

Visvesvaraya Technological University
Belgaum, Karnataka-590 018



A Project Report on

**“OBSTACLE AVOIDANCE ROBOTIC VEHICLE
USING ULTRASONIC SENSOR AND ARDUINO
CONTROLLER”**

*Project Report submitted in partial fulfillment of the requirement for the
award of the degree of*

Bachelor of Engineering
In
Electrical & Electronics Engineering

Submitted by
SHILPA GS (1CR16EE077)
NIVEDHITHA CS (1CR17EE408)
VINUTHA V (1CR16EE418)

Under the Guidance of
Mrs. Parvathy Thampi. M S
Assistant Professor, Department of Electrical & Electronics Engineering
CMR Institute of Technology



CMR Institute of Technology, Bengaluru-560 037

Department of Electrical & Electronics Engineering

2020-2021

CMR INSTITUTE OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING
AECS Layout, Bengaluru-560 037



Certified that the project work entitled “**OBSTACLE AVOIDANCE ROBOTIC VEHICLE USING ULTRASONIC SENSOR AND ARDUINO CONTROLLER**” carried out by Ms. SHILPA GS, USN 1CR16EE077, NIVEDHITHA CS USN-1CR17EE408, VINUTHA V USN-1CR16EE418 are bonafied students of CMR Institute of Technology, Bengaluru, in partial fulfillment for the award of Bachelor of Engineering in Electrical & Electronics Engineering of the Visvesvaraya Technological University, Belgaum, during the year 2020-2021. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Signature of the Guide

Signature of the HOD

Signature of the Principal

Mrs. Parvathy ThampiMS
Assistant Professor
EEE Department
CMRIT, Bengaluru

Dr. K. Chitra
Professor & HOD
EEE Department
CMRIT, Bengaluru

Dr. Sanjay Jain
Principal,
CMRIT, Bengaluru

External Viva

Name of the Examiners

Signature & Date

- 1.
- 2.

CMR INSTITUTE OF TECHNOLOGY
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING
AECS Layout, Bengaluru-560 037



DECLARATION

We, **SHILPA GS,(1CR16EE077), NIVEDHITHA CS (1CR17EE408), VINUTHA V (1CR16EE418)** hereby declare that thereport entitled “**OBSTACLE AVOIDANCE ROBOTIC VEHICLE USING ULTRASONIC SENSOR AND ARDUINO CONTROLLER**” has been carried out by us under the guidance of **Mrs, Parvathy Thampi**, Assistant Professor, Department of Electrical & Electronics Engineering, CMR Institute of Technology, Bengaluru, in partial fulfillment of the requirement for the degree of **BACHELOR OF ENGINEERING in ELECTRICAL & ELECTRONICS ENGINEERING**, of Visveswaraya Technological University, Belagaum during the academic year 2020-21.The work done in this report is original and it has not been submitted for any other degree in any university.

Place: Bengaluru

Date:

SHIIPA GS

(1CR16EE077)

NIVEDHITHA CS

(1CR17EE408)

VINUTHA V

(1CR16EE418)

Abstract

In today's world ROBOTICS is a fast growing and interesting field. ROBOT has sufficient intelligence to cover the maximum area of provided space. Introduces the design and implementation of an autonomous obstacle -avoiding robot car using ultrasonic wave sensor in this thesis. By sending pulses, the obstacle avoidance distance can be measured. At the same time, we can control steering gear to realize the obstacle avoidance function. The robot car uses front axle steering, rear wheel drive arrangement. Two drive tires are driven by two DC motors with gear reduction mechanisms. Using Arduino MCU chip as the control core of the Robot car. Through the design of the hardware and software system, we build the robot car platform and obtain good experimental effect.

Keywords: Wheeled ROBOT, Autonomous, Intelligent, Arduino microcontroller, Artificial Intelligence

Acknowledgement

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people, who are responsible for the completion of the project and who made it possible, because success is outcome of hard work and perseverance, but steadfast of all is encouraging guidance. So with gratitude we acknowledge all those whose guidance and encouragement served us to motivate towards the success of the project work.

*We take great pleasure in expressing our sincere thanks to **Dr. Sanjay Jain, Principal, CMR Institute of Technology, Bengaluru** for providing an excellent academic environment in the college and for his continuous motivation towards a dynamic career. We would like to profoundly thank **Dr. B Narasimha Murthy, Vice-principal of CMR Institute of Technology** and the whole **Management** for providing such a healthy environment for the successful completion of the project work.*

*We would like to convey our sincere gratitude to **Dr. K Chitra, Head of Electrical and Electronics Engineering Department, CMR Institute of Technology, Bengaluru** for her invaluable guidance and encouragement and for providing good facilities to carry out this project work.*

*We would like to express our deep sense of gratitude to **Parvathy Thampi MS, Assistant Professor, Electrical and Electronics Engineering, CMR Institute of Technology, Bengaluru** for his/her exemplary guidance, valuable suggestions, expert advice and encouragement to pursue this project work.*

*We are thankful to all the faculties and laboratory staffs of **Electrical and Electronics Engineering Department, CMR Institute of Technology, Bengaluru** for helping us in all possible manners during the entire period.*

Finally, we acknowledge the people who mean a lot to us, our parents, for their inspiration, unconditional love, support, and faith for carrying out this work to the finishing line. We want to give special thanks to all our friends who went through hard times together, cheered us on, helped us a lot, and celebrated each accomplishment.

*Lastly, to the **Almighty**, for showering His Blessings and to many more, whom we didn't mention here.*

Table of Contents

1	Chapter One: Introduction and Background	1
1.1	Introduction	1
1.2	The main Objectives	2
1.3	Methodology	2
1.4	Project Layout	2
2	Chapter Two: History and Literature Review.....	3
2.1	Artificial Intelligence	3
2.2	Robotics and Robots	3
2.3	Robot Working	3
2.4	The Actuator	4
2.5	Robot Learning	4
2.6	Autonomous Robot	5
3	Chapter Three: Introduction to Arduino.....	7
3.1	Arduino Overview	7
3.2	Arduino History	7
3.3	Arduino Development	8
3.4	Arduino Evolution	8
3.5	Arduino Past and Present	9
3.6	What are Arduino Boards?	9
3.7	Why Arduino Boards?	10
3.8	Features of Arduino Boards.	11
4	Chapter Four: The Final Project Construction, Testing, Results	13
4.1	Block Diagram (Hardware Implementation)	13
4.1.1	Arduino UNO R3	14
4.1.2	HC-SR04 Ultrasonic Sensor.....	17
4.1.3	L298D Motor Driver Shield	21

4.1.4	SG-90 Servo Motor	22
4.1.5	DC motor	26
4.2	robot working principle	28
4.3	Hardware components:	28
4.4	Necessary Tools and Machines:	28
4.5	the project chassisDesign	29
4.6	Constructing the Chassis	30
4.7	programing the robot code	37
4.7.1	Processes and flow of program	38
4.7.2	Arduino Coding	39
4.7.3	Servo System	41
4.7.4	SensorSystem	43
4.7.5	Driving system	43
5	Chapter five: Conclusion and Recommendation.	47
5.1	Conclusion	47
5.2	FutureRecommendations	47
5.3	FutureWorks	47
II		
5.4	Advantages	48
5.5	Limitations	48
5.6	Problems Faced	48

Table of Figures

Figure 2.1 Autonomous Urbie developed by NASA, is designed for various urban operations, including military reconnaissance and rescue operations.	6
Figure 3.1 Types of Arduino Boards	10
Figure 4.1 Block diagram depicting the Hardware Implementation for the project	13
Figure 4.2 Arduino UNO R3 board	14
Figure 4.3 Diagram of the basic ultrasonic sensor operation	17
Figure 4.4 Ultrasonic Sensor Pins	19
Figure 4.5 Block Diagram of Ultrasonic Sensor work	20
Figure 4.6 L298D Motor Driver Shield	21
Figure 4.7 SG-90 Servo Motor	22
Figure 4.8 diagram of Servo Closed Loop System	23
Figure 4.9 Position of motor shaft when PWM is generated.	24
Figure 4.10 DC Motors	26
Figure 4.11 DC Motor – Gear Motor	27
Figure 4.12 Block diagram of Robot working principle	28
Figure 4.13 The final Design of the Robot Chassis	29
Figure 4.14 Chassis Robot Components	30
Figure 4.15 3D Print chassis plate	30
Figure 4.16 Soldering the DC Motors	31
Figure 4.17 attach the front wheel	31
Figure 4.18 attach the rear wheels to the chassis	32
Figure 4.19 schematic circuit diagram of the switch and the power source	32
Figure 4.20 Soldering the ON/OFF Switch	33
Figure 4.21 The hardware model depicting my project model (Obstacle Avoiding Robot) ..	34
Figure 4.22 Schematics Block Diagram of wire up the components	34
Figure 4.23 connect ultrasonic sensor Pins to the Motor Shield	35
Figure 4.24 Block Diagram of connect ultrasonic sensor Pins to the Motor Shield	35
Figure 4.25 Wire up the Servo Motor to Motor Shield (Servo _ port 2)	36
Figure 4.26 Wire up the DC motors to Motor Shield	36
Figure 4.27 Arduino integrated development environment (IDE)	37
Figure 4.28 The Algorithm/ Flowchart Diagram of The Obstacle-avoiding Robot Car Based on Arduino Microcontroller	38
Figure 4.29 Turning function by servo motor to detect obstacles	42
Figure 4.30 Ultrasonic Sensor	43
Figure 4.31 Robot Driving System	44
Figure 5.1 Ultrasonic Sensor Problems Faced	48

1 Chapter One: Introduction and Background

1.1 Introduction

Robotics is part of Today's communication. In today's world ROBOTICS is fast growing and interesting field. It is simplest way for latest technology modification. Now a day's communication is part of advancement of technology, so I decided to work on ROBOTICS field, and design something which will make human life simpler in today aspect.

An autonomous robot is a robot that is capable of moving on its own in an unknown and unstructured environment. An autonomous robot is equipped with software intelligence to sense its environment, detect obstacles in its path and move around an unknown environment overcoming the obstacles. There are many robotic designs that are employed in designing of autonomous robots. These designs are usually developed considering the physical environment in which the robot has to be deployed. There are autonomous robots like snake robots, walking robots, autonomous drones and autonomous robotic cars or rovers.

This ROBOT has sufficient intelligence to cover the maximum area of provided space. It has an infrared sensor which are used to sense the obstacles coming in between the path of ROBOT.

It will move in a particular direction and avoid the obstacle which is coming in its path. The main motto of designing such type of Robot or the technology is that this technology can be used in today's very fast transportation to avoid the accident generally happen in congested by applying emergency break. If we use this technology in the car or any vehicle, it will automatically sense the obstacles then it will take a side to the available free space. An obstacle may be a living things or any object.

Autonomous Intelligent Robots are robots that can perform desired tasks in unstructured environments without continuous human guidance. Thus, by using this technology in vehicles we make the drive safe.

1.2 The main Objectives

The main objectives of the project are comprehended as follows:

- the obstacle avoidance robot is able to move around in an unknown environment without colliding with surrounding objects.
- The robot would have the capacity to detect obstacles in its path based on a predetermined threshold distance.
- After obstacle detection, the robot would change its course to a relatively open path by making autonomous decision.
- It would require no external control during its operation.
- It can measure the distance between itself and the surrounding objects in real-time.
- It would be able to operate effectively in unknown environment.
- Obstacle avoiding robots can be used in almost all mobile robot navigation systems
- They can be used for household work like automatic vacuum cleaning.
- They can also be used in dangerous environments, where human penetration could be fatal.

