MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Obstacle Avoidance Robot Car

Group 5			
Group Members	Nguyen Minh Nhat Cao Dinh Nguyen Khoa	– Leader – Member	– SE61103 – 60043
Supervisor	Mr. Do Duc Minh Quan		
Ext Supervisor	N/A		
Capstone Project code	OAR		

- Ho Chi Minh City, September 2014 –

This page is intentionally left blank

Acknowledgements

First of all, we would like to give special thanks to our teacher and also our mentor in this capstone project, Mr. Do Duc Minh Quan for his professional guidance, his conscientious, and his recommendations during every phase of this capstone project.

We would like to say thank you to other people for their valuable help and ideas. Particularly we also want to say thanks to our great family. Some time we have been overloaded because of hardware issue, software issue and some other factors, we might not go on without their encouragement.

List of Contents

ACKNOWLEDGEMENTS 3

LIST OF CONTENTS 4

LIST OF TABLES 10

LIST OF FIGURES 11

DEFINITIONS, ACRONYMS AND ABBREVIATION 14

INTRODUCTION 15

I. Project Information 15

II. Scenario 15

III. Existing Solution 19

1. Obstacle Avoiding Robot Using Infrared Radiation Sensor (IR Sensor) 19

2. Obstacle Avoiding Robot Using Camera 20

IV. Proposed Solution and Approach 21

V. Project Overview 22

1. Hardware Overview 22

1.1 Arduino Uno R3 22

1.1.1 Specifications of arduino uno r3 23

1.1.2 Functions. 23

1.1.3 Reason why we use this component 23

1.2 Ultrasonic Sensor SRF-05 24

1.2.1 Specifications of ultrasonic sensor 24

1.2.2 Functions 24

1.2.3 Reason why we use this component. 24

1.2.4 Connector 25

1.3 Servo Motor SG-90 26

1.3.1 Specifications of servo motor sg90 26

1.3.2 Functions 26

1.3.3 Reason why we use this component. 27

1.3.4 Connector 27

1.4 DC Gear Motor. 28
1.4.1 Specifications of dc motor 28
1.4.2 Specifications of gear box 28
1.4.3 Functions 29
1.4.4 Reason why we use this component. 29
1.4.5 Connector 29
1.5 H-Bridge L298N 30
1.5.1 Specifications of l298n 30
1.5.2 Functions 30
1.5.3 Reason why we use this component. 31
1.5.4 Connector 31
1.6 Bluetooth HC-05 32
1.6.1 Specifications of hc-05 32
1.6.2 Functions 33
1.6.3 Reason why we use this component. 33
1.6.4 Connector 33
1.7 Battery Wild Scorpion 11.1V – 1500mAh 35C 34
1.7.1 Specification of wild scorpion 34
1.7.2 Functions 35
1.7.3 Reason why we use this component. 35
2. Development Environment Overview. 35
2.1 Arduino IDE 1.0.6 for C development. 35
2.2 Eclipse Android Development Toolkit – 20140702 version with JDK 8 update 20. . 36
2.3 Fritzing.0.9.0b.64.pc 37
2.4 Operating System 37

VI. Team Introduction 37

PROJECT MANAGEMENT PLAN (PMP) 38

I. Problem Definition 38
1. Capstone Project Name 38
2. Problem Abstract 38
3. Project Overview 38
3.1 The Current System 38
3.2 Proposed System 39
3.3 Boundary Of The System 40
3.4 Development Environment 40

II. Project Organization 41
1. System Project Model 41
2. Roles and Responsibilities 41
3. Tools and Techniques 42
3.1 Development Environment 42

3.2 Technique **42**
3.3 Management Environment **42**
3.4 Communication Environment **43**

III. Project Management Plan **43**
1. Project Iteration ***43***
2. Task Sheet ***45***

IV. Coding Convention. **47**

SYSTEM REQUIREMENT SPECIFICATION (SRS) 48

I. Software **48**
1. Software Requirement ***48***
2. GUI Requirement ***48***

II. Hardware **49**
1. Hardware Requirement ***49***
1.1 Arduino Uno R3. ..... **49**
1.2 Ultrasonic Sensor HY SRF-05. ..... **49**
1.3 Servo Motor SG90. ..... **50**
1.4 DC Gear Motor. **50**
1.5 H-Bridge L298N. **50**
1.6 Bluetooth HC-05 ..... **50**
1.7 Battery Wild Scorpion 11.1V – 1500mAh 35C **51**
2. Hardware Interface Requirement ***51***
2.1 Hardware Interface **51**
2.1.1 Ultrasonic Sensor Interface 51
2.1.2 Servo Motor Interface 52
2.1.3 Bluetooth HC-05 53
2.1.4 H-Bridge L298N 54
2.2 Hardware Communication Protocol **54**
2.2.1 Ultrasonic Sensor 55
2.2.2 Servo Motor 56
2.2.3 Bluetooth HC-05 57
2.2.4 H-Bridge L298N 58

