**Software Requirement**

**Specification For**

**Android Controlling Car**

**Version 1.0**

**Prepared By**

**Team Triple Zero**

**Department of Software Engineering**

**Daffodil International University**

****

**<18.04.2017>**

Copyright @2017 By Sakib, Shamol, Parvez

**Team Name**: **Team Triple Zero**.

**Project Name**: **Android Controlling Car.**

**Members of the Project**:

1. **Md. Shamol Ali (162-35-1636)**
2. **Md. Sakib Al Islam (162-35-1680)**
3. **Md. Fokhrul Islam Parvez (162-35-1700)**

**Supervisors of the Project**

**Md. Shohel Arman**

**Lecturer,**

**Department of Software Engineering**

**Daffodil International University**

**TABLE OF CONTENTS**

1. Project Overview……………………………………………………………………………………………………………..…04
2. Obstruction………………………………………………………………………………………………………………………..05
3. Introduction……………………………………………………………………………………………………………………….05
4. Technology Requirement………………………………………………………………………………..………………...06
   1. Hardware………………………………………………………………………………………………………..……….….06
      1. DC Motor…………………………………………………………………………………………………………...06
      2. HC Serial Bluetooth…………………………………………………………………………………………….06
      3. Microcontroller……………………………………………………………………………………………..……07
      4. HC-SR04 Ultrasonic Sensor………………………………………………………………………………….08
      5. Bread Board………………………………………………………………………………………………….….…09
      6. L293D Motor Driver IC………………………………………………………………………………….…....10
      7. Car Body……………………………………………………………………………………………………….….…13
      8. Dc Motor……………………………………………………………………………………………………….…...14
      9. Jumper…………………………………………………………………………………………………………….....14
      10. USB cable…………………………………………………………………………………………………………….15
      11. Battery………………………………………………………………………………………………………………..16
   2. Software Programming Platform………………………………………………………………………………..…16
5. Milestones and Reporting……………………………………………………………………………………………….…..17
   1. Analysis…………………………………………………………………………………………………………..……………..17
   2. Development…………………………………………………………………………………………………….…………..17
   3. Testing………………………………………………………………………………………………………………..………...17
   4. Deployment………………………………………………………………………………………………………..………...17
6. Support……………………………………………………………………………………………………………………….…….…17
7. Training………………………………………………………………………………………………………………………….….…18
8. Requirement Elicitation…………………………………………………………………………………………………..…..18
   1. Task of Developer……………………………………………………………………………………………………….…18
   2. Task of User………………………………………………………………………………………………………..…………18
9. Use case Diagram…………………………………………………………………………………………….....................19
10. Use Case Description…………………………………………………………………………………………………..…….…20
11. Advantage of Application…………………………………………………………………………………………….…….…26
    1. Advantages…………………………………………………………………………………………………….….…26
    2. Drawbacks of Existing System ………………………………………………………………………….….26
12. Conclusion………………………………………………………………………………………………………………..………...26
13. Contract Us…………………………………………………………………………………………………………………..….….27

**ACKNOWLEDGEMENT**

We take this occasion to thank Allah, almighty for blessing us with his grace and taking our endeavor to a successful culmination. We extend our sincere and heartfelt thanks to our course teacher, **Md. Shohel Arman** , for providing us with the right guidance and advice at the crucial junctures and for showing me the right way. We also take this opportunity to express a deep sense of gratitude to our respected ma’am, **Mrs. Afsana Begum** and **Mrs. Lamisha Rawshan** for their cordial support, valuable suggestions and guidance. We extend our sincere thanks to our respected Head of deferment, **Dr. Touhid Bhuiyan**, for allowing us to use the facilities available. We would like to thank the other faculty members also, at this occasion. Last but not the least, we would like to thank our friends and family for the support and encouragement they have given us during the course of our work.

1. **Project Overview**

Robotics is the branch of technology that deals with the design, construction, operation, and application of robots. A machine capable of carrying out a complex series of actions automatically, esp. one programmable by a computers is defined as a robot. The project is to develop a robot that will move according to the code assigned but find a free space, navigating from any obstacle on its way. This kind of obstacle is very useful in industries where automatic supervision is needed, for example, in places where it might be risky for humans to be. This robot can also be made by putting other sensors like light sensors or line sensors, ultrasonic sensors and ultrasound sensor depending on the need.

1. **Obstruction**

Abstract - Now day’s many industries are using robots due to their high level of performance and reliability and which is a great help for human beings. The obstacle avoidance robotics is used for detecting obstacle and avoiding the collision. This is an autonomous robot. The design of obstacle avoidance robot requires the integration of many sensors according to their task. The obstacle detection is primary requirement of this autonomous robot. The robot gets the information from surrounding area through mounted sensors on the robot. Some sensing devices used for obstacle detection like bump sensor, infrared sensor, ultrasonic sensor etc. Ultrasonic sensor is most suitable for obstacle detection and it is of low cost and has high ranging capability. Arduino robot that can be controlled by an android mobile or tablet, with the help of an android app that can be downloaded from Google Play store. The android application gets connected to the Bluetooth module and sends desired commands. This app controlled robot is capable to move in any direction. Though there are lots of similar apps out there, we have programmed this project to be used with ANDROID app.

1. **Introduction**

The project is designed to build an obstacle avoidance robotic vehicle using ultrasonic sensors for its movement. An Arduino Uno is used to achieve the desired operation. A robot is a machine that can perform task automatically or with guidance. Robotics is generally a combination of computational intelligence and physical machines (motors). Computational intelligence involves the programmed instructions. The project proposes robotic vehicle that has an intelligence built in it such that it guides itself whenever an obstacle comes ahead of it. This robotic vehicle is built, using an Arduino Uno. An ultrasonic sensor is used to detect any obstacle ahead of it and sends a command to the Arduino.

In today’s world ROBOTICS is a fast growing and interesting field. ROBOT has sufficient intelligence to cover the maximum area of provided space. Autonomous Intelligent Robots are robots that can perform desired tasks in unstructured environments without continuous human guidance. The obstacle detection is primary requirement of this autonomous robot. The robot gets the information from surrounding area through mounted sensors on the robot.

1. **Technology Requirement (Software and Hardware)**

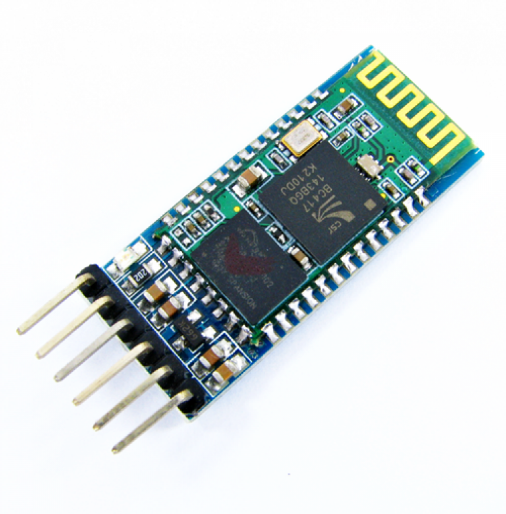
* 1. **Hardware :**
     1. **Dc Motor**

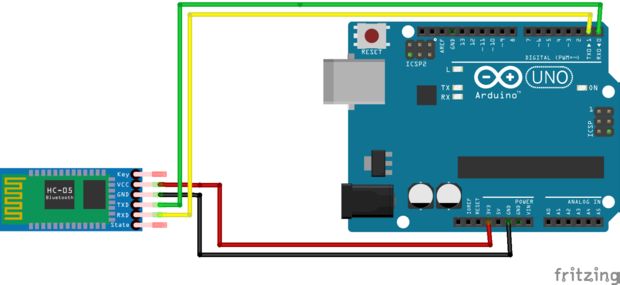
Almost every mechanical movement that we see around us is accomplished by an electric motor. Electric machines are means of converting energy. Motors take electrical energy and produce mechanical energy. Here we use a DC motor.

