CHAPTER -1

INTRODUCTION

In our world today, ROBOTICS is a very interesting research area, which is fast growing as it is the simplest way for modifying modern day technology. Robotics plays amajor role in technology advancement, which is why I decided to work on the robotics field and design something intelligent to make human life simpler.

An autonomous robot is one which can move without any external interference in an environment which is unstructured and unknown to the robot. The robot is able to do this because of the software intelligence embedded inside it in order for it to be able to sense the environment, detect any obstacle which is in its path and move round the environment by avoiding these obstacles [1]. In the designing of an autonomous robot, there are many robotic designs that can be used. To make a selection of the design to be used, the main factor to be put into consideration is the physical environment in which the robot will be operated. Examples of autonomous robots: walking robots, drones, robotic cars, and snake robots.

1.1 AIM & OBJECTIVE

The aim of this project is to design and implement a robot car that is able to move around an unknown environment without running into obstacles in its path.

The Objectives of the project are as follows:

- The robot car should have the capacity to detect obstacles in its path based on a predetermined threshold distance.
- After detection of an obstacle, the robot should be able to change its direction to a relatively open path by making an autonomous decision.

1.2 EMBEDDED SYSTEM

An Embedded System can be best described as a system which has both the hardware and software and is designed to do a specific task. A good example for an Embedded System, which many households have, is a Washing Machine. We use washing machines almost daily but wouldn’t get the idea that it is an embedded system consisting of a Processor (and other hardware as well) and software. Embedded Systems can not only be stand-alone devices like Washing Machines but also be a part of a much larger system.

1.3 PROGRAMMING IN EMBEDDED SYSTEMS

As mentioned earlier, Embedded Systems consists of both Hardware and Software. If we consider a simple Embedded System, the main Hardware Module is the Processor. The Processor is the heart of the Embedded System and it can be anything like a Microprocessor, Microcontroller, DSP, CPLD (Complex Programmable Logic Device) and FPGA (Field Programmable Gate Array).

1.4 REAL TIME APPLICATIONS IN EMBEDDED SYSTEMS

- Latest Smart Tv’s
- GPS Navigation System
- Almost all modern-day smart phones
- Missile Guidance System
- Space Exploration (Rovers)
- Automobiles (ABS, Airbags)
- Industries (Assembly Robots)
- Road Safety Systems (Traffic Monitoring and Collision Alert Systems) .

1.5 INTERNET OF THINGS (IOT)

The Internet of things (IOT) describes physical objects (or groups of such objects) that are embedded with sensors, processing ability, software, and other technologies that connect and exchange data with other devices and systems over the Internet or other communications networks. Internet of Things has been considered a misnomer because devices do not need to be connected to the public internet, If they only connected to a network and be individually addressable. The field has evolved due to the convergence of multiple technologies, including ubiquitous computing, commodity sensors, increasingly powerful embedded systems, and machine learning.

IoT is also used in healthcare systems. There are a number of concerns about the risks in the growth of IoT technologies and products, especially in the areas of privacy and security, and consequently, industry and governmental moves to address these concerns have begun, including the development of international and local standards,guidelines, and regulatory frame works.

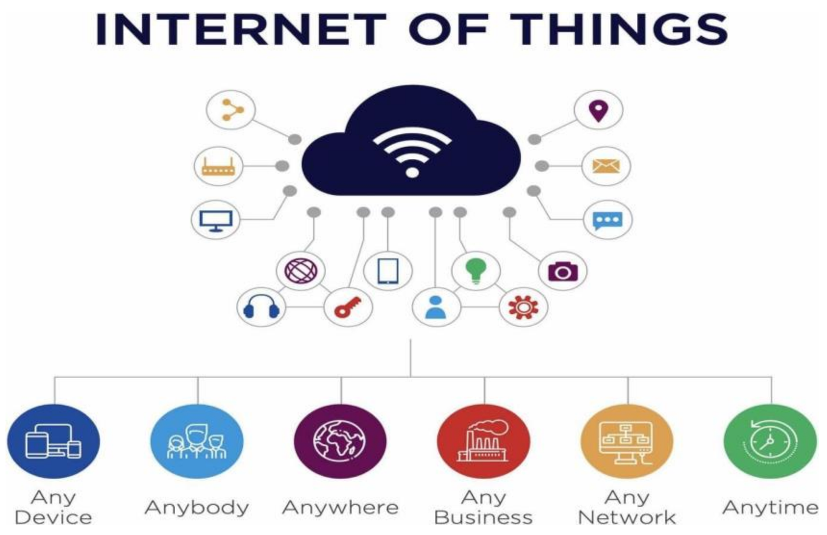

Fig 1.1 Internet of Things

1.6 APPLICATIONS OF IOT

- Smart City
- Self-driven Cars
- IOT Retail Shops
- Smart Grids
- Industrial Internet