III. Controller **59**
1. Controller Requirement ***59***
1.1 Automatic Obstacles Avoidance Mode. **59**
1.2 Manual Control Mode. **59**
2. System Overall Use Case ***60***
3. List of Use Cases ***61***
3.1 Connect Bluetooth **61**
3.1.1 Connect Bluetooth use case specification 61

3.2 Choose Manual Control Mode **63**
3.2.1 Choose manual control mode use case specification 63
3.3 Navigates Automatically **65**
3.3.1 Navigates automatically use case specification 65
3.4 Go Forward **67**
3.4.1 Go forward use case specification 67
3.5 Go Backward **69**
3.5.1 Go backward use case specification 69
3.6 Turn Left **71**
3.6.1 - Turn left use case specification 71
3.7 Turn right **73**
3.7.1 Turn right use case specification 73
4. Nonfunctional Requirement ***74***
4.1. Reliability **74**
4.1.1 Arduino Uno R3 74
4.1.2 Ultrasonic Sensor SR-05 75
4.1.3 H-bridge L298N 75
4.1.4 Servo Motor SG90 75
4.1.5 DC Gear Motor 75
4.2 Availability. **75**
4.3 Security **76**
4.4. Maintainability. **76**
4.5 Portability. **76**
4.6 Performance. **76**
4.6.1 Arduino Uno R3 76
4.6.2 Ultrasonic Sensor SR-05 76
4.6.3 H-bridge L298N 76
4.6.4 Servo Motor SG90 76
4.6.5 DC Gear Motor 76

SYSTEM DESIGN DESCRIPTION (SDD) 77

I. System Architecture Design **78**

II. System Component Diagram **79**

III. Software **80**
1. User Interface Design ***80***

IV. Hardware **82**
1. Hardware Component Diagram ***82***
2. Hardware Details ***82***
2.1 Ultrasonic Sensor Working Principle **82**
2.2 Servo Motor Working Principle **84**

2.3 DC Geared Motor Working Principle 87
2.4 H-Bridge L298N Working Principle 89
2.5 Bluetooth Working Principle 96
2.5.1 Bluetooth radio transmission 96
2.5.2 Bluetooth network 97
2.5.3 Bluetooth device 98
2.5.4 Bluetooth connect process 98

V. Controller 100
1. Sequence Diagram 100
2.1 Connect Bluetooth Sequence Diagram 100
2.2 Choose Operating Mode Sequence Diagram 101
2.3 Manual Control Sequence Diagram 102
2. Controlling Components and Algorithm 103
2.1 Controlling Ultrasonic Sensor 103
2.1.1 Get Distance from Ultrasonic Sensor 103
2.1.2 Calculate Distance from Sensor to an Object 103
2.2 Controlling Servo Motor 104
2.3 Controlling DC Gear Motor by PWM 105
2.4 Controlling Bluetooth HC-05 106
2.4 Steering Mechanism 108
2.4.1 Normal Car Steering Mechanism. 108
2.4.2 OAR Steering Mechanism 108
2.5 Obstacles Avoidance Algorithm 110
2.6 Back-tracking Algorithm 114
2.6.1 Path Saving 114
2.6.2 Back Tracking 115
7.3 Memory Management 116
7.3.1 Where do we store the values? 116
7.3.2 How do we store data in EEPROM? 117
7.3.3 How much values can be store? 117

SOFTWARE IMPLEMENTATION AND TESTING (SIT) 118

I. Implementation 118

II. Testing 118
1. Testing Overview 118
2. Test Approach 118

III. Test Plan 118
1. Features To Be Tested. 118
2. Testing Environment. 119
2.1 Operating System: 119

2.2 Tools: **119**

IV. Test Case **120**
1 Test Hardware Components ***120***
2. Test Obstacles Avoidance Algorithm ***132***
3. Test on Android application ***137***
4. Test Back-Tracing Algorithm Application ***144***

