A Mini Project report submitted on

**AN OBSTACLE DETECTION ROBOT**

*A partial fulfilment of the requirement for the Award of the*

*Degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**SUBMITTED**

By

**ANDELA SANJAY YADAV (21671A0405)**

**K MAHESH (21671A427**



**Department of Electronics and Communication Engineering**

**J.B. INSTITUTE OF ENGINEERING & TECHNOLOGY**

**UGC AUTONOMOUS**

(Accredited by NAAC & NBA, Approved by AICTE & Permanently affiliated by JNTUH) Yenkapally, Moinabad mandal, R.R. Dist-75 (TS) 2021-2025

A Mini Project report submitted on

**AN OBSTACLE DETECTION ROBOT**

*A partial fulfilment of the requirement for the Award of the*

*Degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**SUBMITTED**

By

**ANDELA SANAY YADAV (21671A0405)**

**K MAHESH (21671A0427)**

Under the esteemed guidance of

**Mrs. K. Sneha Latha**

**Associate Professor-Department of ECE**



**Department of Electronics and Communication Engineering**

**J.B. INSTITUTE OF ENGINEERING & TECHNOLOGY**

**UGC AUTONOMOUS**

(Accredited by NAAC & NBA, Approved by AICTE & Permanently affiliated by

JNTUH) Yenkapally, Moinabad mandal, R.R. Dist-75 (TS) 2021-2025

**J.B. INSTITUTE OF ENGINEERING & TECHNOLOGY**

**UGC AUTONOMOUS**

(Accredited by NAAC & NBA, Approved by AICTE & Permanently affiliated by JNTUH) Yenkapally, Moinabad mandal, R.R. Dist-75 (TS)



**CERTIFICATE**

This is to certify that the dissertation work entitled **AN OBSTACLE DETECTION ROBOT** was carried out by **ANDELA SANJAY YADAV (21671A0405), K MAHESH (21671A0427),** in partial fulfilment of the requirements for the degree of Bachelor of Technology in Electronics and Communication Engineering of the J.B. Institute of Engineering and Technology, Hyderabad, during the academic year 2024-25, is a bonafide record of work carried out under our guidance and supervision. The results embodied in this report have not been submitted to any other University or Institution for the award of any degree or diploma.

**Mrs. K. SNEHA LATHA** **Dr. TOWHEED SULTHANA**

**Associate Professor Professor**

**Internal guide HOD-ECE**

# 

# **ACKNOWLEDGEMENT**

This is to acknowledgement of the intensive drive and technical competence of many individuals who have contributed to the success of our dissertation.

We would like to sincerely thank to our internal guide**, Mrs. K. Sneha Latha Associate Professor** who stimulated many thoughts for this project and Staff-Members of Department of ECE for their goodwill gestures towards me.

We are very grateful **to Dr. Towheed Sulthana, Professor & HOD, ECE** who has not only shown at most patience, but fertile in suggestions, vigilant in directions of error and who have been infinitely helpful.

We wish to express deepest gratitude and thanks to Principal **Dr. P.C. KRISHNAMA CHARY** for his constant support and encouragement in providing all the facilities in the college to do the project work.

**ANDELA SANJAY YADAV (21671A0405)**

**K MAHESH (21671A0427)**

# 

# **ABSTRACT**

The obstacle detection robot project is all about creating a cool robot that can move around on its own! This little guy is made to spot and steer clear of things in its way—super smart, right? It uses different sensors like ultrasonic, infrared, LIDAR, & cameras. These help the robot see what's around it. A microcontroller—like an Arduino or Raspberry Pi—takes this information and tells the robot how to move.

Motors & actuators help it zoom around based on what the microcontroller tells it. This makes moving through spaces easy peasy! The whole project needs careful planning, designing, putting together hardware, and writing software. All of this has to work together to make sure the robot can detect obstacles and avoid them really well.

There are some tricky parts too. It’s not always easy for the robot to figure out what’s in front of it or to process info fast enough. Plus, keeping it powered up efficiently is important. But after lots of testing and tweaking things here & there, we can make the robot better at its job.

This project is not just about having fun with robots! It also helps us learn more about robotics & programming. Plus, we gain experience bringing hardware and software together. This knowledge can be used in all sorts of exciting areas like self-driving cars, delivery bots, & even search-and-rescue teams!

# **INDEX**

**CONTENTS**

[**ACKNOWLEDGEMENT** IV](#_Toc178679697)

[**ABSTRACT** V](#_Toc178679698)

[**INDEX** VI](#_Toc178679699)

[**LIST OF FIGURES** VII](#_Toc178679700)

[**CHAPTER-1** 1](#_Toc178679701)

[**1.1** **INTRODUCTION** 1](#_Toc178679702)

[**CHAPTER-2** 3](#_Toc178679703)

[**2.1 LITERATURE SURVEY** 3](#_Toc178679704)

[**2.2 EXISTING SYSTEM** 3](#_Toc178679705)

[**CHAPTER-3** 4](#_Toc178679706)

[**3. PROJECT DESCRIPTION** 4](#_Toc178679707)

[**3.1BLOCK DIAGRAM** 4](#_Toc178679708)

[**3.2 WORKING** 4](#_Toc178679709)

[**3.3 HARDWARE COMPONETS** 5](#_Toc178679710)

[**CHAPTER-4** 6](#_Toc178679711)

[**4.1 DESIGN OF HARDWARE** 6](#_Toc178679712)

[**4.2 ARDUINO** 6](#_Toc178679713)

[**4.3 USB PLUG & EXTERNAL POWER SUPPLY PLUG** 8](#_Toc178679714)

[**4.4 VOLTAGE REGULATOR** 8](#_Toc178679715)

[**4.5 POWER PINS** 8](#_Toc178679716)

[**4.5.1GROUND PINS** 8](#_Toc178679717)

[**4.5.2 IOREF Pin** 9](#_Toc178679718)

[**4.5.3 INPUT AND OUTPUT PINS** 9](#_Toc178679719)

[**4.5.4 Serial Out (TX) & Serial In (RX)** 9](#_Toc178679720)

[**4.5.5 EXTERNAL INTERRUPTS** 9](#_Toc178679721)

[**4.5.6RESET PIN** 10](#_Toc178679722)

[**4.6 LED INDICATOR** 10](#_Toc178679723)

[**4.7 POWER SUPPLY** 10](#_Toc178679724)

[**4.8 ULTRASONIC SENSOR** 11](#_Toc178679725)

[**4.9 SERVO MOTOR** 12](#_Toc178679726)

[**4.10 MOTOR DRIVER(L293D)** 13](#_Toc178679727)

[**4.11 GEAR MOTORS** 15](#_Toc178679728)

[**4.12 BATTERY(12v, 1.5A)** 15](#_Toc178679729)

[**4.13 SWITCH** 17](#_Toc178679730)

[**4.14 JUMPER WIRES** 18](#_Toc178679731)

[**CHAPTER-5** 20](#_Toc178679732)

[**5.0 DESIGN OF SOFTWARE** 20](#_Toc178679733)

[**5.1 INTRODUCTION TO ARDUINO IDE SOFTWARE** 20](#_Toc178679734)

[**5.2.1 SOFTWARE STEPS** 20](#_Toc178679735)

[**5.2.3 WORKING CODE OF PROJECT** 23](#_Toc178679736)

[**CHAPTER-6** 30](#_Toc178679737)

[**6.1 RESULT** 30](#_Toc178679738)

[**6.2 OUTPUT** 30](#_Toc178679739)

[**CHAPTER-7** 33](#_Toc178679740)

[**7.1 CONCLUSION** 33](#_Toc178679741)

**7.2**[**FUTURE SCOPE** 34](#_Toc178679742)

**7.3**[**REFERENCES** 35](#_Toc178679743)

# **LIST OF FIGURES**

[FIG-3.1 BLOCK DIAGRAM 4](#_Toc178679683)

[FIG 4.2 POWER SUPPLY 11](#_Toc178679684)

[FIG 4.3 ULTRASONIC SENSOR 11](#_Toc178679685)

[FIG 4.4 SERVO MOTOR 12](#_Toc178679686)

[FIG 4.5 MOTOR DRIVER 14](#_Toc178679687)

[FIG 4.6 GEAR MOTORS 15](#_Toc178679688)

[FIG 4.7 LITHIUM BATTERIES 16](#_Toc178679689)

[FIG 4.7.1 LITHIUM BATTERIES 17](#_Toc178679690)

[FIG 4.7.2 LITHIUM BATTERIES 17](#_Toc178679691)

[FIG 4.8 SWITCH 18](#_Toc178679692)

[FIG 4.9 JUMPER WIRES 18](#_Toc178679693)

[FIG OUTPUT 6.2.1 31](#_Toc178679694)

[FIG OUTPUT 6.2.2 31](#_Toc178679695)

[FIG OUTPUT 6.2.3 32](#_Toc178679696)

# **CHAPTER-1**

## **INTRODUCTION**

Creating an obstacle detection robot is such an exciting adventure! It’s all about building a robot that can roam around on its own. It finds obstacles in its way & cleverly avoids them. These robots can do so many things—like help with self-driving cars, deliver packages, & even rescue people when needed! The main idea here is to let the robot move from one spot to another by itself—no humans needed! It must make quick choices to dodge anything in its path.

Now, let’s talk about what makes this project tick. The key part is picking and putting together different sensors. For instance, ultrasonic sensors send out sound waves. They measure how long it takes for those waves to bounce back. That's how they figure out how far away obstacles are! Infrared sensors work similarly but use infrared light instead. Then there’s LIDAR (Light Detection and Ranging), which uses lasers to measure distances. It even helps create 3D maps of where the robot is cruising around! Plus, cameras give visual info that the robot processes with smart image recognition tools to spot hurdles & navigate just right.

