi-colon symbol. However, conditionals (such as if statements and for loops) do not need a semi-colon.
- Variables are storage compartments for numbers. You can pass values into and out of variables.
- Variables must be defined (stated in the code) before they can be used and need to have a data type associated with it.

8 SCHEMATIC DIAGRAM

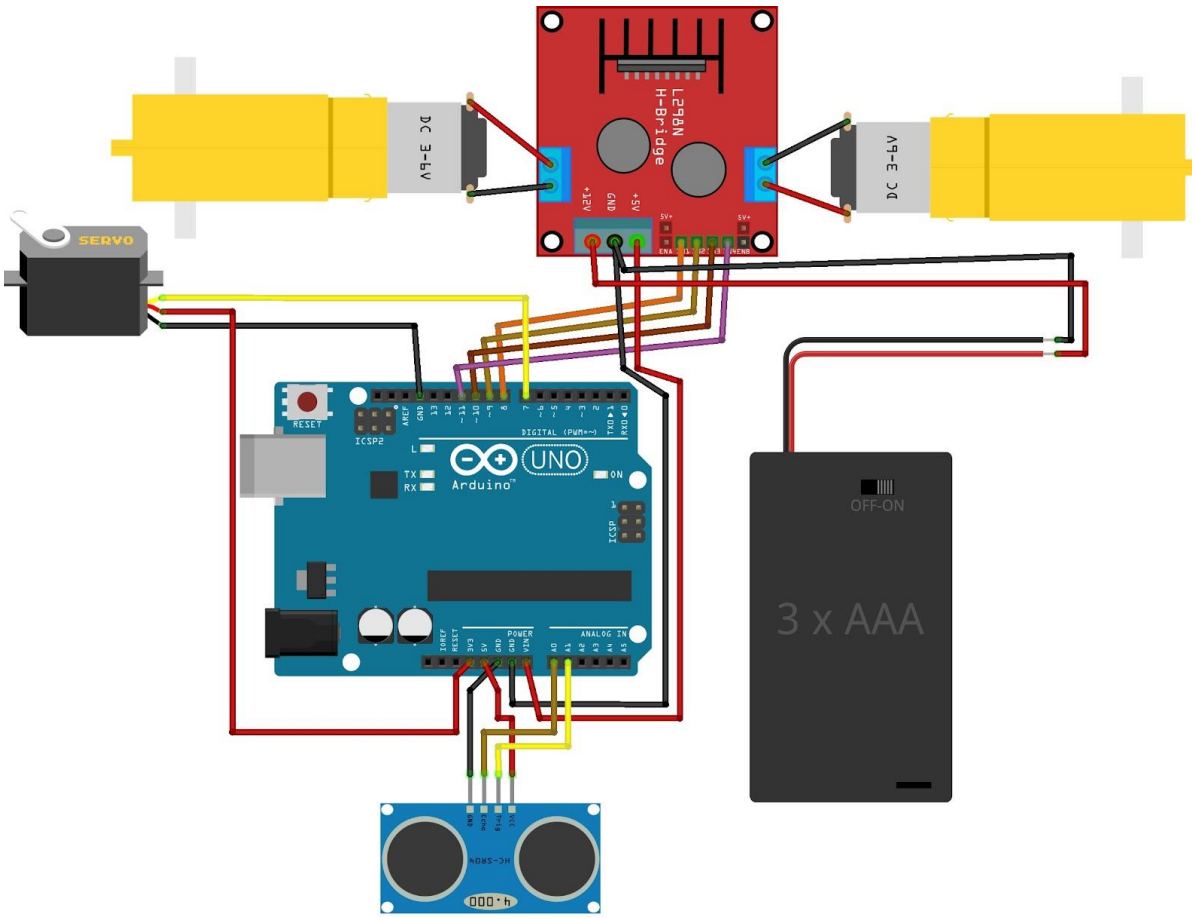
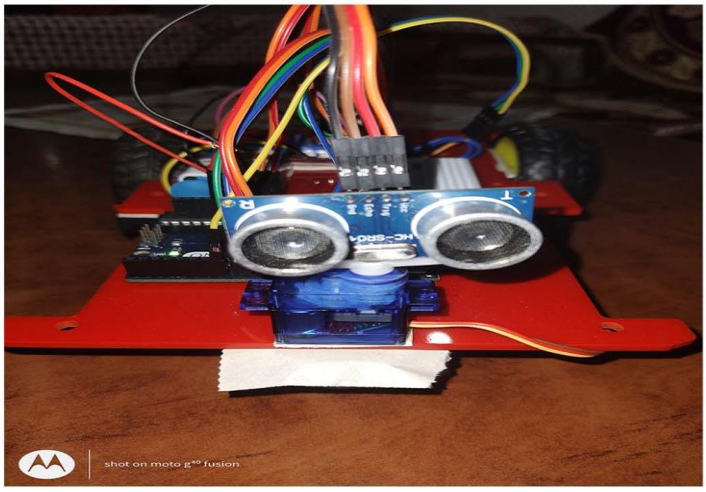