SYSTEM USER'S MANUAL (SUM) 151

I. System Requirement **151**

II. Install/Uninstall Program **151**
1. Install ***151***
2. Uninstall ***153***
3. Configuration ***153***
3.1 Paring with Bluetooth HC-05 ..... **153**
4. How To Use Application ***155***

REFERENCES 158

List of Tables

Table 1 – Definitions, Acronyms and Abbreviation 14
Table 2 – Project Information 15
Table 3 – Arduino Uno R3 Specification 23
Table 4 – Ultrasonic sensor HY-SRF05 Specification 24
Table 5 – Assign pins of Ultrasonic sensor to Arduino Uno R3 25
Table 6 – Servo Motor TowerPro SG90 26
Table 7 – DC Motor Specification 28
Table 8 – Gear Box Specification 28
Table 9 – H-Bridge L298N Specification 30
Table 10 – Bluetooth HC-05 Specification 32
Table 11 – Battery Wild Scorpion Specification 34
Table 12 – Team Introduction 37
Table 13 – Roles and Responsibilities 42
Table 14 – Project Iteration 45
Table 15 – Task Sheet 47
Table 16 – HY-SRF05 Interfaces 52
Table 17 – Servo Motor SG90 Interfaces 53
Table 18 – Bluetooth HC-05 Interfaces 53
Table 19 – H-Bridge L298N Interfaces 54
Table 20 – HY-SRF05 Connects to Arduino 55
Table 21 – Servo SG90 Connects to Arduino 56
Table 22 – Bluetooth HC-05 Connects to Arduino 57
Table 23 – H-Bridge L298N Connects to Arduino 58
Table 24 – Connect Bluetooth Use Case 62
Table 25 – Choose Manual Mode Control Use Case 64
Table 26 – Navigates Automatically Use Case 66
Table 27 – Go Forward Use Case 68
Table 28 – Go Backward Use Case 70
Table 29 – Turn Left Use Case 72
Table 30 – Turn Right Use Case 74
Table 31 – Android Application Description 81
Table 32 – H-bridge L298N Usage 90
Table 33 – analogWrite Function in Arduino 106
Table 34 – Reserve Table 116
Table 35 – Features to Be Tested 119
Table 36 – Hardware Components Test Cases 132
Table 37 – Obstacles Avoidance Test Case 136
Table 38 – Android Application Test Cases 144
Table 39 – Back-Tracing Algorithm Test Cases 150
Table 40 – Android application buttons and function 157

List of Figures

Figure 1 – Coal Mining Rescue Robot 15
Figure 2 – Mars Rover Robot 16
Figure 3 – Boom Disposal Robot 16
Figure 4 – Wild Cat Robot – US Army 17
Figure 5 – Robot P37 S65 for Caring Elderly People 17
Figure 6 – iRobot Roomba – Vacuum Cleaner 18
Figure 7 – iRobot Mirra – Pool Cleaner 18
Figure 8 – BMW – High Speed Self-driving Car 19
Figure 9 – Obstacles Avoidance Robot Using IR Sensor 19
Figure 10 – Obstacles Avoidance Robot Using Camera 20
Figure 11 – 4 Wheeled Drive Mega Hurtz Robot 21
Figure 12 – Arduino Uno R3 22
Figure 13 – Ultrasonic Sensor HY-SRF05 24
Figure 14 – Ultrasonic Sensor's Sketch 25
Figure 15 – Servo Motor – Tower Pro SG90 26
Figure 16 – Servo Motor's Sketch 27
Figure 17 – DC Motor with Gearbox 28
Figure 18 – DC Motor and H-bridge's Sketch 29
Figure 19 – H-bridge L298N 30
Figure 20 – H-bridge L298N's Sketch 31
Figure 21 – Bluetooth HC-05 32
Figure 22 – Bluetooth HC-05's Sketch 33
Figure 23 – Battery Wild Scorpion 11.1v 1500mAg 34
Figure 24 – Discharge Capacity 35
Figure 25 – Arduino IDE 36
Figure 26 – Eclipse ADT 36
Figure 27 – Fritzing Sketch Drawing Tools 37
Figure 28 – Obstacle Avoidance Robot Using Infrared Radiation Sensor 39
Figure 29 – Obstacle Avoidance Robot Using Camera 39
Figure 30 – Boundary of the System 40
Figure 31 – Waterfall System Development Model 41
Figure 32 – Overall Hardware Interface 51
Figure 33 – HY-SRF05 Interfaces 52
Figure 34 – SG90 Interfaces 52
Figure 35 – Bluetooth HC-05 Interfaces 53
Figure 36 – H-Bridge L298N Interfaces 54
Figure 37 – HY-SRF05's Sketch 55
Figure 38 – Servo Motor SG90's Sketch 56
Figure 39 – Bluetooth HC-05's Sketch 57
Figure 40 – H-Bridge L298N's Sketch 58