* + 1. **HC Serial Bluetooth**

HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup.

Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature). It has the footprint as small as 12.7mmx27mm. Hope it will simplify your overall design/development cycle.





* + 1. **A microcontroller**

A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. The one we used is Arduino Uno.

Arduino is a single-board microcontroller designed to make the process of using electronics in multidisciplinary projects more accessible. The hardware consists of a simple open source hardware board designed around an 8-bit Atmel AVR microcontroller. An Arduino board consists of an Atmel 8-bit AVR microcontroller with complementary components to facilitate programming and incorporation into other circuits. An important aspect of the Arduino is the standard way that connectors are exposed, allowing the CPU board to be connected to a variety of interchangeable add-on modules known as shields.



* + 1. **HC-SRO4 ultrasonic sensor**



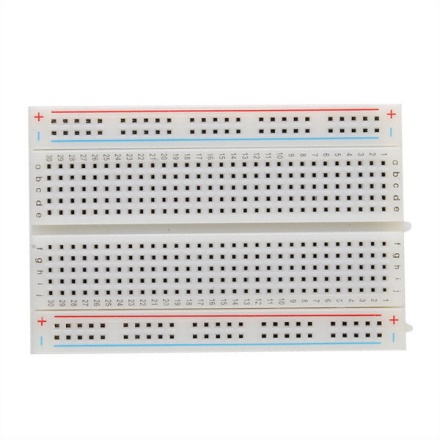
**HCSR04 ultrasonic range finder module:** The HC-SRO4 is an ultrasonic sensor which uses sonar to detect objects at a distance of 2 cm to 4 meters.  This sensor is widely used in robotics to build robots that move and should divert or avoid obstacles. [](http://www.buildcircuit.com.au/ultrasonic-range-finder)The ultrasonic range finder can be used in several ways for range detection and robotics projects.  It is able to detect the distance to obstacles are in front of a mobile robot, allowing a maneuver movements before a collision occurs. Arduino and ultrasonic range finder will give you full control, allowing you to schedule the most convenient way for your project moves your robot. This project uses Arduino UNO R3 and one ultrasonic sensor HC-SRO4. The sensor HC-SRO4 is easy to find and has a good price.

Moreover, it is easy to control it through the Arduino, because it only has 4 pins described below.

**VCC** – 5V (ranging from 4.5 V to 5.5 V)  
**Trig** – Sensor input (trigger)  
**Echo** – Output Sensor (Echo)  
**GND** – Ground

* + 1. **Bread Board**

This is a cute half size breadboard, good for small projects. It's 2.2" x 3.4" (5.5 cm x 8.5 cm) with a standard double-strip in the middle and two power rails on both sides. You can pull the power rails off easily to make the breadboard as thin as 1.4" (3.5cm) and stick it onto an Arduino protoshield. You can also cut these in half with a saw to create 2 tiny breadboards, or "snap" these breadboards together either way to make longer and/or wider breadboards.

****

**Dimensions**

* 2.2" x 3.4" (5.5 cm x 8.5 cm)
* 9.7mm(0.38in) thick, including sticky foam on the bottom
* Weight: 38.9g(1.27oz)

# L293D Motor Driver IC

L293D is a dual [H-bridge](http://www.engineersgarage.com/electronic-circuits/h-bridge-motor-control) motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.

L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result, the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high-impedance state.

****

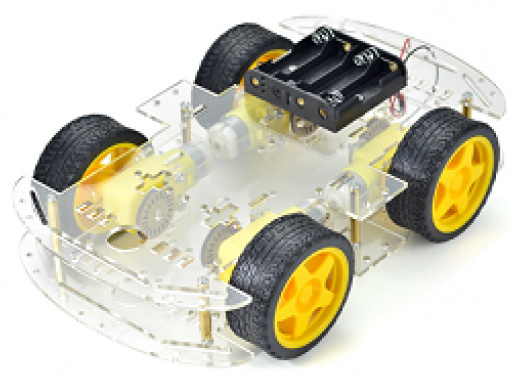
### Pin Diagram:

# L293D Motor Driver IC pin diagram, pinout

### Pin Description:

|  |  |  |
| --- | --- | --- |
| **Pin No** | **Function** | **Name** |
| 1 | Enable pin for Motor 1; active high | Enable 1,2 |
| 2 | Input 1 for Motor 1 | Input 1 |
| 3 | Output 1 for Motor 1 | Output 1 |
| 4 | Ground (0V) | Ground |
| 5 | Ground (0V) | Ground |
| 6 | Output 2 for Motor 1 | Output 2 |
| 7 | Input 2 for Motor 1 | Input 2 |
| 8 | Supply voltage for Motors; 9-12V | Vcc 2 |
| 9 | Enable pin for Motor 2; active high | Enable 3,4 |
| 10 | Input 1 for Motor 1 | Input 3 |
| 11 | Output 1 for Motor 1 | Output 3 |
| 12 | Ground (0V) | Ground |
| 13 | Ground (0V) | Ground |
| 14 | Output 2 for Motor 1 | Output 4 |
| 15 | Input2 for Motor 1 | Input 4 |
| 16 | Supply voltage; 5V | Vcc 1 |

* + 1. **Car Body with tires**

****

* + 1. **DC MOTOR**

****[Remote Control Car Parts](http://www.b2bpakistan.com/12/electronics_electrical-oky_newstar_technology_company-remote_control_car_parts_robot_car_wheel_with_5v_10v_dc_gear_motor-166549.html) , [Robot Car Wheel With 5V - 10V DC Gear Motor](http://www.b2bpakistan.com/12/electronics_electrical-oky_newstar_technology_company-remote_control_car_parts_robot_car_wheel_with_5v_10v_dc_gear_motor-166549.html) Description: 1. Wheel: Diameter: 66mm Width: 28mm Color: Yellow 2. DC gear Motor Dc gear motor, gear ratio 48:1, biaxial can match the speed encoder, convenient measurement speed. DC magnetic motor, gear ratio 48:1, the biaxial motor.

* + 1. **Some Jumper for Connection**

A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire, or DuPont cable – named for one manufacturer of them) is an [electrical wire](https://en.wikipedia.org/wiki/Electrical_wire) or group of them in a cable with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a [breadboard](https://en.wikipedia.org/wiki/Breadboard) or other prototype or test circuit, internally or with other equipment or components, without soldering.

Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the [header connector](https://en.wikipedia.org/wiki/Pin_header#Header_connector) of a circuit board, or a piece of test equipment



* + 1. **An USB Cable**

We have used Serial communication (also known as UART) to send data from arduino to the computer through USB serial COM ports.



**UART:** A Universal Asynchronous Receiver/Transmitter, abbreviated UART, is a piece of computer hardware that translates data between parallel and serial forms. UARTs are commonly used in conjunction with communication standards such as EIA, RS-232, RS-422 or RS-485. We used UART to transfer our data from arduino to the computer through serial COM ports for further processing.

* + 1. **Battery**

Batteries are an essential component of the majority of robot designs. Many types of batteries can be used. Batteries can be grouped by whether or not they are rechargeable.

Batteries that are not rechargeable usually deliver more power for their size, and are thus desirable for certain applications. Various types of alkaline and lithium batteries can be used. Alkaline batteries are much cheaper and sufficient for most uses, but lithium batteries offer better performance and a longer shelf life.