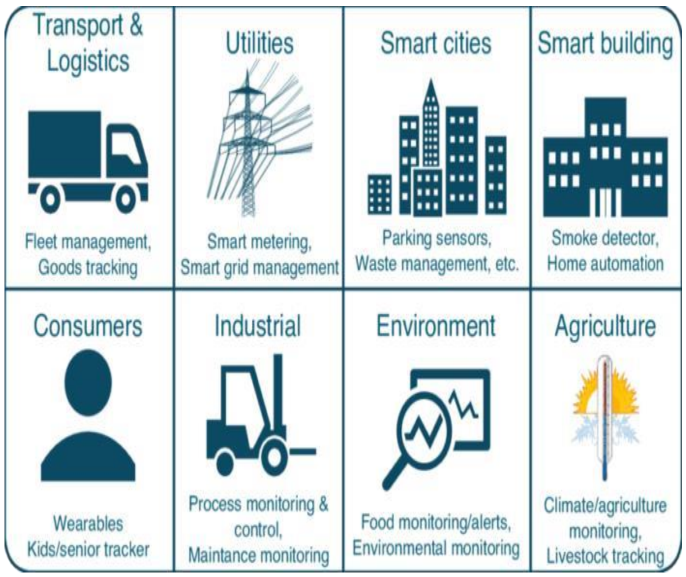
Transport & Logistics
Fleet management, Goods tracking
Utilities
Smart metering, Smart grid management
Smart cities
Parking sensors, Waste management, etc.
Smart building
Smoke detector, Home automation
Consumers
Wearables Kids/senior tracker
Industrial
Process monitoring & control, Maintance monitoring
Environment
Food monitoring/alerts, Environmental monitoring
Agriculture
Climate/agriculture monitoring, Livestock tracking

Fig.1.2 **Applications of IOT**

1.7 PROPOSED SYSTEM

The project proposes a autonomous robotic vehicle, It intelligently detects obstacles present on its path through the sensors, avoid it and take decision on the basis of internal code that we set for it.

CHAPTER-2
MATERIALS & ANALYSIS

2.1 LIST OF COMPONENTS USED

components used for the hardware implementation:

- 2x HC-SR04 Ultrasonic sensor
- 4x 3.7v battery
- 2x 3.7v battery connectors
- 1x SG90 Servo Motor
- 1x Arduino UNOR3
- 1x Power switch
- 1x Mica sheet(6*4)
- Jumper wires
- 4x DC Motors
- 1x L293D Motor Driver

2.2 ANALYSIS

HC-SR04 Ultrasonic Sensor

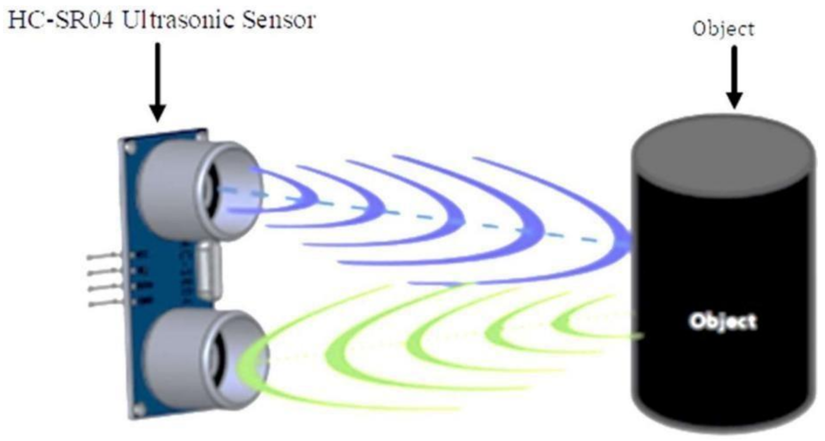

Fig 2.1 operation of the HC-SR04 Ultrasonic Sensor

The word ‘ultra’ means ‘beyond’ and sonic means ‘sound’. Combining the two of them together ultrasonic is a sound which is above the human hearing range (20 KHz).An Ultrasonic sensor is a sensor that can detect ultrasound waves by converting the waves into electric signals or vice versa.

- It has a 5V DC power supply
- Its Quiescent current is less than2mA
- Its working current is15mA
- Its effectual angle is less than 15degrees
- Its ranging distance is between 2cm – 400cm/1” –13ft
- Its Resolution is0.3cm
- Its measuring angle is30degrees
- Its Trigger pulse width is10uS
- Its dimension is 45mm x 20mm x15mm

