"ALL IN ONE ROBOTIC CAR"

Project submitted to

GOVERNMENT POLYTECHNIC, DEHRI ON

in fulfilment of requirement for the award of degree of

DIPLOMA

Under the

Faculty of Engineering and Technology

In the discipline

Electronics Engineering

By,

PARMANAND KUMAR
(611272118026)
DIPLOMA IN ELECTRONICS COMUNICATION

Guide

Mr. Rohit Sir

Asst. Prof. In Electronics Engineering

Department of Electronics Engineering

Government Polytechnic, Dehri On Sone

GOVERNMENT POLYTECHNIC

DEHRI ON SONE

CERTIFICATE

This is to certify that students mentioned below of Diploma Electronics Engineering, Semester 6th have completed the work for the Project having title **"ALL IN ONE ROBOTIC CAR" :-**

1. **PARMANAND KUMAR - 611272118026**
2. **SANNI KUMAR - 611272118049**
3. **VISHNUKANT SINGH - 611272118027**
4. **RITESH SARMA - 611272118018**

Guide	**External committee**	Head of Department
Mr. Rohit Sir		**Mr. SURAJ SIR**

ACKNOWLEDGEMENT

We consider ourselves to be fortunate to get this opportunity to be part of a Project Report on "**ALL IN ONE ROBOTIC CAR"**. We are sincerely grateful to Asst. **Prof. Mr. Rohit Sir** (Guide) for his invaluable guidance, motivation and support at all stages and creating a flexible and enjoyable environment to work in.

Special thanks to **Prof. Mr. SURAJ KUMAR (**Head of Electronics Engineering) for the support and help us in completing this project successfully. Last but not least; We are thankful to the GOD, my dearly beloved Parents, all Faculty Members, my Friends and all who directly or indirectly supported for completion....

PARMANAND KUMAR

Abstract

A remote-control vehicle is defined as any mobile device that is controlled by a means that does not restrict its motion with an origin external to the device. This is often a radio control device, cable between control and vehicle or an infrared or Bluetooth controller. A remote-control vehicle (Also known as RCV) is always controlled by a human and takes no positive action autonomously. It is vital that a vehicle should be capable of proceeding accurately to a target area; maneuvering within that area to fulfill its mission and returning equally accurately and safely to base.

In this project we are using Bluetooth wireless Technogym to control our robot car which is a very simple communication system. The remote in this project is an android device which has Bluetooth feature built in the user has to install an application on his/her mobile from drive of email address **allinoneroboticcar@gmail.com** **(pass: - AllInOneRoboticCar@123)** and turn on the Bluetooth in the mobile phone.

User can perform various actions like

1) moving forward, Backward, move Left and move right using voice commands, joysticks, switching and gravity command
2) Obstacle avoiding
3) Line following
4) Human following
5) Door monitoring that are sent from the android mobile.

The Bluetooth is a serial communication medium through which we can connect two devices wirelessly. Here we have used a Bluetooth module in our robot car which gets connected to the phone's Bluetooth, that allows us to communicate and allows to take command over it. The task of controlling the car is done by the Arduino UNO which houses the micro-controller ATMEGA32. Arduino has played a major role in the robotic section and has made it easier to convert digital and among signal to physical movements. The project is Bluetooth based because it gives us wider range of control and more efficiency. It also gives us the advantage of changing the remote anytime, meaning that we can use any android devices including phones, tablets, computers. Physical barriers like walls, doors, etc. do not affect in controlling the car

Application can be download from drive

Email address: - **allinoneroboticcar@gmail.com**

pass: - **AllInOneRoboticCar@123**

Bluetooth Technology

Bluetooth wireless Technogym is a short-range radio technology, which is developed for Personal Area Network (PAN). Bluetooth is a standard developed by a group of electronics manufacturers that allows any sort of electronic equipment, from computers and cell phones to keyboards and headphones to make its own connections, without wires, cables or any direct action from a user. It is an ad hoc type network operable over a small area such as a room Bluetooth wireless technology makes it possible to transmit signal over short distances between telephones, computers and other devices and thereby simplify communication and synchronization between devices. It is a global standard that eliminates wires and cables between both stationary and mobile devices and facilitates both data and voice communication Bluetooth offer the possibility of ad hoc networks and delvers the ultimate synchronicity between all your personal devices. Bluetooth is a dynamic standard where devices can automatically find each other, establish connections, and discover what they can do for each other on an ad hoc basis. Bluetooth is intended to be a standard that works at two Level:

1) It provides agreement at the physical level - Bluetooth is a radio-frequency

Standard

2) It also provides agreement at the next level up, where products have to agree on

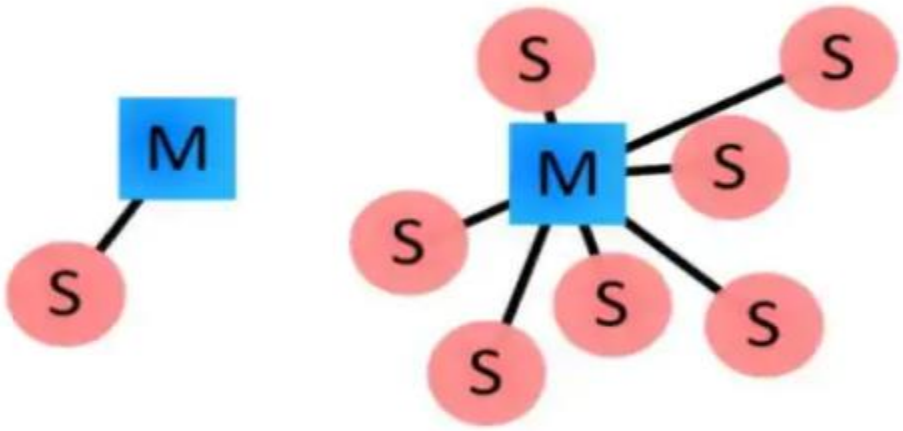
when bis are sent, how many will be sent at a time and how the parties in a conversation can be sure that the message received is the same as the message sent. It is conceived initially by Ericsson, before being adopted by a myriad of other companies, Bluetooth is a standard for a small, cheap radio chip to be plugged into computers, printers, mobile phones, etc. A Bluetooth chip is designed to replace cables by taking the information

normally carried by the cable, and transmitting it at a special frequency to a receiver

Bluetooth chip, which will then give the information received to the computer, phone whatever

Bluetooth topologies:

Bluetooth networks (commonly referred to as Piconets) use a master/slave model to control when and where devices can send data. In this model, a single master device can be connected to up to seven different shave devices as shown in Fig. 1. Any slave device in the Piconet can only be connected to a single master. The master coordinates communication throughout the Piconet. It can send data to any of its slaves and request data from them as well Slaves are only allowed to transmit to and receive from their master. They can't talk to other slaves in the Piconet.



Example of Bluetooth master/slave pin connect topologies

CONTENT

List of figures

CHAPTER 1

INTRODUCTION

Introduction:

Recent advancement in semiconductor material results smart devices such as smart phones and they becomes a basic need in day to day life with massive storage capacities, powerful with reinforced processors, richer entertainment function and vast communicating methodologies.

Robots are electromechanical machine which can be controlled by artificial programming using high speed microcontroller [1]. They found in wide area of application such as industry, manufacturing, production lines, health, etc. The robot is preferred to work in rough industrial environment and designed to reduce human effort, to improve productivity and to reduce overall manufacturing cost.

Bluetooth technology, created by telecom vendor Ericsson in 1994, shows its advantage by integrating with smart phones. Bluetooth raised as one of the popular communications in which, the user can transfer files, commands etc.
Utilizing its flexibility in communication, Bluetooth controlled devices are start occupying its place in the market especially in controlling the robotic vehicle.

Bluetooth controlled robotic vehicle is one solution to design and develop the cheap and rugged robots to perform any task with safe distance operation. Bluetooth communication will enable us to control the robot up to 100 meters without the need for direct sight which means that the robot could be located behind a wall or some other object and the communication would not be lost.

In recent years, an open-source platform called android has been widely used in smart phones which the smart phones have gradually turned into an all-purpose portable device. Android has complete software package consisting of an operating system, middleware layer and core applications [2]. Using a Smartphone as the “brain” of a robot is already an active research field with several open opportunities and promising possibilities. This paper aims to describe the design and development of Bluetooth controlled robotic which operated using android mobile application (All in one robotic car).

1.1 About all in one robotic car

As its name suggest this car is developed to perform multiple operation that mean it can perform following operation: -

1) Obstacle avoiding
2) Human following
3) Line following
4) Car speed detecting monitoring system
5) And simply riding purpose with the help of
    a. Joysticks
    b. Switching
    c. Voice command
    d. Gravity

All the mode of working of All in one robotic car is controlled by the help of an android application whose name is “All in one robotic car”. This application car be download by the allinoneroboticcar@gmail.com and also all the needed software document can be also available here.

CHAPTER 2

Technology
And
Literature survey

Literature Survey

In recent years a great deal of time and effort has been spent of developing systems to enable an autonomous robot to follow a marked path, human and obstacle avoiding, door monitoring system as well as simply riding purpose using a vision system Not surprisingly, the majority of this research has been towards modifying, or designing from a "Hummer Bot" project a full-sized road vehicle so that it can drive on ordinary roads without human supervision. Due to the large amount of space available in an ordinary road vehicle, high performance computers can be used to perform complex image processing and, typically, to maintain a mathematical model of the vehicle and the environment Research into autonomous driving

2.1 Working of All in "One Robotic car": -

All in one robotic car is multifunctional car so it can perform many tasks that is: -

1) Joystick mode
2) Switch mode
3) Gravity mode
4) Voice command mode
5) Obstacle avoiding mode
6) Line following mode
7) Human following mode
8) Door monitoring system

As we know that "all in one robotic car" is remote controlled robot, so before to perform any work we need to connect car to android application from car.

Android application is downloaded from drive of email address given above.

How to connect android application to robot: -

- For connect the application from the car first step is: we need to pair "all in one robotic car" Bluetooth device

  For to pair the device do the following step

  - Step 1) Switch ON the power
  - Step2) swap the upper tap of your smart phone.
  - Step3) long press the Bluetooth icon
  - Step4) Turn on Bluetooth
  - Step 3) Refresh the scanning (for find new Bluetooth device)

- ❖ Step4) In new Rarely used devices we see that “all in one robotic car” click them
- ❖ Step 5) After clicking he say for enter your password, enter the password 1234 and then press OK.

∇ *Now* after pairing the “all in one robotic car” Bluetooth device open application do the following work

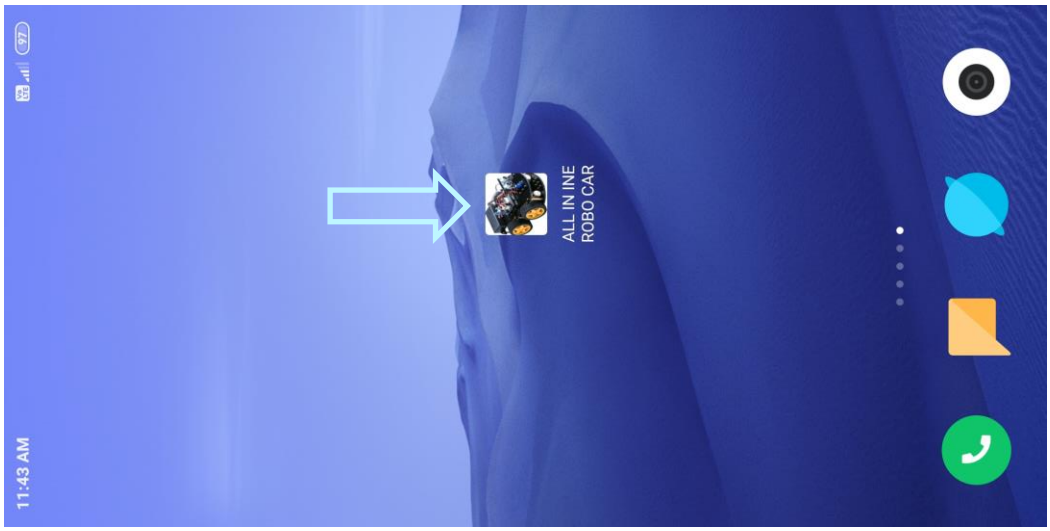


Fig 2.10 All in One robotic Car app

∇ When you click the application, we see that a tab which is introduce my facilities of our college.

Fig 2.11 All facilities of electronics

- Wait for 5 second, after 5 second automatically a anther tab is open that is shows that all group member that is developed this project.

Fig 2.12 All project developer

- Once again wait for 5 second after 5 second automatically a new page is open that is give option to connect the application to car. This tab says the user to give me permission to connect the android application to all in one robotic car. Click the Bluetooth icon which is indicated below by a arrow head.