At the center of our robot is a microcontroller or processor—think Arduino, Raspberry Pi, or similar devices. This little guy is super important because it makes sense of all the sensor data & helps the robot decide what to do in real time. The microcontroller takes data from the sensors, runs clever algorithms, & controls motors and other parts that get the robot moving. Speaking of moving, motors & actuators are key for getting around! They turn the commands from the microcontroller into actual motions—letting the robot travel through its space. Choosing which motors to use—DC, stepper, or servo—depends on how precise we need things to be for each unique task.

A reliable power source is also super crucial for our robot's life. It could be simple batteries or even fancier setups that manage power well. The goal? To ensure there’s enough energy to run all parts like sensors, microcontroller, and motors for a good amount of time.

Don’t forget about software development—it’s a major piece too! Writing code helps process sensor data, implement those smart obstacle detection techniques, & control movement. We need these algorithms to be fast and effective for quick decision-making! Sometimes we keep it simple with distance thresholds for 2 avoiding obstacles. Other times we use more complex path planning like A\* or Dijkstra’s to help get around things.

So where do we start with this project? We kick off with designing & planning things out. This early phase means setting goals and figuring out what we need. Careful planning helps us pick just the right sensors & parts and sketch out how our robot will look! After planning wraps up, it’s time for hardware assembly— putting everything together!

Next comes software development and programming our microcontroller so it can work with sensor data & control movement really well. Testing those obstacle-dodging algorithms is really important here too! We need to keep tweaking the software until it does a great job in different settings.

Testing never stops throughout this project! The robot needs trials in various scenarios so we can find any hiccups along the way. This may mean adjusting where sensors go, tuning up those algorithms, or making sure both hardware & software are tough enough.

One big challenge? Getting accurate readings from different obstacles! Sensors must spot objects of all sizes, shapes, and materials reliably. Plus, processing information quickly is super important; delays could lead to crashes! We also need to think about power management; our little friend needs to be efficient without needing a recharge all the time.

In summary, this fun project boosts skills in robotics as well as programming and putting systems together! It gives hands-on experience in mixing hardware & software into something that works on its own! Completing an obstacle detection robot project doesn't just show off technical skills; it also opens doors for new ideas and uses in many fields ahead!

# **CHAPTER-2**

## **2.1 LITERATURE SURVEY**

Obstacle detection is one of the ways of enhancing the visual mechanism of any robotic system. Obstacle detection is a primary requirement of any autonomous mobile robot. Obstacle detection and avoidance Robot is designed to allow robot to navigate in unknown environment by avoiding collisions. Obstacle avoiding robot senses obstacles on its path, avoid it and resumes its running.

There are some very famous methods for obstacle detection robot. Some sensing devices used for obstacle detection like bump sensor, infrared sensor, camera, and ultrasonic sensor. A more general and commonly employed method for obstacle avoidance is based on ultrasonic sensing mechanism. The choice of the mechanism depends on the nature and effectiveness of the image required as well as the physical environment in which the detection is to be applied. The obstacle detector works on the principle of SONAR.

The autonomous surface vehicle (ASV) developed by Heidarsson and Sukhatme in 2011 employed a single-beam mechanically scanning profiling sonar to detect obstacles under water. The profiling sonar has the ability to produce cone-shaped beam, which is ideal for detecting near surface obstacles. One of the objectives of their work was to investigate the suitability of using sonar near the water-air boundary for which the study found promising results.

## 

## **2.2 EXISTING SYSTEM**

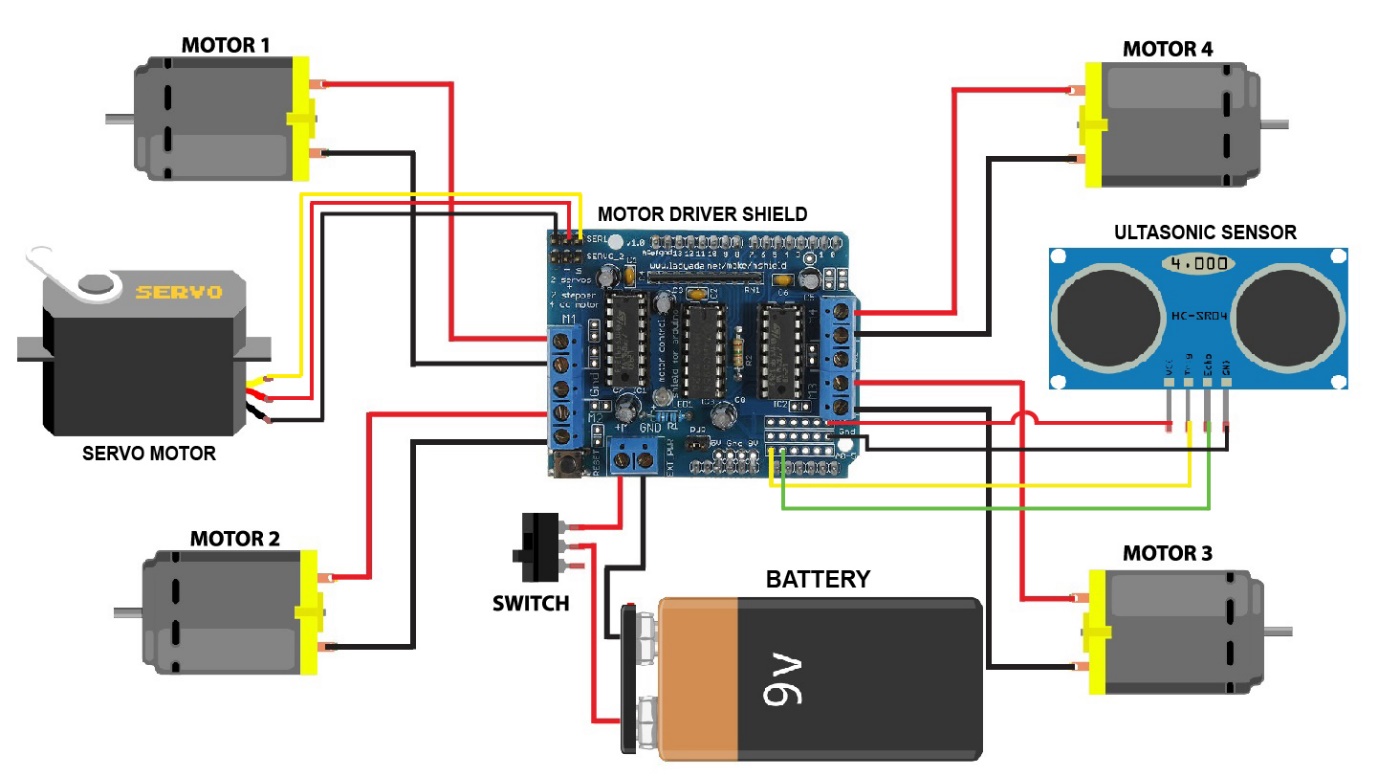
Obstacle detection robots use a mix of different sensors, microcontrollers, & smart algorithms. There have been many advancements, sure! But challenges still exist. Things like sensor limits, processing speed, and power use are ongoing hurdles that researchers are trying to overcome. These systems play a big role in many areas – from fun hobby projects to advanced self-driving cars and even search-and-rescue missions. It shows just how versatile & important these obstacle detection technologies really are!

# **CHAPTER-3**

## **3. PROJECT DESCRIPTION**

This chapter deals with creating an obstacle detection robot that can roam around on its own. It finds obstacles in its way & cleverly avoids them. . It can be simply understood by its block diagram &circuit diagram.

## **3.1BLOCK DIAGRAM**



#### FIG-3.1 BLOCK DIAGRAM

## **3.2 WORKING**

The Arduino Uno acts as a central processing unit, coordinating functions of sensors and actuators for real-time decision making and acceleration Hardware integration.

The core of the system processes data from Arduino Uno's ultrasonic sensors, which are strategically placed around the robot to cover multiple channels. These sensors generate ultrasonic waves, measure the time it takes for the echo to hit the obstacle and return, calculate the distance to the object and then send the data collected from these sensors to the Arduino so that it can be processed.

For excitation, the robot is equipped with two DC motors connected to the control motors. The motor drivers receive commands from the Arduino and control the speed and direction of the motors. The motor drivers are important because they can handle the high current requirements of the motors, which the Arduino alone cannot provide. The orientation of the robot is controlled by varying the speed of the motors, allowing them to turn smoothly and straight.

A servomotor is used to change the orientation of the ultrasonic sensors, allowing the robot to better explore its environment. By rotating the sensors, servomotors can collect distance information from multiple robots, improving the ability to detect obstacles from different angles, this rotation can be controlled with specific angles, and provides the robot has detected a sharp object.

Power supplies to the robot are provided by batteries, which are selected based on the power and current requirements of the motor and Arduino. The battery pack is connected to the motors and the Arduino, ensuring that all parts receive stable power. The power management system includes voltage management

## **3.3 HARDWARE COMPONETS**

* Arduino UNO R3 ATMEGA 328P
* Ultra Sonic Sensor
* Servo motor
* Motor Driver(L293d)

* Robot Wheels
* Gear Motors(4)
* LI – ion Battery
* Jumper wires
* LI- ion Battery holder
* Switch

# **CHAPTER-4**

## **4.1 DESIGN OF HARDWARE**

This chapter briefly explains about the Hardware implementation of solar monitoring and cleaning. It discuss the circuit diagram of each module in detail.

## **4.2 ARDUINO**

The most common version of Arduino is the Arduino Uno. This board is what most people are talking about when they refer to an Arduino. The Uno is one of the more popular boards in the Arduino family and a great choice for beginners. There are different revisions of Arduino Uno, below detail is the most recent revision (Rev3 or R3).