****LiPo batteries are made up of multiple LiPo cells, each producing a nominal 3.7 volts of power. So a 2 cell (2S) LiPo battery gives 7.4v, and a 3 cell (3S) LiPo battery delivers 11.1v.

* 1. **Software Programming Platforms**

For programming the arduino, we have used the default arduino IDE. The code running on the computer which receives the data from serial port, and decodes it to perform various functions is written in C.

**5. Milestones and Reporting**

* + Analysis
  + Development
  + Testing
  + Deployment

|  |  |  |  |
| --- | --- | --- | --- |
| Milestone | Tasks | Hours | Date |
| 1 | Analysis | 90 | 13/01/2017 |
| 2 | Development | 100 | 15/02/2017 |
| 3 | testing | 70 | 18/03/2017 |
| 4 | Deployment | 40 | 16/04/2017 |

**5.1 Deployment**

This car is easy to control by android phone or tab. User can buy this smart car in low price. We provide a long lasting Li-po battery and if user charge the battery once he/she can drive it for 45 minutes.

**5.2 Testing**

The testing process shall be as follows:

Functional Testing System

Testing Performance

User/Acceptance Testing

Field Testing

Installation or Production Testing

1. **Support**

Provide the full support for maintenance, please contact with us if you feel any type of problem

1. **Training**

Give proper training to control the car. To maintain properly without any issue.

1. **Requirement Elicitation**

List of Stakeholder:

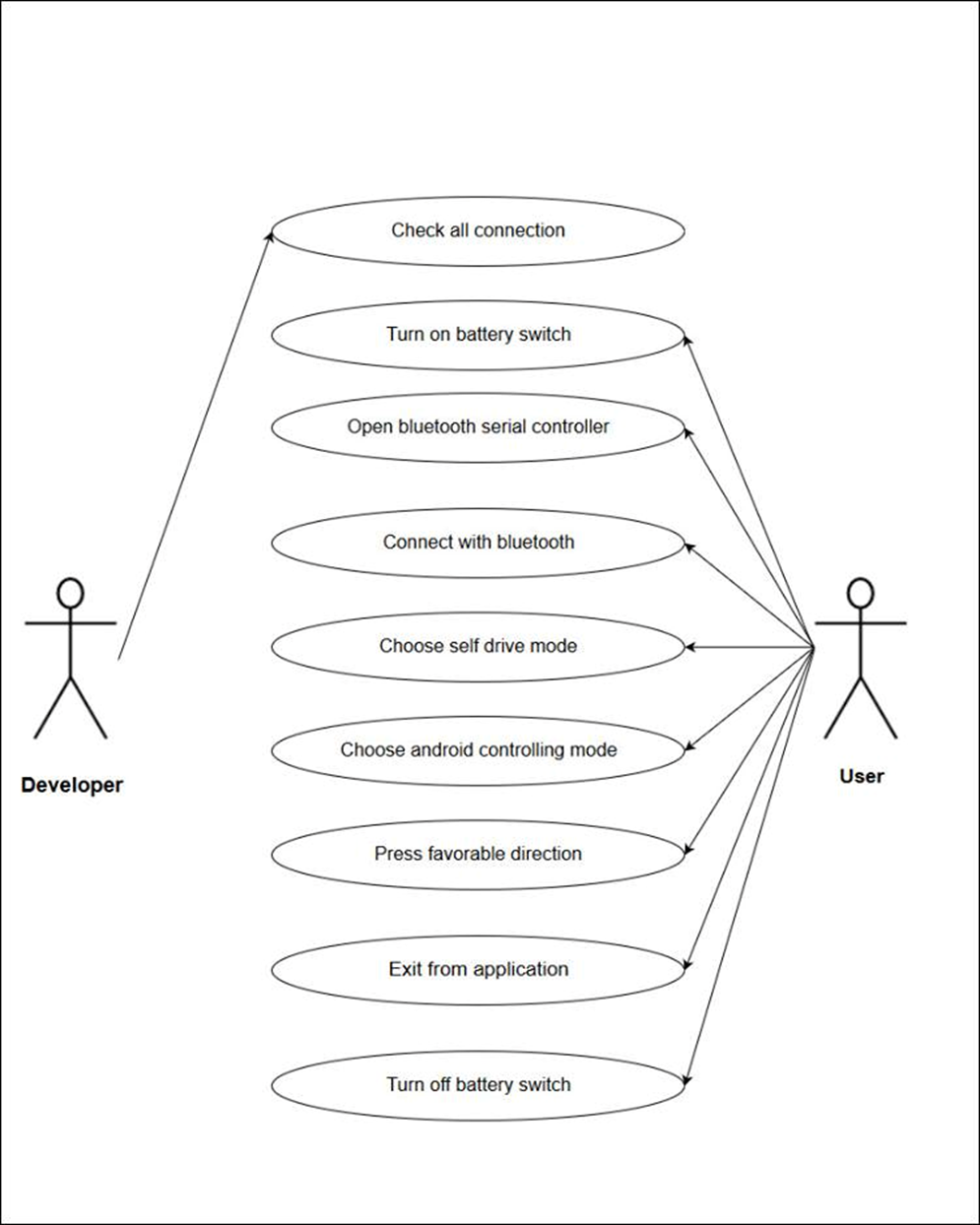
* 1.Developers
* 2.Users
  1. **Task of Developers**

Developers wrote a program for this car. To get connection, controlling system, obstacle find our programmers wrote code, which code helps to run this car . Our developers also create this car by using many type of hardware.

* 1. **Task of Users**

At first Turn on battery switch, then open Bluetooth serial controller and Connect with Bluetooth. After that choose self-drive mode then choose android controlling mode and press favorable direction. If user want to exit from this work then he/she should exit from application and at end of the driving he/she needs to turn off battery switch.

**9. Use case Diagram**

****

**10. Use Case Description**

• Check all connection

* Turn on battery switch
* Open Bluetooth serial controller
* Connect with Bluetooth
* Choose self-drive mode
* Choose android controlling mode
* Press favorable direction
* Exit from application
* Turn off battery switch

|  |  |  |
| --- | --- | --- |
| Use case | Check all connection | |
| Goal | Go to next step and Turn on battery switch | |
| Preconditions | N/A | |
| Success End  Condition | Then turn on battery switch | |
| Failed End  Condition | Again, check your connection | |
| Primary Actors; | user | |
| Secondary Actors | N/A | |
| Trigger | N/A | |
| Description / Main  Success Scenario  <the steps of the  scenario from  trigger to goal  delivery and any  clean up after> | step | Action |
| 1 | Start your next step |
| 2 | N/A |
| 3 | N/A |
| Alternative Flows  <a: condition  causing  branching>  <a1: action or  name of sub use  case> | step | Branching Action |
| 1 | N/A |
| 2 | N/A |

|  |  |  |
| --- | --- | --- |
| Use case | Turn on battery switch | |
| Goal | Start you all engine and connection your all task | |
| Preconditions | Check all connection | |
| Success End  Condition | successfully your connection. | |
| Failed End  Condition | Change your battery | |
| Primary Actors; | User | |
| Secondary Actors | N/A | |
| Trigger | N/A | |
| Description / Main  Success Scenario  <the steps of the  scenario from  trigger to goal  delivery and any  clean up after> | Step | Action |
| 1 | Use 9v battery |
| 2 | Make sure that the battery is good. |
| 3 | Get a successful notice |
| Alternative Flows  <a: condition  causing  branching>  <a1: action or  name of sub use  case> | Step | Branching Action |
| 1 | and turn on your power light. |
| 2 | N/A |