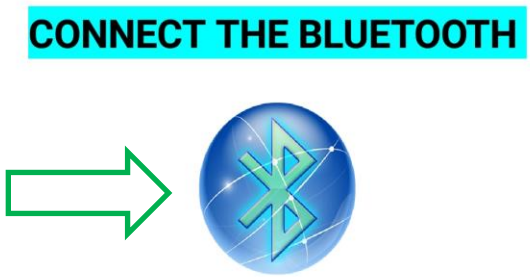


Fig 2.13 Bluetooth connection tab of app

After click the Bluetooth icon it will automatically connect the car to the application. And a new tab called ' all mode selection tap' is open. (note point is also shows that how to connect the application to car.) If any error is coming it can show which type is error is and how to solve them.

- This is scrollable tab that show all the function which all in one robotic car is perform. In this scrollable tap there are four tab selecting tab is present, each tab consists of more than two function that 'all in one car' easily perform. Now time to use to select the mode which they want to do from the car.

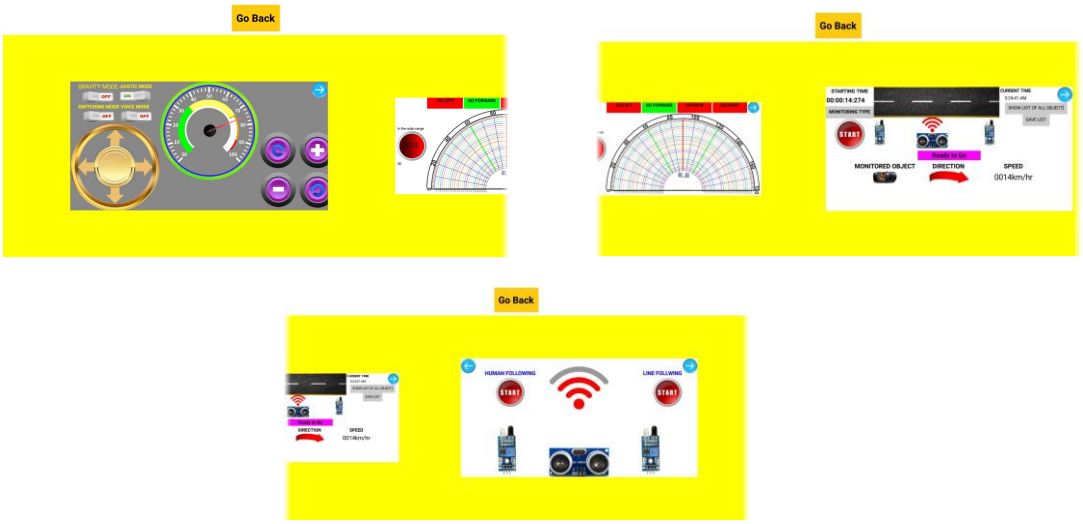


Fig 2.14 All function selection tab

In this tab we see that a back button is also given this button is use if Bluetooth of car is not connected to the android application and we need to go back to Bluetooth tab for reconnection purpose. When Bluetooth is reconnected, it is automatically directed to 'all mode selection' tab.

These are the following four tabs

1) **Drive mode tab**
2) **Obstacle avoiding tab**
3) **Door monitoring tab**
4) **Line and human following tab**

1)Drive mode tab

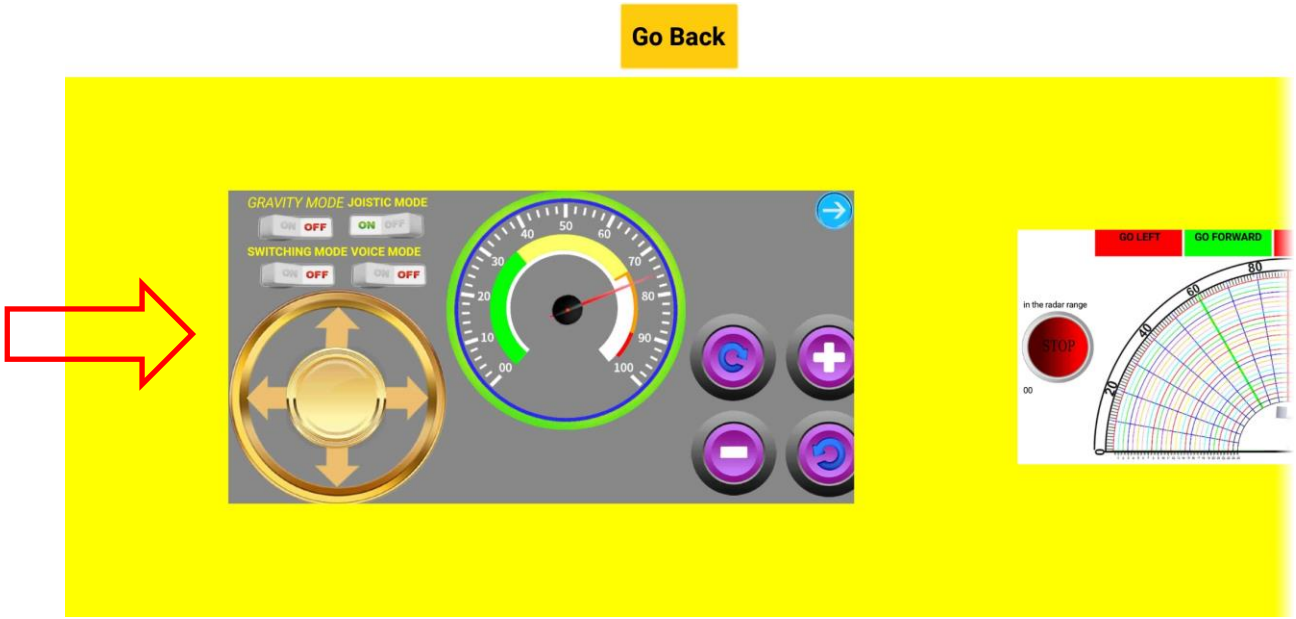


Fig 2.15 All driving mode selection tab

After clicking, a new tab is open, in this tab there are four function include which is robotic car perform one by one.

Fig 2.16 All driving mode tab

➢ Gravity mode

- For to operate gravity mode tab the switch of gravity mode after clicking this mode is activated.

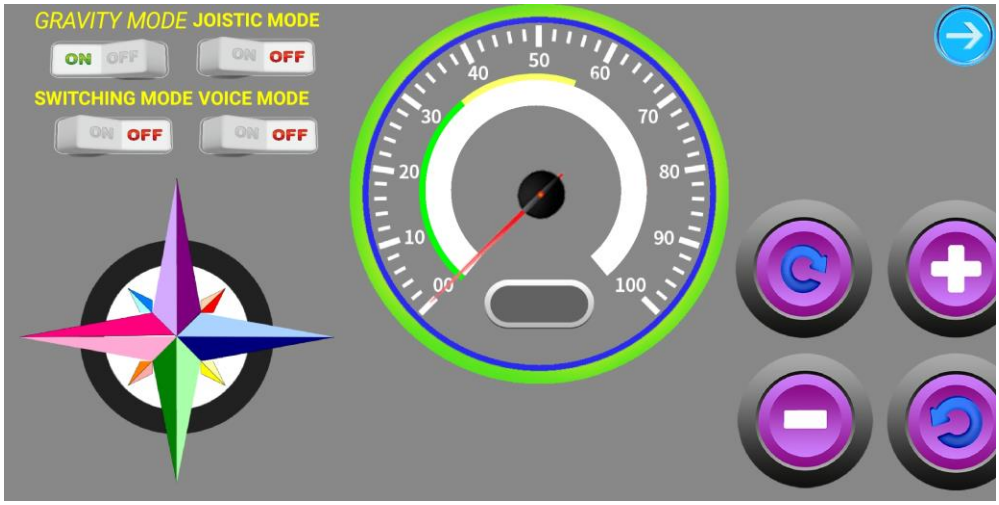


Fig 2.17 Gravity mode tab

- After activate this mode give some speed by the help of plus and minus button.
- Now if we tilt mobile to forward, backward, left and right direction robot car goes in ahead, back, left and right direction with given speed
- Speed meter shows the virtual speed of car

➢ Joystick mode

- For to operate Joystick mode tab the switch of Joystick mode after clicking this mode is activated

Fig 21.8 Joystick mode tab

- In this mode robot go all direction according to movement of Joysticks.
- No need to give any external speed, robot can automatically detect the speed according to joystick's scroll bar.
- Virtual speed is shows on speedometer

➢ Switching mode

- For to operate Switching mode tab the switch of Switching mode after clicking this mode is activated

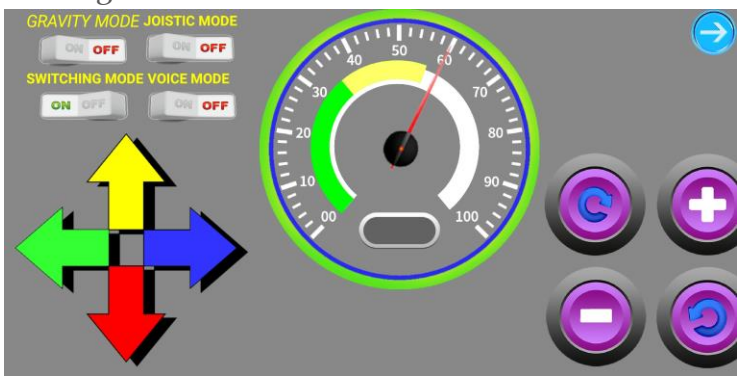


Fig 2.19 Switching mode tab

After giving some speed from speed button if we press up, down, left, right direction button robot car Go ahead, back, left and right direction respectively.

➢ voice command mode

- For to voice command mode tab the switch of voice command mode after clicking this mode is activated

Fig 2.20 voice control mode tab

Give some speed and then Click the mic, after clicking voice recognizer is open and wait for your voice command.

These are following some voice command and their description

All voice controlling command

Voice command	Description
GO FORWARD	Car go ahead
GO BACKWARD	Car go back
TURN LEFT	Car turn left
TUNRN RIGHT	Car turn right
STOP	Car stop their motion
BREAK	Car stop and close the voice recognizer's tab

- If any one press the left and right rotate button car can rotate at given direction at given speed
- In this tab we see that a left arrow head located on the right up side of the screen, their work is to close the present tab.

2) Obstacle avoiding tab

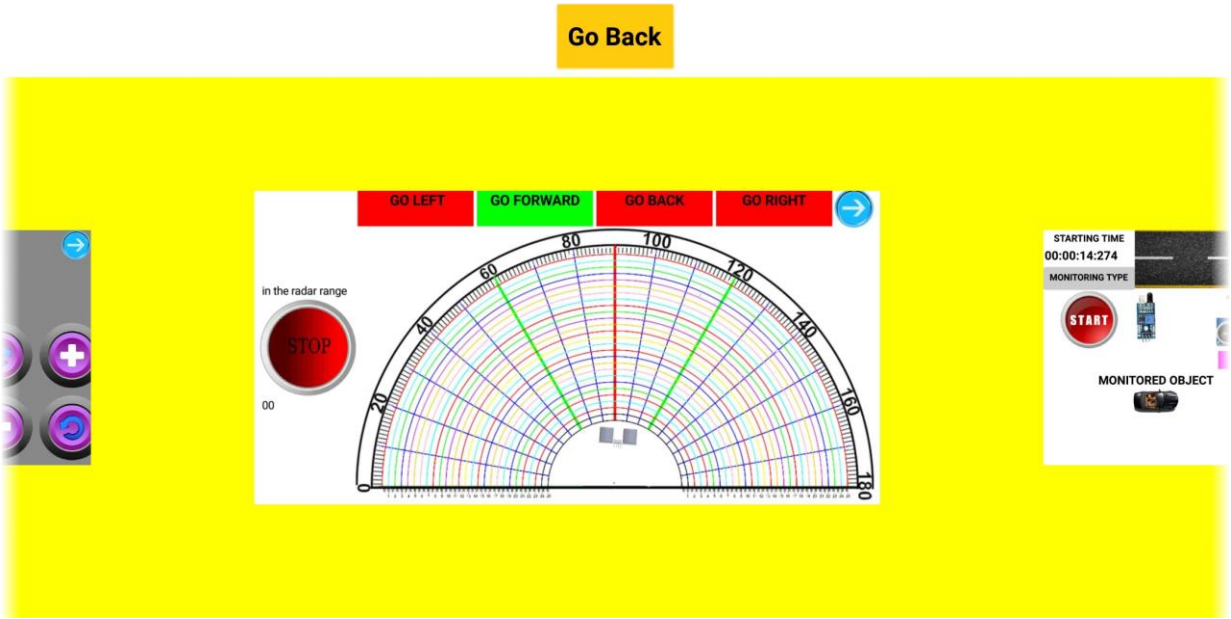


Fig 2.21 Obstacle avoiding selectin tab

After clicking obstacle avoiding tab, a new interface is open, which shows the obstacle of left, right and forward direction in real time.