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

Microcontroller : ATmega328

Operating Voltage : 5V

Input Voltage (recommended) : 7-12V

Input Voltage (limits) : 6-20V

Digital I/O Pins output) : 14

Analog Input Pins : 6

DC Current per I/O Pin : 40 mA

DC Current for 3.3V Pin : 50 mA

Flash Memory KB used by bootloader : 2 KB (ATmega328) of which 0.5

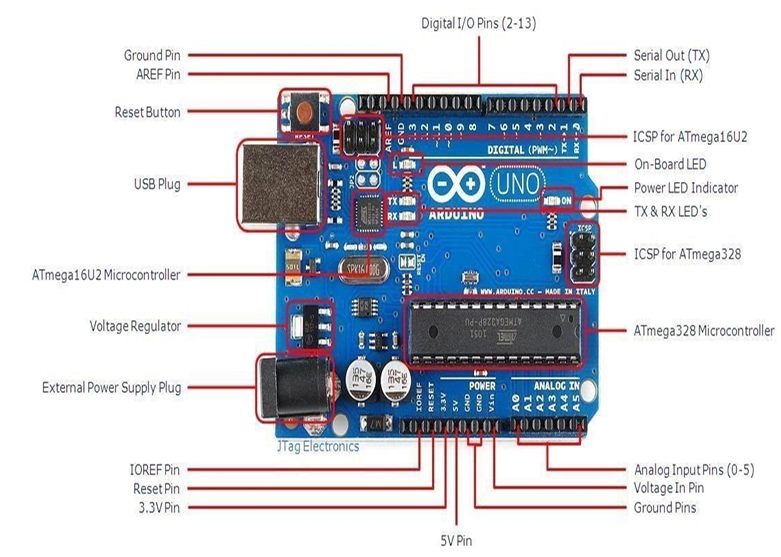
SRAM : 1KB (ATmega328)

EEPROM : 2KB(ATmega328)

Clock Speed : 16MHZ

Length : 68.6MM

Width : 53.4MM

Figure 4.1 Arduino Uno R3 Board

## **4.3 USB PLUG & EXTERNAL POWER SUPPLY PLUG**

Every Arduino board needs a way to be connected to a power source. The Arduino Uno can be powered from a USB cable coming from your computer or a wall power supply that is terminated in a barrel jack. The power source is selected automatically. The USB connection is also how you will load code onto your Arduino board. Please on my other post on how to program with Arduino can be found in Installing and Programming Arduino.

NOTE: The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

## **4.4 VOLTAGE REGULATOR**

The voltage regulator is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it’s for. The voltage regulator does exactly what it says – it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course,it has its limits, so don’t hook up your Arduino to anything greater than 20 volts.

## **4.5 POWER PINS**

Voltage In Pin – The input voltage to the Arduino board when it’s using an external power source(as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. 5V Pin – This pin outputs a regulated 5V from the regulator on the board.

The board can be supplied with power either from the DC power jack (7 – 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. It’s not recommended.3.3V Pin – A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

### **4.5.1GROUND PINS**

There are several GND pins on the Arduino, any of which can be used to ground your circuit.

### **4.5.2 IOREF Pin**

This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power SOURCE or enable voltage translators on the outputs for working with the 5V or 3.3V.

### **4.5.3 INPUT AND OUTPUT PINS**

Each of the 14 digital pins on the Uno can be used as an input or output. They operate at 5 volts. These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED). Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-5k Ohms. In addition, some pins have specialized functions.

### **4.5.4 Serial Out (TX) & Serial In (RX)**

Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

### **4.5.5 EXTERNAL INTERRUPTS**

Pins 2 and 3 can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.

PWM – You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). Think of these pins as being able to simulate analog output (like fading an LED in and out). SPI – Pins 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).

SPI stands for Serial Peripheral Interface. These pins support SPI communication using the SPI library.

Analog Input Pins – Labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read. By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF Pin (Stands for Analog Reference. Most of the time you can leave this pin alone). Additionally, some pins have specialized functionality:

TWI – Pins A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

### **4.5.6RESET PIN**

Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

## **4.6 LED INDICATOR**

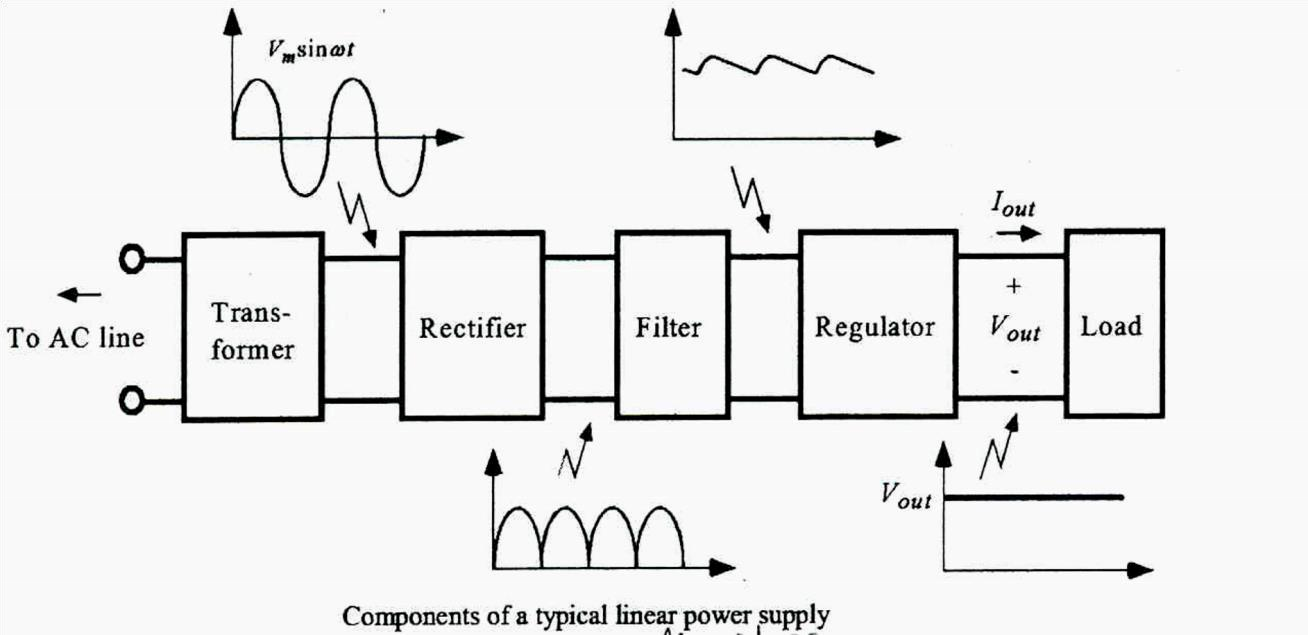
Power LED Indicator – Just beneath and to the right of the word “UNO” on your circuit board, there’s a tiny LED next to the word ‘ON’. This LED should light up whenever you plug your Arduino into a power source. If this light doesn’t turn on, there’s a good chance something is wrong. Time to re check your circuit!

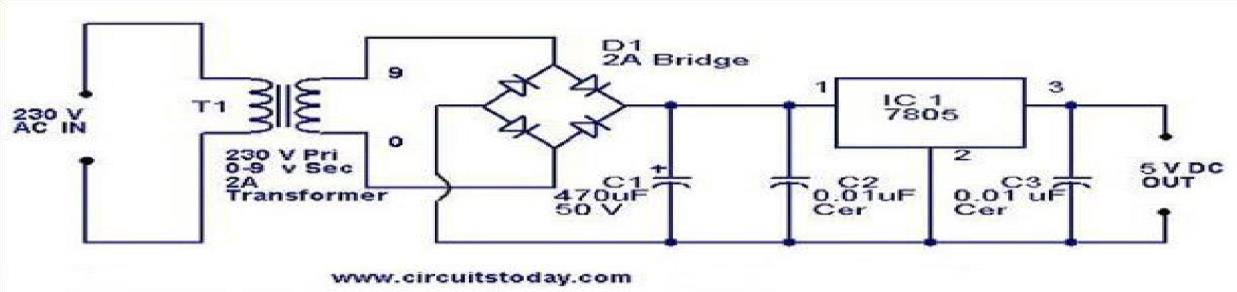
On-Board LED – There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it’s off. This useful to quickly check if the board has no problem as some boards has a preloaded simple blinking LED program in it.

TX & RX LEDs – These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we’re loading a new program onto the board).

## **4.7 POWER SUPPLY**

The power supplies are designed to convert high voltage AC mains electricity to a suitable low voltage supply for electronic circuits and other devices. A power supply can by broken down into a series of blocks, each of which performs a particular function. A DC power supply which maintains the output voltage constant irrespective of AC mains fluctuations or load variations is known as “Regulated DC Power Supply”.

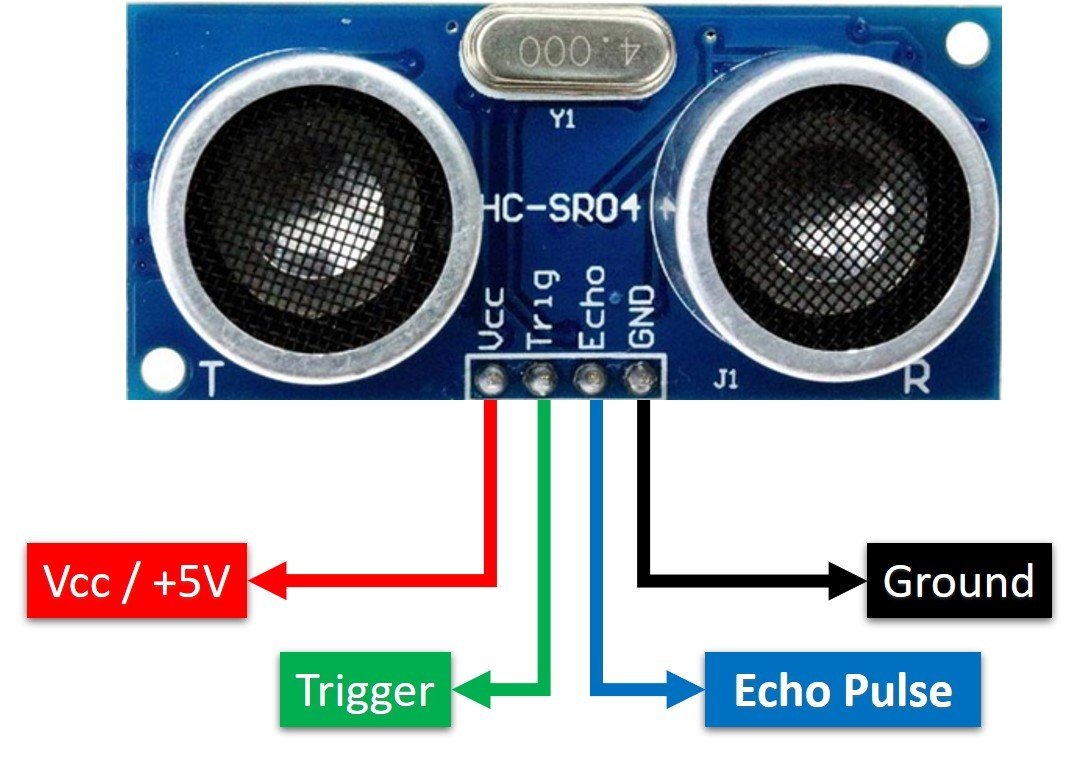




#### FIG 4.2 POWER SUPPLY

## **4.8 ULTRASONIC SENSOR**

An ultrasonic sensor is a device used to measure the distance to an object by emitting ultrasonic sound waves and measuring the time it takes for the echo to return. It operates on the principle of echolocation, similar to how bats navigate. These sensors are widely used in robotics for obstacle detection and distance measurement due to their accuracy, cost effectiveness, and ease of use.