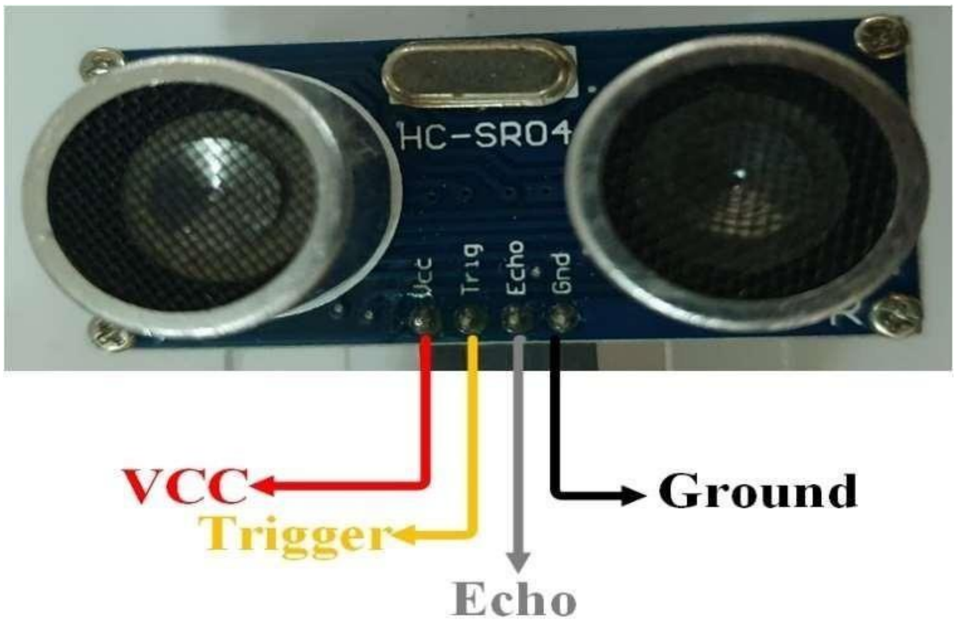

Fig 2.2 Ultrasonic sensor Pin out

SG-90 Servo Motor

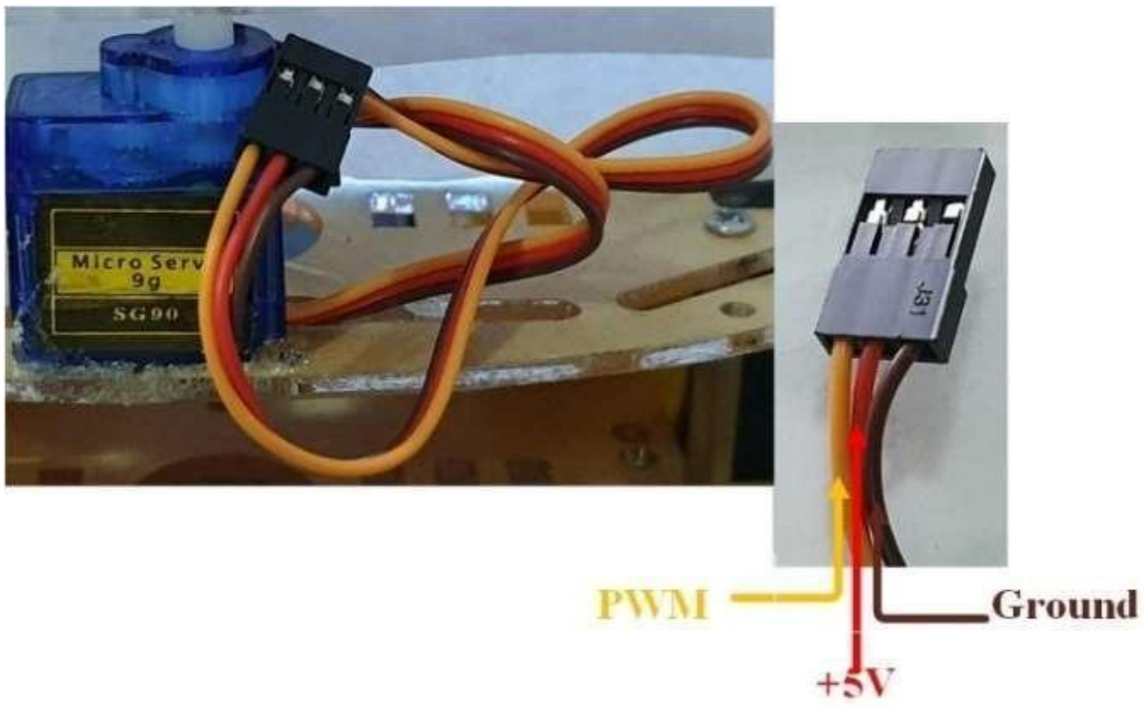

Figure 2.3 SG-90 Servo Motor.

A servo motor is an electrical device that pushes or rotates objects with high precision. If there is need for an object to be rotated as a specific angle or distance, then a servo motor is used. A servo motor consists of a motor that uses servo mechanism. The two types of servo motors are the

DC servo motors (DC powered) and the AC servo motor (AC powered) where the difference between the m is the input power. A very high torque can be obtained from a small and light weight servo motor which allows the servo motors to be used in applications like robots, to y cars, etc. The main reason why a servo motor is used is because of it sight angle precision, i.e. after rotating , it will stop and wait for the next instruction to happen unlike   a normal electric motor which rotates as long as it is being supplied power and stops rotating when power supply is turned off.