|  |  |  |
| --- | --- | --- |
| Use case | Open Bluetooth serial controller | |
| Goal | Control your car with android phone | |
| Preconditions | Must connect battery | |
| Success End  Condition | Your phone and car Bluetooth connected successfully. | |
| Failed End  Condition | Check your Bluetooth connection | |
| Primary Actors; | user | |
| Secondary Actors | N/A | |
| Trigger | N/A | |
| Description / Main  Success Scenario  <the steps of the  scenario from  trigger to goal  delivery and any  clean up after> | Step | Action |
| 1 | Confirm Bluetooth serial name with your phone |
| 2 | Make sure that the name is correct. |
| 3 | Get a successful notice in your phone |
| Alternative Flows  <a: condition  causing  branching>  <a1: action or  name of sub use  case> | Step | Branching Action |
| 1 | N/A |
| 2 | N/A |

|  |  |  |
| --- | --- | --- |
| Use case | Choose android controlling mode. | |
| Goal | Control your car with your android phone | |
| Preconditions | Must be connect Bluetooth system | |
| Success End  Condition | Start android control system | |
| Failed End  Condition | Check your Arduino code | |
| Primary Actors; | user | |
| Secondary Actors | N/A | |
| Trigger | N/A | |
| Description / Main  Success Scenario  <the steps of the  scenario from  trigger to goal  delivery and any  clean up after> | Step | Action |
| 1 | User need to Choose android controlling mode. |
| 2 | Make sure android controlling code is write. |
| 3 | Get a successful notice and rc car follow your comment. |
| Alternative Flows  <a: condition  causing  branching>  <a1: action or  name of sub use  case> | Step | Branching Action |
| 1 | User must press comment button. |
| 2 | Make sure your comment is write. |

|  |  |  |
| --- | --- | --- |
| Use case | Choose self-drive mode. | |
| Goal | Control your car with its own controlling system. | |
| Preconditions | N/A | |
| Success End  Condition | Perfectly work ultrasonic sensor | |
| Failed End  Condition | Check your ultrasonic sensor connection or change it | |
| Primary Actors; | user | |
| Secondary Actors | N/A | |
| Trigger | N/A | |
| Description / Main  Success Scenario  <the steps of the  scenario from  trigger to goal  delivery and any  clean up after> | Step | Action |
| 1 | Must confirm its connection. |
| 2 | Make sure that ultrasonic sensor was good. |
| 3 | Get a successful and maintain its own comment. |
| Alternative Flows  <a: condition  causing  branching>  <a1: action or  name of sub use  case> | Step | Branching Action |
| 1 | Make sure your Arduino code is write |
| 2 | N/A |

|  |  |  |
| --- | --- | --- |
| Use case | Exit from application | |
| Goal | Whole system was off. | |
| Preconditions | N/A | |
| Success End  Condition | System does not work. | |
| Failed End  Condition | Check your exit pin | |
| Primary Actors; | user | |
| Secondary Actors | N/A | |
| Trigger | N/A | |
| Description / Main  Success Scenario  <the steps of the  scenario from  trigger to goal  delivery and any  clean up after> | Step | Action |
| 1 | Off your power light |
| 2 | N/A |
| 3 | Get a successful off your whole system. |
| Alternative Flows  <a: condition  causing  branching>  <a1: action or  name of sub use  case> | Step | Branching Action |
| 1 | N/A. |
| 2 | N/A. |

**11. Advantages & Application**

**11.1 Advantages**

* It can be used as a movable Surveillance System.
* It can be controlled remotely.
* It does not require Man Power.
* It can be used for critical application like flood, bomb disposal, Fire, Terrorist attack, Earth quake, Spying.

**11.2 Drawbacks of Existing System**

* It is time consuming project.
* It is use for short distance only.
* It is not in human control.
* It is not recommended to keep the range very long because this would cause the robot to keep moving forward and backward as it senses obstacle, even far away from it.

**12. Conclusion**

Enormous amount of work has been done on wireless gesture controlling of robots. In this paper, various methodologies have been analyzed and reviewed with their merits and demerits under various operational and functional strategies. Thus, it can be concluded that features like user friendly interface, light weight and portability of android OS based smart phone has overtaken the sophistication of technologies like programmable glove, static cameras etc., making them obsolete. Although recent researches in this field have made wireless gesture controlling a ubiquitous phenomenon, it needs to acquire more focus in relevant areas of applications like home appliances, wheelchairs, artificial nurses, table top screens etc. in a collaborative manner.

**13. Contact Us**

You can get in touch with us in any of the below ways:

**Md. Sakib Al Islam**

Email: [sakibalislam12345@gmail.com](file:///E:\C%20Project%20Documentation\sakibalislam12345@gmail.com)

Phone: +8801711275277

**Md. Shamol Ali**

Email: [shamol35-1636@diu.edu.bd](file:///E:\C%20Project%20Documentation\shamol35-1636@diu.edu.bd)

Phone: +8801799031771

**Md. Fokhrul Islam Parvez**

Email: [parvezdiu16@gmail.com](mailto:parvezdiu16@gmail.com)

Phone: +8801726407234

User Manual Guide

**Contents**

**Chapter 1: Introduction ……………………………….03**

**Chapter 2: Overviews……………………………………04**

**Chapter 3: Controlling ………………………………….06**

**Chapter 1 : Introduction**

The project is designed to build an obstacle avoidance robotic vehicle using ultrasonic sensors for its movement. An Arduino Uno is used to achieve the desired operation. A robot is a machine that can perform task automatically or with guidance. Robotics is generally a combination of computational intelligence and physical machines (motors). Computational intelligence involves the programmed instructions. The project proposes robotic vehicle that has an intelligence built in it such that it guides itself whenever an obstacle comes ahead of it. This robotic vehicle is built, using an Arduino Uno. An ultrasonic sensor is used to detect any obstacle ahead of it and sends a command to the Arduino.

In today’s world ROBOTICS is a fast growing and interesting field. ROBOT has sufficient intelligence to cover the maximum area of provided space. Autonomous Intelligent Robots are robots that can perform desired tasks in unstructured environments without continuous human guidance. The obstacle detection is primary requirement of this autonomous robot. The robot gets the information from surrounding area through mounted sensors on the robot.