Figure 16 Schematic Diagram

9 WORKING

- The robot is switched on by giving it 9V DC power from an external battery. The motors start rotating and thus the robot starts moving forward.
- During this time, the ultrasonic sensors continuously keep calculating the distance between the robot and the reflective surface.
- This information gets processed by the ARDUINO.
- If the distance between the robot and the obstacle is less than the specified value, the robot changes its path (moves towards the back). Here, we have kept the minimum distance to be of 15 cm.
- This process continues forever and the robot keeps moving without danger.



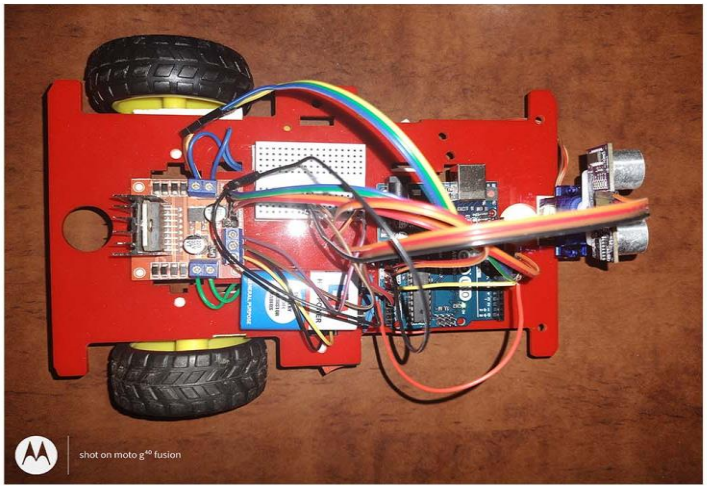


Figure 17 Working Model

Figure 18 Demonstration of Car turning when object is detected

10 APPLICATIONS

- The modification of this logic code is used in vacuum cleaners.
- This robot can be used for avoiding concealed paths, such as an industrial robot in a factory is expected to avoid workers so that it won't hurt them.
- It will be very useful in parking system.
- It can also be used in assembling automobiles and in chemical industries.
- If there is an obstacle in the root of the robot, it can detect and avoid it. Thus, it can move without having damaged by any obstacle which makes it more reliable.
- They have great importance in scientific exploration and emergency rescue, there may be places that are dangerous for humans or even impossible for humans to reach directly, then we should use robots to help us gather information to about their surrounding challenging environments.

10.1 FURTHER IMPLEMENTATIONS

- This technique can also be used as a vision belt for blind people by changing the ultrasonic sensor by a kinetic sensor which is a type of microwave sensor whose sensing range is very high and the output of this sensor vary in according to the object position changes. This technique enables blind people to navigate obstacles easily by placing three vibratos in left, right and the center of a belt named as Vision Belt.
- On top of obstacle avoiding robot temperature/ pressure sensors can be added to monitor the atmospheric conditions around. This is useful in places where the environment is not suitable for humans. Same technology can be used in various applications by modifying the program for example Line/ Path finder Robot, automatic vacuum cleaner etc.
- They can be used as services robots, for the Low-Cost Obstacle Avoidance Robot, purpose of household work and so many other indoor applications.