Arduino UNO R3

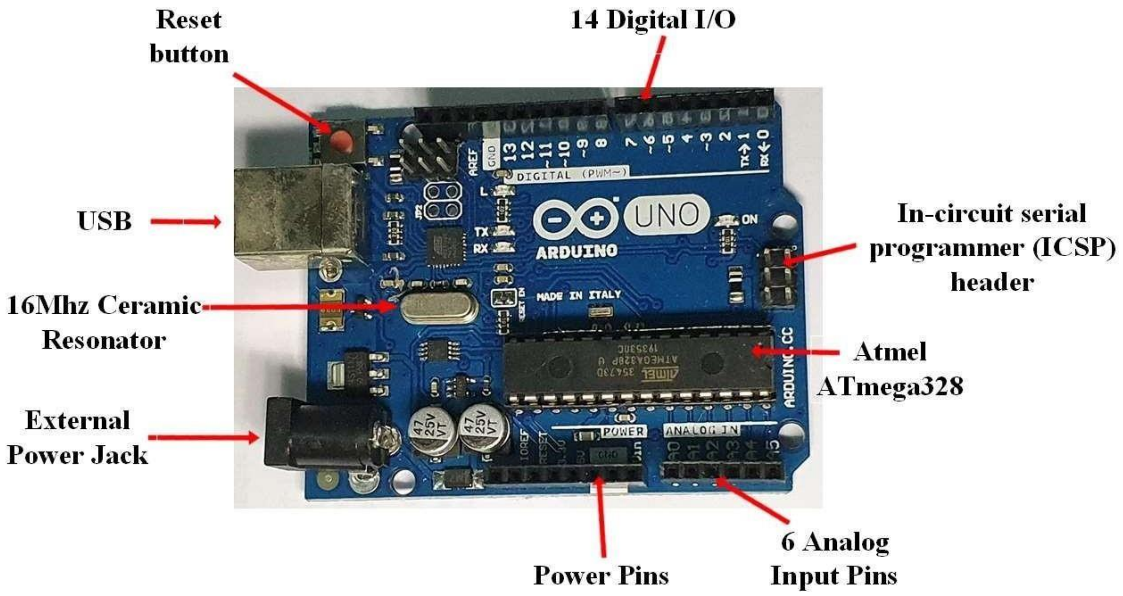

Figure 2.4 Arduino UNO R3 Board.

Arduino UNO Digital Input/ Output Overview

- It has 14 Digital Inputs and Outputs.
- It has a logic level of 5V.
- Its Sink/Source is 20mA per pin.
- It utilizes Universal Asynchronous Serial Receiver/Transmitter (UART) for which pin 0 is for receiving and pin 1 is for transmitting.
- It has 8-bit PWM.
- It has a Serial Peripheral Interface.

Arduino UNO Analog Inputs Overview

- Analog Input pins are A0, A1, A2, A3, A4,A5.
- Its sampled input is from 0V to 5V and it has a 10 bits resolution.
- The AREF pin can be used to adjust the upper limit of the input range.
- It has Two-wire Interface COM.
- Pin A4 is SDA pin for data.

Arduino UNO Power Overview

- External power supply voltage: 7V to 12VDC.
- USB power or externally via barrel jack connector.

Arduino UNO R3 Communication Overview

- It can communicate with a computer, another Arduino, shields, sensors.
- Asynchronous communication (No clock).
- UART TTL(5V)On board ATmegaU16 programmed to function as USB to serialchipATmegaU16 uses standard USB COM drivers so that no external drivers are required for communication with the board.
- Synchronous communication (Clock) uses SPI Pins 10, 11, 12, 13, and TWI pins A4 andA5.

Arduino Programming Overview

- The boot loader firmware comes already preinstalled with the Arduino UNO
- New sketches can be Uploaded via the USB.

- It has 32KB of Flash memory where the sketches are stored.
- It has 2KB of SRAM where the variables are stored until power cycled.
- It has 1KB of EEPROM which stores long term information.
- External programmer can be used to upload sketches via the In-circuit serial programmer pins (ICSP).

Arduino Programming Overview

- The boot loader firmware comes already preinstalled with the Arduino UNO
- New sketches can be Uploaded via the USB.

- It has 32KB of Flash memory where the sketches are stored.

- It has 2KB of SRAM where the variables are stored until power cycled.
- It has 1KB of EEPROM which stores long term information.
- External programmer can be used to upload sketches via the In-circuit serial programmer pins (ICSP).

DC Gear Motor

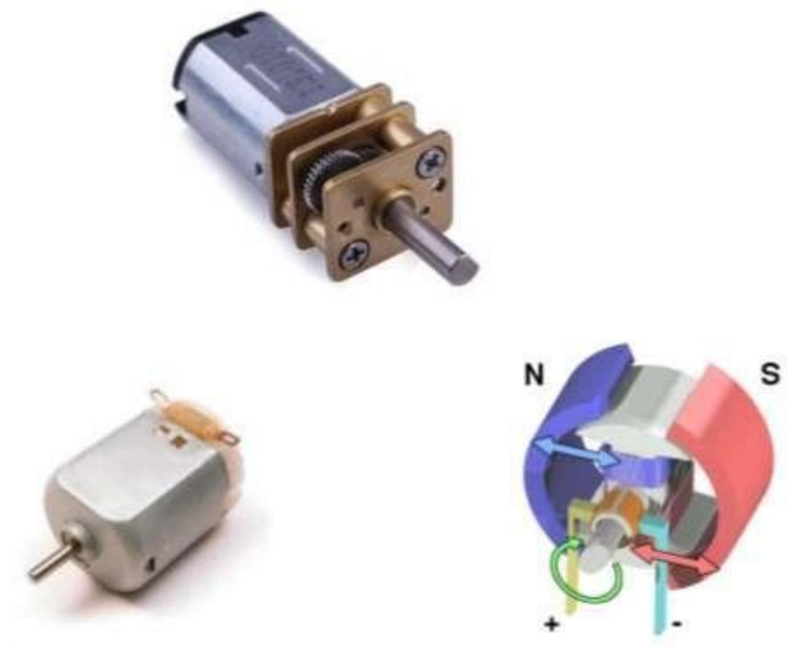

Figure 2.5 DC Gear Motor

DC Gear Motors are the most common kind of motors which convert electrical to mechanical energy. A gear motor consists of a gearbox and a motor. The addition of a gear head to a motor reduced the speed while increasing the torque output. Important parameters in gear motors include:

- Speed (rpm)
- Efficiency (%)
- Torque (lb.-in)

The factors to consider before selecting a gear motor for any application include:

- Speed
- Load
- Torque requirements

L293D Motor Driver Shield

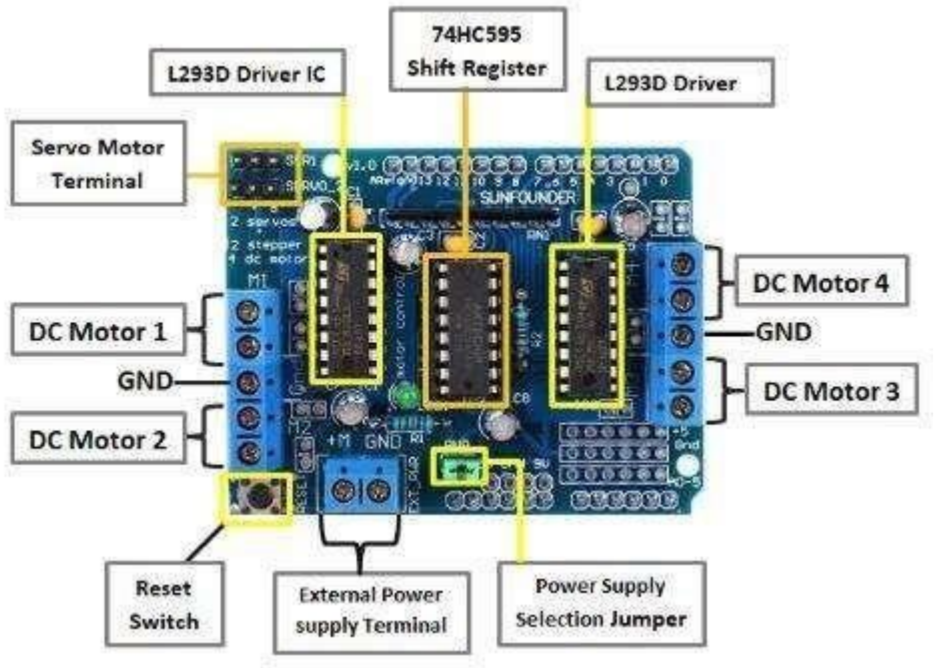

Fig 2.6 L293D Motor Driver Shield

The L293D is a high voltage, high current Integrated circuit which is used to drive DC motors with a power supply of up to 36V. This chip is able to supple a maximum of 600mA per channel. This chip is also known as a type of H-Bridge as it enables a voltage to be applied across a load in either direction to an output, e.g. a motor.