1.3 Methodology

- simulations of Obstacle-avoiding Robot will be performed before and after optimization in Proteus Simulink, and others tools if needed.
- The methodology may include research, surveys and other research techniques, and could include both present and historical information.
- Prepare the required components to do a practical simulation by Arduino microcontrollers to make the Robot
- Start to design the final project

1.4 Project Layout

This project has been divided to five chapters as following:

chapter one is gives the introduction and the objective and methodology of proposed Obstacle Avoiding Robot design. Chapter two consist of History and Literature Review. Chapter three shows the introduction to Arduino, Overview, History, Development, Evolution, Arduino past and present, Arduino Boards, Features of Arduino Boards. Chapter four shows The Final Project Construction, Testing, Results.

Chapter five contains conclusion and recommendation.

2 Chapter Two: History and Literature Review

2.1 Artificial Intelligence

Artificial intelligence is the branch of computer science that develops machines and software with human-like intelligence. It is the intelligence exhibited by software or machines. The central goals of artificial intelligence research include knowledge, reasoning, learning, planning, perception, the ability to manipulate and move objects and natural language processing.

The field was founded on the claim that a central property of humans is intelligence, and that it can be sufficiently well described to the extent that a machine can simulate it.

2.2 Robotics and Robots

It is a branch of technology and deals with designing, construction, operation, and application of robots. It also deals with the computer systems for their sensory, control, information processing and feedback. These technologies deal with automated machines that can replace humans in manufacturing processes or dangerous environments. These robots resemble humans in behavior, appearance, and/or cognition. Robotics requires a working knowledge of mechanics, electronics, and software [15].

Robots are machines and are of a wide range. The common feature of robots is their capability to move. They perform physical tasks. Robots have many different forms. They range from industrial robots, whose appearance is dictated by the function they are to perform. Or they can be humanoid robots, which mimic the human movement and our form.

Robots can be grouped generally as:

- Manipulator robots (for e.g. industrial robots)
- Mobile robots (for e.g. autonomous vehicles),
- Self-reconfigurable robots, the robots that can conform themselves to the task at hand.

Robots may act according to their own decision-making ability, provided by artificial intelligence or may be controlled directly by a human, such as remotely-controlled bomb-disposal robots and robotic arms; or. However, the majority of robots fall in between these extremes, being controlled by pre-programmed computers[2].

2.3 Robot Working

Human beings on a basic level are made of five major components:

-
- A muscle system that can move the body structure
 - A body structure itself
 - A power source that can activate the muscles and sensors
 - A sensory system which can receive information about the body and the surrounding environment
 - A brain system which can process sensory information and tell the muscles what to do.

Robots are made up of the same components as above. A typical autonomous robot has a sensor system, a movable physical structure, a power supply and a computer brain that controls all of these elements. Basically, robots are man-made versions of the animal life. They are machines that can replicate human and animal behavior.

2.4 The Actuator

All robots have a movable body (almost all). Some have motorized wheels only, while others may have a dozen of movable parts (that are typically made of plastic or metal). Like bones in a human body, the individual segments are connected together with the help of joints.

Robots use actuators to spin wheels and jointed pivot. Some robots use solenoids and electric motors as actuators; others some use a pneumatic system (a system driven by compressed gases); yet others use a hydraulic system. A robot may even use all of these actuator types together.

Robots need a power source to be able to drive the actuators. Most robots have a battery or they plug into an electricity source. Pneumatic robots need air compressors or compressed air tanks and hydraulic robots need a pump that pressurizes the hydraulic fluid. The actuators are wired to an electrical circuit. The circuit powers these electrical motors and solenoids directly. It also activates the hydraulic system by manipulating electrical valves. The valves determine the pressurized fluid's path through the machine.

2.5 Robot Learning

Robot learning is an intersecting research field between robotics and machine learning. It studies techniques that allow robots to acquire skills and adapt to its environment by learning various algorithms. Learning can take place either by self-exploration or through guidance (from a human teacher), like in robot learning that learns by imitation.

2.6 Autonomous Robot

Autonomous robots are independent of any controller and can act on their own. The robot is programmed to respond in a particular way to an outside stimulus. The bump-and-go robot is a good example. This robot uses bumper sensors to detect obstacle. When the robot is turned on, it moves in a straight direction and when it hits an obstacle, the crash triggers its bumper sensor. The robot gives a programming instruction that asks the robot to back up, turn to the right direction and move forward. This is its response to every bump. In this way, the robot can change direction every time, it encounters an obstacle.

A more elaborate version of the same idea is used by more advanced robots. Roboticists create new sensor systems and algorithms to make robots more perceptive and smarter. Today, robots are able to effectively navigate a variety of environments. Obstacle avoidance can be implemented as a reactive control law whereas path planning involves the pre-computation of an obstacle-free path which a controller will then guide a robot along.

Some mobile robots also use various ultrasound sensors to see obstacles or infrared. These sensors work in a similar fashion to animal echolocation. The robot sends out a beam of infrared light or a sound signal. It then detects the reflection of the signal. The robot locates this distance to the obstacles depending on how long it takes the signal to bounce back.

Some advanced robots also use stereo vision. Two cameras provide robots with depth perception. Image recognition software then gives them the ability to locate, classify various objects. Robots also use smell and sound sensors to gain knowledge about its surroundings.

More advanced robots are able to analyze unfamiliar environments and adapt to them. They even work on areas with rough terrain. This kind of robots can associate particular terrain patterns with particular actions.

For example, a rover robot constructs a land map with the help of its visual sensors. If the map depicts a bumpy terrain pattern, the robot decides to travel some other way. Such kinds of system are very useful for exploratory robots and can be used to operate on other planets. Figure 2.1 shows the bot developed by NASA called Urbie. It is designed for various military purposes and is able to move through stairs and other such paths .

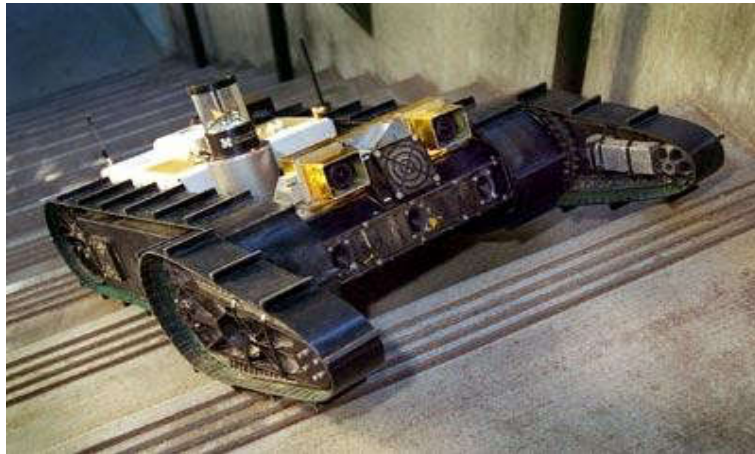


Figure 2.1 *Autonomous Urbie developed by NASA, is designed for various urban operations, including military reconnaissance and rescue operations.*

An ant tries to get over an obstacle, it does not decide when it needs to get over an obstacle. It simply keeps trying different things until it gets over the obstacle. An alternative robot design takes a similar less structured approach, which can also be termed as a randomness approach. When the robot gets stuck, it moves its appendages in every way until something works out. Force sensors work very closely with the actuators, instead of the computer directing everything based on a program.

3 Chapter Three: Introduction to Arduino

3.1 Arduino Overview

Arduino is a popular programmable board used to create projects. It consists of a simple hardware platform as well as a free source code editor which has a “one click compiles or upload” feature. Hence it is designed in way that one can use it without necessarily being an expert programmer (Kushner 1987). Arduino offers an open-source electronic prototyping platform that is easy to use and flexible for both the software and hardware. Arduino is able to sense the environment through receiving input from several sensors.

It is also able to control its surrounding through controlling motors, lights and other actuators. The Arduino programming language that is based on the wiring and the Arduino development environment that is based on the processing are used to program the microcontroller found on the board (Banzi, 2005). Due to its open-source environment, one is able to easily write and upload codes to the I/O board. It is also worth to note that Arduino can be run on Linux, Mac OSX and Windows as its environment is written in Java



3.2 Arduino History

Arduino was released in 2005 by students from the Interaction Design Institute Ivrea (IDII) as a modest tool for Mac OSX and Windows. Since then, Arduino has been able to initiate an international-do-it yourself revolution at the electronics industry. The open source microcontroller hardware has been designed in a way that it can easily interface with various sensors (registering user inputs) and driving the behaviors and responses of the external components such as speakers, motors, and LED (responding to the user inputs). The most important feature of Arduino is the ease of programmability hence users with little expertise are able to use it. This aspect has made Arduino one of the most popular tools of choice for designers and artists in creating interactive spaces and objects (Arduino Team).

3.3 Arduino Development

While discussing the development of Arduino, it is worth introducing a brief history of microcontrollers. A revolutionary leap in the computing industry was seen in the 1960s following the development of solid state computers (including the IBM 1401), that used transistors to process its operations and a magnetic core memory for its storage (instead of vacuum tubes), and these enabled an increase in the compactness of the computer hardware. In addition, Jack Kilby's invention of integrated circuits in 1959 enabled circuits and transistors to be fused into tiny chips of semiconducting materials (like silicon) as well as further miniaturization of the computer component. The other crucial development made in the same decade was the high-level computer programming languages, written in symbolic languages such as plain English, and this made computer codes somehow easy to learn and read than the earlier machine languages that consisted of letters and numbers only. This development enabled individuals with few years of expertise to carry out the basic operations on a computer.

FORTRAN (for the scientific calculators) and COBOL (for business application) were the two main languages that were introduced in that period.

The microprocessor was one of the greatest innovations in the history of the modern computer in the 1970's. Initially, the microprocessor miniaturized all the hardware components of CPU to fit into one, tiny, integrated circuit, popularly known as the microchip. The microchip became the major driving component of the microcontrollers including Arduino which is made up of a microchip, input/output hardware and memory storage hardware for sensors. The microprocessor, due to the small form factor, was incorporated into a surfeit of electronic devices ranging from personal computers to calculators and are still used up to date. More programming languages were also developed in the 1970s and 80s including C, C++ and Java for applications in science and business. (Massimo, 2005)

3.4 Arduino Evolution

Having looked at the evolution of microcontrollers, there have been recent incarnations of the microcontrollers that have been designed in a way to fulfill the needs of hobbyists and casual users who happen to have a limited technical knowledge. In other words, the microcontrollers have moved from the more complex requirements in the scientific, business or commercial fields. Before the invention of Arduino, the PIC microcontroller board that was introduced by general instruments in 1985 was one of the most used tools for the electronic enthusiasts. The reasons as to why the PIC microcontroller board was preferred were the speed and ease of its

programming through simple languages including PBASIC. An additional reason was that it was able to store programs on a flash memory chip that enabled the instructions on the board to be reprogrammed or erased at will with an infinite number of possibilities. It also supported output devices such as LEDs and motors as well as input sensors. There are other popular boards for the hobbyists including BASIC Stamp and wiring which are some of the microcontroller boards that were designed for tangible media explorations and electronic art. The two boards share the advantages of ease of rapid prototyping and simplicity of programming.

It was in 2005 when the Arduino team was formed in Italy and it consisted of Barragan Massimo, David Cuartielles, Gianluca Marino, Dave Mellis and Nicholas Zambetti. The main goal of this team was to develop an electronic prototyping platform that would simplify the wiring platform and make it accessible to the non-technical users especially in the creative fields. The Arduino, therefore, incorporated several characteristics including a programming environment that is based on the processing language that was conceived by Casey Reas and Ben Fry and other artists and designers. Arduino also incorporated the ability to program its board using a standard USB connection with a low price point (Wheat, 2001).