11 ARDUINO CODE

```
#include <Servo.h>          //Servo motor library. This is standard library
#include <NewPing.h>        //Ultrasonic sensor function library. You must
install this library

//our L298N control pins
const int LeftMotorForward = 8;
const int LeftMotorBackward = 9;
const int RightMotorForward = 10;
const int RightMotorBackward = 11;

//sensor pins
#define trig_pin A1 //analog input 1
#define echo_pin A0 //analog input 2

#define maximum_distance 200
boolean goesForward = false;
int distance = 100;

NewPing sonar(trig_pin, echo_pin, maximum_distance); //sensor function
Servo servo_motor; //our servo name


void setup(){

  pinMode(RightMotorForward, OUTPUT);
  pinMode(LeftMotorForward, OUTPUT);
  pinMode(LeftMotorBackward, OUTPUT);
  pinMode(RightMotorBackward, OUTPUT);

  servo_motor.attach(7); //our servo pin

  servo_motor.write(115);
  delay(2000);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
}

void loop(){
```

```
  int distanceRight = 0;
  int distanceLeft = 0;
  delay(50);

  if (distance <= 20){
    moveStop();
    delay(300);
    moveBackward();
    delay(400);
    moveStop();
    delay(300);
    distanceRight = lookRight();
    delay(300);
    distanceLeft = lookLeft();
    delay(300);

    if (distance >= distanceLeft){
      turnRight();
      moveStop();
    }
    else{
      turnLeft();
      moveStop();
    }
  }
  else{
    moveForward();
  }
    distance = readPing();
}

int lookRight(){
  servo_motor.write(50);
  delay(500);
  int distance = readPing();
  delay(100);
  servo_motor.write(115);
  return distance;
}

int lookLeft(){
  servo_motor.write(170);
  delay(500);
  int distance = readPing();
  delay(100);
  servo_motor.write(115);
  return distance;
```

```
  delay(100);
}

int readPing(){
  delay(70);
  int cm = sonar.ping_cm();
  if (cm==0){
    cm=250;
  }
  return cm;
}

void moveStop(){

  digitalWrite(RightMotorForward, LOW);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(RightMotorBackward, LOW);
  digitalWrite(LeftMotorBackward, LOW);
}

void moveForward(){

  if(!goesForward){

    goesForward=true;

    digitalWrite(LeftMotorForward, HIGH);
    digitalWrite(RightMotorForward, HIGH);

    digitalWrite(LeftMotorBackward, LOW);
    digitalWrite(RightMotorBackward, LOW);
  }
}

void moveBackward(){

  goesForward=false;

  digitalWrite(LeftMotorBackward, HIGH);
  digitalWrite(RightMotorBackward, HIGH);

  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(RightMotorForward, LOW);

}

void turnRight(){
```

```
  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(RightMotorBackward, HIGH);

  digitalWrite(LeftMotorBackward, LOW);
  digitalWrite(RightMotorForward, LOW);

  delay(500);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(RightMotorForward, HIGH);

  digitalWrite(LeftMotorBackward, LOW);
  digitalWrite(RightMotorBackward, LOW);



}

void turnLeft(){

  digitalWrite(LeftMotorBackward, HIGH);
  digitalWrite(RightMotorForward, HIGH);

  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(RightMotorBackward, LOW);

  delay(500);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(RightMotorForward, HIGH);

  digitalWrite(LeftMotorBackward, LOW);
  digitalWrite(RightMotorBackward, LOW);
}
```

12 REFERENCES

- A. Davids, "Urban Search and Rescue Robots: From Tragedy to Technology", IEEE Intelligent systems 17.2, 2002., pp. 81-83.
- T. Rakib, and M.A. Rashid Sarkar, "Design and fabrication of an autonomous fire fighting robot with multisensor fire detection using PID controller.", 5th International Conference on IEEE Informatics, Electronics and Vision (ICIEV), 2016.
- T. Lozano-Perez, "Autonomous robot vehicles", Eds. Ingemar J. Cox, and Gordon T. Wilfong, Springer Science & Business Media, 2012.
- G. Song, Y. Kaijian, Z. Yaoxin, and C. Xiuzhen, "A surveillance robot with hopping capabilities for home security", IEEE Transactions on Consumer Electronics 55.4, 2009.
- H. Durrant-Whyte, and T. Bailey, "Simultaneous localization and mapping: part I.", EEE robotics & automation magazine 13.2, 2006, pp. 99-110.
- HCSRO4 datasheet: http://www.micropik.com/PDF/HCSR04.pdf
- H. Choset, K. Nagatani, and N.A. Lazar, "The arc-transversal median algorithm: a geometric approach to increasing ultrasonic sensor azimuth accuracy", IEEE Transactions on Robotics and Automation 19.3, 2003, pp. 513-521.
- R.O. Duda, P.E. Hart, and D.G. Stork, "Pattern classification", John Wiley & Sons, 2012.
- R.C. de Amorim, and C. Hennig, "Recovering the number of clusters in data sets with noise features using feature rescaling factors", Information Sciences 324, 2015, pp. 126-145.
- B. lias, S. A. A. Shukor, A. H. Adom, M. F. Ibrahim andS. Yaacob, "A novel indoor mobile robot mapping with USB-166 ultrasonic sensor bank and NWA optimization algorithm," 2016 EEE Symposium on Computer Applications& Industrial Electronics (ISCAIE), Penang, 2016, pp. 189-194.