CHAPTER-3

DIAGRAMATIC REPRESENTATION

3.1 BLOCK DIAGRAM

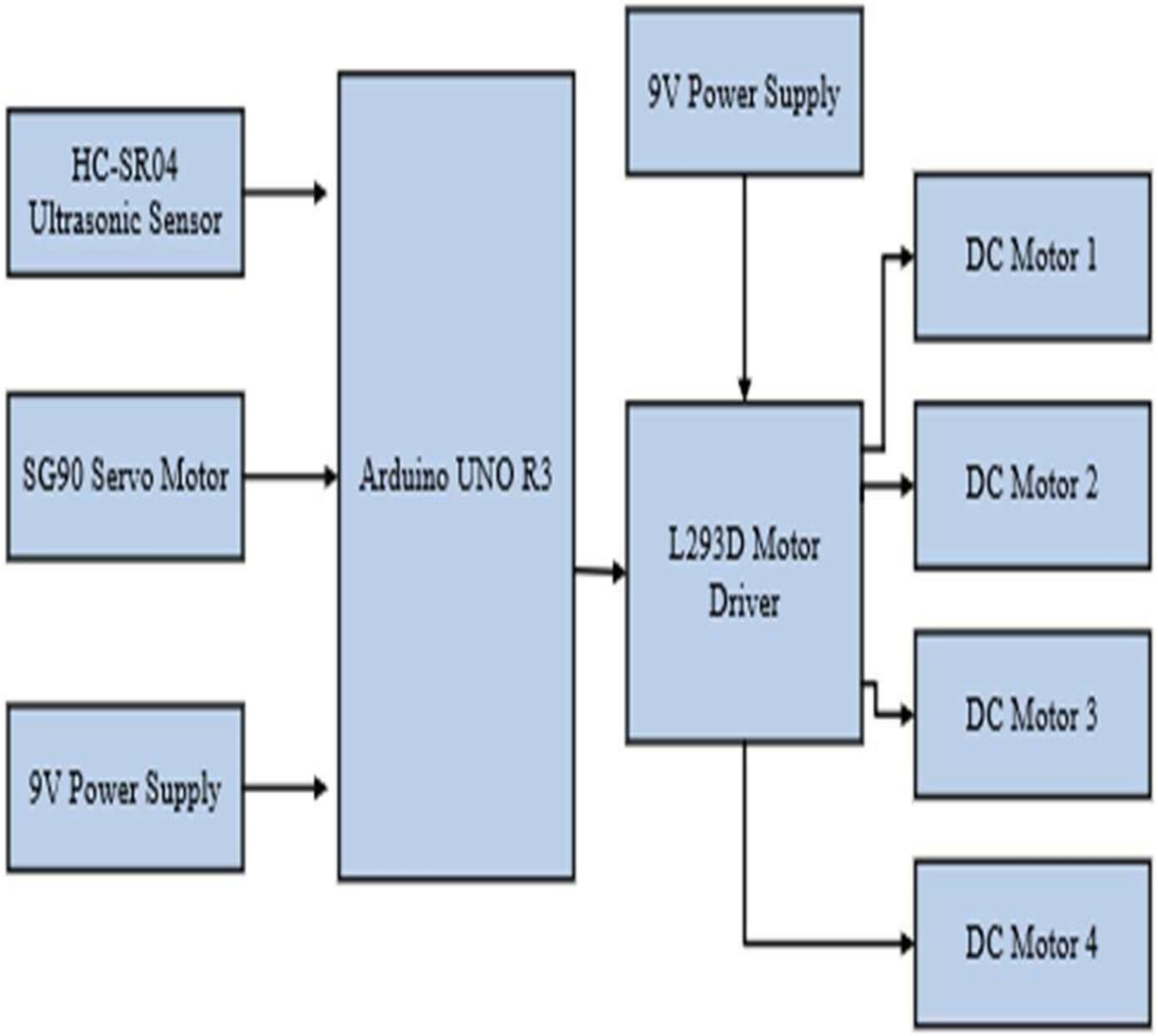

Fig 3.1 Block Diagram Obstacle Avoidance Robot

3.2 HARDWARE DIAGRAM

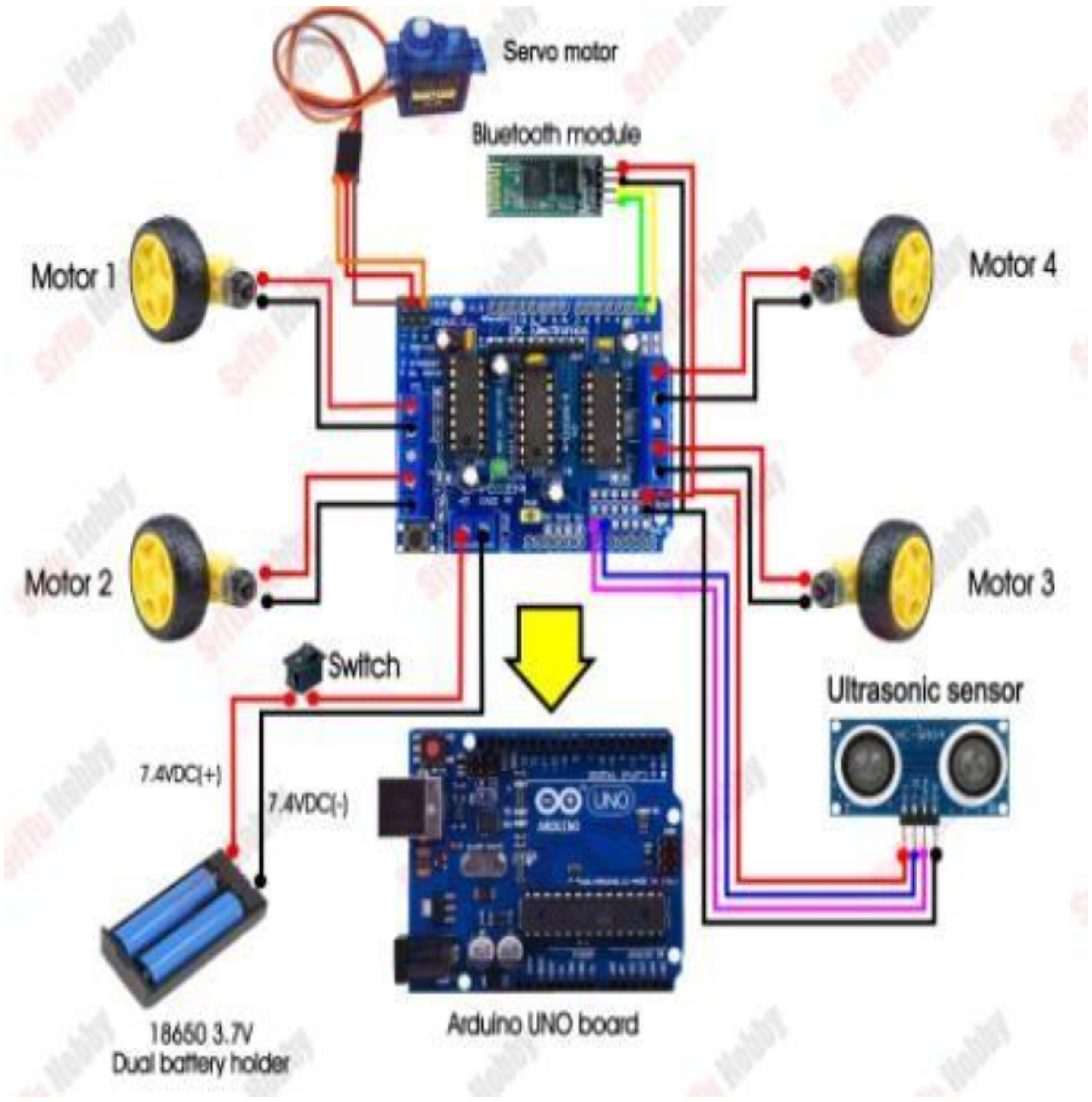

Fig 3.2 Circuit Diagram of Obstacle Avoidance Robot

CHAPTER-4
SOFTWARE REQUIREMENT

4.1 ARDUINO IDE

Fig 4.1 Arduino IDE Software

The open soursce Arduino software (IDE) makes it easy to write code and uplode it to the board. This software can be used with any Arduino board. The Arduino Microcontroller communicates with the PC via the USB connection. Data is transferred between the board and the PC bit by bit. An adaptor is used for power supply to the board and a USB programmer is used to burn the hardware program (written in Arduino IDE) into the board.