3.5 Arduino Past and Present

Within its first 2 years of existence, Arduino achieved rapid success where more than 50,000 boards were sold. By 2009, Arduino had more than 13 different incarnations with each having a specialized application. For instance, Arduino Mini was a miniature to be used in small interactive objectives, Arduino BT was built with Bluetooth capabilities, and Arduino LilyPad for wearable technologies projects. Today, the Arduino microcontroller is a popular prototyping platform across the world and it is a good example of how software and hardware technologies that were originally created for business, military or scientific applications have been repurposed so as to serve the needs of people developing projects in new media and arts and design.

3.6 What are Arduino Boards?

Arduino board is an open-source platform used to make electronics projects. It consists of both a microcontroller and a part of the software or Integrated Development Environment (IDE) that runs on your PC, used to write & upload computer code to the physical board. The platform of an Arduino has become very famous with designers or students just starting out with electronics, and for an excellent cause.

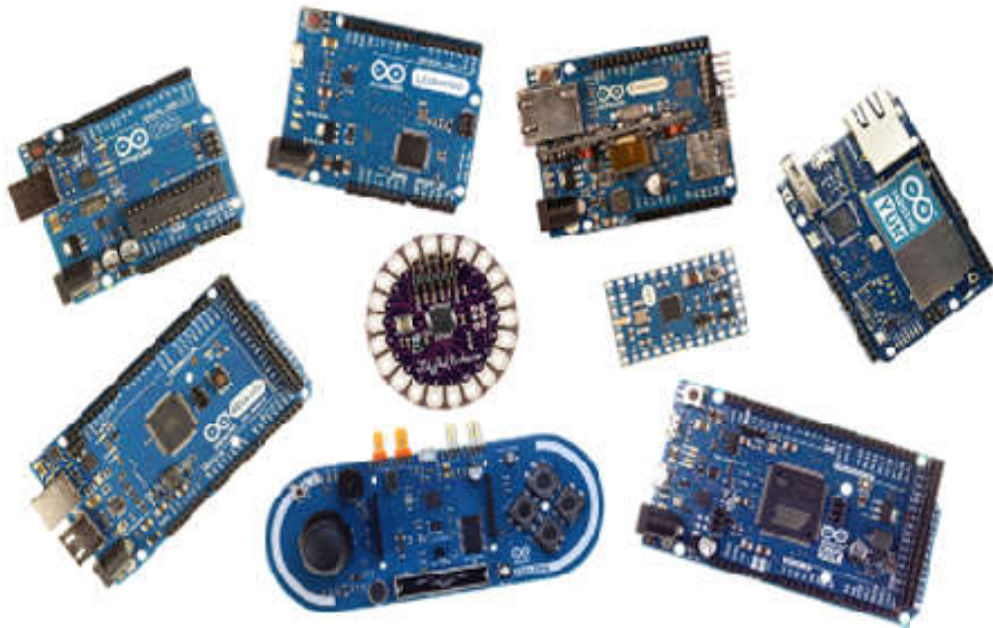


Figure 3.1 Types of Arduino Boards

3.7 Why Arduino Boards?


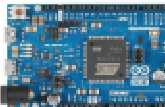
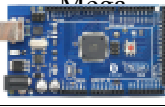

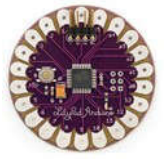

Arduino board has been used for making different engineering projects and different applications. The Arduino software is very simple to use for beginners, yet flexible and adequate for advanced users. It runs Windows, Linux, and Mac.




Teachers and students in the schools utilize it to design low-cost scientific instruments to verify the principles of physics and chemistry. There are numerous other microcontroller platforms obtainable for physical computing. The Netmedia's BX-24, Parallax Basic Stamp, MIT's Handyboard, Phidget, and many others present related functionality.

Arduino also makes simpler the working process of a microcontroller, but it gives some advantages over other systems for teachers, students, and beginners.

- Inexpensive
- Cross-platform
- Simple, clear programming environment
- Open source and extensible software
- Open source and extensible hardware

3.8 Features of Arduino Boards.

Arduin o Board					
	5V	16Mhz ATmega328			
		84MHz AT91SAM3X8E			
	5V	16MHz ATmega2560			
			SRAM, 32KB flash		
	2.7-5.5 V	ATmega 168v or 328v			
	5V	Atmel ATmega328	2KB SRAM, 32KB Flash	14	8 inputs, output

<p>Arduino</p> 	<p>3.3V</p>	<p>ATmega328P</p>	<p>3.3KB SRAM, 32KB Flash</p>	<p>14</p>	<p>8 inputs, output</p>
<p>Arduino Mini</p> 	<p>5v</p>	<p>ATmega328</p>	<p>2KB SRAM, 32KB Flash</p>	<p>14</p>	<p>outputs</p>
<p>Arduino Pro Mini</p> 	<p>3.3V or 5V</p>	<p>ATmega328</p>	<p>2KB SRAM, 16KB Flash</p>	<p>14</p>	<p>output</p>

4 Chapter Four: The Final Project Construction, Testing, Results

4.1 Block Diagram (Hardware Implementation)

Figure 4.1 below shows the block diagram that describes the process for the designing and development of the hardware.

The hardware implementation phase is divided into six components.

The components are:

- HC-SR04 Ultrasonic Sensor
- SG-90 Servo Motor
- Arduino UNO R3
- L298D Motor Driver IC
- Power Supply
- Left DC Motor and Right DC Motor

Each component has been explained below along with their circuitual representation as used in the building of the robot. The software implementation is included in the microcontroller phase

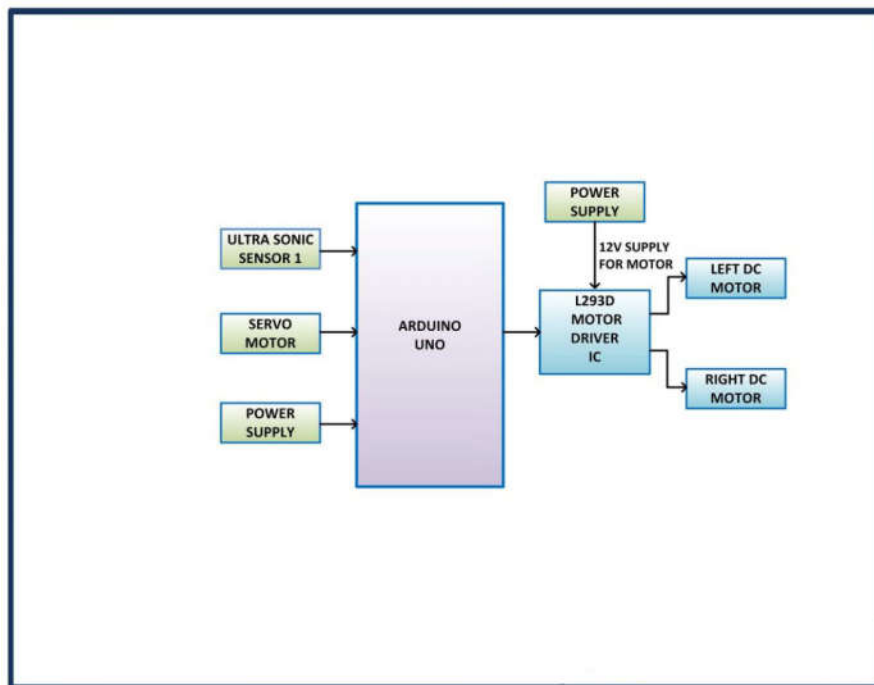


Figure 4.1 *Block diagram depicting the Hardware Implementation for the project*

4.1.1 Arduino UNO R3

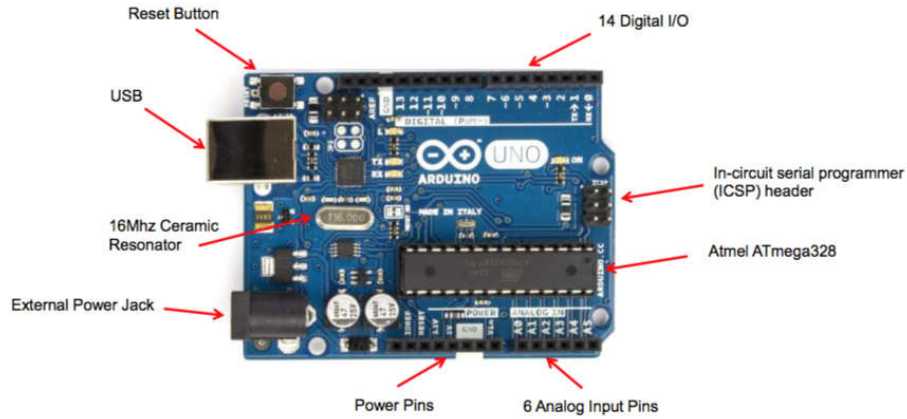


Figure 4.2 Arduino UNO R3

board4.1.1.1 Arduino Uno Overview – Power

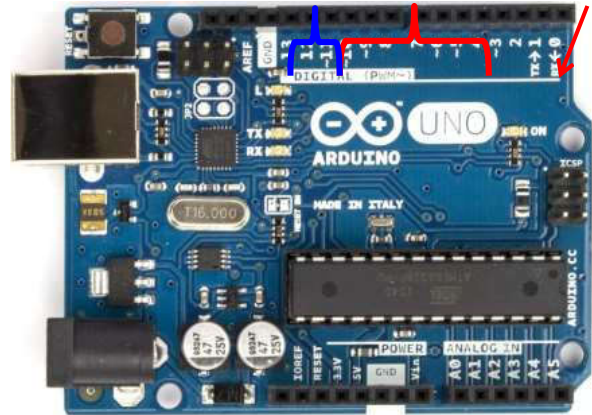
- USB power or externally via barrel jack connector
- External supply voltage: 7 to 12 DC

P i n	Function		
V i n	Voltage from Ext. Power jack		
5 V	5V outp	ut from on-board Voltage regulator chip	
3.3 V	3.3V out	put from on-board Voltage regulator chip	
G n d	3 pins		
I O R	EF	Tiedto5V,tellsArduino	shieldsvoltagelevelfromwhich Arduino boardoperates
R e s	et	FromResetpinonMC	U,tiedtoVCCthrough10K resistor,pulltoGNDtoreset

4.1.1.2 Arduino Uno Overview – Digital Input / Output

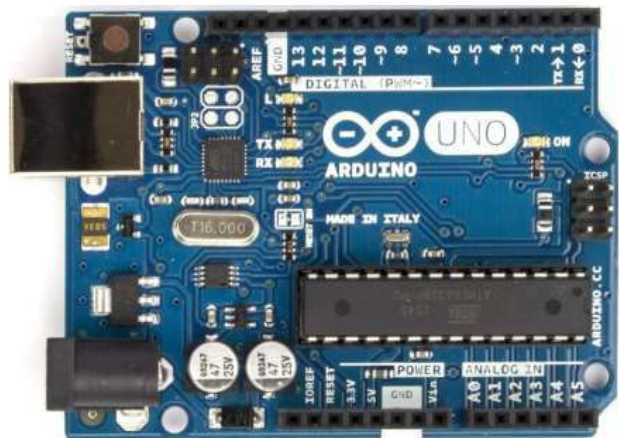
- 14 Digital I/O
- 5V logic level
- Sink/source 20mA per pin
- Universal Asynchronous Serial Receiver / Transmitter (UART)
 - ❑ Pin 0 – Receive (RX)
 - ❑ Pin 1 – Transmit (TX)
 - ❑ MCU INT pins brought out to pins 2, 3
- 8-bit PWM
 - ❑ Pins 3, 5, 6, 9, 10, 11
- Serial Peripheral Interface (SPI)
 - ❑ Pins 10, 11, 12, 13