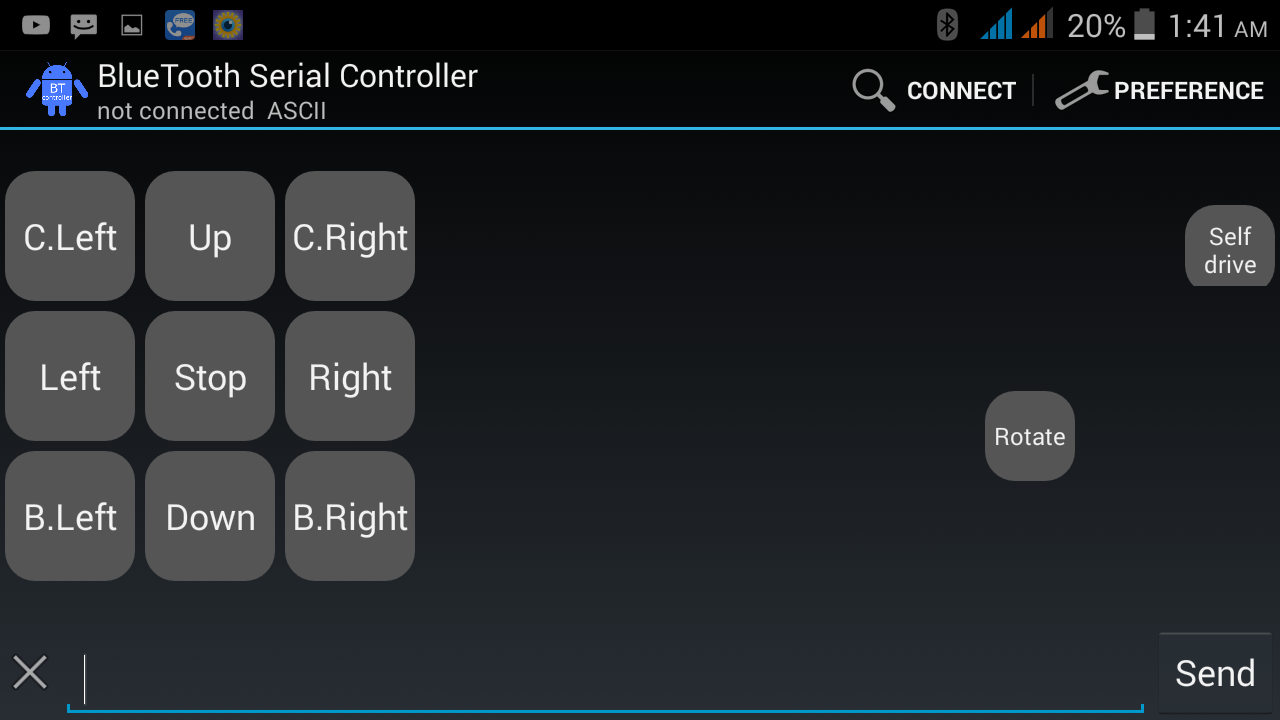
**Chapter 2 : Overviews**

Robotics is the branch of technology that deals with the design, construction, operation, and application of robots. A machine capable of carrying out a complex series of actions automatically, esp. one programmable by a computer is defined as a robot. The project is to develop a robot that will move according to the code assigned but find a free space, navigating from any obstacle on its way. This kind of obstacle is very useful in industries where automatic supervision is needed, for example, in places where it might be risky for humans to be. This robot can also be made by putting other sensors like light sensors or line sensors, ultrasonic sensors and ultrasound sensor depending on the need.

**Chapter 3: Controlling**

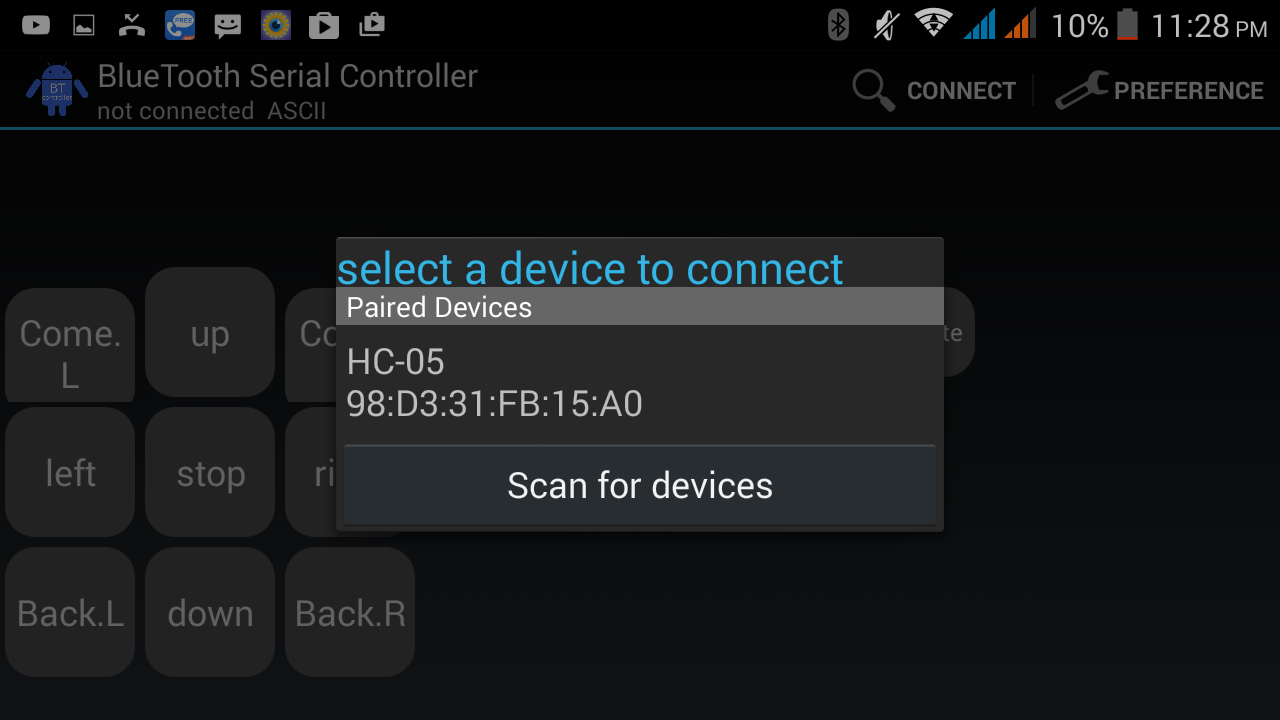
**To use this android controlling car you have to install a software first on your android. We provide you a software, which is BlueTooth Serial Controller.**

**After Installing , you have to open your Bluetooth connection .Then you should open your software. If you does not open your bluetooh and you run this software from your android then its show:**



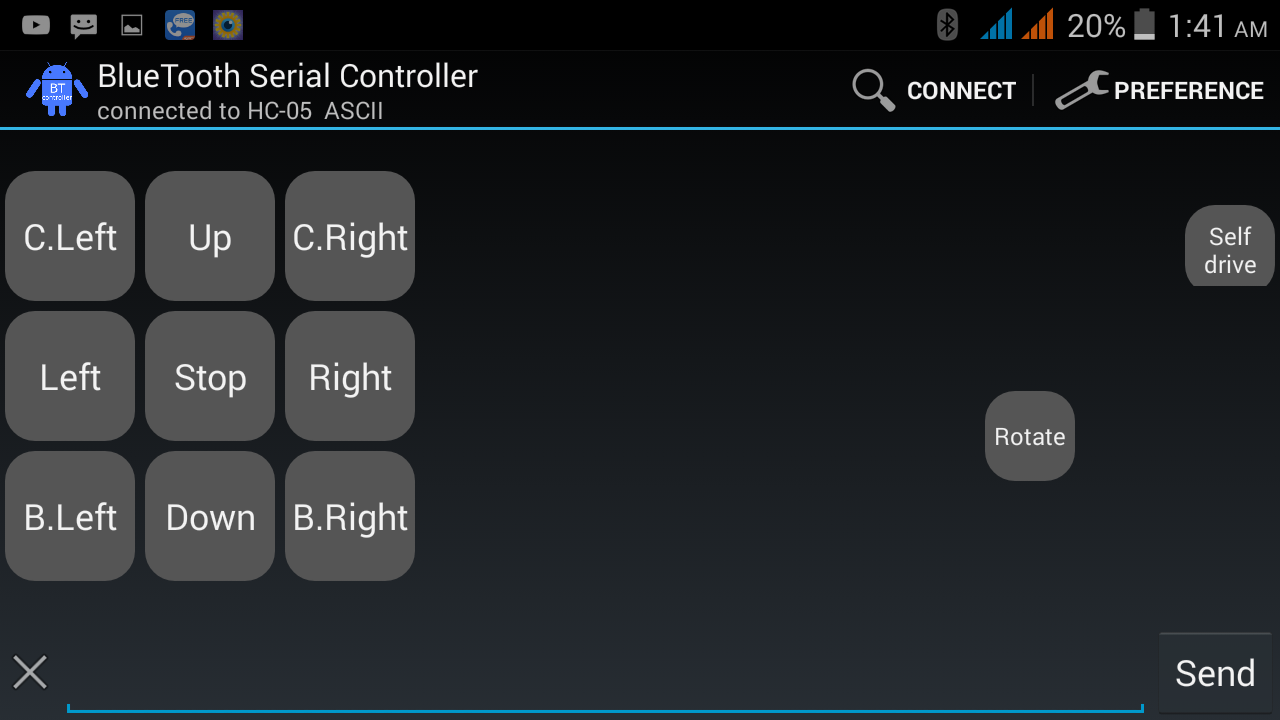
**You have press ‘Connect’ button for activation of your Bluetooth.**