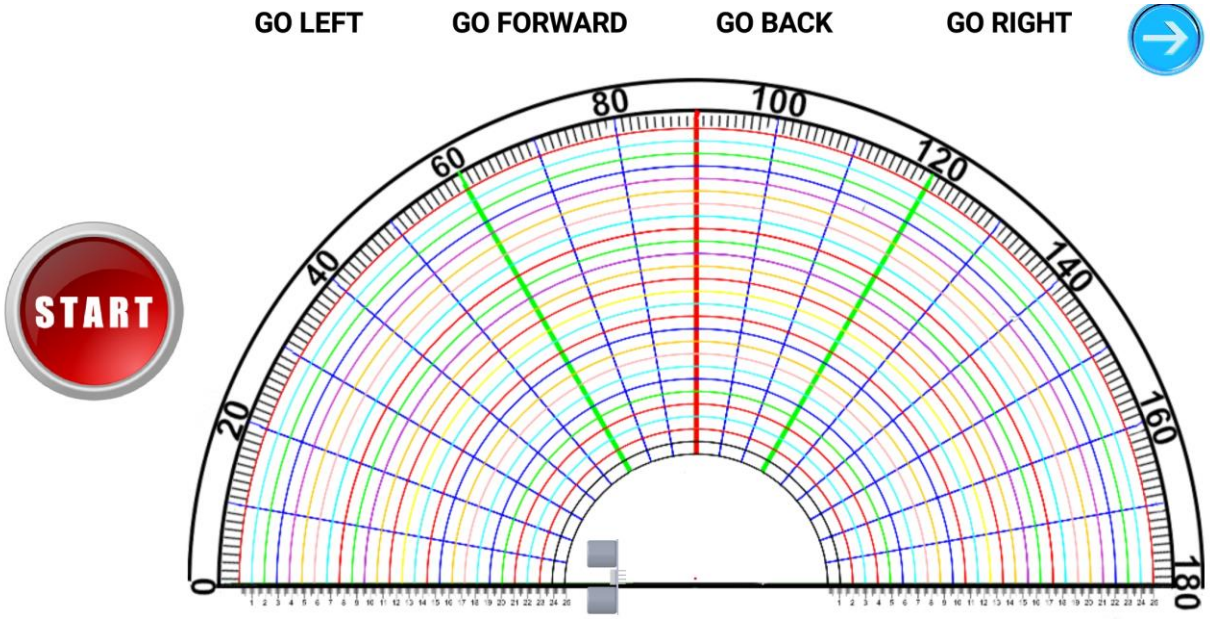
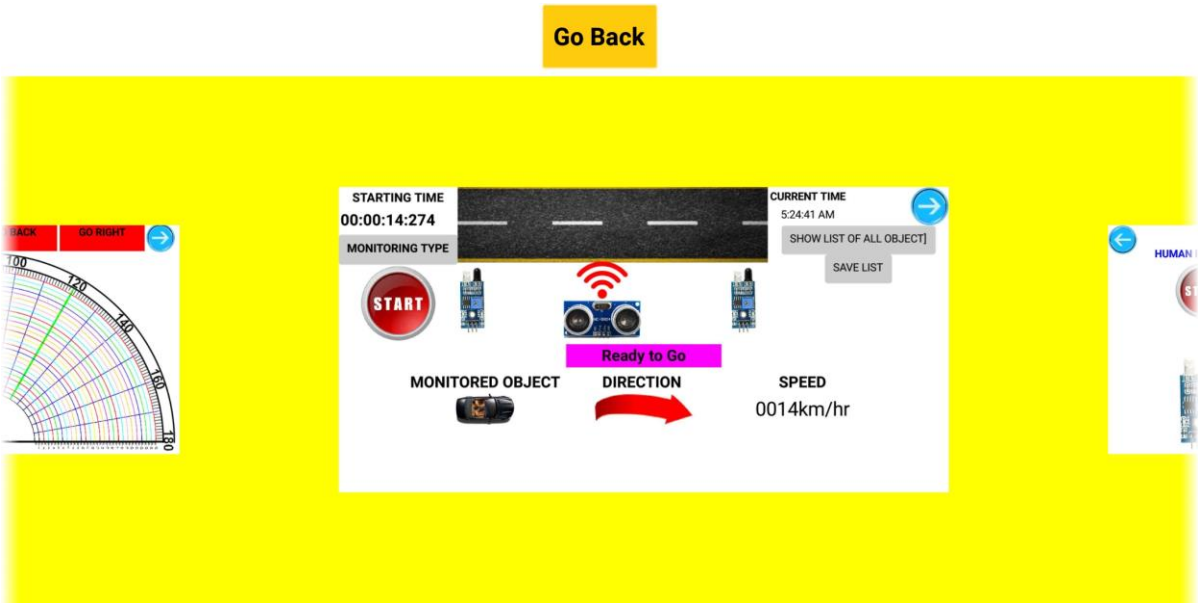


Fig 2.22 obstacle avoiding mode

For to obstacle avoid click on start button, after clicking it show the shows the obstacle of left, right and forward direction, if any obstacle is very close to robot then robot stop and scan three direction.In any of three direction where any obstacle is not found (i.e. where robot can go) robot chose that direction automatically and go that direction.

5) Door monitoring tab



6) Fig 2.23 Door monitoring selection tab

After clicking Door monitoring tab, a new interface is open which is given below: -

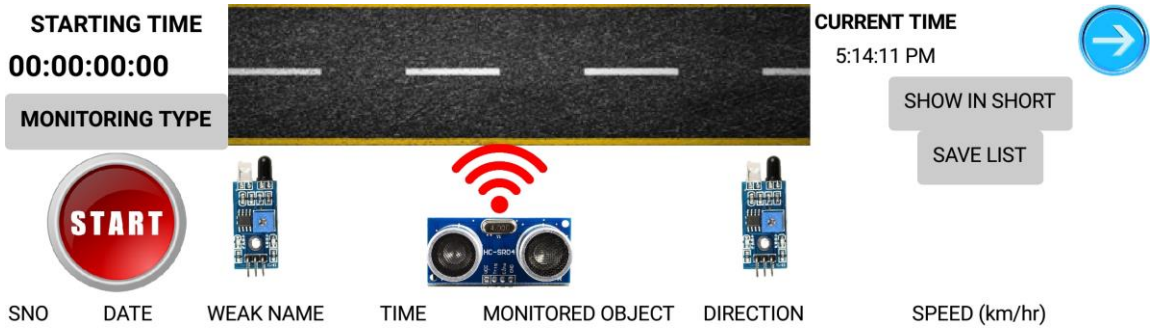


Fig 2.24 Door monitoring tab

After clicking start, it start the monitoring , that mean if any one is pass in the range of robot it can detect their speed and direction and make list of all thing (van, or man) in the real time .

Save List:- this button is use to save the list of all monitored object.

Monitored type: This button is use to says which type of object is monitring.we can select man, car and Bus. Dafaul it is car.

Show in short/ Show in long button:-

This button shows the list in short and with complete dicription

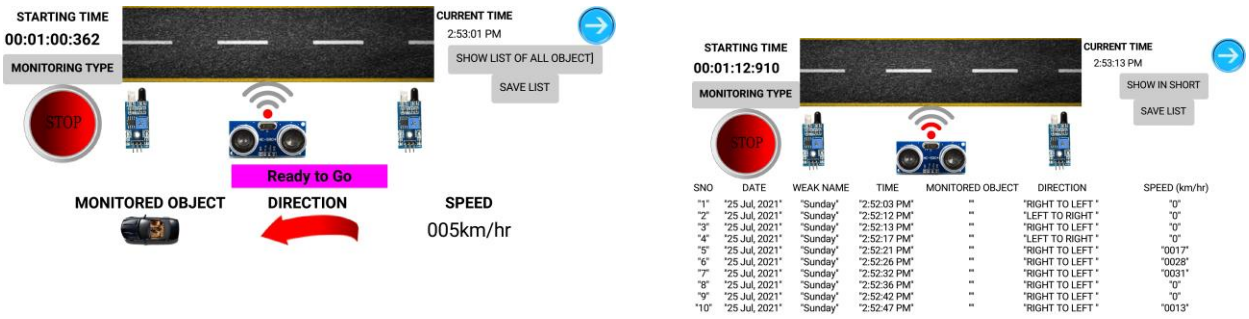


Fig 2.25 Door monitoring list

7) Line and human following tab

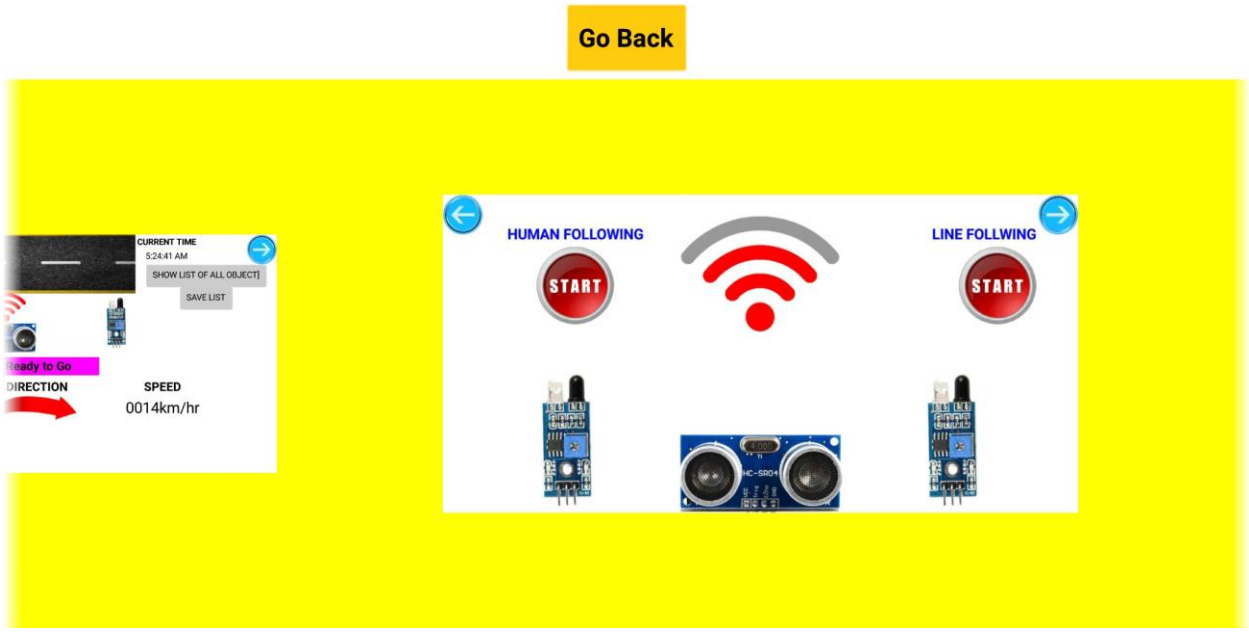


Fig 2.26 Line and human following Selection tab

After clicking Line and human following, a new interface is open which is given below: -

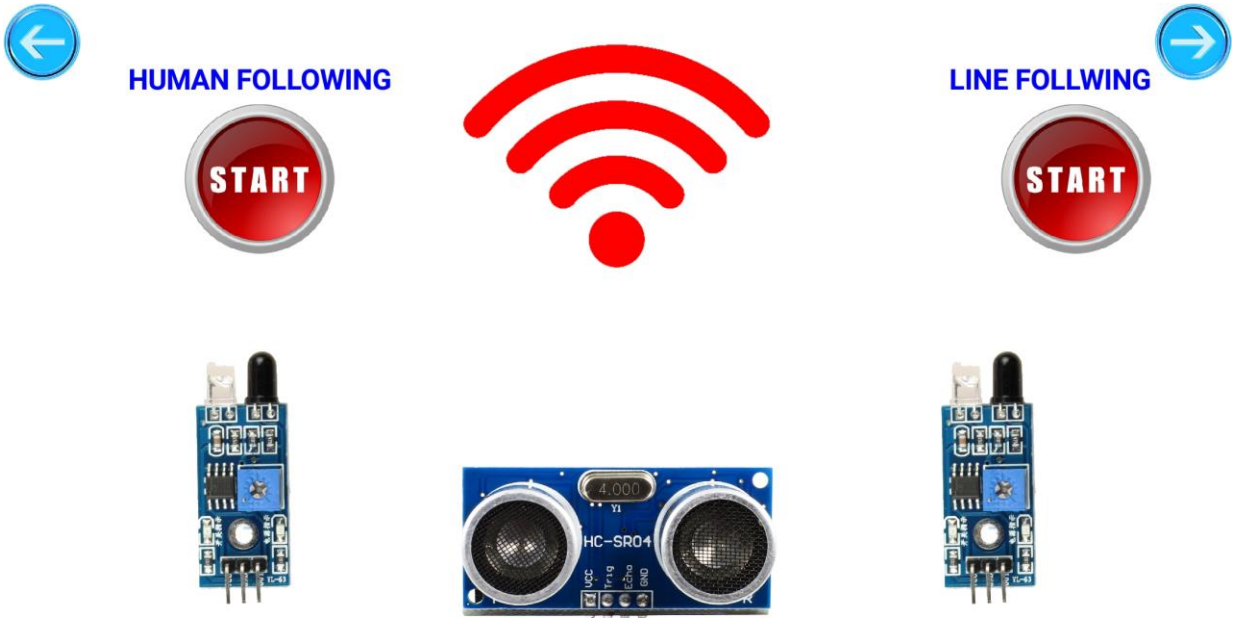


Fig 2.27 Line and human following tab

In this tab ther are two mode of function that is :-

1) Line follower
    a. When we click start button of line follwer it follows the line path which can be made from the black paint and black wiring tape.
2) Humman follower
    a. When we click start button of Human follwer it follows the hunam, but practically it is not possible to shows that is it follwing human due reason that its shape is very small as compare to human.
    b. So for demonstate the mode you use your hand
    c. If take your hand in the range(30cm) it detact your hand and follow them.(i.e. If take the hand in leftword,backword, forward and right direction robot follow that all direction easily.)