SPI PWM ~ TX / RX



4.1.1.3 Arduino Uno Overview – Analog

- Analog Inputs (A0 to A5)
- Sampled input:
 - ❑ 0 to 5V
 - ❑ 10 bits resolution
- AREF pin can be used to adjust
 - upper limit of input range
- Two-wire Interface COM (TWI)
 - Pin A4 – SDA pin (data)
 - Pin A5 – SCL pin (clock)



Analog Pins

4.1.1.4 Arduino Uno Overview – Communication

- Can communicate with a PC, another Arduino, shields, sensors
- Asynchronous communication (No clock):
 - UART TTL (5V)
 - Digital Pin 0 (RX) / Digital Pin 1 (TX)
 - On-board ATmegaU16 programmed to function as USB to serial chip
 - UART => USB format and chip appears as virtual COM port to PC
 - ATmegaU16 uses standard USB COM drivers, so no external drivers are needed
- (Windows needs .inf file)
 - TX / RX LEDs flash when data is being transmitted to and from Uno via USB
- to serial chip
- Synchronous communication (Clock) using:
 - SPI
 - Pin10–SlaveSelect(SS)
 - Pin11–MasterOut/SlaveIn(MOSI)
 - Pin12–MasterIn/SlaveOut(MISO)
 - Pin13–Clock(SCK)
 - TWI
 - PinA4–SDA pin (data)
 - PinA5–SCL pin (clock)

4.1.1.5 Arduino Uno Overview – Programming

- Bootloader firmware preinstalled with Uno
- Allows you to upload new sketches via USB
- Can also use external programmer to upload sketches via the In-circuit serial programmer (ICSP) pins
- Uno Memory
 - 32KB of Flash memory (Sketches stored here)
 - 2KB of SRAM (variables stored until power cycled)
 - 1KB of EEPROM (store long term info: calibration constants, serial #, etc.)

4.1.2 HC-SR04 Ultrasonic Sensor



Figure 4.3 Diagram of the basic ultrasonic sensor operation

4.1.2.1 HC-SR04 Ultrasonic Sensor – Overview

- What is an ultrasonic?
 - ULTRA = BEYOND
 - SONIC = SOUND The sound beyond human hearing range (20000Hz) is known as ultrasonic.
- What is Ultrasonic sensor?
 - Ultrasonic sensors are sensors that convert ultrasound waves to electrical signals or vice versa.

An Ultrasonic sensor is a device that can measure the distance to an object by using sound waves. It measures distance by sending out a sound wave at a specific frequency and listening for that sound wave to bounce back. By recording the elapsed time between the sound wave being generated and the sound wave bouncing back, it is possible to calculate the distance between the sonar sensor and the object.

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1” to 13 feet. Its operation is not affected by sunlight or black material like sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

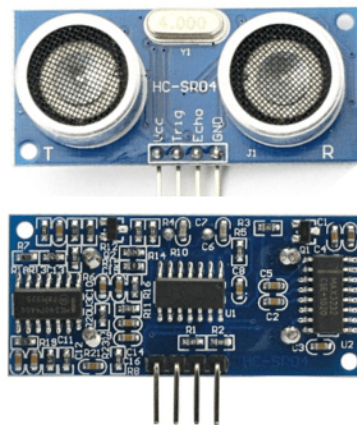
Since it is known that sound travels through air at about 344 m/s (1129 ft/s), you can take the time for the sound wave to return and multiply it by 344 meters (or 1129 feet) to find the total

round-trip distance of the sound wave. Round-trip means that the sound wave traveled 2 times the distance to the object before it was detected by the sensor; it includes the 'trip' from the sonar sensor to the object AND the 'trip' from the object to the Ultrasonic sensor (after the sound wave bounced off the object). To find the distance to the object, simply divide the round-trip distance in half.

$$distance = \frac{speed\ of\ sound \times time\ taken}{2}$$

4.1.2.2 HC-SR04 Ultrasonic Sensor – Features

- Power Supply: +5V DC
- Quiescent Current: <2mA
- Working Current: 15mA
- Effectual Angle: <15°
- Ranging Distance: 2cm – 400 cm/1”
– 13ft
- Resolution: 0.3 cm
- Measuring Angle: 30 degrees
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm



4.1.2.3 HC-SR04 Ultrasonic Sensor Pin Configuration

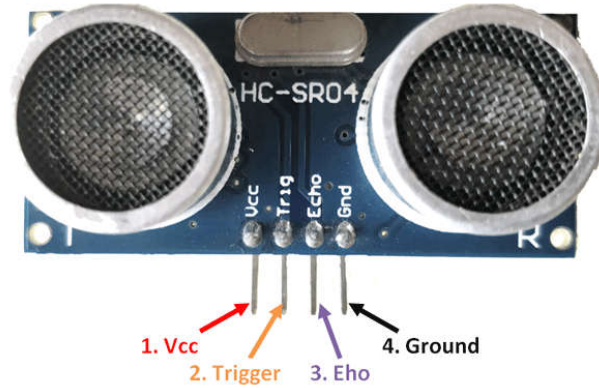


Figure 4.4 Ultrasonic Sensor Pins

Pin Number	Pin Name	Description
1	VCC	The VCC pin powers the sensor, typically with +5V
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor.
4	Ground	This pin is connected to the Ground of the system.

4.1.2.4HC-SR04 Ultrasonic Sensor – Work

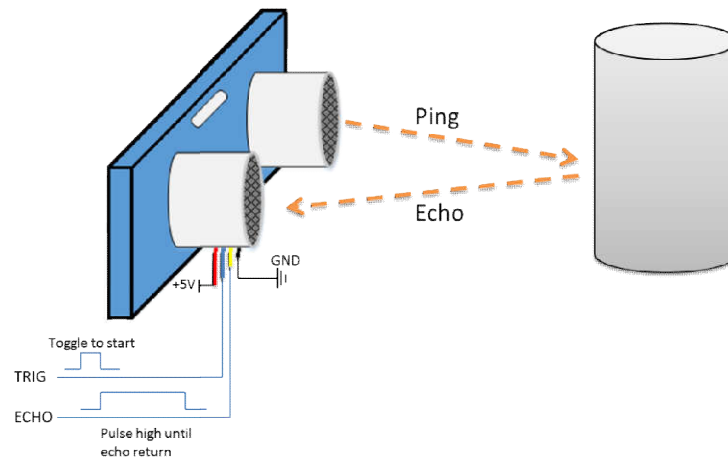


Figure 4.5 Block Diagram of Ultrasonic Sensor work

The ultrasonic sensor uses sonar to determine the distance to an object. Here's what happens:

- the transmitter (trig pin) sends a signal: a high-frequency sound
- when the signal finds an object, it is reflected and
- the transmitter (echo pin) receives it.

For example, if the object is 10 cm away from the sensor, and the speed of the sound is 340 m/s or 0.034 cm/ μ s the sound wave will need to travel about 294 μ seconds. But what we will get from the Echo pin will be double that number because the sound wave needs to travel forward and bounce backward. So, in order to get the distance in cm we need to multiply the received travel time value from the echo pin by 0.034 and divide it by 2 as shown above.

4.1.2.5HC-SR04 Ultrasonic Sensor - Applications

- Used to avoid and detect obstacles with robots like biped robot, obstacle avoider robot, path finding robot etc.
- Used to measure the distance within a wide range of 2cm to 400cm
- Can be used to map the objects surrounding the sensor by rotating it
- Depth of certain places like wells, pits etc. can be measured since the waves can penetrate through water

4.1.3 L298D Motor Driver Shield

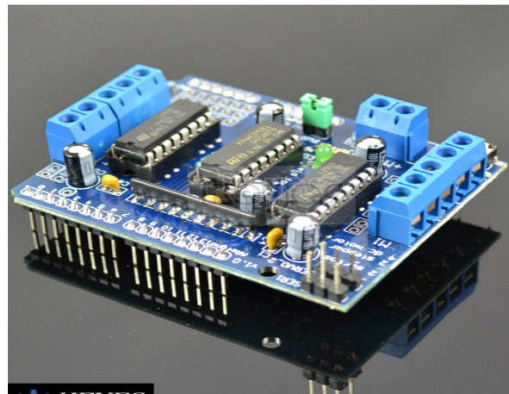


Figure 4.6 *L298D Motor Driver Shield*

4.1.3.1L298D Motor Driver Shield – Overview

L293D is a monolithic integrated, high voltage, high current, 4-channel driver. Basically, this means using this chip we can drive DC motors with power supplier up to 36 Volts, and the chip can supply a maximum current of 600mA per channel. L293D chip is also known as a type of H-Bridge. The H-Bridge is typically an electrical circuit that enables a voltage to be applied across a load in either direction to an output, e.g. motor.

4.1.3.2L298D Motor Driver Shield – Features

- 2 connections for 5V 'hobby' servos connected to the Arduino's high-resolution dedicated timer - no jitter! □
- Up to 4 bi-directional DC motors with individual 8-bit speed selection (so, about 0.5% resolution) □
- Up to 2 stepper motors (unipolar or bipolar) with single coil, double coil, interleaved or micro-stepping. □

- 4 H-Bridges: L293D chipset provides 0.6A per bridge (1.2A peak) with thermal shutdown protection, 4.5V to 12V □
- Pull down resistors keep motors disabled during power-up □
- Big terminal block connectors to easily hook up wires (10-22AWG) and □power □
- Arduino reset button brought up top □
- 2-pin terminal block to connect external power, for separate logic/motor □supplies □
- Tested compatible with Mega, UNO & Duemilanove □
- Dimensions: 69mm x 53mm x 14.3mm (2.7in x 2.1in x 0.6in) □

4.1.3.3 L298D Motor Driver Shield – Application

- It is used in Robotics projects requiring stepper motor interface.
- Multiple DIY projects.

4.1.4 SG-90 Servo Motor



Figure 4.7 SG-90 Servo Motor

4.1.4.1 Servo Motors – Overview

A servo motor is an electrical device which can push or rotate an object with great precision. If we want to rotate an object at some specific angles or distance, then we use a servo motor. It is just made up of a simple motor which runs through a servo mechanism. If a motor is used is DC powered then it is called a DC servo motor, and if it is AC powered motor then it is called an AC servo motor. We can get a very high torque servo motor in a small and light weight package. Due to these features they are being used in many applications like toy cars, RC helicopters and planes, Robotics, Machine etc. The position of a servo motor is decided by electrical pulse and its circuitry is placed beside the motor.

Nowadays servo systems have huge industrial applications. Servo motor applications are also commonly seen in remote controlled toy cars for controlling direction of motion and it is also very commonly used as the motor which moves the tray of a CD or DVD player. Besides these there are other hundreds of servo motor applications we see in our daily life. The main

reason behind using a servo is that it provides angular precision, i.e. it will only rotate as much we want and then stop and wait for next signal to take further action. This is unlike a normal electrical motor which starts rotating as and when power is applied to it and the rotation continues until we switch off the power. We cannot control the rotational progress of electrical motor; but we can only control the speed of rotation and can turn it ON and OFF.

4.1.4.2 Servo Mechanism

It consists of three parts:

- Controlled device
- Output sensor
- Feedback system

It is a closed loop system where it uses positive feedback system to control motion and final position of the shaft. Here the device is controlled by a feedback signal generated by comparing output signal and reference input signal.