**After activation you can see this type of picture:**

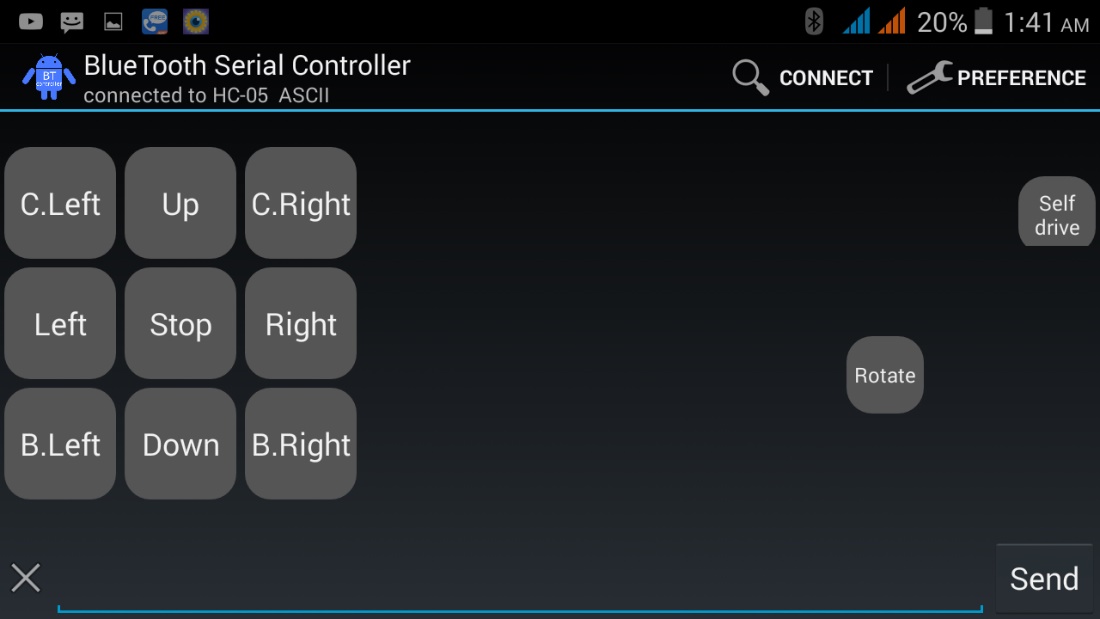


**Then you have to pair with HC-05 .**

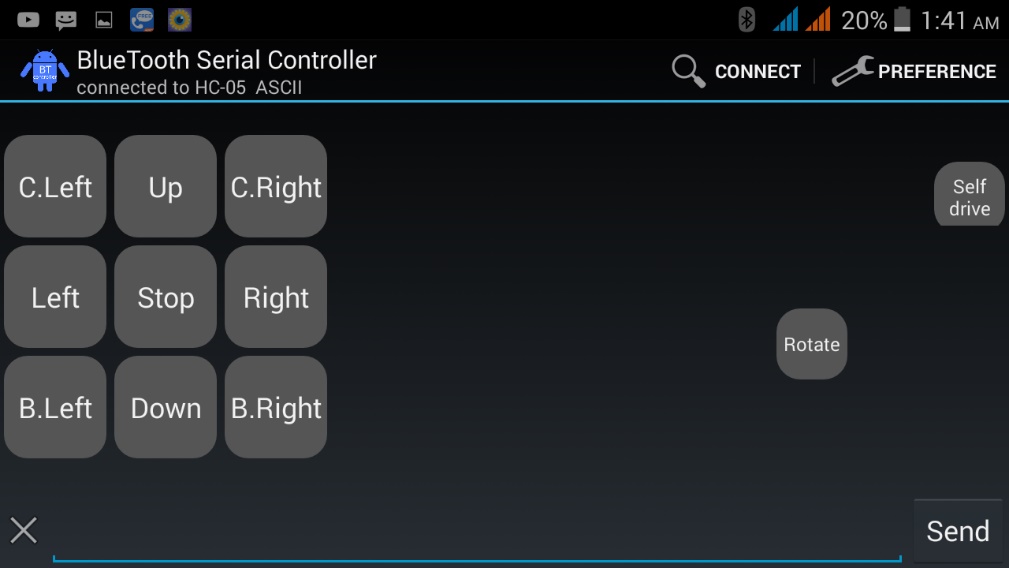
**That means you are connected your android controlling car.HC-05 is the Bluetooth pair connection of your car.**

**After this work you can get your controlling system like this :**

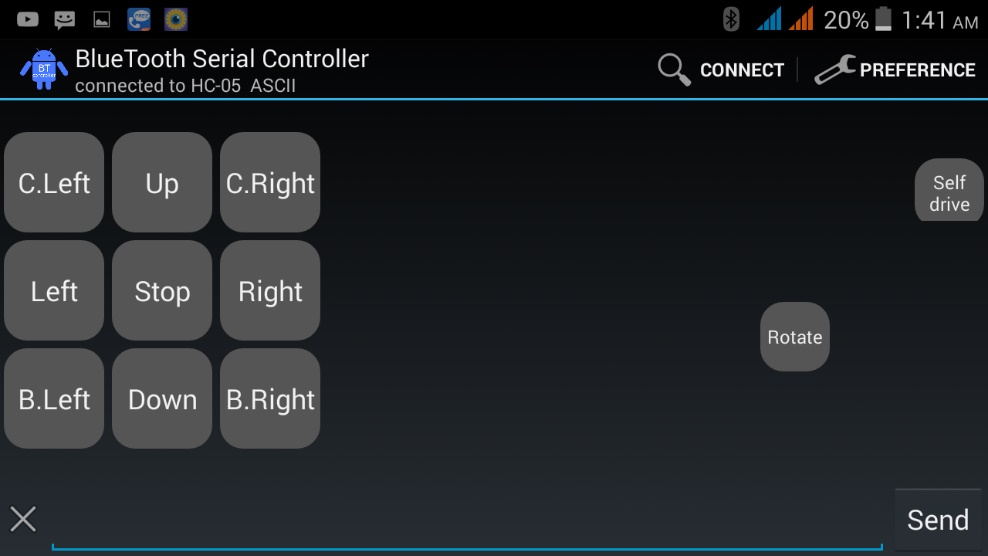
**1.UP: If you press ”up” button,your car will run for front.**