2.2 Hardware Required

These are following component and their description which is required to make all in one robotic car.

2.201 Arduino uno

Arduino Uno is a microcontroller board developed by Arduino.cc which is an open-source electronics platform mainly based on AVR microcontroller Atmega328.
The first Arduino project was started in Interaction Design Institute Ivrea in 2003 by David Cuartielles and Massimo Banzi with the intention of providing a cheap and flexible way for students and professionals for controlling a number of devices in the real world. The current version of Arduino Uno comes with a USB interface, 6 analog input pins, 14 I/O digital ports that are used to connect with external electronic circuits. Out of 14 I/O ports, 6 pins can be used for PWM output.
It allows the designers to control and sense the external electronic devices in the real world.

This board comes with all the features required to run the controller and can be directly connected to the computer through a USB cable that is used to transfer the code to the controller using IDE (Integrated Development Environment) software, mainly developed to program

Arduino. IDE is equally compatible with Windows, MAC or Linux Systems,

however, Windows is preferable to use. Programming languages like C and C++ are used in IDE.

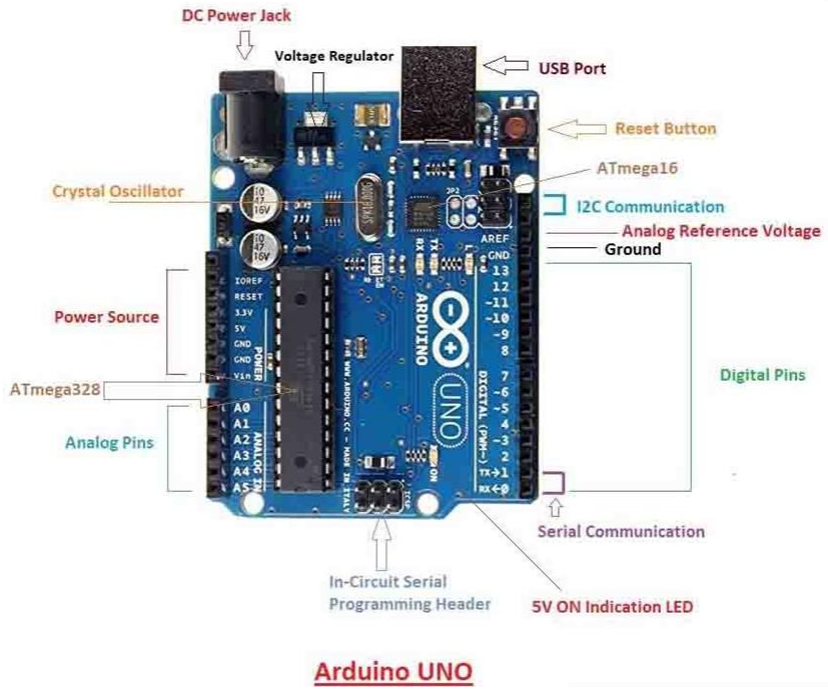


Fig 2.28 Arduino Uno R3

Apart from USB, a battery or AC to DC adopter can also be used to power the board.

Arduino Uno boards are quite similar to other boards in the Arduino family in terms of use and functionality, however, Uno boards don't come with FTDI USB to Serial driver chip.

There are many versions of Uno boards available, however, Arduino Nano V3 and Arduino Uno are the most official versions that come with the Atmega328 8-bit AVR Atmel microcontroller where RAM memory is 32KB.

When the nature and functionality of the task go complex, a Mirco SD card can be added to the boards to make them store more information.

Arduino Uno comes with a USB interface i.e. USB port is added on the board to develop serial communication with the computer.

Atmega328 microcontroller is placed on the board that comes with a number of features like timers, counters, interrupts, PWM, CPU, I/O pins and based on a 16MHz clock that helps in producing more frequency and number of instructions per cycle.

It is an open-source platform where anyone can modify and optimize the board based on the number of instructions and tasks they want to achieve.

This board comes with a built-in regulation feature that keeps the voltage under control when the device is connected to the external device.

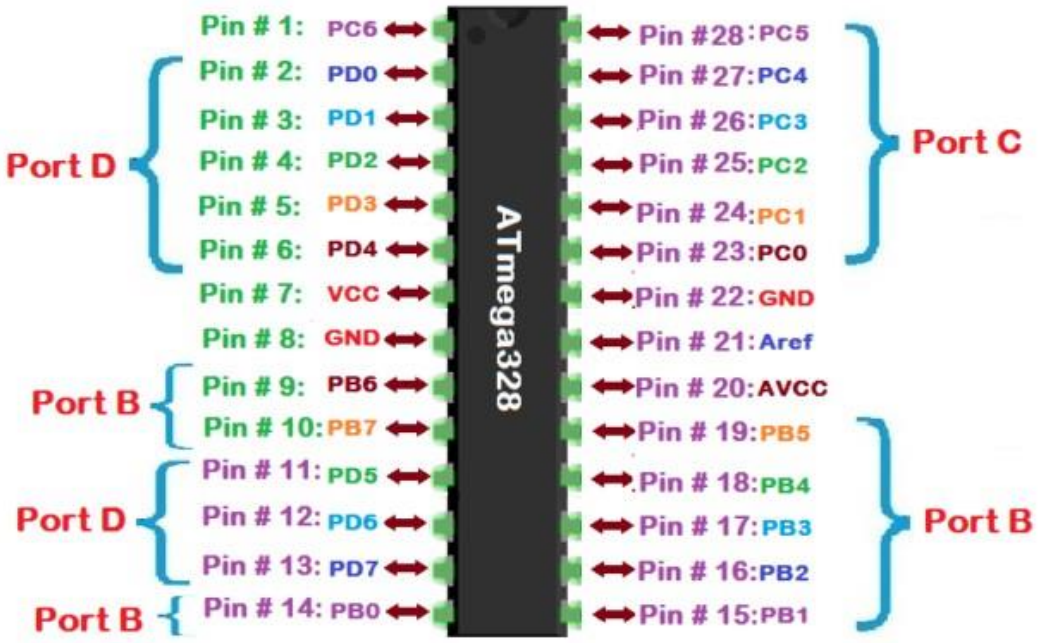


Fig 2.29 AT mega328p microcontroller

A reset pin is present in the board that resets the whole board and takes the running program in the initial stage. This pin is useful when the board hangs up in the middle of the running program; pushing this pin will clear everything up in the program and starts the program right from the beginning.

There are 14 I/O digital and 6 analog pins incorporated in the board that allows the external connection with any circuit with the board. These pins provide flexibility and ease of use to the external devices that can be connected through these pins. There is no hard and fast interface required to connect the devices to the board. Simply plug the external device into the pins of the board that are laid out on the board in the form of the header.

The 6 analog pins are marked as A0 to A5 and come with a resolution of 10bits. These pins measure from 0 to 5V, however, they can be configured to the high range using analogReference() function and AREF pin.

13KB of flash memory is used to store the number of instructions in the form of code.

Only 5 V is required to turn the board on, which can be achieved directly using a USB port or external adopter, however, it can support an external power source up

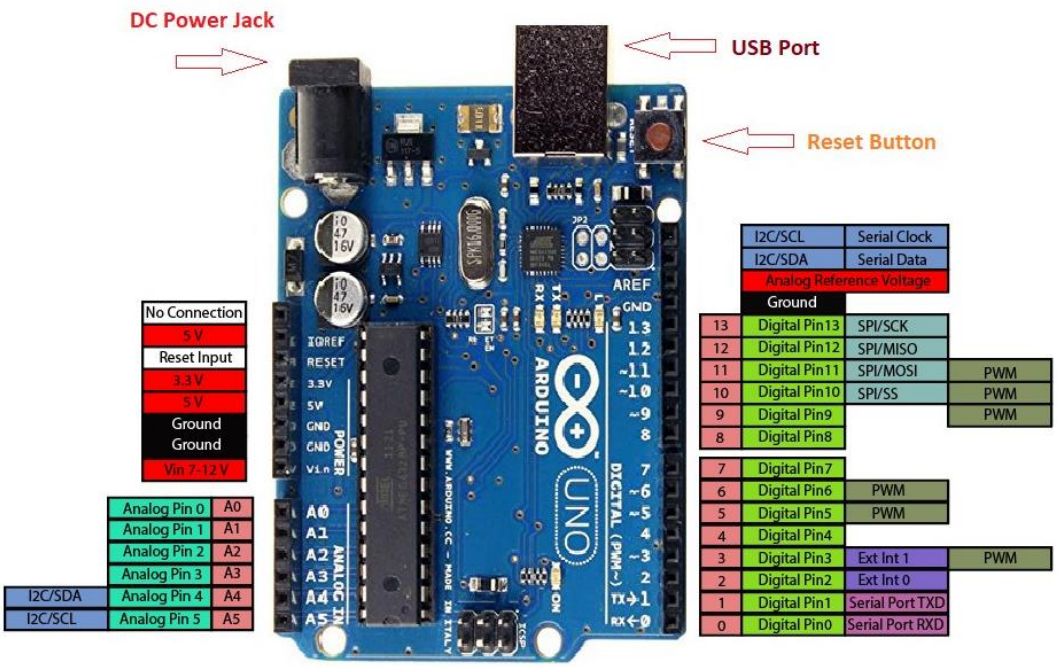


Fig 2.30 Arduino uno pinout

LED. Arduino Uno comes with a built-in LED which is connected through pin 13. Providing HIGH value to the pin will turn it ON and LOW will turn it OFF.

Vin. It is the input voltage provided to the Arduino Board. It is different than 5 V supplied through a USB port. This pin is used to supply voltage. If a voltage is provided through a power jack, it can be accessed through this pin.

5V. This board comes with the ability to provide voltage regulation. 5V pin is used to provide output regulated voltage. The board is powered up using three ways i.e. USB, Vin pin of the board or DC power jack.

USB supports voltage around 5V while Vin and Power Jack support a voltage ranges between 7V to 20V. It is recommended to operate the board on 5V. It is important to note that, if a voltage is supplied through 5V or 3.3V pins, they result in bypassing the voltage regulator that can damage the board if the voltage surpasses its limit**.**

GND. These are ground pins. More than one ground pins are provided on the board which can be used as per requirement.

Reset. This pin is incorporated on the board which resets the program running on the board. Instead of physical reset on the board, IDE comes with a feature of resetting the board through programming.

IOREF. This pin is very useful for providing voltage reference to the board. A shield is used to read the voltage across this pin which then selects the proper power source.

PWM. PWM is provided by 3,5,6,9,10, 11pins. These pins are configured to provided 8-bit output PWM.

SPI. It is known as Serial Peripheral Interface. Four pins 10(SS),

11(MOSI), 12(MISO), 13(SCK) provide SPI communication with the help of the SPI library.
AREF. It is called Analog Reference. This pin is used for providing a reference voltage to the analog inputs.

AREF. It is called Analog Reference. This pin is used for providing a reference voltage to the analog inputs.

TWI. It is called Two-wire Interface. TWI communication is accessed through Wire Library. A4 and A5 pins are used for this purpose.

Serial Communication. Serial communication is carried out through two pins called Pin 0 (Rx) and Pin 1 (Tx).