4.2 **PROGRAMMING**

The following steps to program the obstacle avoidance robot

STEP 1: Download the Arduino IDE (new version).
STEP 2: Install the IDE software in your PC.
STEP 3: GO to File and click new.
STEP 4: Start to program your robot in new tab.
STEP 5: Compile the code and verify the code was compile done.
STEP 6: Connect the Arduino Board in your PC & Uploading the code.

APPENDIX

Arduino coding for Obstacle avoidance robot

```
//Include the motor driver library
#include <AFMotor.h>
#include <Servo.h>
//Define the sensor pins
#define const int S1
#define const int S2
#define S1Trig A0
#define S2Trig A1
#define S1Echo A3
#define S2Echo A4
//Set the speed of the motors
#define Speed 160
#define MAX_DISTANCE 300

//Create objects for the motors
AF_DCMotor motor1(1);
AF_DCMotor motor2(2);
AF_DCMotor motor3(3);
AF_DCMotor motor4(4);
int sonar(MAX_DISTANCE); //TRIGGER_PIN, ECHO_PIN,
Servo myservo;
String voice;
```

```
 void setup() {
   Serial.begin(9600);
   myservo.attach(10);
   myservo.write(90);

//Set the Trig pins as output pins
pinMode(S1Trig, OUTPUT);
pinMode(S2Trig, OUTPUT);

//Set the Echo pins as input pins
pinMode(S1Echo, INPUT);
pinMode(S2Echo, INPUT);

   //Set the speed of the motors
   motor1.setSpeed(Speed);
   motor2.setSpeed(Speed);
   motor3.setSpeed(Speed);
   motor4.setSpeed(Speed);
 }
void loop() {
    if(S1 < 50){
    Stop();
    voice="";
    if(S2 < 50){
    Stop();
    voice="";
   }
  }

if(Serial.available()>0) {
voice="";
   delay(2);
   voice = Serial.readString();
   delay(2);
   Serial.println(voice);
   if (voice == "turn left") {
     left();
   }else if (voice == "left") {
     left();
   }else if(voice == "turn right") {
     right();
   }else if(voice == "right") {
     right();
   }
 }

 while(voice == "move forward") {
   forward();
 }
 while(voice == "move backward") {
   backward();
 }
 }
//Get the sensor values
int S1() {
   //pulse output
   digitalWrite(S1Trig, LOW);
   delayMicroseconds(4);
   digitalWrite(S1Trig, HIGH);
   delayMicroseconds(10);
   digitalWrite(S1Trig, LOW);

   long t = pulseIn(S1Echo, HIGH);//Get the pulse
   int cm = t / 29 / 2; //Convert time to the distance
   return cm; // Return the values from the sensor
 }

 //Get the sensor values
 int S2() {
   //pulse output
   digitalWrite(S2Trig, LOW);
   delayMicroseconds(4);
   digitalWrite(S2Trig, HIGH);
   delayMicroseconds(10);
   digitalWrite(S2Trig, LOW);
```

```
  long t = pulseIn(S2Echo, HIGH);//Get the pulse
  int cm = t / 29 / 2; //Convert time to the distance
  return cm; // Return the values from the sensor
}

/******************Motor functions*********************/
void forward() {
  motor1.setSpeed(255);
  motor1.run(FORWARD);
  motor2.setSpeed(255);
  motor2.run(FORWARD);
  motor3.setSpeed(255);
  motor3.run(FORWARD);
  motor4.setSpeed(255);
  motor4.run(FORWARD);
}

void backward() {
  motor1.setSpeed(255);
  motor1.run(BACKWARD);
  motor2.setSpeed(255);
  motor2.run(BACKWARD);
  motor3.setSpeed(255);
  motor3.run(BACKWARD);
  motor4.setSpeed(255);
  motor4.run(BACKWARD);
}
```

```
void right() {
  myservo.write(0);
  delay(500);
  myservo.write(90);
  delay(500);
  motor1.run(FORWARD);
  motor1.setSpeed(255);
  motor2.run(FORWARD);
  motor2.setSpeed(255);
  motor3.run(BACKWARD);
  motor3.setSpeed(255);
  motor4.run(BACKWARD);
  motor4.setSpeed(255);
  delay(700);
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  motor3.run(RELEASE);
  motor4.run(RELEASE);
}

void left() {
  myservo.write(180);
  delay(500);
  myservo.write(90);
  delay(500);
  motor1.run(BACKWARD);
  motor1.setSpeed(255);
  motor2.run(BACKWARD);
  motor2.setSpeed(255);
  motor3.run(FORWARD);
  motor3.setSpeed(255);
  motor4.run(FORWARD);
  motor4.setSpeed(255);
  delay(700);
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  motor3.run(RELEASE);
  motor4.run(RELEASE);
}

void Stop() {
  motor1.run(RELEASE);
  motor2.run(RELEASE);
  motor3.run(RELEASE);
  motor4.run(RELEASE);
}
```