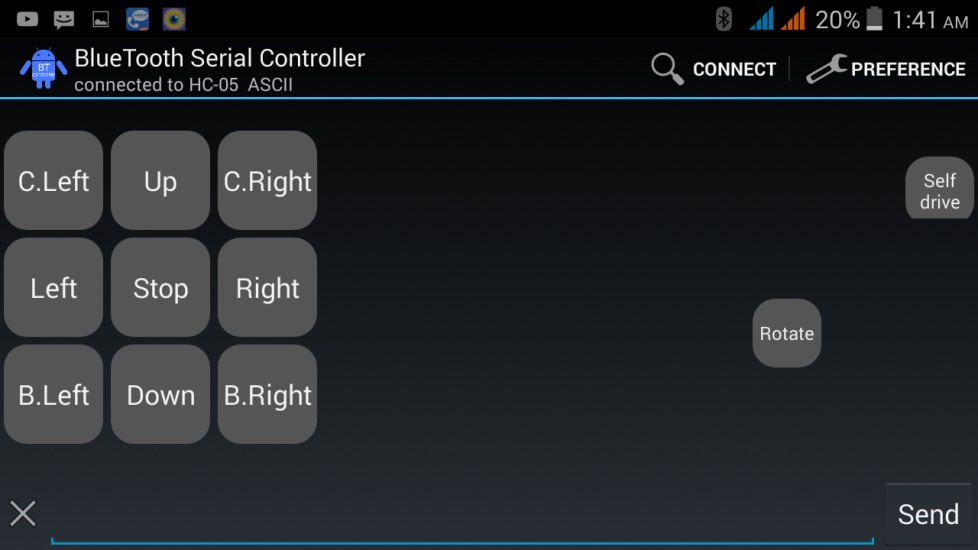
**2.STOP : If you press “stop” button,your car will Stop. (Does not move)**



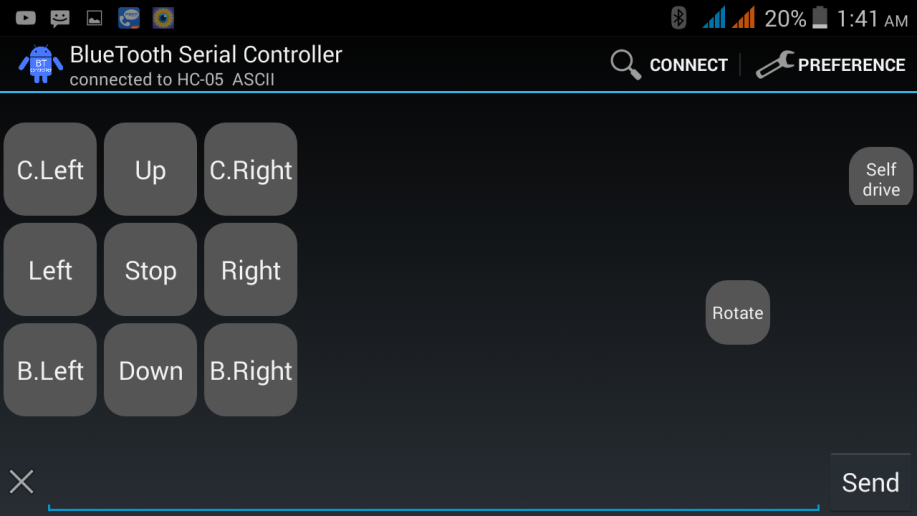
**3.Left: If you press “left” button, your car will turn into left and run .**



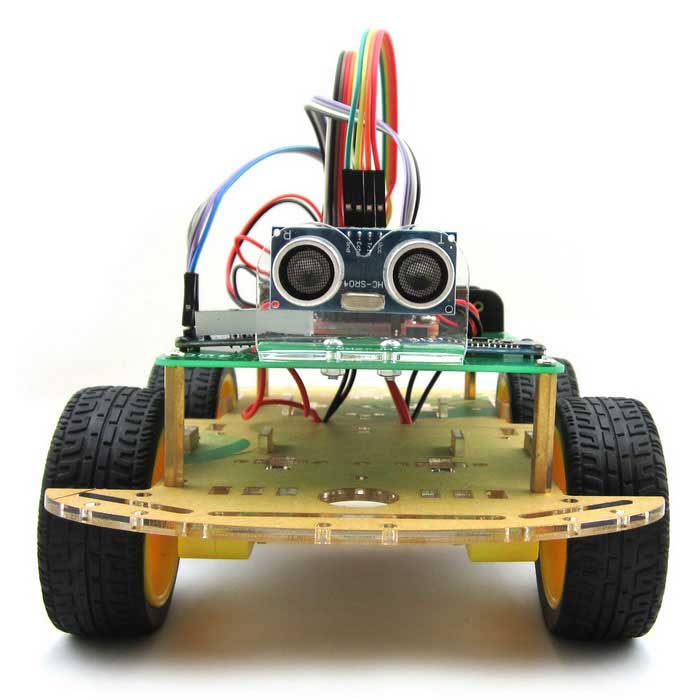
**4.Right: If you press “right” button, your car will turn into right and run.**



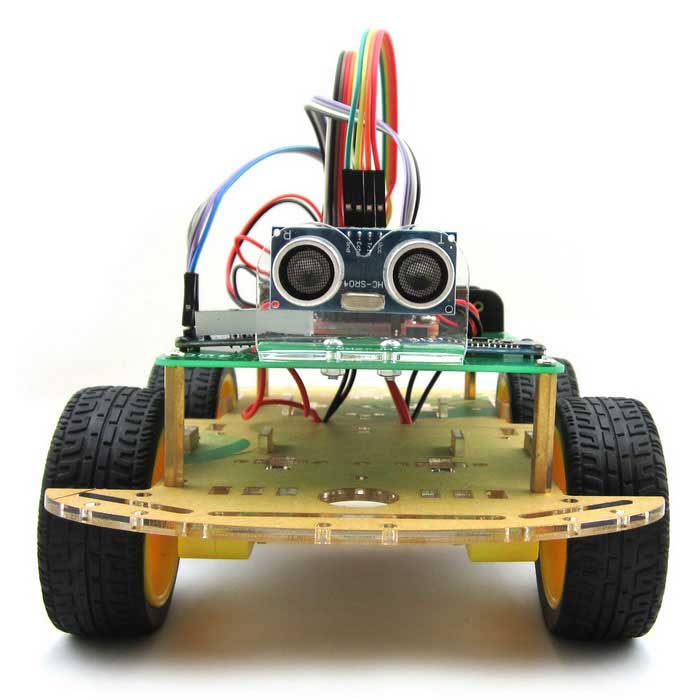
**5.Down: If you press Down button,your car will will run for back.**



**6.Come.L: when the car come to you on that time if you want to move the car on your left then press “Come.L”.**

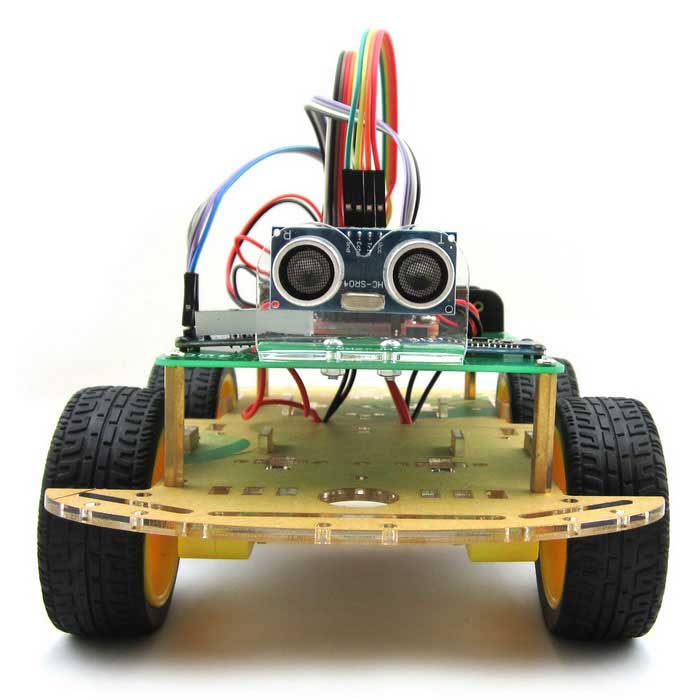


1. **Come.R: when the car come to you on that time if you want to move the car on your right then press “Come.R”.**