****

#### FIG 4.3 ULTRASONIC SENSOR

## **4.9 SERVO MOTOR**

A servo motor is a type of motor commonly used in robotics and automation for precise control of angular position, velocity, and acceleration. It is designed to provide high torque and accuracy in a compact form.



#### FIG 4.4 SERVO MOTOR

**Description of Servo Motor**

1. Components:

* Motor: The core component that provides rotational motion.
* Gearbox: Reduces the speed of the motor and increases the torque. The gearbox ensures that the motor's high-speed rotation is converted into precise, controlled movements.
* Control Circuit: Receives the input signal (usually from a controller or microcontroller) and adjusts the motor’s position accordingly.
* Feedback Mechanism: Typically includes a potentiometer or encoder that provides feedback on the motor's position, enabling precise control.

2. Functionality:

* Position Control: Servo motors are commonly used for precise positioning applications. They can move to a specific angle and hold that position with high accuracy.
* Speed Control: They can also control rotational speed, though this is secondary to their positional control capabilities.
* Torque: Servo motors provide high torque relative to their size, which is beneficial for moving parts in robotic arms, steering systems, and other applications requiring force.

3. Types:

* Standard Servo: Used in most hobbyist and light-duty applications. Typically provides 180 degrees of rotation.
* Continuous Rotation Servo: Allows for continuous rotation in both directions, useful for applications requiring variable speed and direction control.
* High-Torque Servo: Designed to deliver more torque for heavy-duty applications, such as large robotic arms or high-load mechanisms.

4. Control Signal:

* Servo motors are controlled using a Pulse Width Modulation (PWM) signal.
* The width of the pulse determines the position of the servo. For instance, a 1.5 ms pulse might correspond to the center position, while 1.0 ms and 2.0 ms pulses might move it to the extremes.

5. Applications:

* Robotics: For controlling joints and actuators in robotic arms and other systems.
* RC Vehicles: Used in remote-controlled cars, boats, and airplanes for steering and control surfaces.
* Automation: Employed in industrial automation for precise control of machinery. Example In a robotic arm, a servo motor might be used to control the rotation of the arm’s joints. The motor receives a PWM signal from the controller, which adjusts the arm’s position based on the desired angle, allowing for intricate movements and precise tasks.

## **4.10 MOTOR DRIVER(L293D)**

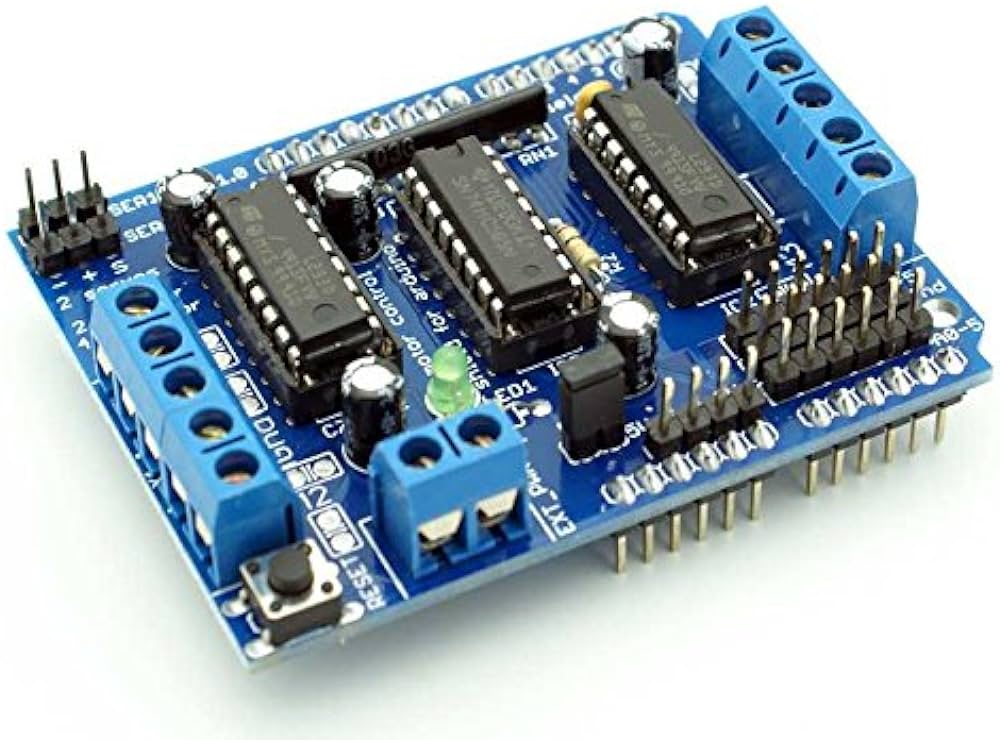
The L293D is a versatile motor driver IC that can simultaneously control two DC motors in both directions (forward and reverse). It is commonly used in robotics, automation systems, and embedded applications where control of motor direction and speed is required. The IC is capable of driving inductive loads such as relays, solenoids, and small DC motors.

**Key Features:**

* **Dual H-Bridge Configuration**: The L293D contains two H-bridges, allowing for the independent control of two DC motors. Each H-bridge allows current to flow in either direction through the motor, enabling forward and reverse operation.
* **High Current Output**: It can supply up to 600mA of continuous current per channel and handle peaks of up to 1.2A per channel, making it suitable for small motors.
* **Wide Operating Voltage**: The motor driver operates between 4.5V and 36V, allowing for flexibility in the power source.
* **Internal Diodes**: The IC includes diodes for back EMF protection, which safeguard the circuit from voltage spikes generated by motors.
* **Enable Pins for Speed Control**: The IC includes enable pins that can be used with pulse width modulation (PWM) to control the motor's speed by adjusting the duty cycle of the PWM signal.
* **Thermal Shutdown**: The L293D has thermal shutdown protection to prevent overheating and damage to the IC.

**Pin Configuration:**

1. **Input Pins (1A, 2A, 3A, 4A)**: These are control inputs that determine the direction of motor rotation.
2. **Output Pins (1Y, 2Y, 3Y, 4Y)**: Connected to the motor terminals, these provide the current to drive the motor in the desired direction.
3. **Enable Pins (1, 9)**: These pins enable or disable the respective H-bridge. They are used to control the speed of the motors via PWM.
4. **Power Supply (VCC1, VCC2)**: VCC1 powers the logic circuitry, while VCC2 powers the motor.
5. **Ground (GND)**: Common ground for both logic and motor power.



#### FIG 4.5 MOTOR DRIVER

The L293D motor driver works by using an H-bridge configuration to control the direction and speed of two DC motors independently. Each motor is controlled through a pair of input pins (1A, 2A for Motor 1 and 3A, 4A for Motor 2) which determine the flow of current, allowing the motors to rotate either forward or reverse. The driver also includes enable pins (1 and 9) that can be used to control motor speed by applying Pulse Width Modulation (PWM). By adjusting the PWM signal's duty cycle, the speed of the motor is controlled, with higher duty cycles resulting in faster motor rotation.

The L293D operates with two power supplies: VCC1 for the internal logic, usually powered by 5V, and VCC2 for the motors, supporting a range of 4.5V to 36V, making it versatile for different motor applications. The IC also features internal diodes to protect against back EMF (electromotive force) generated by the motors, as well as a thermal shutdown mechanism that prevents damage due to overheating during high-current operations. This makes the L293D ideal for motor control in robotics, automation, and embedded system

## **4.11 GEAR MOTORS**

## 

In a robotic car, gear motors play a crucial role in driving the vehicle's wheels and enabling precise control over its movement. Here’s how gear motors are utilized in a robotic car: Role of Gear Motors in a Robotic Car

* Driving Wheels: Gear motors are typically used to power the wheels of the robotic car. They provide the necessary torque to move the car and overcome obstacles.
* Speed and Torque Adjustment: The gearbox in the gear motor reduces the motor’s high speed while increasing the torque. This allows the robotic car to move slowly and steadily with sufficient force to handle different terrains and obstacles.
* Directional Control: By using differential steering (where each wheel or set of wheels can be driven at different speeds), gear motors help control the direction of the robotic car. For example, driving one wheel faster than the other can make the car turn.



#### FIG 4.6 GEAR MOTORS

## **4.12 BATTERY(12v, 1.5A)**

Lithium-ion (Li-ion) batteries are a type of rechargeable battery widely used in modern electronics and energy storage applications due to their high energy density, lightweight design, and long cycle life. They have become the preferred choice for powering devices such as smartphones, laptops, electric vehicles, and renewable energy storage systems. The basic components of a lithium-ion battery include a positive electrode (cathode), a negative electrode (anode), an electrolyte, and a separator. The high energy density of Li-ion batteries allows them to store more energy per unit weight than other types of rechargeable batteries, making them ideal for portable and space-constrained applications. Their ability to recharge multiple times while maintaining performance has made them integral to the advancement of technology in various industries.