Rx pin is used to receive data while Tx pin is used to transmit data.

External Interrupts. Pin 2 and 3 are used for providing external interrupts. An interrupt is called by providing LOW or changing value.

2.202 L293D Motor driver

L293D shield is a driver board based on L293 IC, which can drive 4 DC motors and 2 stepper or Servo motors at the same time.

Fig 2.30 Arduino uno pinout

Fig 2.31 L293D motor Driver shield

Each channel of this module has the maximum current of 1.2A and doesn't work if the voltage is more than 25v or less than 4.5v. So be careful with choosing the proper motor according to its nominal voltage and current. For more features of this shield let's mention compatibility with Arduini UNO and MEGA, electromagnetic and thermal protection of motor and disconnecting circuit in case of unconventional voltage raise.

How to Use Arduino L293D Motor Driver Shield?

While using this shield 6 analog Pins (which can be used as digital pins too), pin 2 and pin 13 of arduino are free.
In the case of using Servo motor, pins 9, 10, 2 are in use.
In the case of using DC motor, pin11 for #1, pin3 for #2, pin5 for #3, pin6 for #4 and pins 4, 7, 8 and 12 for all of them are in use.
In the case of using Stepper motor, pins 11 and 3 for #1, pins 5 and 6 for #2 and pins 4, 7, 8 and 12 for all of them are in use.

How to Use Arduino L293D Motor Driver Shield?

While using this shield 6 analog Pins (which can be used as digital pins too), pin 2 and pin 13 of arduino are free.

In the case of using Servo motor, pins 9, 10, 2 are in use.
In the case of using DC motor, pin11 for #1, pin3 for #2, pin5 for #3, pin6 for #4 and pins 4, 7, 8 and 12 for all of them are in use.
In the case of using Stepper motor, pins 11 and 3 for #1, pins 5 and 6 for #2 and pins 4, 7, 8 and 12 for all of them are in use.
You can use free pins by wired connections.

If you are applying separate power supply to Arduino and shield, make sure you have disconnected the jumper on the shield.

2.203 Ultrasonic sensor

At its core, the HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers. The one acts as a transmitter which converts electrical signal into 40 KHz ultrasonic sound pulses. The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse travelled. As simple as pie!

The sensor is small, easy to use in any robotics project and offers excellent non-contact range detection between 2 cm to 400 cm (that's about an inch to 13 feet) with an accuracy of 3mm. Since it operates on 5 volts, it can be hooked directly to an Arduino or any other 5V logic microcontrollers.

Here are complete specifications:

Operating Voltage	DC 5V
Operating Current	15mA
Operating Frequency	40KHz
Max Range	4m
Min Range	2cm
Ranging Accuracy	3mm
Measuring Angle	15 degree
Trigger Input Signal	10µS TTL pulse
Dimension	45 x 20 x 15mm

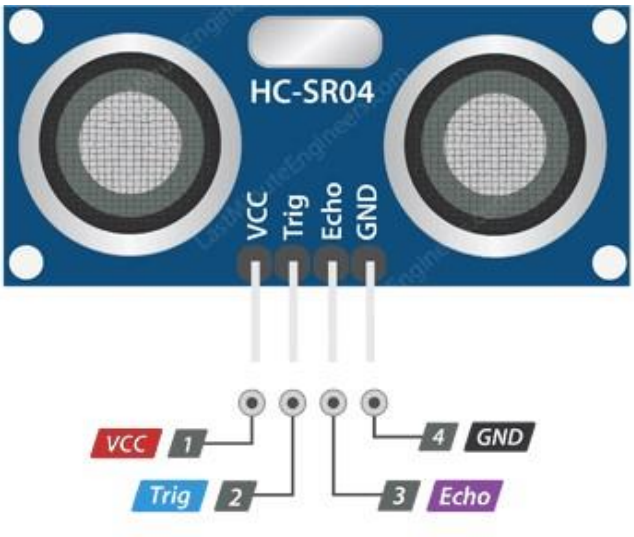


Fig 2.32 ultrasonic sensor

VCC is the power supply for HC-SR04 Ultrasonic distance sensor which we connect the 5V pin on the Arduino.

Trig (Trigger) pin is used to trigger the ultrasonic sound pulses.

Echo pin produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected.

GND should be connected to the ground of Arduino.

How Does HC-SR04 Ultrasonic Distance Sensor Work?

all starts, when a pulse of at least 10 µS (10 microseconds) in duration is applied to the Trigger pin. In response to that the sensor transmits a sonic burst of eight pulses at 40 KHz. This 8-pulse pattern makes the "ultrasonic signature" from the device unique, allowing the receiver to differentiate the transmitted pattern from the ambient ultrasonic noise.

The eight ultrasonic pulses travel through the air away from the transmitter. Meanwhile the Echo pin goes HIGH to start forming the beginning of the echo-back signal.

In case, If those pulses are not reflected back then the Echo signal will timeout after 38 mS (38 milliseconds) and return low. Thus a 38 mS pulse indicates no obstruction within the range of the sensor.

If those pulses are reflected back the Echo pin goes low as soon as the signal is received. This produces a pulse whose width varies between 150 µS to 25 mS, depending upon the time it took for the signal to be received.

The width of the received pulse is then used to calculate the distance to the reflected object. This can be worked out using simple distance-speed-time equation, we learned in High school. In case you forgot, an easy way to remember the distance, speed and time equations

2.204 Infrared sensor

What is an IR Sensor?

IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations.

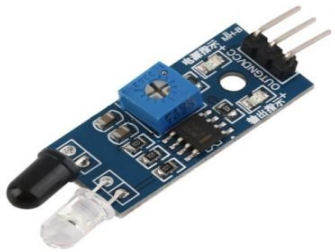


Fig 2.33 Infrared sensor

The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode . Photodiode is sensitive to IR light of the same wavelength which is emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

There are five basic elements used in a typical infrared detection system: an infrared source, a transmission medium, optical component, infrared detectors or receivers and signal processing. Infrared lasers and Infrared LED's of specific wavelength used as infrared sources.

The three main types of media used for infrared transmission are vacuum, atmosphere and optical fibres. Optical components are used to focus the infrared radiation or to limit the spectral response.

Types of IR Sensor

There are two types of IR sensors are available and they are,

a) Active Infrared Sensor

b) Passive Infrared Sensor

Active Infrared Sensor

Active infrared sensors consist of two elements: infrared source and infrared detector. Infrared sources include the LED or infrared laser diode. Infrared detectors include photodiodes or phototransistors. The energy emitted by the infrared source is reflected by an object and falls on the infrared detector.

Passive Infrared Sensor

Passive infrared sensors are basically Infrared detectors. Passive infrared sensors do not use any infrared source and detector. They are of two types: quantum and thermal. Thermal infrared sensors use infrared energy as the source of heat. Thermocouples, pyroelectric detectors and bolometers are the common types of thermal infrared detectors. Quantum type infrared sensors offer higher detection performance. It is faster than thermal type infrared detectors. The photo sensitivity of quantum type detectors is wavelength dependent.

IR Sensor Working Principle

There are different types of infrared transmitters depending on their wavelengths, output power and response time. An IR sensor consists of

an IR LED and an IR Photodiode, together they are called as PhotoCoupler or OptoCoupler.

IR Transmitter or IR LED

Infrared Transmitter is a light emitting diode (LED) which emits in- frared radiations called as IR LED's. Even though an IR LED looks like a normal LED, the radiation emitted by it is invisible to the human eye.

The picture of an Infrared LED is shown below.

Fig 2.34 Infrared transmitter

IR Receiver or Photodiode

Infrared receivers or infrared sensors detect the radiation from an IR transmitter. IR receivers come in the form of photodiodes and phototransistors. Infrared Photodiodes are different from normal photo diodes as they detect only infrared radiation. Below image shows the picture of an IR receiver or a photodiode,

Fig 2.35 Infrared receiver

different types of IR receivers exist based on the wavelength, voltage, package, etc. When used in an infrared transmitter – receiver combination, the wavelength of the receiver should match with that of the transmitter.

The emitter is an IR LED and the detector is an IR photodiode. The IR photodiode is sensitive to the IR light emitted by an IR LED. The photo-diode's resistance

and output voltage change in proportion to the IR light received. This is the underlying working principle of the IR sensor.

When the IR transmitter emits radiation, it reaches the object and some of the radiation reflects back to the IR receiver. Based on the intensity of the reception **by** the IR receiver, the output of the sensor defines.

2.205 Sg90 servo motor

Servo motors are high torque motors which are commonly used in robotics **and** several other applications due to the fact that it's easy to control their rotation. Servo motors have a geared output shaft which can be electrically **controlled** to turn one (1) degree at a time. For the sake of control, unlike normal DC motors, servo motors usually have an additional pin aside the

two power pins (Vcc and GND) which is the signal pin. The signal pin is used to control the servo motor, turning its shaft to any desired angle.

TowerPro SG-90 Features

Operating Voltage is +5V typically
Torque: 2.5kg/cm

Operating speed is 0.1s/60°
Gear Type: Plastic

Rotation : 0°-180°
Weight of motor : 9gm
Package includes gear horns and screws

Wire Configuration

Wire Number	Wire Colour	Description
1	Brown	Ground wire connected to the ground of system
2	Red	Powers the motor typically +5V is used
3	Orange	PWM signal is given in through this wire to drive the motor

Fig 2.36 Sg90 servo Motor and their wire configuration

2.206 Dc gear Bo motor

Fig 2.37 Dc Gear Bo Motor

2.207 Car wheal

Fig 2.38 car Wheel

2.208 Hc 05 Bluetooth module

HC-05 Pinout Configuration

Pin Number	Pin Name	Description
1	Enable / Key	This pin is used to toggle between Data Mode (set low) and AT command mode (set high). By default it is in Data mode
2	Vcc	Powers the module. Connect to +5V Supply voltage
3	Ground	Ground pin of module, connect to system ground.
4	TX – Transmitter	Transmits Serial Data. Everything received via Bluetooth will be given out by this pin as serial data.

5	RX – Receiver	Receive Serial Data. Every serial data given to this pin will be broadcasted via Bluetooth
6	State	The state pin is connected to on board LED, it can be used as a feedback to check if Bluetooth is working properly.
7	LED	Indicates the status of Module • Blink once in 2 sec: Module has entered Command Mode • Repeated Blinking: Waiting for connection in Data Mode • Blink twice in 1 sec: Connection successful in Data Mode
8	Button	Used to control the Key/Enable pin to toggle between Data and command Mode

HC-05 Default Settings

Default Bluetooth Name: "HC-05"

Default Password: 1234 or 0000

Default Communication: Slave

Default Mode: Data Mode

Data Mode Baud Rate: 9600, 8, N, 1

Command Mode Baud Rate: 38400, 8, N, 1

Default firmware: LINVOR

HC-05 Technical Specifications

- Serial Bluetooth module for Arduino and other microcontrollers
- Operating Voltage: 4V to 6V (Typically +5V)
- Operating Current: 30mA
- Range: <100m
- Works with Serial communication (USART) and TTL compatible
- Follows IEEE 802.15.1 standardized protocol
- Uses Frequency-Hopping Spread spectrum (FHSS)
- Can operate in Master, Slave or Master/Slave mode

- Can be easily interfaced with Laptop or Mobile phones with Bluetooth
- Supported baud rate: 9600,19200,38400,57600,115200,230400,460800.

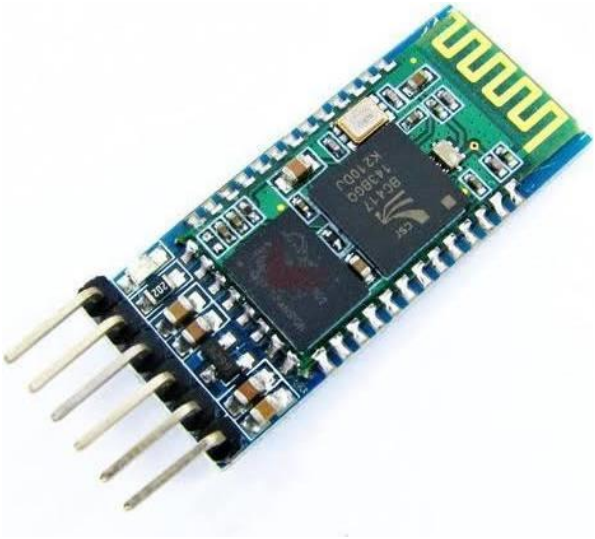


Fig 2.39 Hc05 Bluetooth Module

2.210 Dual Battery holder

Fig 2.40 Battery Holder

2.211 Dual state on off switch

Fig 2.41 Dual state On Off swithch

2.212 Double layer tape

Fig 2.42 Double layer tape

2.213 Male to female connection wire

Fig 2.43 Male to Female connection Wire

2.214 Soldering iron

Fig 2.44 Soldering iron with soldering wire and soldering paste

2. 3 Software Required

2.31 Arduino ide

The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

It is also download form drive of email address allinoneroboticcar@gmail.com

Fig 2.45 Arduino ide interface

2.32 MIT app inventor

MIT App Inventor is an intuitive, visual programming environment that allows everyone even children to build fully functional apps for smartphones and tablets. Those new to MIT App Inventor can have a simple first app up and running in less than 30 minutes. And what's more, our blocks-based tool facilitates the creation of complex, high-impact apps in significantly less time than traditional programming environments. The MIT App Inventor project seeks to democratize software development by empowering all people, especially young people, to move from technology consumption to technology creation.