Figure 41 – System Overall Use Case 60
Figure 42 – Connect Bluetooth Use Case 61
Figure 43 – Manual Mode Control Use Case 63
Figure 44 – Navigates Automatically Use Case 65
Figure 45 – Go Forward Use Case 67
Figure 46 – Go Backward Use Case 69
Figure 47 – Turn Left Use Case 71
Figure 48 – Turn Right Use Case 73
Figure 49 – Typical Saturation Voltage vs Output Current. 75
Figure 50 – System Architecture Design 78
Figure 51 – Component Diagram 79
Figure 52 – Android Application 80
Figure 53 – Hardware Component Diagram 82
Figure 54 – Ultrasonic Sensor HY-SRF05 82
Figure 55 – HC-SR04 Timing Diagram 83
Figure 56 – Ultrasonic Pulse – Conical Form 84
Figure 57 – Reflected Wave When First Ultrasonic Wave Hits the Obstacles 84
Figure 58 – Servo Motor SG90 Disassembly 85
Figure 59 – Servo Motor SG90's Main Components 85
Figure 60 – Servo Motor SG90's Cog Wheel 86
Figure 61 – Three Wires of Servo Motor 87
Figure 62 – DC Gear Motor and Wheel 87
Figure 63 – Gear Box Mechanism 88
Figure 64 – H-bridge L298N Usage 89
Figure 65 – H-Bridge 91
Figure 66 – Current Flow of H-Bridge 91
Figure 67 – NPN Transistor 92
Figure 68 – PNP Transistor 92
Figure 69 – General H-Bridge Circuit Using BJT Transistor 93
Figure 70 – General H-bridge Circuit Using BJT Transistor 93
Figure 71 – Switch Q1, Q4 close, Q2, Q3 open. 94
Figure 72 – General H-bridge Circuit Using BJT Transistor 94
Figure 73 – Switch Q1, Q4 open, Q2, Q3 close. 95
Figure 74 – General H-bridge Circuit Using BJT Transistor 95
Figure 75 – General H-bridge Circuit Using BJT Transistor 96
Figure 76 – Bluetooth Radio Transmission 97
Figure 77 – Bluetooth network 97
Figure 78 – Bluetooth device 98
Figure 79 – Connect Bluetooth Sequence Diagram 100
Figure 80 – Choose Operating Mode Sequence Diagram 101
Figure 81 – Manual Control Mode Sequence Diagram 102
Figure 82 – Measure Distance from Sensor to an Obstacle 104
Figure 83 – Servo SG90 Pulse Position Modulation Graph 105
Figure 84 – PWM Duty Cycle 106

Figure 85 – UART Wiring 107
Figure 86 – Normal Car Steering Mechanism 108
Figure 87 – OAR Turn Right Mechanism 109
Figure 88 – OAR Steer Left Mechanism 109
Figure 89 – Obstacles Avoidance Algorithm Flow Chart 111
Figure 90 – Ultrasonic Sensor at Center Angle – 90 degree 112
Figure 91 – Ultrasonic Sensor at Left Sidelong Angle – 50 degree 112
Figure 92 – Ultrasonic Sensor at Left Angle – 10 degree 112
Figure 93 – Ultrasonic at Right Sidelong Angle – 140 degree 113
Figure 94 – Ultrasonic Sensor at Right Angle – 190 degree 113
Figure 95 – Phone Setting Menu 151
Figure 96 – Phone Security Menu 151
Figure 97 – Unknown Source Option 152
Figure 98 – Installation File 152
Figure 99 – Installation Process 152
Figure 100 – App Menu 153
Figure 101 – Installed Application 153
Figure 102 – Uninstall Application 153
Figure 103 – Phone Setting Menu 154
Figure 104 – Turn on Bluetooth 154
Figure 105 – Scan New Device 154
Figure 106 – Paring Request 154
Figure 107 – Paring Code 155
Figure 108 – Paired Device Menu 155
Figure 109 – Application 155
Figure 110 – Main Application Layout 155
Figure 111 – Application Layout Details 156