#### FIG 4.7 LITHIUM BATTERIES

**Working Principle and Operation**

Lithium-ion batteries operate by moving lithium ions between the cathode and anode through an electrolyte during charging and discharging cycles. During charging, lithium ions are driven from the cathode (commonly made of materials like lithium cobalt oxide or lithium iron phosphate) to the anode (typically made of graphite), where they are stored. During discharge, the ions move back from the anode to the cathode, generating an electric current that powers the connected device. The electrolyte, a lithium salt dissolved in an organic solvent, facilitates the movement of ions, while the separator keeps the electrodes apart to prevent short circuits. The efficient transfer of ions between the electrodes ensures that lithium-ion batteries can deliver a high level of energy consistently over many charge cycles, providing reliable and long-lasting power for a wide range of applications.



#### FIG 4.7.1 LITHIUM BATTERIES



#### FIG 4.7.2 LITHIUM BATTERIES

**Applications and Considerations**

Lithium-ion batteries are used in diverse applications due to their superior performance characteristics, such as higher energy density, longer lifespan, and lower self-discharge compared to other battery types. They are extensively used in consumer electronics, electric vehicles, medical devices, and large-scale renewable energy storage systems. Despite their many advantages, lithium-ion batteries come with certain challenges, such as safety risks related to overheating and the potential for thermal runaway, which can lead to fires or explosions if the battery is damaged. To mitigate these risks, modern batteries include safety features like protective circuits and temperature sensors. Additionally, the sourcing of materials like lithium and cobalt raises environmental and ethical concerns, prompting ongoing research into sustainable alternatives. Nevertheless, lithium-ion batteries remain critical for modern technology and are expected to continue evolving, with future innovations aimed at improving their safety, efficiency, and sustainability.

## **4.13 SWITCH**

A switch is a fundamental electrical component used to control the flow of current in a circuit. It allows users to open or close the circuit, thereby turning electrical devices or systems on or off.



#### FIG 4.8 SWITCH

## **4.14 JUMPER WIRES**

Jumper wires are fundamental components used in electronics prototyping and circuit development, offering a flexible and efficient method for making temporary connections between different elements of a circuit. These wires are designed to connect components on breadboards or development boards like Arduino, Raspberry Pi, and others without the need for soldering. They play a crucial role in enabling quick assembly, modification, and testing of circuits, which is why they are a staple in electronics laboratories, hobbyist projects, and educational settings. Jumper wires come in a variety of connector types and lengths, providing versatility for a wide range of applications.



#### FIG 4.9 JUMPER WIRES

**Types of Jumper Wires**

Jumper wires are primarily available in three types based on the connectors at their ends: **male-to-male**, **male-to-female**, and **female-to-female**. **Male-to-male** jumper wires feature pin connectors at both ends and are commonly used to connect two points on a breadboard or between a component and a microcontroller. **Male-to-female** jumper wires have a pin on one end and a socket on the other, making them ideal for connecting pins on a module to a breadboard or microcontroller. Finally, **female-to-female** jumper wires have sockets at both ends, often used to connect two pin headers. These wires are typically color-coded, making it easier to track connections in complex circuits and reduce the likelihood of wiring mistakes.

**Applications and Uses**

Jumper wires are indispensable in prototyping environments where circuits need to be designed, tested, and modified frequently. They are widely used in projects involving microcontrollers, sensors, displays, and various other electronic modules. Their ability to create quick, solder-free connections makes them perfect for temporary circuits where designers need to adjust configurations on the fly. Educational kits for students often include jumper wires for hands-on learning experiences, where breadboards and development boards are used to teach the basics of circuit design. In professional prototyping, jumper wires facilitate the rapid development of new electronic designs by allowing engineers to connect and test different components without permanent connections. They are especially useful when experimenting with different configurations or debugging circuits before finalizing a design.

**Advantages and Limitations**

The primary advantage of using jumper wires is their flexibility, allowing engineers and hobbyists to quickly build and alter circuits without the need for soldering, which is time-consuming and not easily reversible. Jumper wires can be reused across multiple projects, making them an economical choice for prototyping. Their use in breadboard setups also eliminates the risk of damaging components through improper soldering. However, because the connections are not permanent, they may come loose over time or with movement, which could lead to intermittent connections or faults in a circuit. For long-term or production-level applications, soldered connections are typically preferred due to their reliability and durability. Despite this, jumper wires remain a critical tool in the development phase of electronic projects, offering speed, ease of use, and flexibility that are essential in circuit prototyping.

# **CHAPTER-5**

## **5.0 DESIGN OF SOFTWARE**

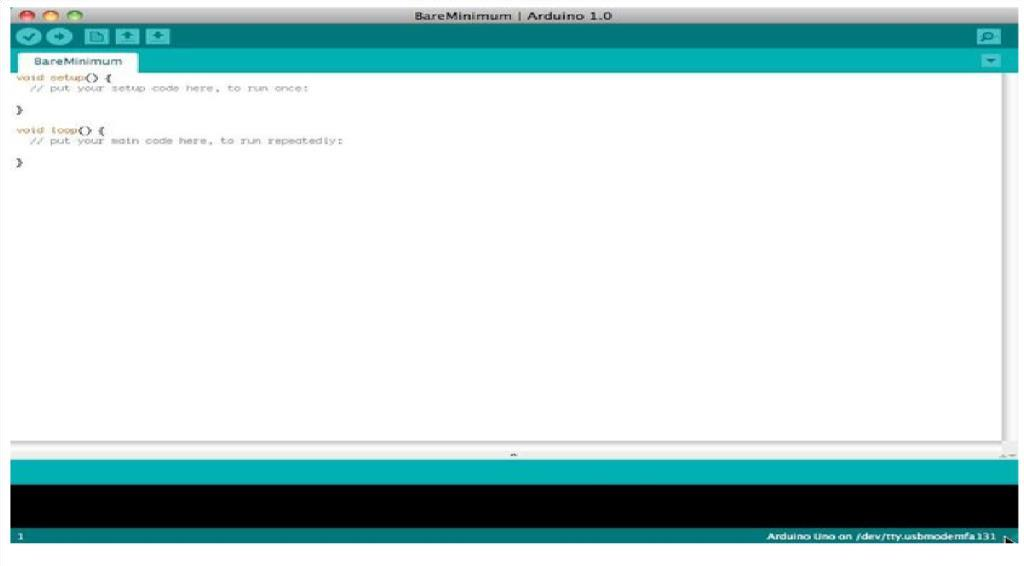
## **5.1 INTRODUCTION TO ARDUINO IDE SOFTWARE**

This is free software (evaluation version) which solves many of the pain points for an embedded system developer. This software is an Integrated Development Environment(IDE), which integrated text editor to write program, a compiler and it will convert your source code into HEX file. Here is simple guide to start working with Arduino IDE Vision which can be used for:

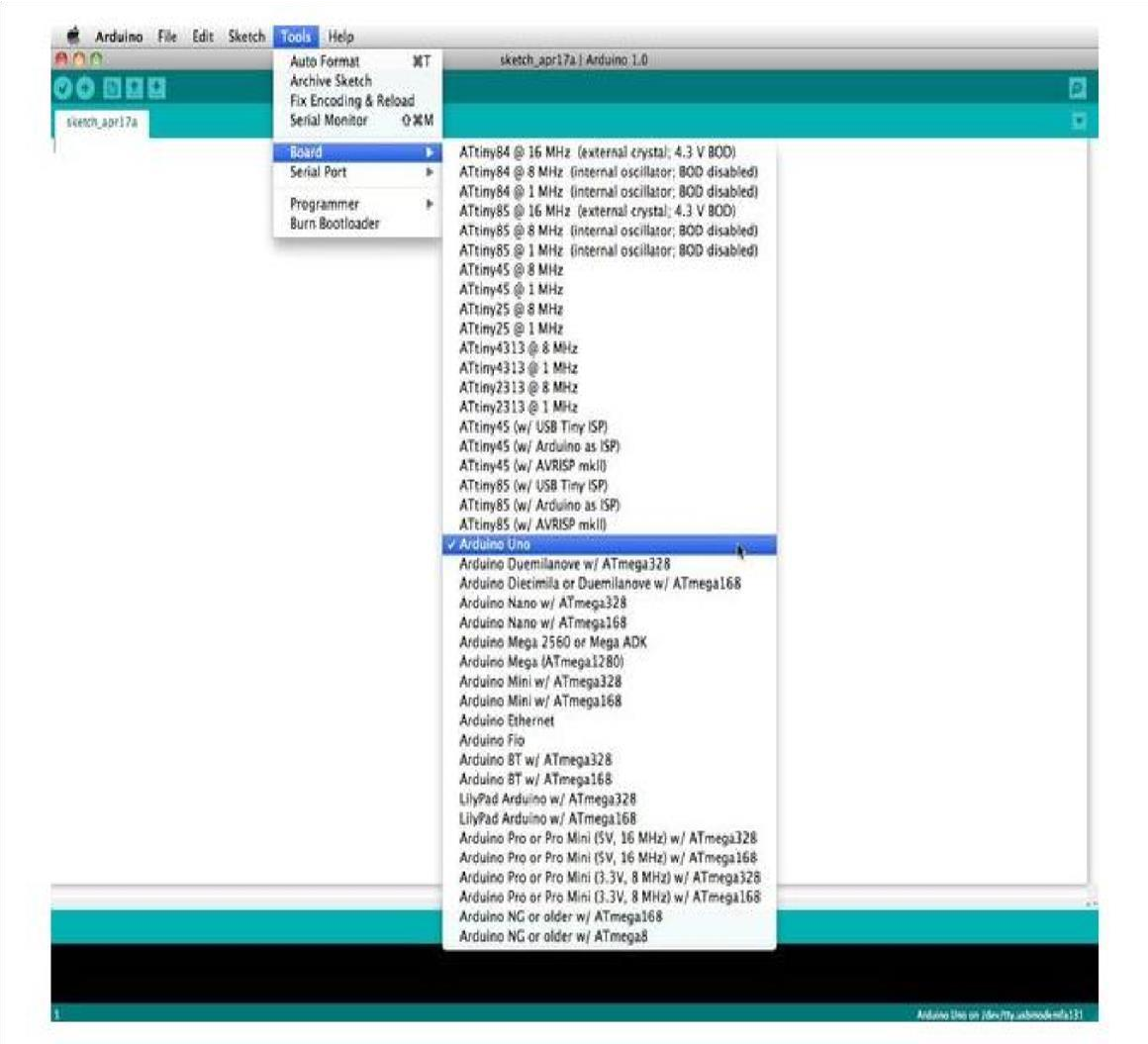
* Writing programs in Arduino IDE
* Compiling and assembling programs
* Debugging programs

### **5.2.1 SOFTWARE STEPS**

Before you can start doing anything with the Arduino, you need to download and install the Arduino IDE (integrated development environment).