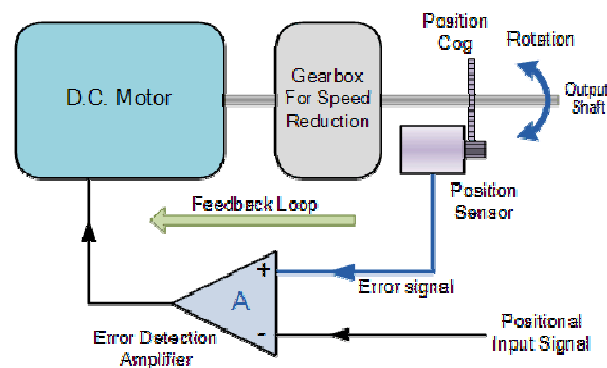


Figure 4.8 diagram of Servo Closed Loop System

Here reference input signal is compared to reference output signal and the third signal is produced by feedback system. And this third signal acts as input signal to control device. This signal is present as long as feedback signal is generated or there is difference between reference input signal and reference output signal. So, the main task of servomechanism is to maintain output of a system at desired value at presence of noises.

4.1.4.3 Servo Motors - Working principle

A servo consists of a Motor (DC or AC), a potentiometer, gear assembly and a controlling circuit.

First of all, we use gear assembly to reduce RPM and to increase torque of motor. Say at initial position of servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. Now an electrical

signal is given to another input terminal of the error detector amplifier. Now difference between these two signals, one comes from potentiometer and another comes from other source, will be processed in feedback mechanism and output will be provided in term of error signal. This error signal acts as the input for motor and motor starts rotating. Now motor shaft is connected with potentiometer and as motor rotates so the potentiometer and it will generate a signal. So as the potentiometer's angular position changes, its output feedback signal changes. After sometime the position of potentiometer reaches at a position that the output of potentiometer is same as external signal provided. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer, and in this situation motor stops rotating.

4.1.4.4 Servo Motors – Controlling

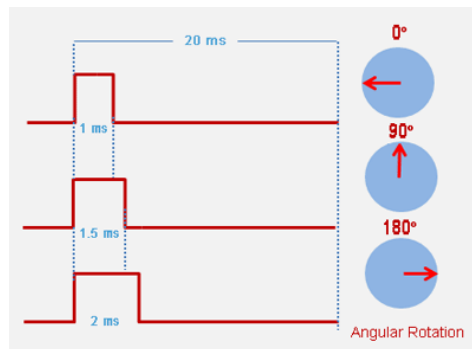


Figure 4.9 Position of motor shaft when PWM is generated.

Servo motor is controlled by PWM (Pulse with Modulation) which is provided by the control wires. There is a minimum pulse, a maximum pulse and a repetition rate. Servo motor can turn 90 degrees from either direction from its neutral position. The servo motor expects to see a pulse every 20 milliseconds (MS) and the length of the pulse will determine how far the motor turns. For example, as shown above Figure 4.9 a 1.5ms pulse will make the motor turn to the 90° position, such as if pulse is shorter than 1.5ms shaft moves to 0° and if it is longer than 1.5ms than it will turn the servo to 180°. Servo motor works on PWM (Pulse width modulation) principle, means its angle of rotation is controlled by the duration of applied pulse to its Control PIN.

Basically, servo motor is made up of DC motor which is controlled by a variable resistor (potentiometer) and some gears. High speed force of DC motor is converted into torque by Gears. We know that $WORK = FORCE \times DISTANCE$, in DC motor Force is less and distance (speed) is high and in Servo, force is High and distance is less. Potentiometer is

connected to the output shaft of the Servo, to calculate the angle and stop the DC motor on required angle.

To recap, there are two important differences between the control pulse of the servo motor versus the DC motor. First, on the servo motor, duty cycle (on-time vs. off-time) has no meaning whatsoever—all that matters are the absolute duration of the positive-going pulse, which corresponds to a commanded output position of the servo shaft. Second, the servo has its own power electronics, so very little power flows over the control signal. All power is drawn from its power lead, which must be simply hooked up to a high-current source of 5 volts.

4.1.4.5 Servo Motors – Applications

- Robotics
- Animatronics
- Radio Control Cars/Boats/Planes

4.1.4.6 Servo Motors – Advantages

- Low cost - (RC Servos) Smaller sized servos can be purchased for just a few dollars.
- Variety - There is a wide range of sizes and torque ratings
- Simple to control - using logic level pulses from a microcontroller or a dedicated servo controller

4.1.5 DC motor

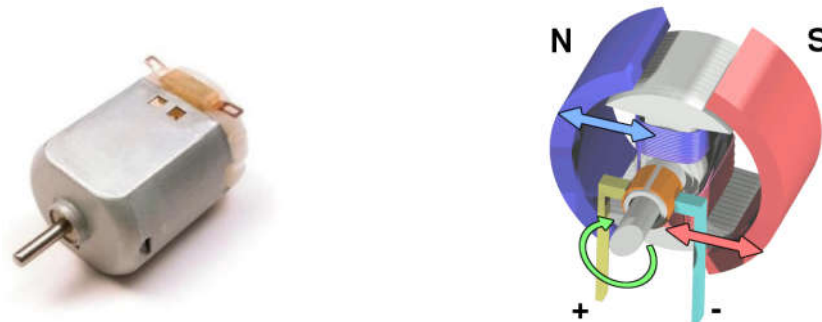


Figure 4.10 *DC Motors*

4.1.5.1 DC Motor – Overview

A Direct Current (DC) motor is a rotating electrical device that converts direct current, of electrical energy, into mechanical energy. An Inductor (coil) inside the DC motor produces a magnetic field that creates rotary motion as DC voltage is applied to its terminal. Inside the motor is an iron shaft, wrapped in a coil of wire. This shaft contains two fixed, North and South, magnets on both sides which causes both a repulsive and attractive force, in turn, producing torque.

4.1.5.2 DC Motor – Types

These types of motors are powered by direct current (DC).

- Brushed Motors
- Brushless Motors
- Planetary Gear Motors
- Spur Gear Motors
- Stepper Motors
- Coreless & Coreless Brushless Motors
- Servo Motors
- Gear heads Motors

4.1.5.3 DC Motor – Gear Motor



Figure 4.11 *DC Motor – Gear Motor*

Gear Motors – The most common electrical motors convert electrical energy to mechanical energy.

A gear motor is an all-in-one combination of a motor and gearbox. The addition of a gear head to a motor reduces the speed while increasing the torque output. The most important parameters in regards to gear motors are

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

In order to select the most suitable gear motor for our application we must first compute:

- Load
- Speed
- Torque requirements for our application

4.2 robot working principle

The robot uses the Ultrasonic distance sensor to measure the distance in front of it. When this distance reduces to a particular level, the robot interprets it to mean the presence of an obstacle in its path. When the robot detects an obstacle in its path, it stops, goes backward for a few cm, looks around (right and left) then turn towards the direction that shows more free space in front of it.



Figure 4.12 *Block diagram of Robot working*

*principle*4.3 Hardware components:

- 1.1x Arduino Uno
- 2.1x Robot Chassis as designed
- 3.1x Arduino Motor Driver Shield L293D Model
- 4.1x HC-SR04 Ultrasonic Sensor
- 5.1x Micro Servo 9g (SG90)
- 6.2x DC Motor + Gearbox
- 7.2x Wheels
- 8.1x Caster wheel
- 9.2x 9v Battery
- 10.1x 9v Battery Clip
11. Jumper Wires as required
- 12.1x ON/OFF Switch

4.4 Necessary Tools and Machines:

- 1x Soldering Iron (generic)
- Solder as required

-
- 1x Hot Glue Gun (generic)
 - Hot Glue as required
 - Arduino IDE (Integrated Development Environment)
 - Proteus 8 Pro Simulation Software

4.5the project chassis Design

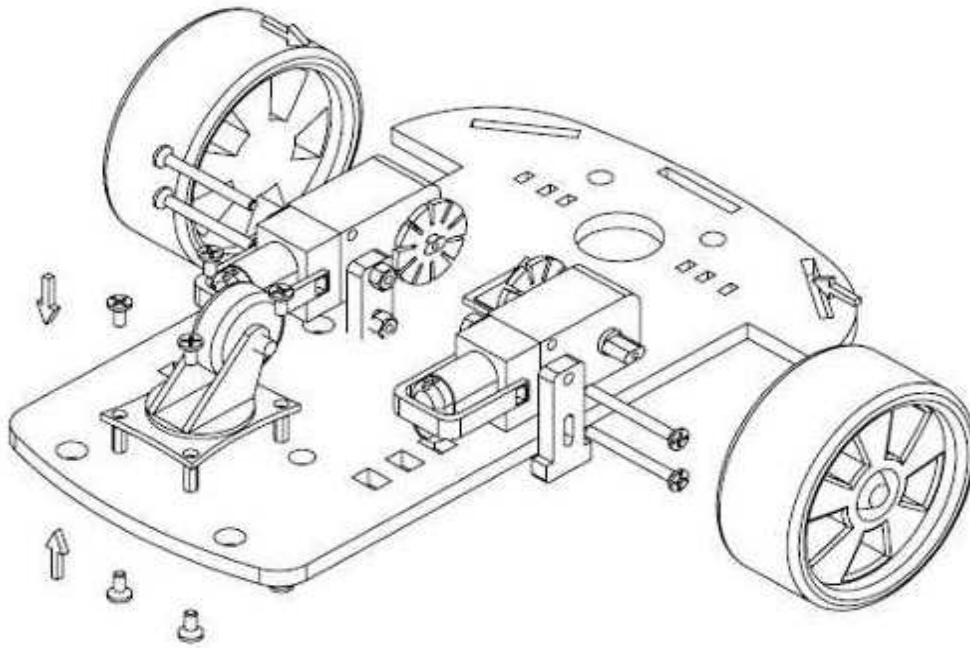


Figure 4.13 *The final Design of the Robot Chassis*

4.6 Constructing the Chassis

To begin, we start by coupling the chassis of the robot. The robot kit, contains the chassis, two geared DC motors, the wheels, the front wheel, battery holder, some screws, and wires.



Figure 4.14 *Chassis Robot Components*

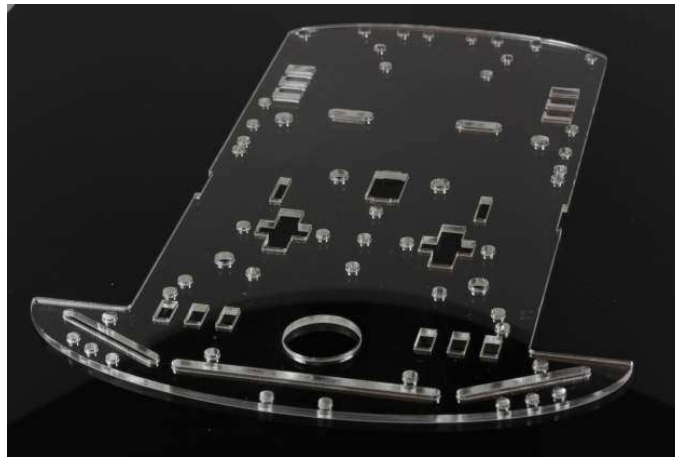


Figure 4.15 *3D Print chassis plate*

➤ Step 1. Connect the motor and wheels to the chassis.

- To complete this step, we start by soldering the thick red and black wires to the positive and negative terminals of the motors as shown in the image below.

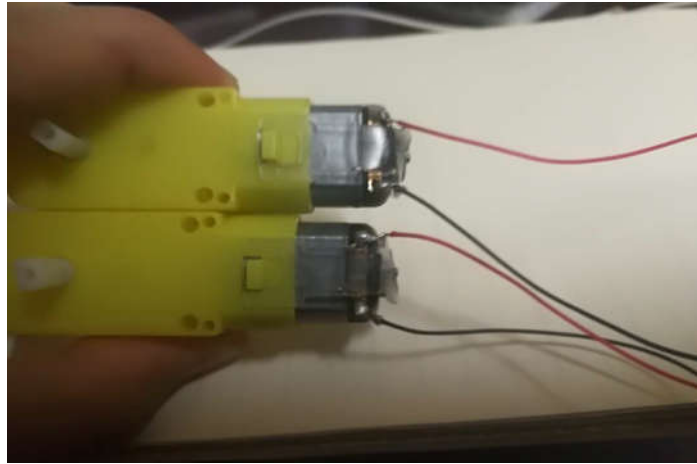


Figure 4.16 *Soldering the DC Motors*

- attach the front wheel as shown in the image below.