Definitions, Acronyms and Abbreviation

Short Form	Meaning
OAR	Obstacles Avoidance Robot
GUI	Graphical User Interface
PWM	Pulse Width Modulation
UART	Universal Asynchronous Receiver Transmitter
DC Motor	Direct Current Motor
IDE	Integrated Development Environment
GPIO	General Purpose Input Output
H-Bridge	Half Bridge
mS	Millisecond
Us	Microsecond
RX	Receiver
TX	Transmitter
SPP	Serial Port Profile
IR	Infrared Radiation
kHz	Kilo
MHz	Megahertz
PPM	Pulse Position Modulation
SRAM	Static random access memory
EEPROM	Electrical Erasable Programmable Read-Only Memory
TTL	Transistor –transistor logic
Mbit	Mega bit

Table 1 – Definitions, Acronyms and Abbreviation

Introduction

I. Project Information

Project name:	**Obstacles Avoidance Robot Car**
Project code	**OAR**
Product type:	**4 wheels robot**
Start date:	**September, 2014**
End date:	**December, 2014**

Table 2 – Project Information

II. Scenario

Nowadays robotics is a fast growing and interesting field. It is also a practical and attractive field. This is the main reason that many universities open and offer classes and programs that combine mechanical engineering and software engineering instead of "just-software engineering".

It allows student to have new field to study, to explorer, match their interest and they can have more opportunity for works after graduate from university.

Robot has brought us new opportunity to do what humans cannot do. Like exploring new planet, exploring Deep Ocean, exploring mineral mines where beyond human reach. It is all about scientific.

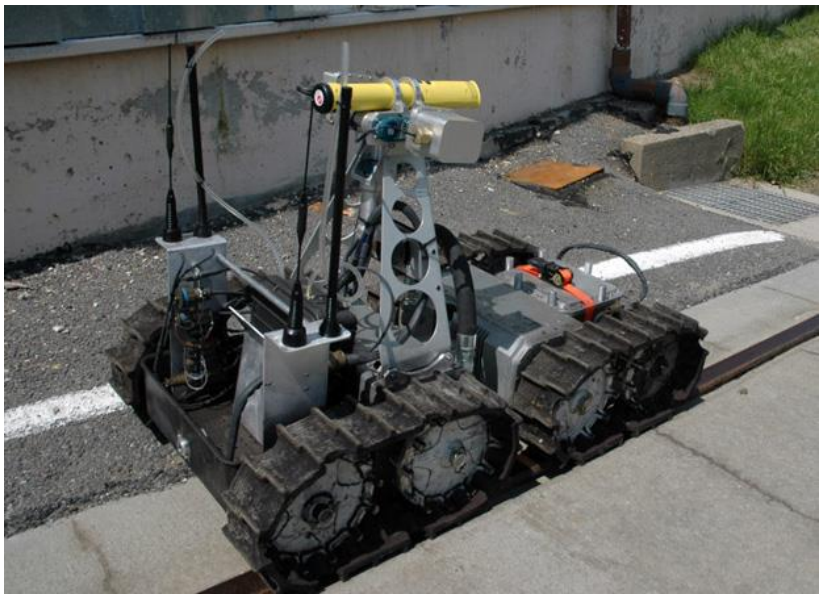


Figure 1 – Coal Mining Rescue Robot

Due to humanity's limitation, we cannot go to other planet (except the moon) and explore it by ourselves. Scientist wants to know and study other planet in our universe for some reasons, but how? Autonomous or manual control wheeled robot is the solution for this problem. Four or six wheels drive make the robot more flexible for various planet's surfaces.