Web address if MIT app inventor: - **http://ai2.appinventor.mit.edu/**

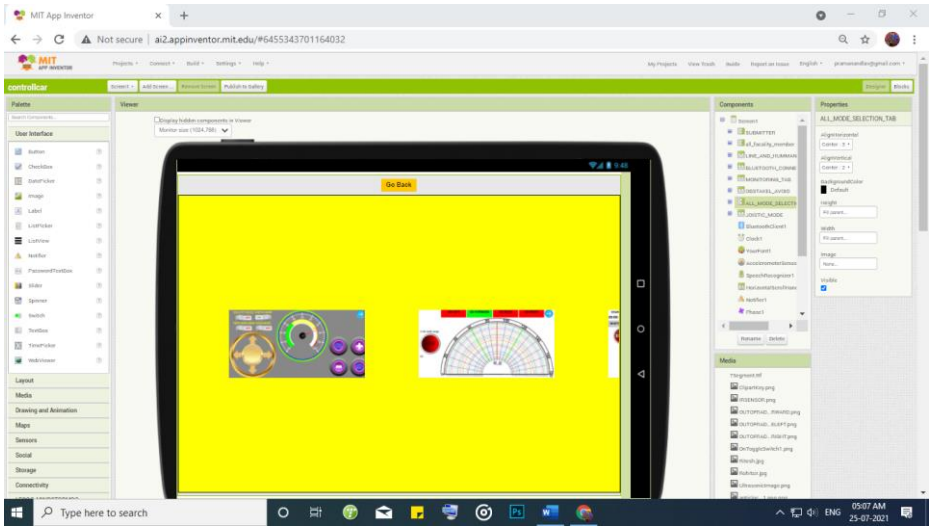


Fig 2.46 MIT app inverter interface

CHAPTER 3

HARDWARE DESIGN AND CONNECTION

3.1 Attachment of motor and wheal

As shown in figure attach the wheel from wheel carefully a slot is given at the saft of motor.

Fig 3.10 attachment of motor and wheel

3.2 Attachment of Arduino uno from l293d motor driver

There is most advantage of l293D motor driver shield is there is no need of external connecting wire to connect together.

There are a lot of inbuilt pin is consisting of shield which make it unique, it is directly paste on Arduino uno.

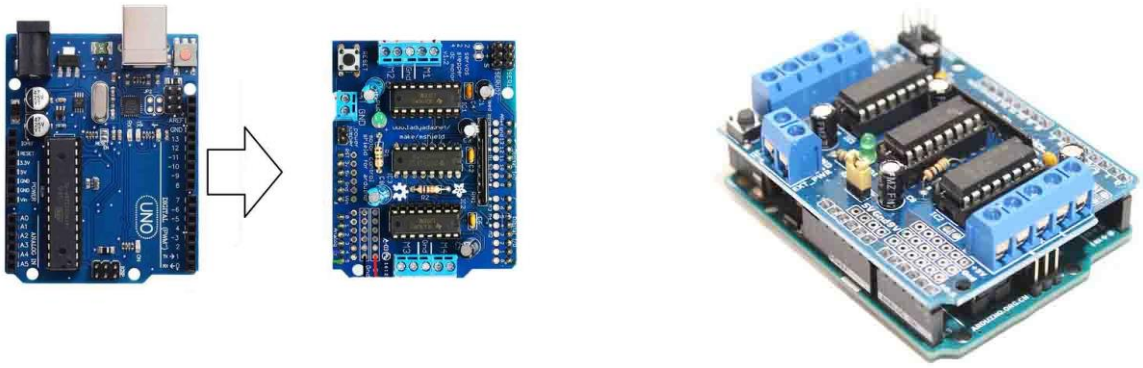


Fig 3.11 attachment of shield with Arduino uno

3.3 Attachment of ultrasonic sensor with servo motor

Adjust the ultrasonic sensor just above the head of servo motor, so that it can move left and right direction. Fig shows how to connect the ultrasonic sensor with servo motor

Fig 3.12 Attachment of ultrasonic sensor with servo motor

3.4 Connection of motor through motor driver

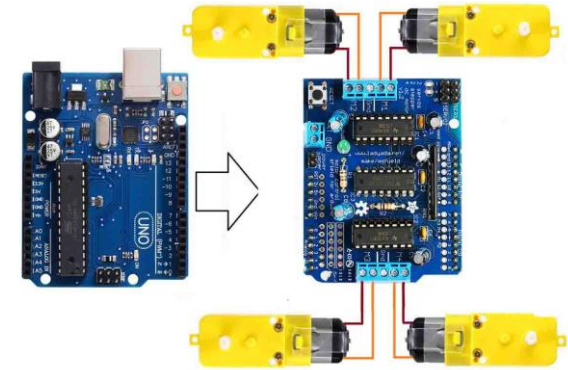


Fig 3.13 Connection of motor through motor driver

3.5 Connection of Bluetooth module

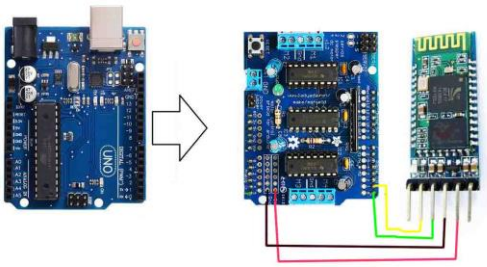


Fig 3.14 Connection of Bluetooth module

Wire connection of Bluetooth

L293D MOTOR DRIVER SHIELD PIN NO	BLUETOOTH PIN NO
0(RX)	TX
1(TX)	RX

3.6 Connection of ultrasonic sensor

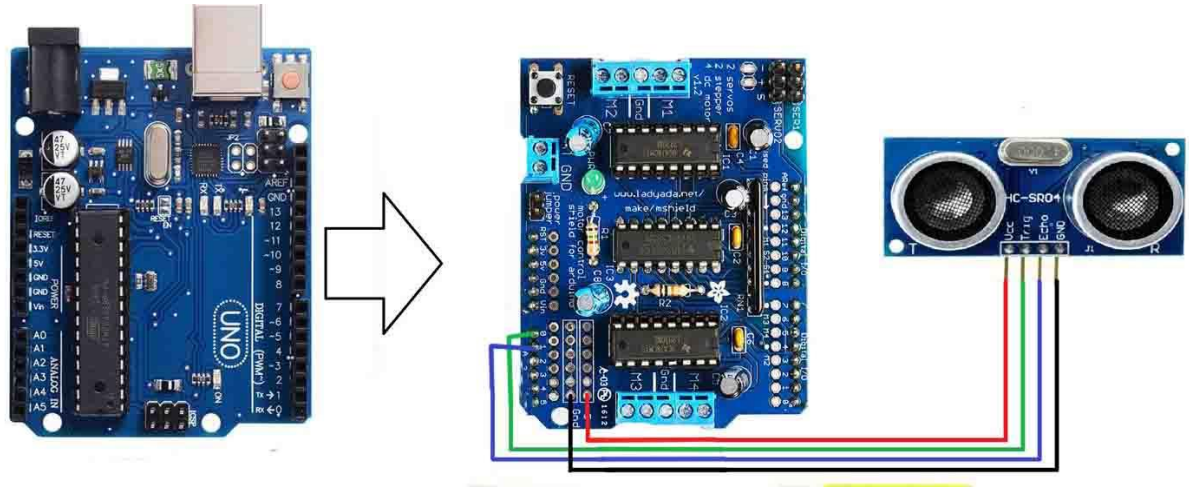


Fig 3.15 Connection of ultrasonic sensor

Connection of ultrasonic sensor

L293D MOTOR DRIVER SHIELD PIN NO	HC05 ULTRASONIC SENSOR PIN NO
A0	TRIGGER
A1	ECHO

3.7 Connection of infrared sensor

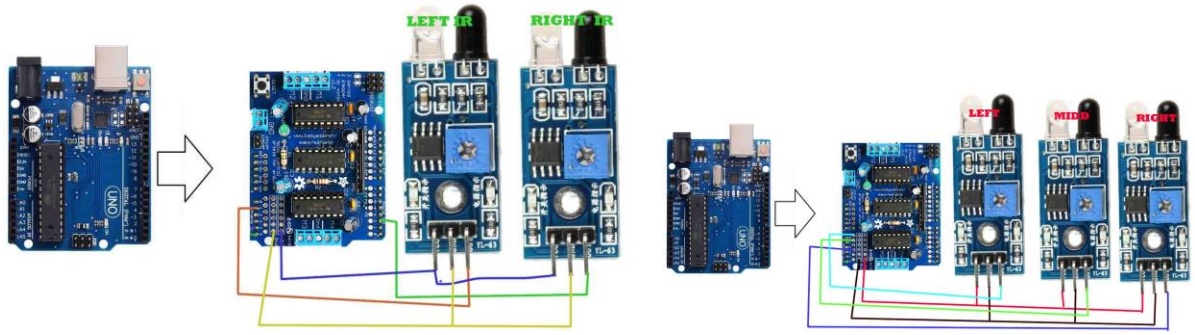


Fig 3.16 Connection of infrared sensor

Connection of upper Ir sensor

L293D MOTOR DRIVER SHIELD PIN NO	UPER IR SENSOR
A2	LEFT IR SESOR PIN
2	RIGHT IR SENSOR OUT PIN

Connection of bottom Ir sensor

L293D MOTOR DRIVER SHIELD PIN NO	UPER IR SENSOR
A3	LEFT IR SESOR PIN
A4	MIDDLE IR SENSOR OUT PIN
A5	RIGHT IR SENSOR OUT PIN

3.8 Connection of servo motor

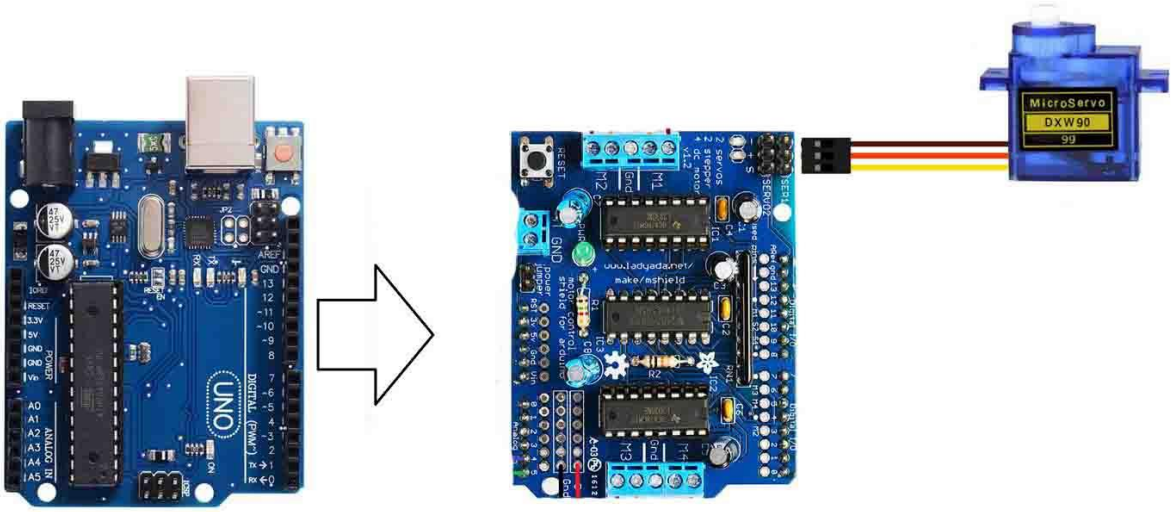


Fig 3.17 Connection of servo motor

9 Connection of power supply

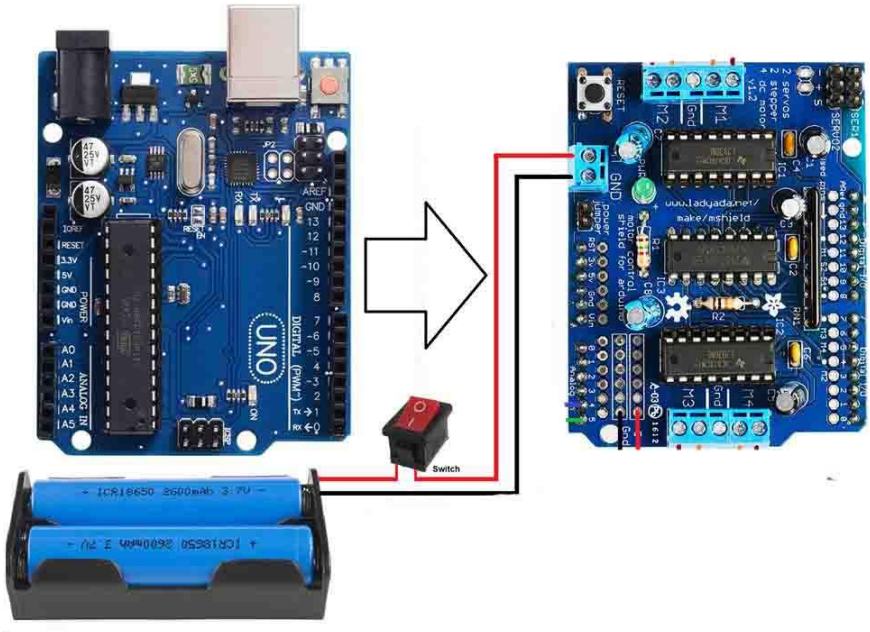


Fig 3.18 Connection of power supply

CHAPTER 4

Program Design

And

Implementation

4.1 Designing of application for android devices

All in one robotic car application is developed by a open source web app called Mit app inventor.