Figure 4.17 *attach the front wheel*

- attach the rear wheels to the chassis

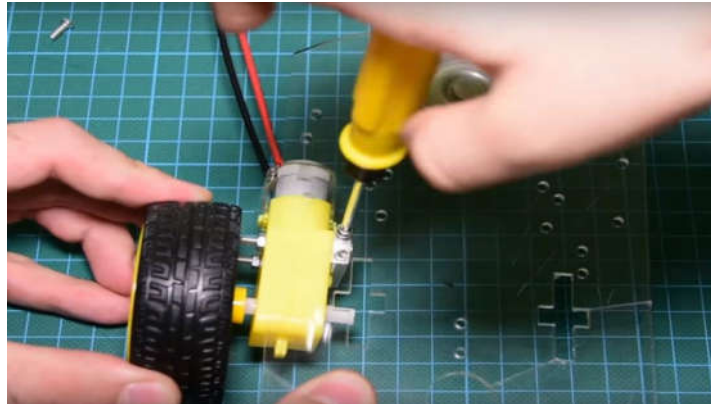


Figure 4.18 *attach the rear wheels to the chassis*

➤ Step 2. Prepare the Switch and connect the Power Source

we add a switch to the battery holder so that we will be able to turn the robot on or off. The switch is connected according to the schematics shown below and attached to the case using a hot glue. The Battery case is attached to the chassis using a double-sided tape to ensure everything sticks together.

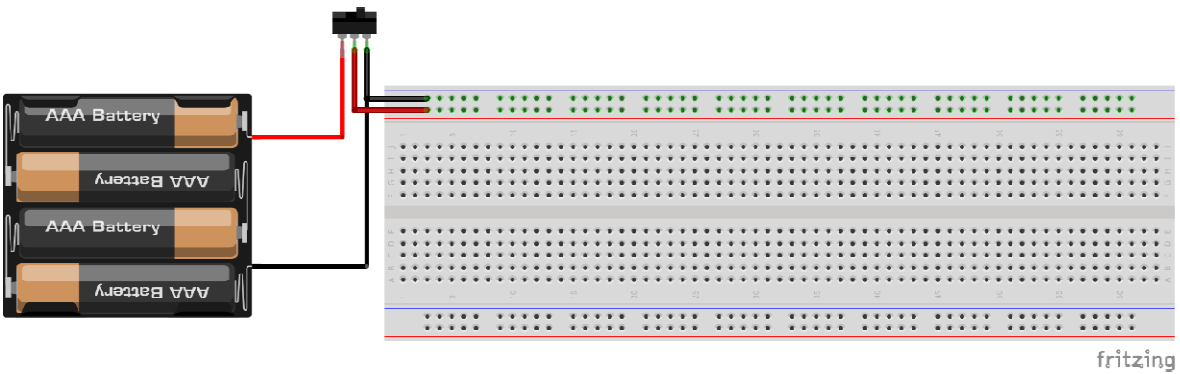


Figure 4.19 *schematic circuit diagram of the switch and the power source*

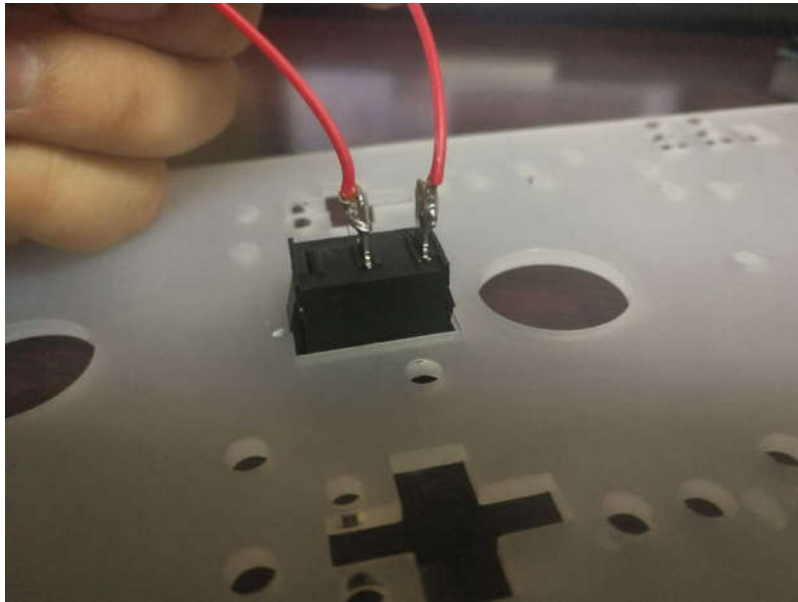


Figure 4.20 *Soldering the ON/OFF Switch*

➤ Step 3. installation the others parts of the robot

This step is to mount other parts of the robot before we start connecting their wires. The motor shield is stacked on the Arduino and it is mounted on the chassis using a double-sided tape. The current requirements of the motors are often higher than what the Arduino can provide, that is why it's important to use the motor shield as it is equipped with additional circuitry to provide up to 600mA current to each of the motors. This shield provides power to the motors and the servo motor and ultrasonic sensor and makes it much easier. The Ultrasonic sensor is also mounted on the top of the servo motor which is then mounted on the chassis using some screws.

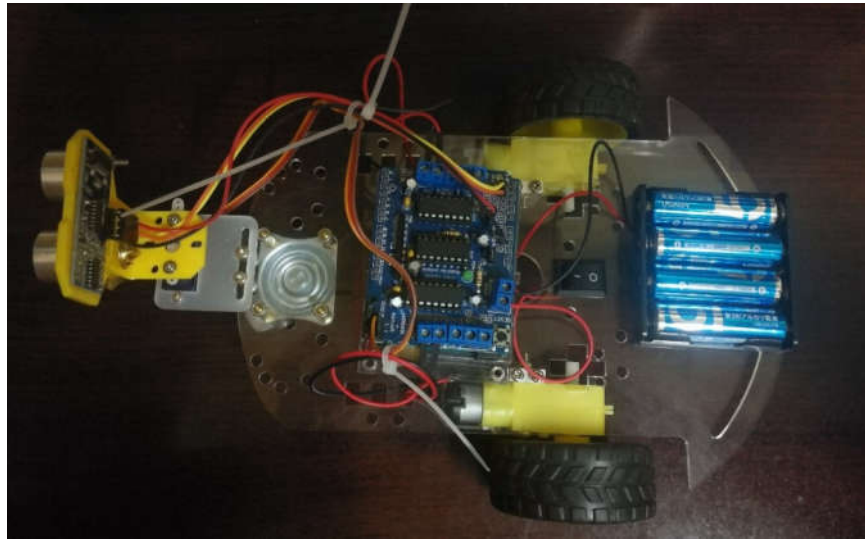


Figure 4.21 *The hardware model depicting my project model (Obstacle Avoiding Robot)*

➤ Step 4. Wire up the components

To simplify the connections, below is a pin map of how the components connect, one to the other.

Wire up the components together as shown in the image below.

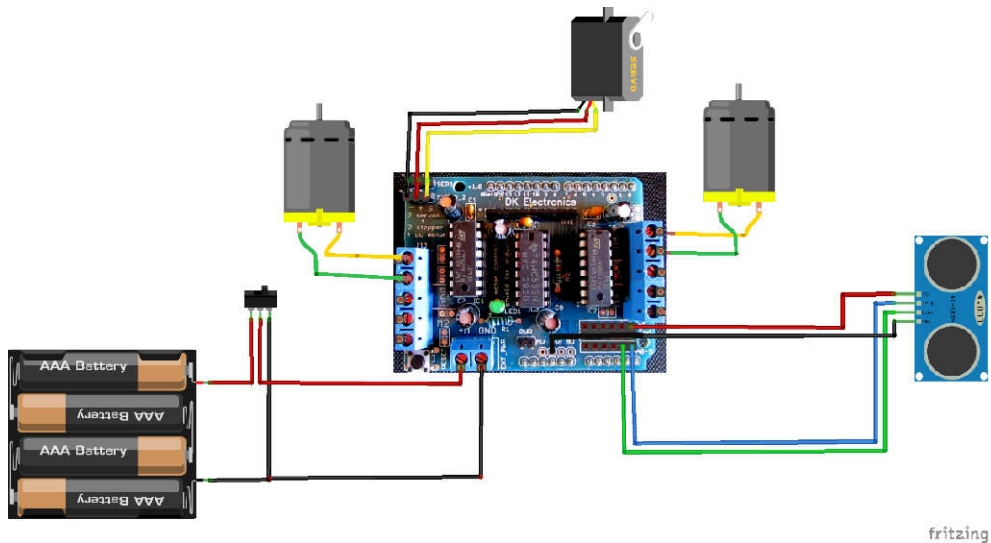


Figure 4.22 *Schematics Block Diagram of wire up the components*

➤ Ultrasonic Sensor □ Motor Shield

- VCC □ 5v
- Gnd □ Gnd
- Trig □ A4
- Echo □ A5

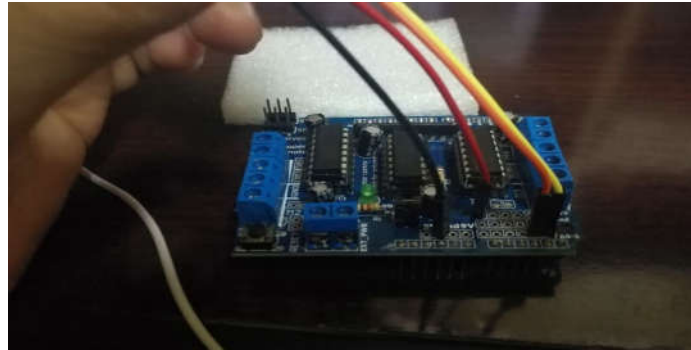


Figure 4.23 connect ultrasonic sensor Pins to the Motor Shield

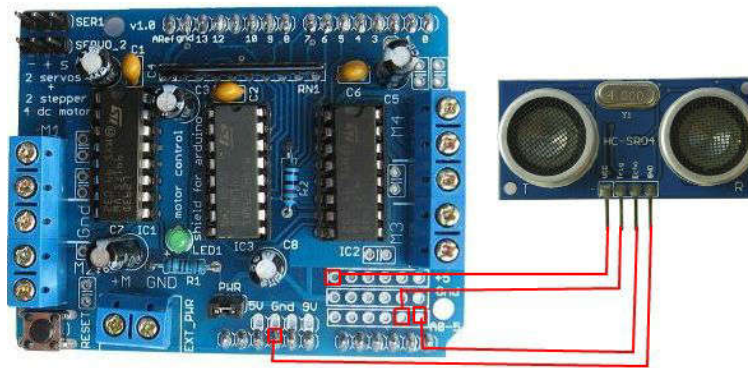


Figure 4.24 Block Diagram of connect ultrasonic sensor Pins to the Motor Shield

➤ Servo □ Motor Shield (Servo_2 port)