After the opening IDE the settings are changed in order to connect to the Arduino.



Before you can start doing anything in the Arduino programmer, you must set

The board-type and serial port.

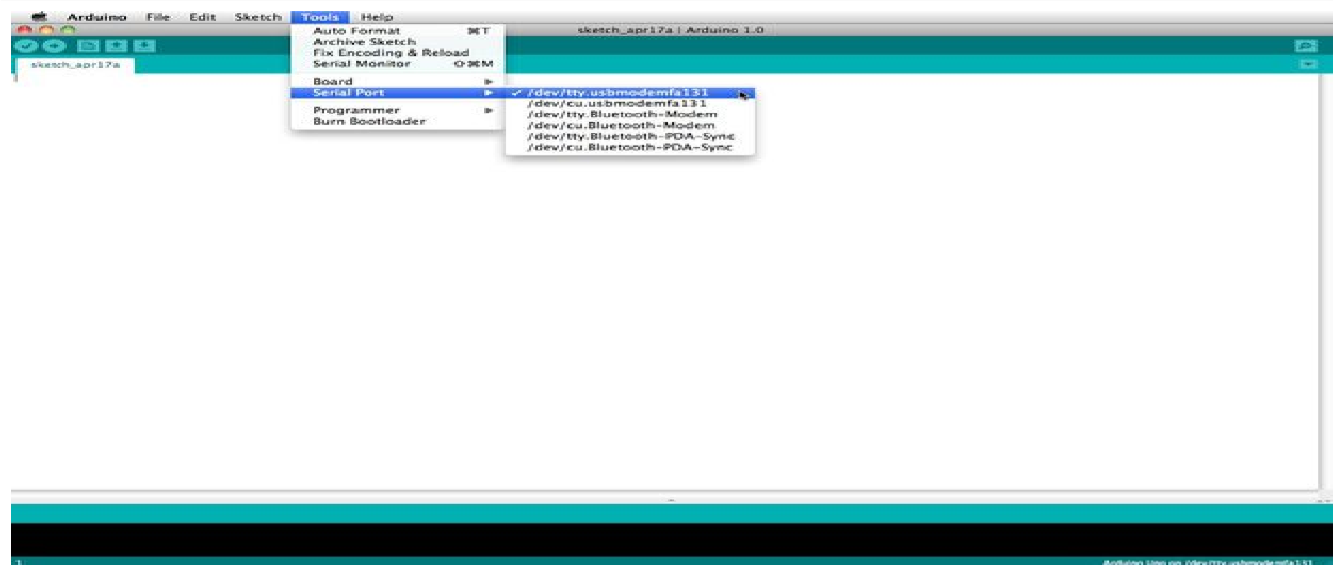
To set the board, go to the following:

Tools --> Boards

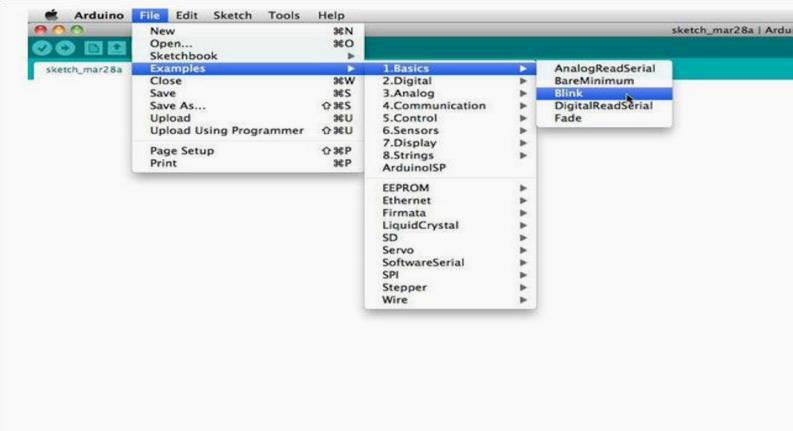
Select the version of board that you are using. Since I have an Arduino

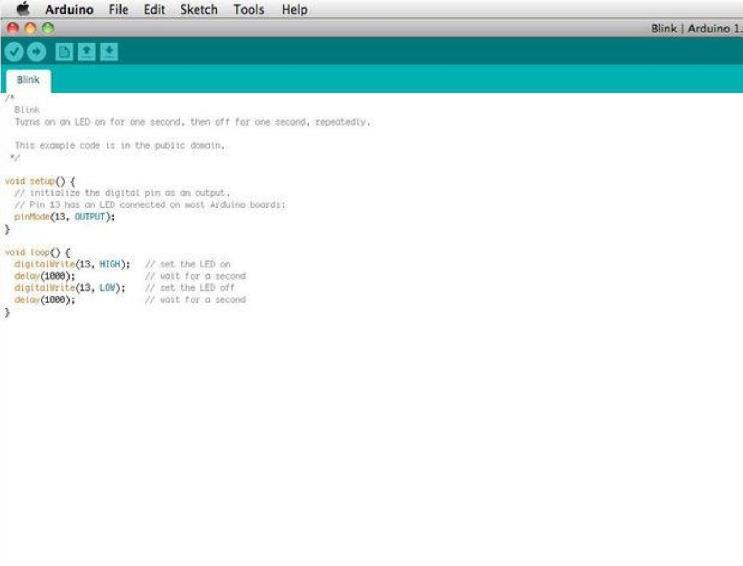
Uno plugged in, I obviously selected "Arduino Uno."

To set the serial port, go to the following: Tools --> Serial Port



Arduino programs are called sketches. The Arduino programmer comes with a ton of example sketches preloaded. This is great because even if you have never programmed anything in your life, you can load one of these sketches and get the Arduino to do something.



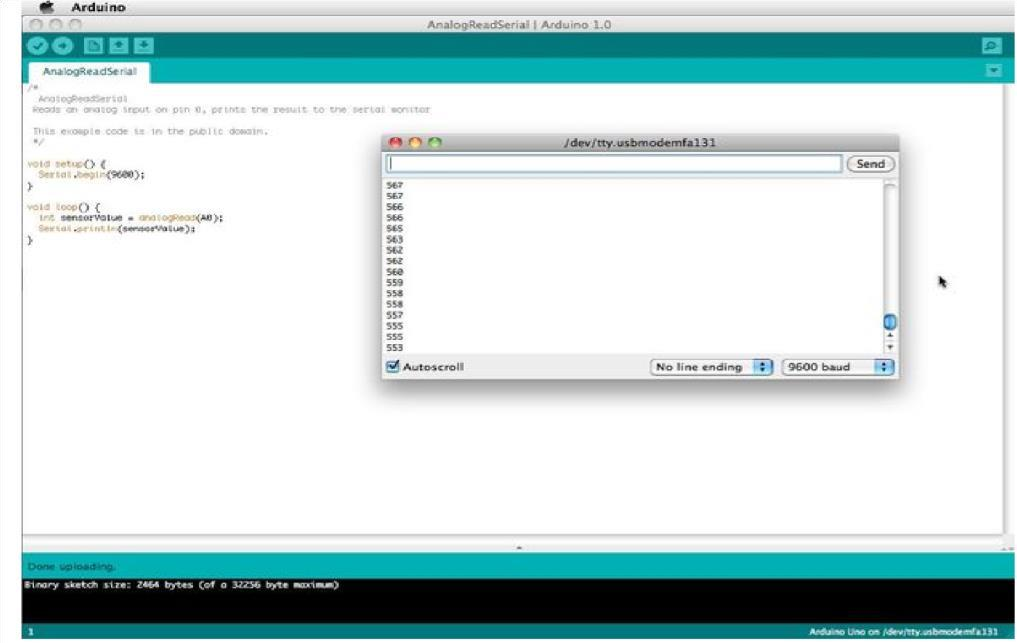


The serial monitor allows your computer to connect serially with the Arduino. This is important because it takes data that your Arduino is receiving from sensors and other devices and displays it in real-time on your computer. Having this ability is invaluable to debug your code and understand what number value the chip is actually receiving.

For instance, connect centre sweep (middle pin) of a potentiometer to A0, and the outer pins, respectively, to 5v and ground.

Next upload the sketch shown below: File --> Examples --> 1. Basics --> Analog Read Serial

Click the button to engage the serial monitor which looks like a magnifying glass. You can now see the numbers being read by the analog pin in the serial monitor. When you turn the knob the numbers will increase and decrease. The numbers will be between the range of 0 and 1023. The reason for this is that the analog pin is converting a voltage between 0 and 5V to a discreet number.