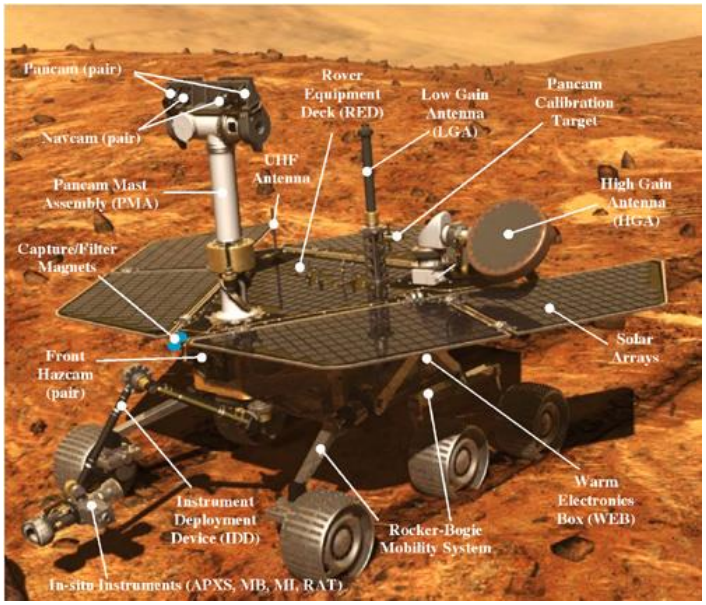


Figure 2 – Mars Rover Robot

In military robot have saved many soldier lives, in the future soldier will not have to risk their life in dangerous mission like hostages rescue, boom or mine disposal.

Figure 3 – Boom Disposal Robot

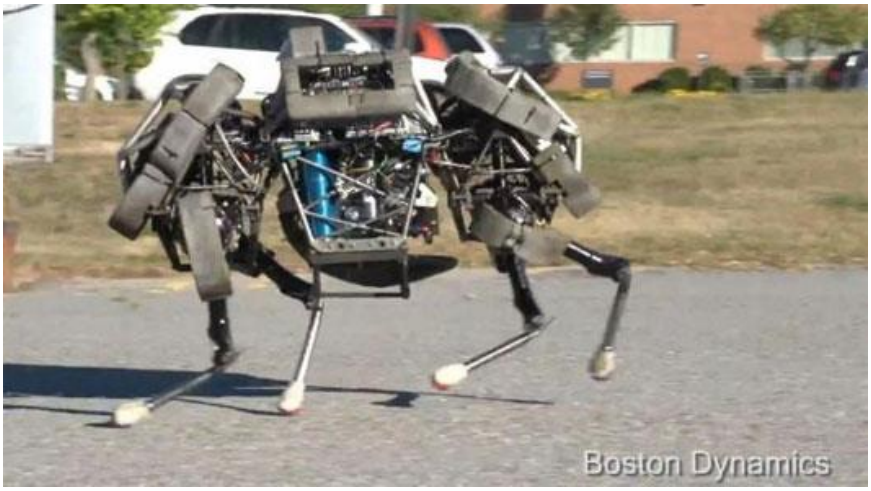


Figure 4 – Wild Cat Robot – US Army

Besides scientific and military, robot also brought us many benefits in our life. Pool cleaning robot, vacuum cleaner robot, robot for elderly people which can take care, acts as a friend, monitoring owner's health and send emergency signal to the hospital when the owner is falling sick, self-driving/parking car and others.

Figure 5 – Robot P37 S65 for Caring Elderly People

Reference: http://www.salford.ac.uk/news/salford-phd-student-develops-revolutionary-elderly-care-robot

People are busy with their lives. We have not much time for doing house works. This robot is an answer for us.

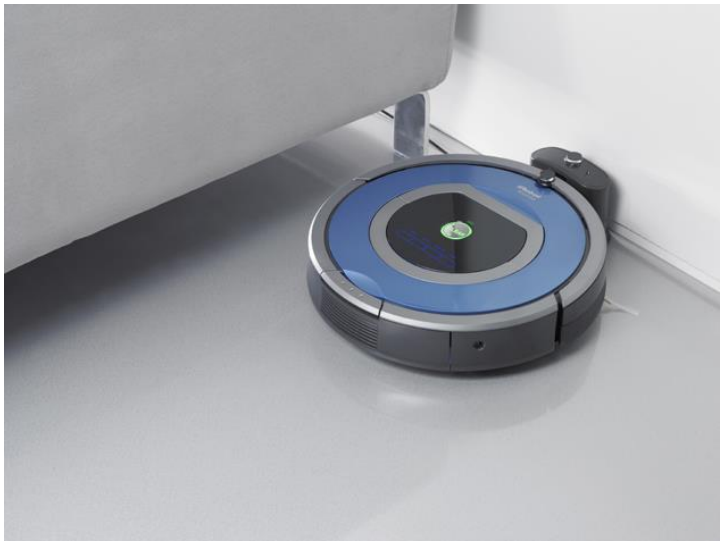


Figure 6 – iRobot Roomba – Vacuum Cleaner

Reference: http://www.irobot.com/For-the-Home/Vacuum-Cleaning/Roomba.aspx

Figure 7 – iRobot Mirra – Pool Cleaner

Reference: http://www.irobot.com/For-the-Home/Vacuum-Cleaning/Roomba.aspx

Figure 8 – BMW – High Speed Self-driving Car

Reference: http://www.huffingtonpost.com/2012/01/26/bmw-self-driving-car_n_1234362.html

There are different types of mobile robot that can be divided into several categories, such as wheeled robot, crawling robot and legged robot. This we choose to deal with wheeled autonomous robot. Our robot uses ultrasonic sensor to detect obstacles on its path and then avoid them to complete its objective.