- Signal (yellow wire) □ S
- Vcc (Red wire) □ +
- Gnd (Black wire) □ -

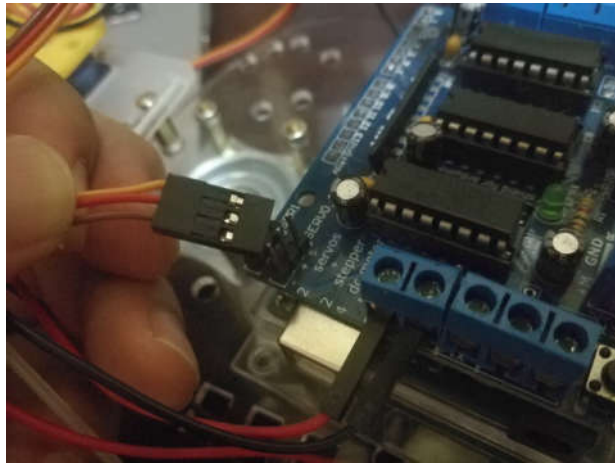


Figure 4.25 *Wire up the Servo Motor to Motor Shield (Servo _port 2)*

➤ DC motors □ Motor Shield

- Left Motor □ M1
- Right Motor □ M2

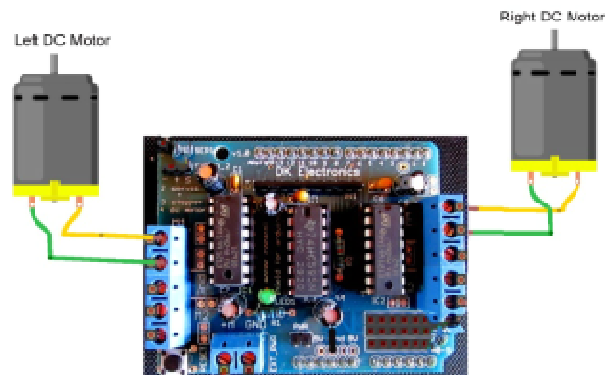


Figure 4.26 *Wire up the DC motors to Motor Shield*

4.7 programming the robot code

The controller board is used to communicate with the PC using serial communicator (USB connection). The data is transferred between them bit by bit. An adaptor is used to supply power to the controller board and a USB programmer is used to burn the hardware program (written in Arduino IDE) into the Arduino board.

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2.

The use of Artificial Intelligence in the designing of the robot is introduced in the programming phase of the robot. The programming for the Obstacle Detecting and Avoiding Robot is done in C Language and uses various pre-defined header file.

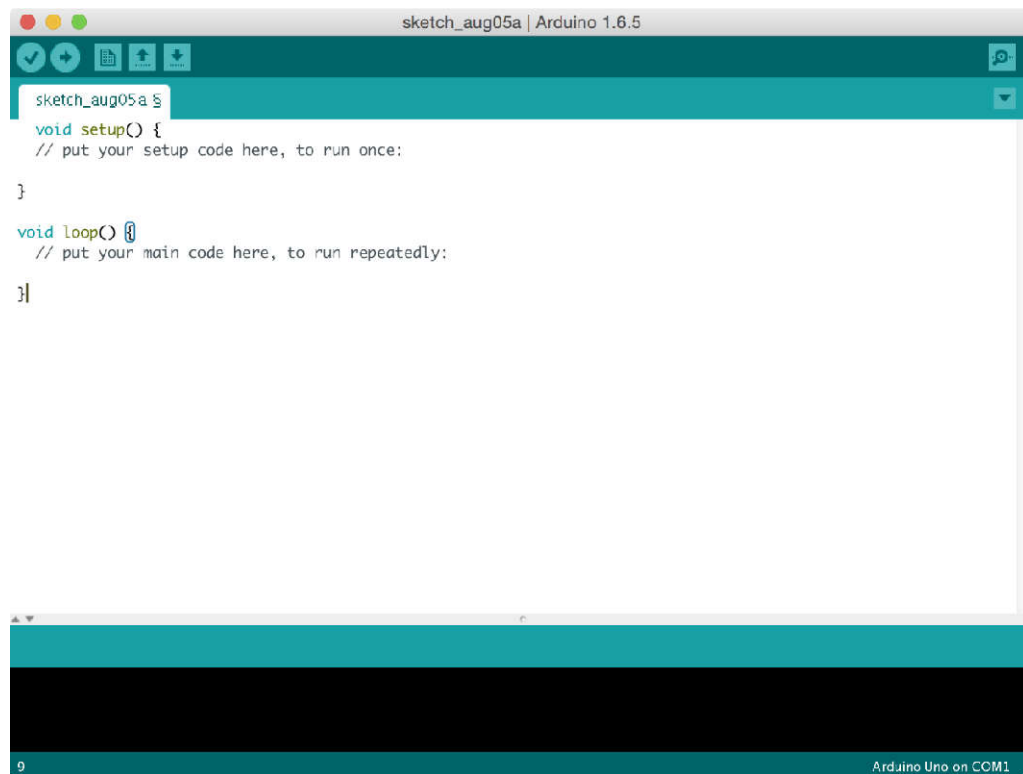


Figure 4.27 *Arduino integrated development environment (IDE)*

4.7.1 Processes and flow of program

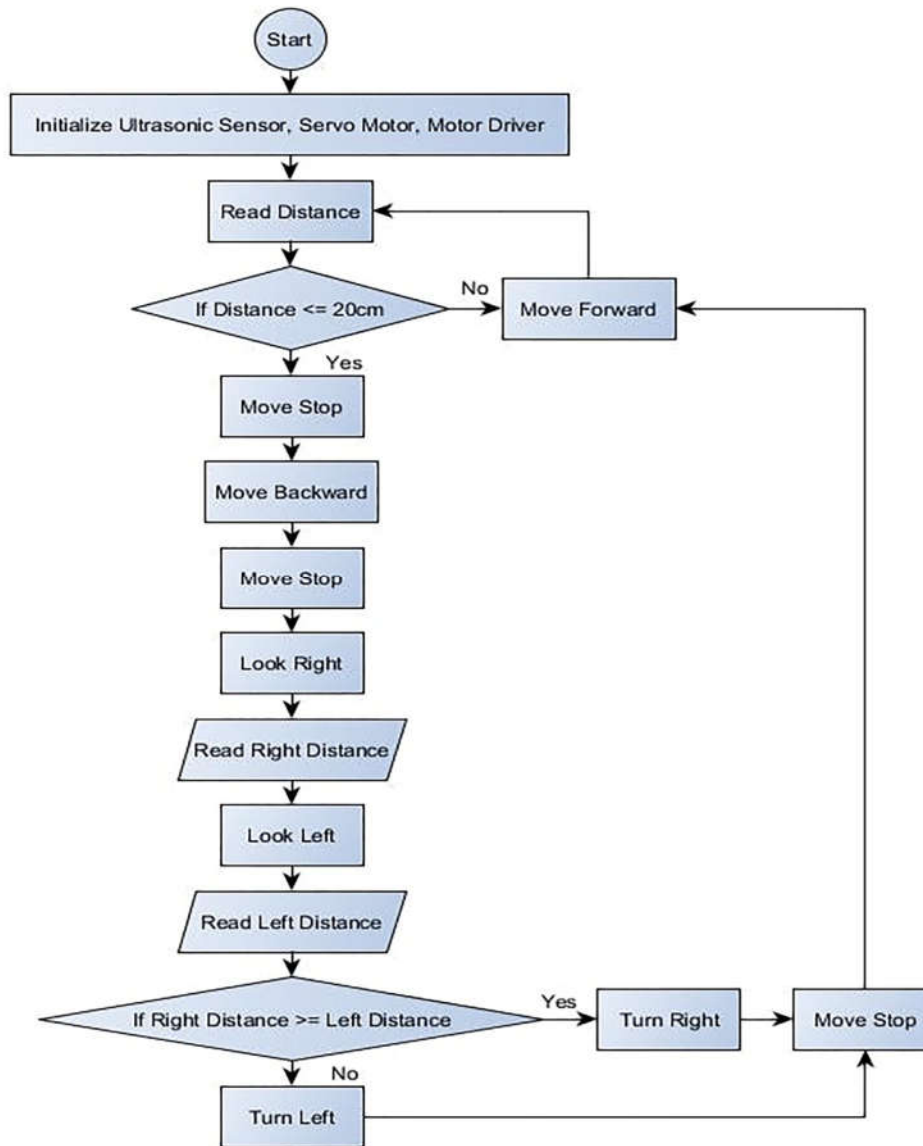


Figure 4.28 The Algorithm/ Flowchart Diagram of The Obstacle-avoiding Robot Car Based on Arduino Microcontroller

4.7.2 Arduino Coding

The code uses three libraries. Two of them must be downloaded in order for the program to compile. The first one is the motor shield driver from Adafruit. The second library is the NewPing library for the supersonic distance sensor.

The first thing we do in the code is to include the libraries that have been downloaded into the code.

- Motor Shield Library
- New Ping Library
- Servo Motor Library

```
#include <AFMotor.h>
#include <NewPing.h>
#include <Servo.h>
```

Next, we declare the pins to which our ultrasonic sensor is connected to, and some variables which will be used to store info as the code runs then we set the speed of the motors. We can set the motors speed in any value up to 255.

```
#define TRIG_PIN A4 // Arduino pin tied to trigger pin on the ultrasonic
sensor.
#define ECHO_PIN A5 // Arduino pin tied to echo pin on the ultrasonic
sensor.
#define MAX_DISTANCE 200 // Maximum distance we want to ping for (in
centimeters). Maximum sensor distance is rated at 250cm.
#define MAX_SPEED 200 // sets speed of DC motors
#define MAX_SPEED_OFFSET 20
```

Next, we initialize the servo motor by creating an object of the servo library. We also initialize the Motors to which the wheels are connected using the AF library.

```
AF_DCMotor motor1(1, MOTOR12_1KHZ);
AF_DCMotor motor2(3, MOTOR12_1KHZ);

Servo myservo;
```

move into the void setup () function where we initialize the servo motor and we set it to “look” straight. In my case, this was at 115 degrees angle.

Next, we read the distance a few times in order to get a valid distance measurement as shown below

We then

```

void setup() {

myservo.attach(9); // Attaches the servo on pin 9 to servo object.
myservo.write(115);
delay(2000); // Wait for 2s.
    distance =readPing(); // Get Ping Distance.
delay(100); // Wait for 100ms.
    distance =readPing();
delay(100);
    distance =readPing();
delay(100);
    distance =readPing();
delay(100);
}

```

Now we go to the loop function which executes every 40 Ms. If the distance we measured is less than or equal to 15 cm, we stop the motors, reverse for 300ms, stop, look right and left and measure the distance in each direction. If the distance in one direction is greater than the other we turn the robot to the direction with the farthest/greatest distance and instruct it to move forward as shown below.

```

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();
}
```

4.7.3 Servo System

this autonomous robot (auto mode) has 5 statuses: move forward, stop, move backward, scan right and left, and turn right or left. Since only 1 distance sensor used in this project, the sensor must have the capability to turn around to scan for alternative way to move so it must attach together with servo. The servo shaft can be turn in 180 degrees to help the sensor scan the obstacle around and choose the clear path to move along. A homemade bracket used to attach the sensor to the servo.

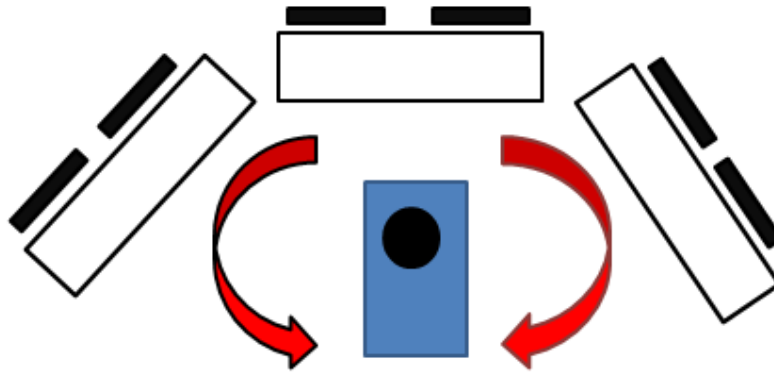


Figure 4.29 Turning function by servo motor to detect obstacles

4.7.3.1 Servo System Coding

```
int lookRight() // Look Right Function for Servo Motor.
{
myservo.write(50);
delay(500);
    int distance = readPing();
delay(100);
myservo.write(115);
    return distance;
}

int lookLeft() // Look Left Function for Servo Motor.
{
myservo.write(170);
delay(500);
    int distance = readPing();
delay(100);
myservo.write(115);
    return distance;
delay(100);
}
```

4.7.4 Sensor System

The obstacle sensor is used avoiding the robot from the clash to any external devices or that is like walls, any obstacle which comes in its way. Here I used the Ultrasonic Sensor as shown below.