### **5.2.3 WORKING CODE OF PROJECT**

#include <AFMotor.h>

#include <NewPing.h>

#include <Servo.h>

#define TRIG\_PIN A0

#define ECHO\_PIN A1

#define MAX\_DISTANCE 200

#define MAX\_SPEED 190 // sets speed of DC motors

#define MAX\_SPEED\_OFFSET 20

NewPing sonar(TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE);

AF\_DCMotor motor1(1, MOTOR12\_1KHZ);

AF\_DCMotor motor2(2, MOTOR12\_1KHZ);

AF\_DCMotor motor3(3, MOTOR34\_1KHZ);

AF\_DCMotor motor4(4, MOTOR34\_1KHZ);

Servo myservo;

boolean goesForward=false;

int distance = 100;

int speedSet = 0;

void setup() {

myservo.attach(10);

myservo.write(115);

delay(2000);

distance = readPing();

delay(100);

distance = readPing();

delay(100);

distance = readPing();

delay(100);

distance = readPing();

delay(100);

}

void loop() {

int distanceR = 0;

int distanceL = 0;

delay(40);

if(distance<=15)

{

moveStop();

delay(100);

moveBackward();

delay(300);

moveStop();

delay(200);

distanceR = lookRight();

delay(200);

distanceL = lookLeft();

delay(200);

if(distanceR>=distanceL)

{

turnRight();

moveStop();

}else

{

turnLeft();

moveStop();

}

}else

{

moveForward();

}

distance = readPing();

}

int lookRight()

{

myservo.write(50);

delay(500);

int distance = readPing();

delay(100);

myservo.write(115);

return distance;

}

int lookLeft()

{

myservo.write(170);

delay(500);

int distance = readPing();

delay(100);

myservo.write(115);

return distance;

delay(100);

}

int readPing() {

delay(70);

int cm = sonar.ping\_cm();

if(cm==0)

{

cm = 250;

}

return cm;

}

void moveStop() {

motor1.run(RELEASE);

motor2.run(RELEASE);

motor3.run(RELEASE);

motor4.run(RELEASE);

}

void moveForward() {

if(!goesForward)

{

goesForward=true;

motor1.run(FORWARD);

motor2.run(FORWARD);

motor3.run(FORWARD);

motor4.run(FORWARD);

for (speedSet = 0; speedSet < MAX\_SPEED; speedSet +=2) // slowly bring the speed up to avoid loading down the batteries too quickly

{

motor1.setSpeed(speedSet);

motor2.setSpeed(speedSet);

motor3.setSpeed(speedSet);

motor4.setSpeed(speedSet);

delay(5);

}

}

}

void moveBackward() {

goesForward=false;

motor1.run(BACKWARD);

motor2.run(BACKWARD);

motor3.run(BACKWARD);

motor4.run(BACKWARD);

for (speedSet = 0; speedSet < MAX\_SPEED; speedSet +=2) // slowly bring the speed up to avoid loading down the batteries too quickly

{

motor1.setSpeed(speedSet);

motor2.setSpeed(speedSet);

motor3.setSpeed(speedSet);

motor4.setSpeed(speedSet);

delay(5);

}

}

void turnRight() {

motor1.run(FORWARD);

motor2.run(FORWARD);

motor3.run(BACKWARD);

motor4.run(BACKWARD);

delay(500);

motor1.run(FORWARD);

motor2.run(FORWARD);

motor3.run(FORWARD);

motor4.run(FORWARD);

}

void turnLeft() {

motor1.run(BACKWARD);

motor2.run(BACKWARD);

motor3.run(FORWARD);

motor4.run(FORWARD);

delay(500);

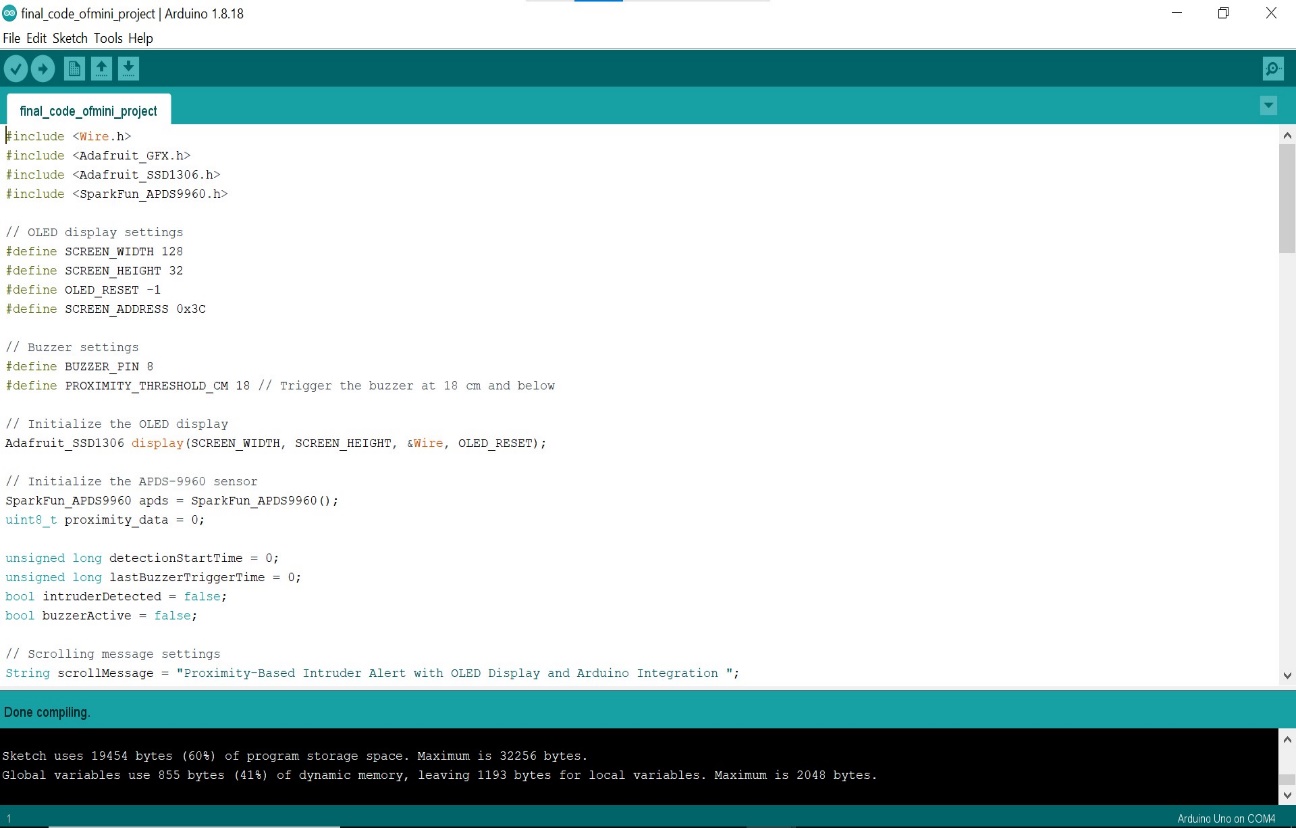
motor1.run(FORWARD);

motor2.run(FORWARD);

motor3.run(FORWARD);

motor4.run(FORWARD);

}



# **CHAPTER-6**

## **6.1 RESULT**

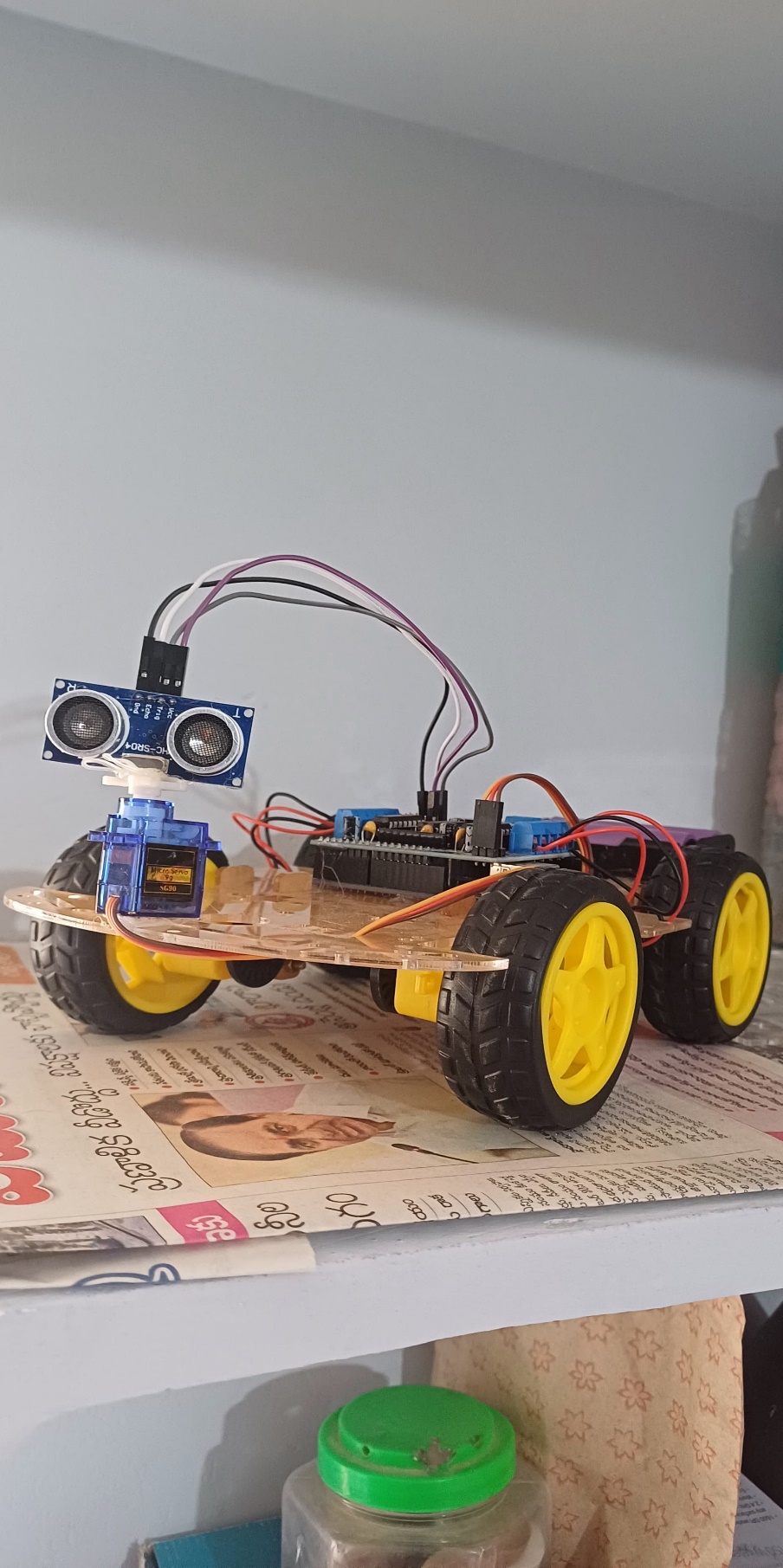
When the power supply is on , the power is supplied to both the Arduino and the motor driver (L293d) there by the led’s present in both Arduino and the motor driver starts glowing and the robot car starts moving forward. The ultra sonic sensor detects an obstacle within a certain range. The distance measurement is sent to the Arduino.