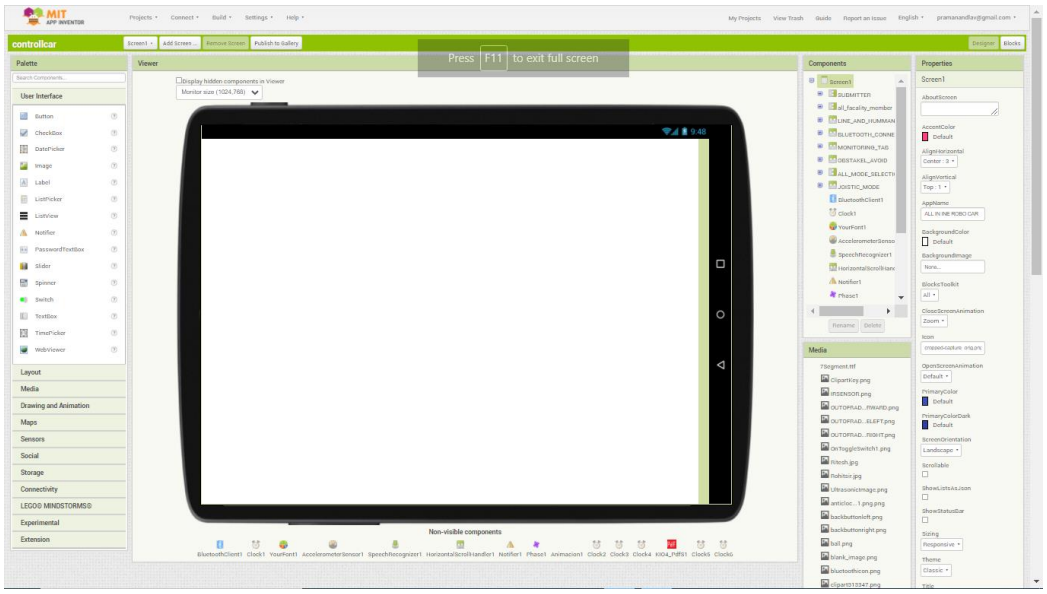


Fig 4.10 Graphical user interface area of MIT app Inventor

By the help of this web all the screen graphics and background block programming is possible.

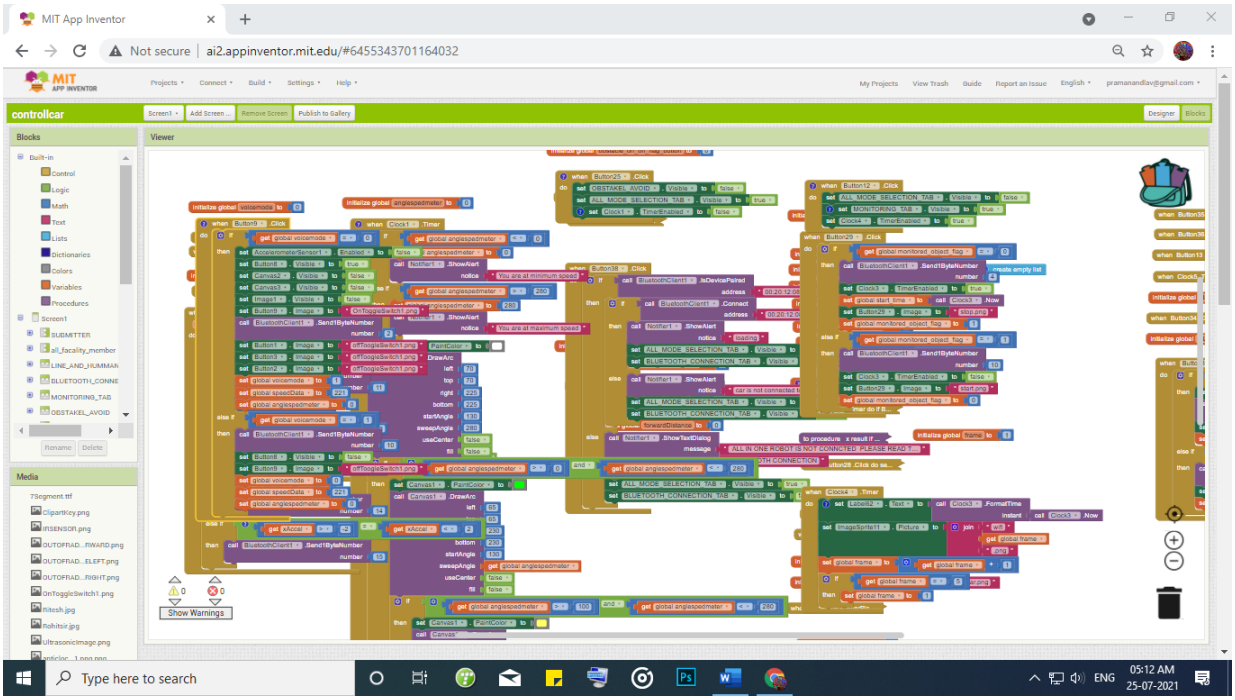
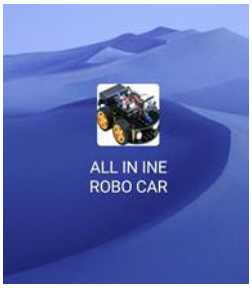


Fig 4.11 java block programming area of MIT app Inventor

There are a lot of program logic, external extension, internal extension is used to developed this application. Some of them are given above in very briefly.

Now finally after programming and compiling the Java code we get my android application that look like this: -