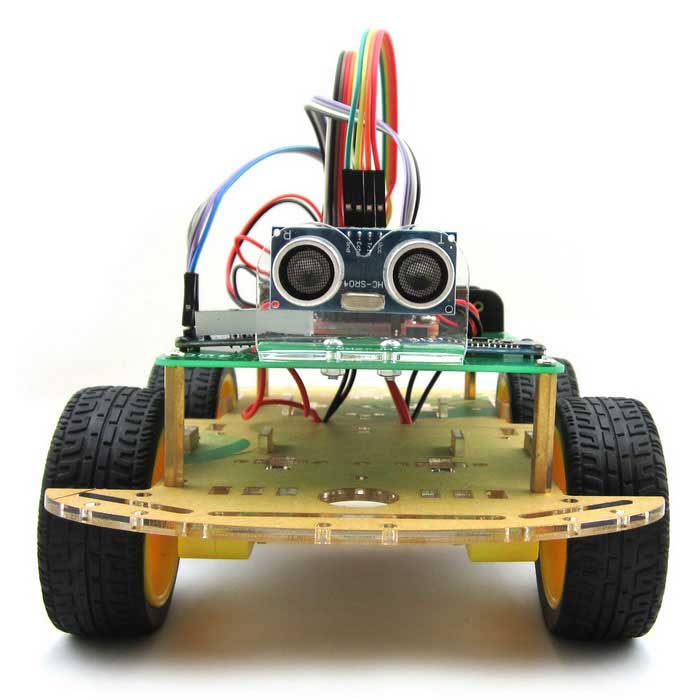


**8.Back.R:**

**If you want to turn your car back at a time right press “Back.R” button.**

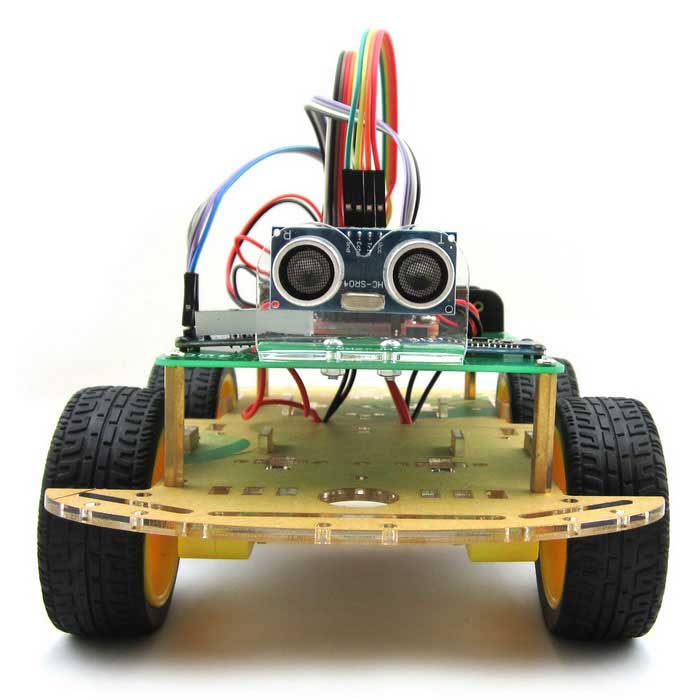
****

**9.Back.L: If you want to turn your car back at a time right press “Back.L” button.**



**10.Rotate:**

**For rotating your car in 360 degree press “Rotate” .**



**11.Self-drive mode:**

**If you want to fix the car in self-drive mode then press “Self drive mode”.**

**In that case, you need to continue pressing on this button till you want.**

**Project Code**

**String voice;**

**int**

**in1 = 2, //Connect To Pin #2**

**in2 = 3, //Connect To Pin #3**

**in3 = 4, //Connect To Pin #4**

**in4 = 5; //Connect To Pin #5**

**int trig = 12;**

**int echo = 11;**

**int duration = 0;**

**int distance = 0;**

**void setup() {**

**Serial.begin(9600);**

**pinMode(trig , OUTPUT);**

**pinMode(echo , INPUT);**

**pinMode(in1, OUTPUT);**

**pinMode(in2, OUTPUT);**

**pinMode(in3, OUTPUT);**

**pinMode(in4, OUTPUT);**

**}**

**void loop() {**

**digitalWrite(trig , HIGH);**

**delayMicroseconds(10);**

**digitalWrite(trig , LOW);**

**duration = pulseIn(echo , HIGH);**

**distance = (duration / 2) / 28.5 ;**

**Serial.println(distance);**

**while (Serial.available())**

**{**

**delay(10); //Delay added to make thing stable**

**char c = Serial.read(); //Conduct a serial read**

**if (c == '#') {break;} //Exit the loop when the # is detected after the word**

**voice += c; //Shorthand for voice = voice + c**

**}**

**if (voice.length() > 0) {**

**Serial.println(voice);**

**//-----------------------------------------------------------------------//**

**//----------Control Multiple Pins----------//**

**if(voice =="\*sensor")**

**{**

**if ( distance < 40)**

**{**

**digitalWrite(in1,HIGH);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,HIGH);**

**digitalWrite(in4,LOW);**

**delay(300);**

**digitalWrite(in1,LOW);**

**digitalWrite(in2,HIGH);**

**digitalWrite(in3,HIGH);**

**digitalWrite(in4,LOW);**

**delay(200);**

**}**

**else if ( distance > 40 )**

**{**

**digitalWrite(in1,LOW);**

**digitalWrite(in2,HIGH);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,HIGH);**

**}**

**}**

**else if(voice =="\*up")**

**{**

**digitalWrite(in1,LOW);**

**digitalWrite(in2,HIGH);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,HIGH);**

**delay(500);**

**}**

**else if(voice=="\*right")**

**{**

**digitalWrite(in1,LOW);**

**digitalWrite(in2,HIGH);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,LOW);**

**delay(500);**

**}**

**else if(voice=="\*come\_right")**

**{**

**digitalWrite(in1, LOW);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,HIGH);**

**delay(500);**

**}**

**else if(voice=="\*back\_right")**

**{**

**digitalWrite(in1, HIGH);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,LOW);**

**delay(500);**

**}**

**else if(voice=="\*down")**

**{**

**digitalWrite(in1, HIGH);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,HIGH);**

**digitalWrite(in4,LOW);**

**delay(500);**

**}**

**else if(voice=="\*left")**

**{**

**digitalWrite(in1, LOW);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,HIGH);**

**delay(500);**

**}**

**else if(voice=="\*come\_left")**

**{**

**digitalWrite(in1,LOW);**

**digitalWrite(in2,HIGH);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,LOW);**

**delay(500);**

**}**

**else if(voice=="\*back\_left")**

**{**

**digitalWrite(in1,LOW);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,HIGH);**

**digitalWrite(in4,LOW);**

**delay(500);**

**}**

**else if(voice =="\*rotate")**

**{**

**digitalWrite(in1,HIGH);**

**digitalWrite(in2,LOW);**

**digitalWrite(in3,LOW);**

**digitalWrite(in4,HIGH);**

**delay(500);**

**}**

**else if(voice == "\*stop")**

**{**

**digitalWrite(in2,LOW);**

**digitalWrite(in4,LOW);**

**digitalWrite(in1,LOW);**

**digitalWrite(in3,LOW);**

**}**

**voice="";}} //Reset the variable after initiating//Reset the variable after initiating**

### ****THANK YOU****