HARDWARE IMPLEMENTION

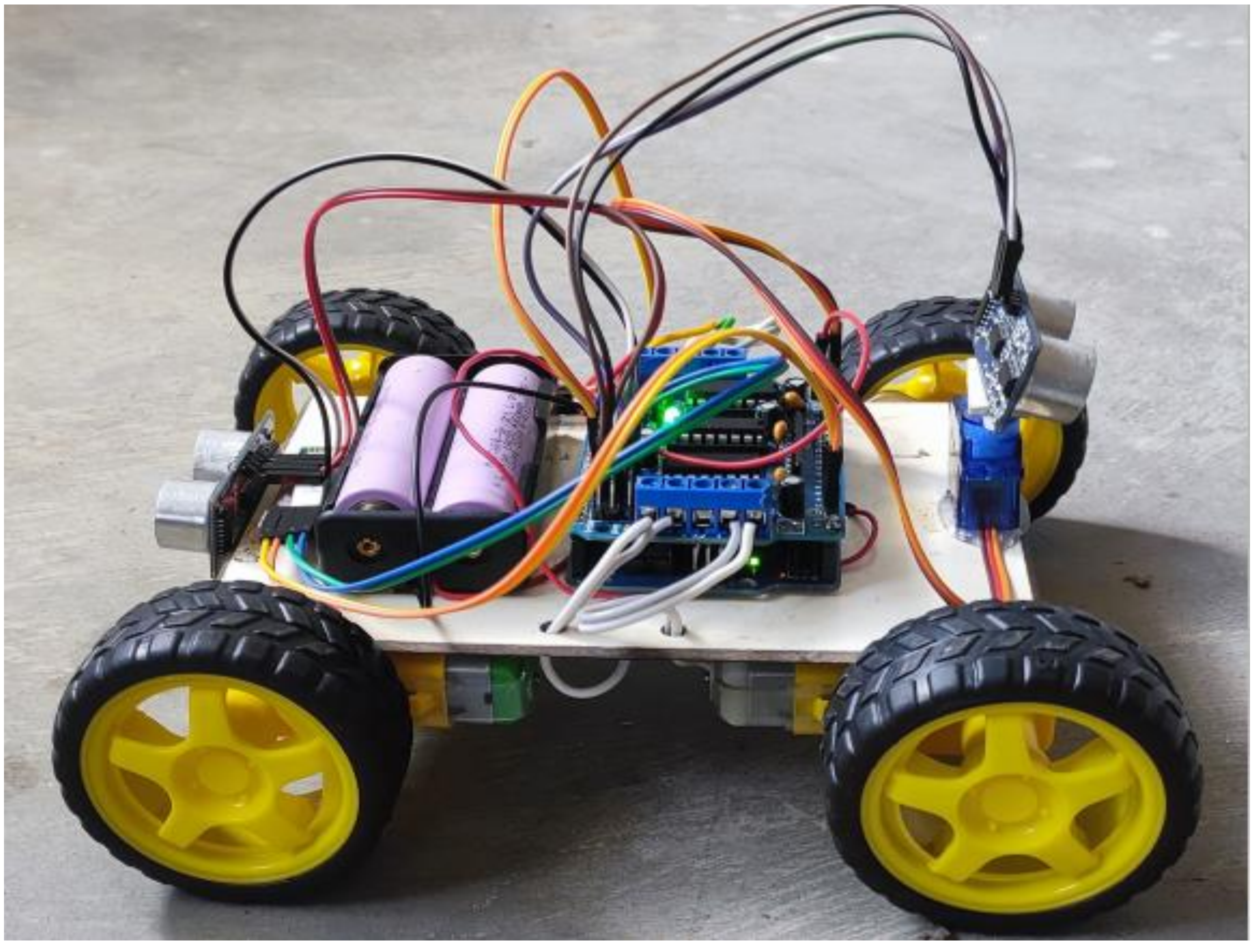

Fig 4.2 Hardware Implementation

After uploading the code, you can make the connections to make the obstacle avoidancerobot and the test its working properly.

CONCLUSION

Today we are in a world of robotics and we use different types of robots daily in our life. This “Obstacle Avoidance Robot Car” project is proved using the Ultrasonic sensor for detecting objects, Motor Driver Shield for driving the DC motors, DC motors for movement of the wheels of the robot with the help of the Arduino Microcontroller. The factors which affect the accuracy of the designed robot include the environment the robot was tested and the number of present obstacles in the test space. These factors mainly affected the sensor which means that the accuracy of the robot is dependent on the sensors. For obstacle detection, three ultrasonic distance sensors were used that provided a wider field of detection. The robot is fully autonomous and after the initial loading of the code, it requires no user intervention during its operation.

REFERENCE

[1] Abayomi O. Agbeyangi ard. ”THE JOURNAL OF COMPUTER SCIENCE ANDITS APPLICATIONS”Vol. 27, No. 1 June 2020.

[2] B. Rakesh , “Journal of Emerging Technologies and Innovative Research”, Vol.8, Issue June 26,2021.

[3] Dr. Bilkis Jamal Ferdosi, Global Journal of Researches in Engineering: HRobotics & Nano-Tech Volume 17 Issue 1 Version 1.0 Year 2017.

[4] Arduino. (2015). Arduino Software (IDE). (Arduino) Retrieved December 27, 2015.

[5] Heidarsson, H. K., & Sukhatme, G. S. (2011). Obstacle Detection and Avoidance for an Autonomous Surface Vehicle using a Profiling Sonar. 2011 IEEE International Conference on Robotics and Automation. Shanghai.

[6] Ryther, C. A., & Madsen, O. B. (2009). Obstacle Detection and Avoidance for Mobile Robots. Technical University of Denmark.

[7] https://www.arduino.cc/en/Guide/Environment.

[8] https://www.frontiersin.org/articles/10.3389/frobt.2019.00057/full

[9] https://www.jetir.org/view?paper=JETIR2106671