The Arduino processes the distance data. If the obstacle is too close, the Arduino sends signals to the motor driver to stop or change direction.

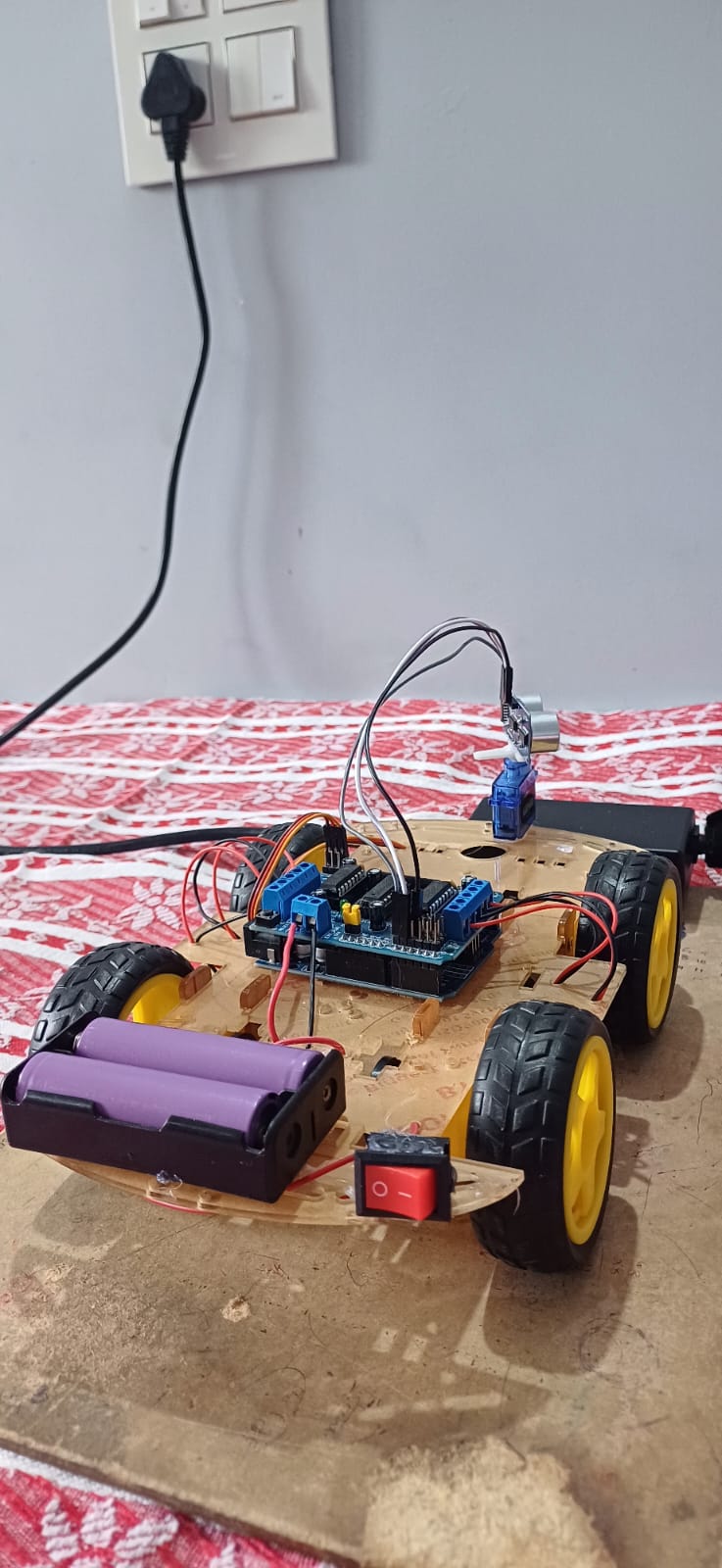
* The motor driver adjusts the gear motors to turn the robot away from the obstacle, allowing it to continue navigating.
* The robot continues to scan for obstacles and adjust its path as needed, navigating autonomously around its environment.

## **6.2 OUTPUT**

The output of the obstacle detection robot project will be a functional autonomous robot capable of navigating its environment by detecting and avoiding obstacles. The system integrates the Arduino Uno, ultrasonic sensor, servo motor, gear motors, and motor driver to achieve smooth and responsive movement, effective obstacle avoidance, and reliable operation based on the power supplied by the battery.



#### FIG OUTPUT 6.2.1



#### FIG OUTPUT 6.2.2



#### FIG OUTPUT 6.2.3

# **CHAPTER-7**

## **7.1 CONCLUSION**

The obstacle detection robot project represents a successful integration of various electronic and mechanical components to achieve autonomous navigation capabilities. By utilizing an Arduino Uno as the central control unit, paired with an ultrasonic sensor for distance measurement, gear motors for movement, and a motor driver (L293D) for controlling motor direction and speed, the robot demonstrates the ability to effectively navigate its environment and avoid obstacles.

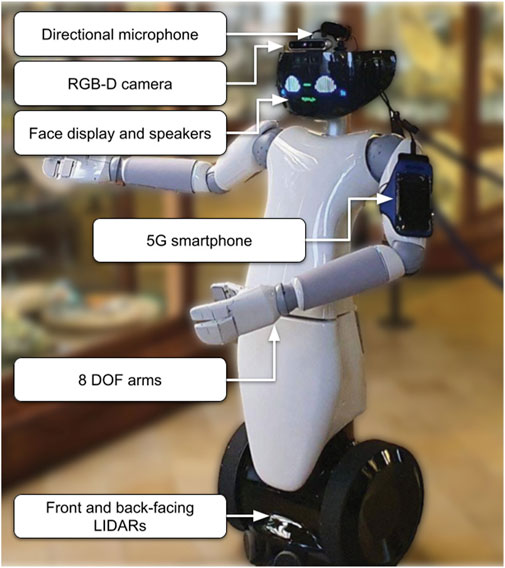
The project's result showcases a functional and reliable robot capable of real-time obstacle avoidance, reflecting the effective coordination between the hardware components and the software algorithms. The use of a servo motor for adjusting the sensor’s angle enhances the robot’s ability to scan a wider area, further improving its navigation accuracy.

In essence, this project not only highlights the practical application of fundamental robotics and electronics principles but also serves as a robust foundation for future enhancements, such as adding advanced sensors, improving navigation algorithms, or incorporating wireless control features. The successful execution of this project underscores the importance of integrating diverse technologies to create intelligent, autonomous systems capable of operating efficiently in dynamic environments.

# **7.2FUTURE SCOPE**

The future scope of obstacle detection robots is quite promising. These robots play a crucial role in various fields like autonomous vehicles, search and rescue operations, surveillance, and even in household applications like robotic vacuum cleaners. As technology advances, obstacle detection robots are expected to become more efficient, accurate, and versatile.

They may incorporate advanced sensors, artificial intelligence algorithms, and machine learning techniques to navigate complex environments with greater precision and safety. Overall, the future looks bright for obstacle detection robots as they continue to evolve and find new applications in different industries.



# **7.3REFERENCES**

 Kirti Bhagat, Sayali Deshmukh, Shraddha Dhonde, Sneha Ghag, "Obstacle Avoidance Robot," Bachelor of Computer Engineering, IJSETR, Volume 5, Issue 2, February 2016.

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

 Sachin Modi, "Comparison of Three Obstacle Avoidance Methods for an Autonomous Guided Vehicle," Master's Thesis, Maharashtra Institute of Technology, University of Pune, India, 1998, pp. 24-27.

 J. Chahl, M. Srinivasan, "Panoramic Vision System for Imaging, Ranging and Navigation in Three Dimensions," Proceedings of the International Conference on Field and Service Robotics, 1999, pp. 127-132.

 R. Bischoff, "Advances in the Development of the Humanoid Service Robot HERMES," Proceedings of the International Conference on Field and Service Robotics, 1999, pp. 156-161.

 D. Apostolopoulos, M. Wagner, W. Whittaker, "Technology and Field Demonstration Results in the Robotic Search for Antarctic Meteorites," Proceedings of the International Conference on Field and Service Robotics, 1999, pp. 185-190.

 L. Navarro-Serment, C. Paredis, P. Khosla, "A Beacon System for the Localization of Distributed Robotic Teams," Proceedings of the International Conference on Field and Service Robotics, 1999, pp. 232-237.

 T. Oomichi, N. Kawauchi, Y. Fuke, "Hierarchy Control System for Vehicle Navigation Based on Information of Sensor Fusion Perception Depending on Measuring Distance Layer," Proceedings of the International Conference on Field and Service Robotics, 1999, pp. 197-201.

 M. Torrie, S. Veeramachaneni, B. Abbott, "Laser-Based Obstacle Detection and Avoidance System," Proceedings of the SPIE Conference on Robotics and Semi-Robotic Ground Vehicle Technology, 1998, pp. 2-7.

 A. Foessel, S. Chheda, D. Apostolopoulos, "Short-Range Millimeter-Wave Radar Perception in a Polar Environment," Proceedings of the International Conference on Field and Service Robotics, 1999, pp. 133-138.