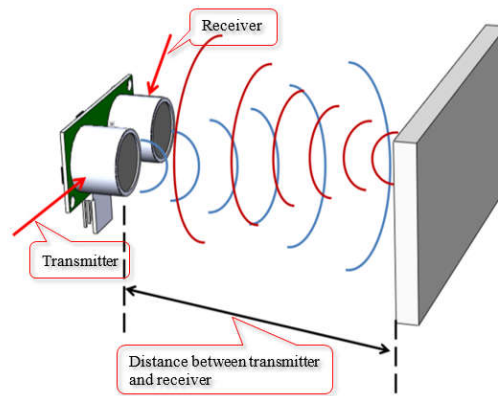


Figure 4.30 Ultrasonic Sensor

4.7.4.1 Sensor System Coding

```
int readPing() // Read Ping Function for Ultrasonic Sensor.
{
delay(70); // Wait 70ms between pings (about 20 pings/sec).
int cm = sonar.ping_cm(); //Send ping, get ping distance in centimeters
(cm).
if(cm==0)
{
cm = 250;
}
return cm;
}
```

4.7.5 Driving system

we will be using the following rotation of wheel for steering and straight motion:

MOTION	RIGHT Wheel	Left Wheel
F ORWARD	Counter Clockwise	Clockwise
BACKWARD	Clockwise	Counter Clockwise
R o t ate RIGHT	Counter Clockwise	Counter Clockwise
Rotate LEFT	Clockwise	Clockwise

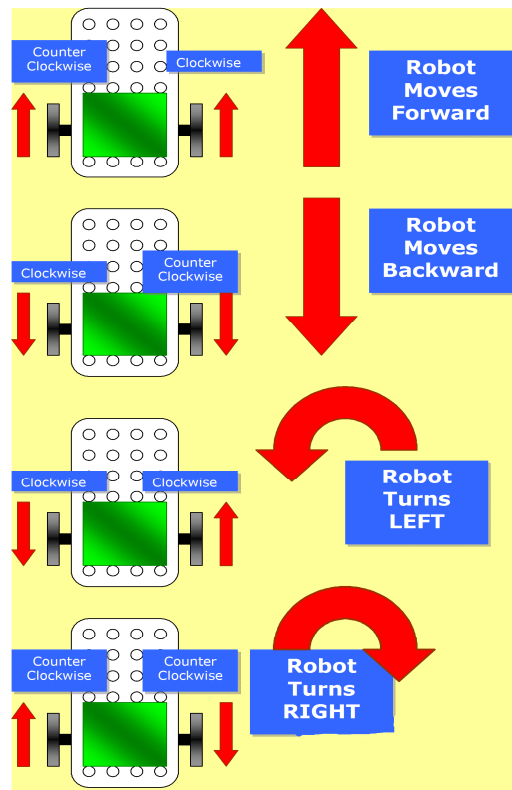


Figure 4.31 Robot Driving System

4.7.5.1 Driving system Coding

```
void moveStop() // Move Stop Function for Motor Driver.
{
  motor1.run(RELEASE);
  motor2.run(RELEASE);
}

void moveForward() // Move Forward Function for Motor Driver.
{
  if(!goesForward)
  {
    goesForward=true;
    motor1.run(FORWARD);
    motor2.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+MAX_SPEED_OFFSET);
    }
    delay(5);
}

void moveBackward() // Move Backward Function for Motor Driver.
{
    goesForward=false;
    motor1.run(BACKWARD);
    motor2.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+MAX_SPEED_OFFSET);
    }
    delay(5);
}

void turnRight() // Turn Right Function for Motor Driver.
{
    motor1.run(FORWARD);
    motor2.run(BACKWARD);
    delay(300);
    motor1.run(FORWARD);
    motor2.run(FORWARD);
}

void turnLeft() // Turn Left Function for Motor Driver.
{
    motor1.run(BACKWARD);
    motor2.run(FORWARD);
}

```

```
delay(300);  
motor1.run(FORWARD);  
motor2.run(FORWARD);  
}
```

5 Chapter five: Conclusion and Recommendation.

5.1 Conclusion

Today we are in the world of robotics. Knowingly or unknowingly, we have been using different types of robots in our daily life.

The project is “obstacle detection and the avoidance robot” is practically proved by using the Ultrasonic sensor for sensing the robot, Motor Shield Driver for the driving the dc motors, dc motor is used for the movement of the robot with the help of the Arduino Microcontroller.

A lot of factors determined the accuracy of the robot we designed. These factors were the environmental phenomenon in which the robot was tested, the number of obstacles present making the test space crowded or relatively less crowded the type and shape of the obstacle (the robot is designed for a uniform shaped obstacle).

These factors majorly affected the sensors. The accuracy of the robot is dependent on the sensors used. Thus, the nature of the sensor and its accuracy defined the accuracy of my robot.

5.2 Future Recommendations

- Adding a Camera: If the current project is interfaced with a camera (e.g. a Webcam) robot can be driven beyond line-of-sight & range becomes practically unlimited as networks have a very large range.
- Use as a fire fighting robot: By adding temperature sensor, water tank and making some changes in programming we can use this robot as firefighting robot.
- We can extend this project with wireless technology by IR (or) RF (or) ZIGBEE.
- We can use the DTMF receiver by using the mobile phone.
- This robot can be used for pick and place the required object by giving directions to the robot but ultrasonic sensor should be replaced depending upon the application.

5.3 Future Works

To enable robots to be able to adapt to its environment is an important domain of robotics research. Whether this environment be underwater, on land, underground, in the air or in

space.

A fully autonomous robot has the ability to

- Work for an extended period of time without intervention from human or a need for power supply.
- Avoid situations that are harmful.

47

-
- Move either all or part of itself throughout its operating environment.

The most effective method to increase the accuracy of my robot is the inclusion of better sensors, although the project cost might increase but the accuracy will definitely increase as well as the problem space where the robot can be used. Better actuators will result in a faster and more efficient robot.

5.4 Advantages

- Collision control.
- It provides Safe Navigation.
- This is the basic of all robot and has a wide scope of extensions

5.5 Limitations

The performance of this robot mainly depends on the sensors and number of sensors.

The ultrasonic sensor used here is of commercial application so it may easily undergo interference.

5.6 Problems Faced

Although the concept and design of the project seemed perfect, there were some problems faced while actual implementation:

The sensor cannot accurately measure distance to an object if:

- Object > 3 meters
- Too shallow of angle
- Object is too small
- object surface is not reflective

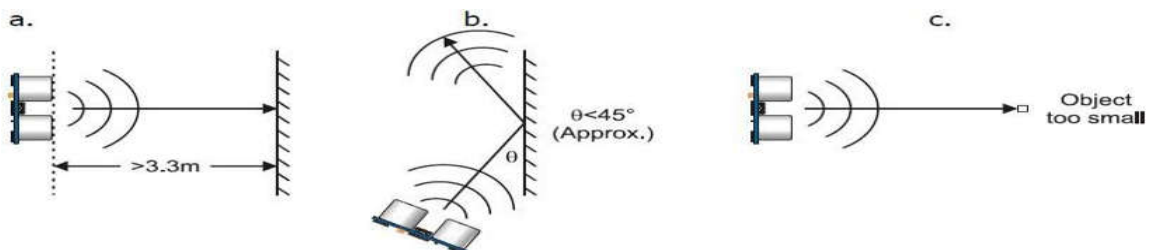


Figure 5.1 Ultrasonic Sensor Problems Faced

REFERENCES

- [1] N. Senthil Kumar, M. Saravanan, S. Jeebanathan, "Microprocessors & Microcontrollers", Oxford University Press, 4th Edition, 2012, ISBN: 978-0-19-806647-7
- [2] *Vision-based obstacles detection for a mobile robot* ElHalawany, B.M.; Abdel-Kader, H.M.; TagEldeen, A.; Ahmed, A.E.S.; Nossair, Z.B. Informatics and Systems (INFOS), 2012 8th International Conference on Year: 2012
- [3] T. Ishida, and Y. Kuroki, "Sensor System of a small Biped Entertainment Robot", Advanced Robotics, © VSP and Robotics Society of Japan, Vol. 18, No. 10, pp 1039-1052, 2004,
- [4] Arduino Step by Step Book
- [5] J. Grefenstette and A. Schultz. "An Evolutionary Approach to Learning in Robots" Machine Learning Workshop on Robot Learning, New Brunswick, 1994. □
- [6] Marija Seder. "Hierarchical Path Planning of Mobile robots in Complex Indoor Environments", June 2011, University of Zagreb.
- [7] B. Ram, "Fundamental of Microprocessors & Microcontrollers", Dhanpat Rai Publication, Seventh Edition, ISBN: 978-81-89928-60-5 □ [3] M. C. Lam, A. S. Prabuwo, H. Arshad, C. S. Chan, "A Real-Time Vision-Based Framework for Human-Robot Interaction", 7066 Lecture Notes in Computer Science (Part 1), 2011, pp. 257-267, IVIC 2011. □
- [8] LINK: <http://science.howstuffworks.com/robot2.htm> □
- [9] LINK: <http://en.wikipedia.org/wiki/Robotics> □
- [10] LINK: http://en.wikipedia.org/wiki/Artificial_intelligence
- [11] D. Floreano and J. Urzelai. "Evolutionary Robots with Online Self-Organization and Behavioral Fitness", June 2000.
- [12] <http://www.alldatasheet.com/>
- [13] <https://www.adafruit.com>
- [14] <https://www.arduino.cc>
- [15] <https://playground.arduino.cc/Code/NewPing>
- [16] <https://www.arduino.cc/en/Reference/Servo>

APPENDIX:

Robot Coding

```
#include <AFMotor.h>
#include <NewPing.h>
#include <Servo.h>
#define TRIG_PIN A4
#define ECHO_PIN A5
#define MAX_DISTANCE 200
#define MAX_SPEED 200 // sets speed of DC motors
#define MAX_SPEED_OFFSET 20
NewPingsonar(TRIG_PIN, ECHO_PIN, MAX_DISTANCE);
AF_DCMotor motor1(1, MOTOR12_1KHZ);
AF_DCMotor motor2(4, MOTOR12_1KHZ);
Servo myservo;
booleangoesForward=false;
int distance = 100;
int speedSet = 0;
void setup() {
myservo.attach(9);
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);
}
```

```

}

void moveForward() {

  if(!goesForward)
  {
    goesForward=true;
    motor1.run(FORWARD);
    motor2.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+MAX_SPEED_OFFSET);
    }
    delay(5);
  }
}

void moveBackward() {
  goesForward=false;
  motor1.run(BACKWARD);
  motor2.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+MAX_SPEED_OFFSET);
  }
  delay(5);
}

void turnRight() {
  motor1.run(FORWARD);
  motor2.run(BACKWARD);
}

```

```
delay(300);
  motor1.run(FORWARD);
motor2.run(FORWARD);
}
void turnLeft() {
  motor1.run(BACKWARD);
motor2.run(FORWARD);
  delay(300);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
}
```