4.2 writing of c++ language to Arduino uno to control all different function

The following c++ language is written for Arduino for controlling all the function and hardware equipment. This is very precious for this project.

```
#include <Servo.h>


#include "AFMotor.h"


#include <NewPing.h>

//-----------------------------------------------------------

//       define upper ir sensor for huamn follower and  monitoring data

//-----------------------------------------------------------

   #define  RIGHT 2    // connect a ir sensor of right uper side of car

   #define  LEFT A2    // connect a ir sensor of left upper side  of car

// ----------------------------------------------------------- //  for monitoring data

//-----------------------------------------------------------

      int flag1 = 0;
```

```
int flag2 = 0;
int timer1;
int timer2;
int Speedm;
float Time;
 int distancem=5 ;



//-------------------------------------------------------------
//           down ir sensor for line follower
//-------------------------------------------------------------
  #define Dleft A5   // connect a ir sensor of left downward side  of car
  #define Dmiddle A4  // connect a ir sensor of left downward  side  of car
  #define Dright  A3  // connect a ir sensor of left downward  side  of car
//-------------------------------------------------------------
 //           ultrasonic sensor data explain
 //------------------------------------------ -------------
   #define TRIGGER_PIN  A0  // connect the ultrasonic sensor's trigger pin
   #define ECHO_PIN     A1 // connect the ultrasonic sensor's echo  pin
  int distance_L, distance_F, distance_R;
   unsigned  distance; // this is use to normally read distance
 NewPing sonar(TRIGGER_PIN, ECHO_PIN);
//-------------------------------------------------------------
//           for control servo motor data explain
//-------------------------------------------------------------
    Servo myservo;
```

```
      int pos =0;
//-------------------------------------------------------------
//       for l293d motor driver controll
//-------------------------------------------------------------
     const int MOTOR_1 = 1;
     const int MOTOR_2 = 2;
     const int MOTOR_3 = 3;
     const int MOTOR_4 = 4;
     int LSpeed = 0; // controll m1 and m2 speed of motor which is left side
     int RSpeed = 0; // controll m2 and m3 speed of motor which is right side
     AF_DCMotor motor1(MOTOR_1, MOTOR12_64KHZ); // create motor object, 64KHz pwm
     AF_DCMotor motor2(MOTOR_2, MOTOR12_64KHZ); // create motor object, 64KHz pwm
     AF_DCMotor motor3(MOTOR_3, MOTOR12_64KHZ); // create motor object, 64KHz pwm
     AF_DCMotor motor4(MOTOR_4, MOTOR12_64KHZ); // create motor object, 64KHz pwm

//-------------------------------------------------------------
//    for incoming all different data from android
//-------------------------------------------------------------
      int xAxis =140, yAxis=140;


      int function;


void setup() {
```

```
// intial possition of servo motor is set
      myservo.attach(10); // servo motor attach at pin 10
      for(pos = 90; pos <= 180; pos += 1){
     myservo.write(pos);
     delay(15);
     }

     for(pos = 180; pos >= 0; pos-= 1) {
     myservo.write(pos);
     delay(15);
     }

     for(pos = 0; pos<=90; pos += 1) {
     myservo.write(pos);
     delay(15);
      }

// for reading value from ultrasonic sensor we have to explain pin mode
          pinMode (TRIGGER_PIN, OUTPUT);
          pinMode (ECHO_PIN, INPUT );
// define pine mode of upper ir sensor

          pinMode(RIGHT, INPUT);
          pinMode(LEFT, INPUT);
// define for line follower ir sensor
```

```
        pinMode(Dleft, INPUT);pinMode(Dmiddle, INPUT);
        pinMode(Dright , INPUT);



 // inital speed of all dc motor is set 0
        motor1.setSpeed(LSpeed);   // set the motor speed to 0-255
        motor2.setSpeed(LSpeed);
        motor3.setSpeed(RSpeed);
        motor4.setSpeed(RSpeed);
// serial monitor is initalize
        Serial.begin(9600);
        delay(500);
} // void setup end colon



void loop(){

//if some date is sent, reads it and saves in state
        while(Serial.available() > 0){
          xAxis= Serial.read(); // use primary for controll all data
          delay(10);
          yAxis= Serial.read(); // use for controll joystic
        }delay(10);


     //Receive data for mode selection that is in the range of 1 to 10
        if(xAxis <= 10){mode = xAxis;  }
```

```
    // Receive the speed that is send by controller in the range of 201 to 256
        else if(xAxis>220){int sp = xAxis; LSpeed = RSpeed = map(sp,221,255,130,255);



       motor1.setSpeed(LSpeed);   // set the motor speed to 0-255
       motor2.setSpeed(LSpeed);
       motor3.setSpeed(RSpeed);
       motor4.setSpeed(RSpeed);
      }

    //Receive function of different controlling method
        else if(xAxis>10 && xAxis<60 ){ function = xAxis;}  ;

 // select different mode and working operation

 if (mode==1){

//====================================
//                          joistic mode
//====================================

      if (xAxis > 130 && xAxis <150 && yAxis > 130 && yAxis <150){Stop();}



  if (yAxis > 130 && yAxis <150){
```

```
        if (xAxis < 130){turnLeft();
        LSpeed = map(xAxis, 130, 60, 130, 255);
        RSpeed= map(xAxis, 130, 60, 130, 255);
        }

        if (xAxis > 150) {turnRight();
        LSpeed = map(xAxis, 150, 220, 130, 255);
        RSpeed = map(xAxis, 150, 220, 130, 255);
        }

        }
else if (xAxis > 130 && xAxis <150){

            if (yAxis < 130){forword();
           LSpeed = map(yAxis, 130, 60, 130, 255);
           RSpeed = map(yAxis, 130, 60, 130, 255);
            }
            if (yAxis > 150){backword();
            LSpeed = map(yAxis, 150, 220, 130, 255);
           RSpeed = map(yAxis, 150, 220, 130, 255);
            }
            }

else{
```

```
      if (yAxis < 130){forword();}
      if (yAxis > 150){backword();}

      if (xAxis < 130){
      LSpeed = map(xAxis, 130, 60, 255, 130);
      RSpeed = 255;
       }

      if (xAxis > 150){
      LSpeed = 255;
     RSpeed = map(xAxis, 150, 220, 255, 130);
       }
       }

    motor1.setSpeed(LSpeed);
    motor2.setSpeed(LSpeed);
    motor3.setSpeed(RSpeed);
    motor4.setSpeed(RSpeed);

Serial.print(LSpeed);Serial.println(RSpeed);
 }

else if ( mode==2){
```

```
//=====================================
//   switch Control//accelerometer control// voice control//four button Command
//=====================================
    if(function == 11){forword(); }  // if the function is '11' the DC motor will go forward
else if(function == 12){backword();}  // if the function is '12' the motor will Reverse
else if(function == 13){turnLeft();}  // if the function is '13' the motor will turn left
else if(function == 14){turnRight();} // if the function is '14' the motor will turn right
else if(function == 15){Stop(); }     // if the function is '15' the motor will Stop
////////////////////////////////////END\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

//=====================================
//              Remain two  Voice Control Command
//=====================================
else if(function == 16){turnLeft();  delay(400);  function = 15;}
else if(function == 17){turnRight(); delay(400);  function = 15;}
}

/////////END\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
//-------------------------------------------------------------
//                          obstakel avoiding robot
//-------------------------------------------------------------
 else if (mode==3){
delay(10);
distance_F = sonar.ping_cm(); // read distance from ultra sonic sensor
delay(10);
 if (distance_F > 35){ Serial.println("210"); Serial.print(":");Serial.println(distance_F); LSpeed =
RSpeed=150;forword();  delay(200);}       // turn the motor forward with 180 speed
```

```
// if not then check condition
 if (distance_F >= 2  && distance_F <= 35  ){ different_direction_distance();}

    motor1.setSpeed(LSpeed);
    motor2.setSpeed(LSpeed);
    motor3.setSpeed(RSpeed);
    motor4.setSpeed(RSpeed);


}
 ////////////////////////end//////////////////////
//-------------------------------------------------------
//                              monitored object
//-------------------------------------------------------
else if(mode==4){


if(digitalRead (LEFT) == LOW && digitalRead (RIGHT) == HIGH &&
flag1==0){Serial.print("207");Serial.print(":"); Serial.print("0") ;timer1 = millis(); flag1=1;}

if(digitalRead (RIGHT) == LOW && digitalRead (LEFT) == HIGH &&
flag2==0){Serial.print("208");Serial.print(":"); Serial.print("0"); timer2 = millis();flag2=1;}

if (flag1==1 && flag2==1){
   if(timer1 > timer2){Time = timer1 - timer2;}
```

```
 if(timer2 > timer1){Time = timer2 - timer1;}

 Time=Time/1000;//convert millisecond to second

 Speedm=(distancem/Time);//v=d/t

 Speedm=Speedm*3600;//multiply by seconds per hr

 Speedm=Speedm/1000;//division by meters per Km

 if (Speedm<50) {Serial.print("0");Serial.print(Speedm); Serial.print(":"); }

delay(1000);

Serial.print("209");Serial.print(":");Serial.print("0") ;

flag2=0;

flag1=0;

}

}

//-------------------------------------------------------------

//                                human follower

//-------------------------------------------------------------

 // for resettion purpose all the previous condition make sure zero

else if(mode==5){

// speed of both motor zero

 myservo.write(90); // set the ulrasonic sensor in forward direction

int Right_Value = digitalRead(RIGHT);

int Left_Value  = digitalRead(LEFT);

distance = sonar.ping_cm();

   if((Right_Value==1) && (distance>=10 && distance<=30)&&(Left_Value==1)){LSpeed = RSpeed=200;
forword();}

else if((Right_Value==0) && (Left_Value==1)){ LSpeed = RSpeed=230; turnRight();}

else if((Right_Value==1) && (Left_Value==0)){LSpeed = RSpeed=230; turnLeft();}

else if((Right_Value==1) && (Left_Value==1)){ LSpeed = RSpeed=0;Stop();}
```

```
else if(distance > 5 && distance < 10){LSpeed = RSpeed=0; Stop();}

else if(distance < 5){LSpeed = RSpeed=230; backword();}



    motor1.setSpeed(LSpeed);

    motor2.setSpeed(LSpeed);

    motor3.setSpeed(RSpeed);

    motor4.setSpeed(RSpeed);

    }

//-------------------------------------------------------------

//                        line follower

//-------------------------------------------------------------


else if(mode==6){


if ( (digitalRead(Dleft) == HIGH)&&(digitalRead(Dmiddle) == HIGH)&&(digitalRead(Dright) ==
HIGH)){LSpeed = RSpeed=0 ; Stop();}  // no path is found

else if ((digitalRead(Dleft) ==LOW)&&(digitalRead(Dmiddle) == LOW)&&(digitalRead(Dright)
==LOW)){LSpeed = RSpeed=0 ; Stop(); }   //path finish



 else if ((digitalRead(Dleft) == LOW)&&(digitalRead(Dmiddle) == HIGH)&&(digitalRead(Dright) ==
LOW)){LSpeed = RSpeed=150 ; forword();}  // fo go ahead


 else if ((digitalRead(Dleft) == HIGH)&&(digitalRead(Dmiddle) == LOW)&&(digitalRead(Dright) ==
LOW)){LSpeed = RSpeed=150 ; turnLeft();}  //sharp left

else if ((digitalRead(Dleft) == HIGH)&&(digitalRead(Dmiddle) ==HIGH)&&(digitalRead(Dright) == HIGH))
{LSpeed = RSpeed=150 ; turnLeft();}  // soft left
```

```
else if ((digitalRead(Dleft) ==LOW)&&(digitalRead(Dmiddle) == LOW)&&(digitalRead(Dright) == HIGH)){LSpeed = RSpeed=150 ; turnRight();} // sharp right

else if ((digitalRead(Dleft) == LOW)&&(digitalRead(Dmiddle) ==HIGH)&&(digitalRead(Dright) == HIGH)){LSpeed = RSpeed=150 ; turnRight();} //soft right




    motor1.setSpeed(LSpeed);

    motor2.setSpeed(LSpeed);

    motor3.setSpeed(RSpeed);

    motor4.setSpeed(RSpeed);


}

 //-------------------------------------------------------

 //                        default mode reset mode

 //-------------------------------------------------------


else if(mode==10){

 myservo.write(90);

  Stop();

 xAxis=125;yAxis=125;

 LSpeed = RSpeed=0;

 motor1.setSpeed(LSpeed);

    motor2.setSpeed(LSpeed);

    motor3.setSpeed(RSpeed);

    motor4.setSpeed(RSpeed);
```

```
 Stop() ;
}
 // this syntax is switch case

} // this syntax of void loop
 void different_direction_distance(){

   LSpeed = RSpeed=0;
   motor1.setSpeed(LSpeed);
     motor2.setSpeed(LSpeed);
     motor3.setSpeed(RSpeed);
     motor4.setSpeed(RSpeed);
  Stop();
  // stop the motor and check distance

   myservo.write(0);
   delay(300);
   distance_L = sonar.ping_cm(); Serial.print("211"); Serial.print(":");Serial.print( distance_L);

   myservo.write(170);
   delay(500);
   distance_R = sonar.ping_cm();Serial.print("212"); Serial.print(":");Serial.print( distance_R);



   myservo.write(90);
   delay(300);
```

```
  compareDistance();


}


void compareDistance(){
 if (distance_L > distance_R){turnRight();
 // turn it on going left
LSpeed = RSpeed=255;
   motor1.setSpeed(LSpeed);
     motor2.setSpeed(LSpeed);
     motor3.setSpeed(RSpeed);
     motor4.setSpeed(RSpeed);
 delay(500);
 }
 else if (distance_R > distance_L){turnLeft();
  // the other right
 LSpeed = RSpeed=255;
  motor1.setSpeed(LSpeed);
     motor2.setSpeed(LSpeed);
     motor3.setSpeed(RSpeed);
     motor4.setSpeed(RSpeed);
 delay(500);
 }
 else{
 backword();    // the other way
 LSpeed = RSpeed=250;
```

```
   motor1.setSpeed(LSpeed);
     motor2.setSpeed(LSpeed);
     motor3.setSpeed(RSpeed);
     motor4.setSpeed(RSpeed);
  delay(300);
  // after getting practical change this case
 turnLeft(); // turn it on going left
LSpeed = RSpeed=250;
   motor1.setSpeed(LSpeed);
     motor2.setSpeed(LSpeed);
     motor3.setSpeed(RSpeed);
     motor4.setSpeed(RSpeed);
  delay(500);
 }
}
void forword(){
motor1.run(FORWARD); // turn it on going forward
motor2.run(FORWARD);
motor3.run(FORWARD);
motor4.run(FORWARD);
}
void backword(){
motor1.run(BACKWARD); // the other way
motor2.run(BACKWARD);
motor3.run(BACKWARD);
motor4.run(BACKWARD);
```

```
}
void turnRight(){
motor1.run(FORWARD); // the other right
motor2.run(FORWARD);
motor3.run(BACKWARD);
motor4.run(BACKWARD);
}
void turnLeft(){
motor1.run(BACKWARD); // turn it on going left
motor2.run(BACKWARD);
motor3.run(FORWARD);
motor4.run(FORWARD);
}

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

CHAPTER 5

Advantages

And

disadvantage

ADVANTAGES: -

Robot cars can eliminate distracted drivers, you can text, sleep, watch a movie, play video games while driving, they offer better accessibility, the blind people can now drive, there are no more parking issues, there will be no bad drivers and there will be less mistakes on the roads.

The robot cars are programmed not to break the laws, If they are safer than the human drivers, they may be less dangerous to the bikes on the road, There may be an opportunity to create wider bicycle lanes and better infrastructure.

The computer is an ideal motorist, most of car crashes are the result of human error, the computers use the complicated algorithms to determine the appropriate stopping distance which reduces the chances of car accidents, the human driven cars come at a very high cost in terms of danger.

There is a cost savings associated with the time, When the computer takes over the driving responsibilities, the drivers can use that time to do the other things such as catch

DISADVANTAGE: -

Cyber security effects on robot cars, the car's computer can be compromised, as could the communication system between the cars, the security behind self-driving cars would be the major obstacle, especially the hacker's effect on the vehicle's software and control its operation.

Robots cars cause loss of driving-related jobs, they cause the criminal and terrorist activities, they can be loaded with the explosives and they used to drive the bomb to its destination or they can be used in the other criminality acts such as the getaway vehicles.

The ethical problems arise when the robot car is in the unavoidable crash and it must choose between multiple harmful actions that could lead to the death, the self-driving car doesn't eliminate the likelihood of the car accident and there's no legal precedent for how the case would be handled.

CHAPTER 6

Application

Application

The main application of this robot is that

where we need to make following system base device: -

1)HUMAN FOLLOWING SYSTEM

2)LINE FOLLWING SYSYTEM

3)OBSTASTAKEL AVOID SYSYTEM

4)VOICE CONTROL SYSYTEM

5)DOOR MONITORING SYSTEM

CHAPTER 7

Future Enhancements

Future Enhancements

We completed our project successfully with the available sources. But the results and modifications are not up to the expectations. This can be further improved by incorporating the following modifications to obtain better results.

The mechanism which we used i.e. scotch yoke mechanism does not give expected efficiency. This efficiency can be increased by using some other mechanism and speed of motor is reduced because we have used heavy material and this material can be replaced by using light weight material and design of blades should be done based on types of grass used to cut.

The project which we have done surely reaches the average families because the grass can be trimmed with minimum cost and with minimum time Finally this project may give an inspiration to the people who can modify and obtain better results

CHAPTER 8

CONCLUSION

Conclusion

The Arduino is an open source device that has been the brain for numerous projects. The Arduino has everything that is required by the user which includes its inbuilt converters, i/o pins etc, This project provides a design method of an ALL IN ONE ROBOTIC CAR, whose task is to perform line following, human following, voice control driving, door monitoring system and simple riding purpose with the help of android application “All in one robotic car” by using joystick, switch, and gravity sensor. This task is expected to be made possible by using sensors to provide an Arduino with controlling. The obstacle is automatically avoided, here for obstacle avoidance the ultrasonic sensor is used.

CHAPTER 9

Bibliography

REPORT ON ALL IN ONE ROBOTIC CONTROL CAR