III. Existing Solution

1. Obstacle Avoiding Robot Using Infrared Radiation Sensor (IR Sensor)

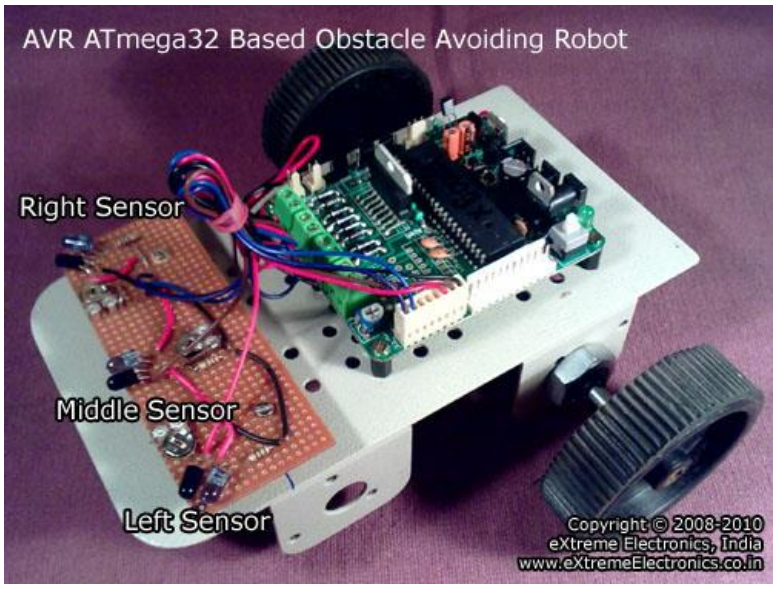


Figure 9 – Obstacles Avoidance Robot Using IR Sensor

Advantages:

- Small, simple to implement.
- Cheap hardware.

Disadvantages:

- IR sensor has narrow beam width, cannot use outside the sun, ambient light would literally distract IR, not highly accurate.
- Short range – about 1.5m.

2. Obstacle Avoiding Robot Using Camera

Figure 10 – Obstacles Avoidance Robot Using Camera

Advantages:

- Precisely to measure obstacles' shape, color and distance from robot to obstacles.

Disadvantages:

- Long processing time.
- If camera lens is dirty, it is hard for robot to detect obstacles.
- Robot will not work properly in low-light condition or camera lens contains dust.
- Complicated to build.

IV. Proposed Solution and Approach

Figure 11 – 4 Wheeled Drive Mega Hurtz Robot

A **4 wheeled robot** provides us high level of mobility and maneuverability, our OAR has ability to avoid obstacle while it is moving. Our goal is to make this robot successfully detect and avoid different kinds of obstacles such as:

- Wall.
- Desk.
- Bottle.
- Other object with various material or shapes.

Next features is that we can control out robot manually by an android smartphone through Bluetooth connection. We are able to control OAR to go forward, backward, turn left, right or stop. User can switch from “Auto Mode” to “Manual Mode” and the opposite.

Other features is our OAR can remember path which it has gone by and ability to re-route and tracing the old path back to the place where it started.

To implement the robot, we need to have following hardware components.

- Hardware that can detect obstacles, we will choose **Ultrasonic Sensor.**
- For movement, we need four **DC motors**.
- To enhance OAR’s flexibility, we will choose **Servo Motor**.
- Bluetooth module for OAR manual control.

- For receiving input from ultrasonic sensor, control DC motor, Servo motor and Bluetooth communication, we will choose Arduino board.

Our robot has some advantages such as:

- Good looking.
- Easy to setup.
- Low power consumption.
- Flexible with great steer ability.
- Response quickly from commands.
- Navigate autonomously.
- Easy to extend new features.

V. Project Overview

1. Hardware Overview

1.1 Arduino Uno R3

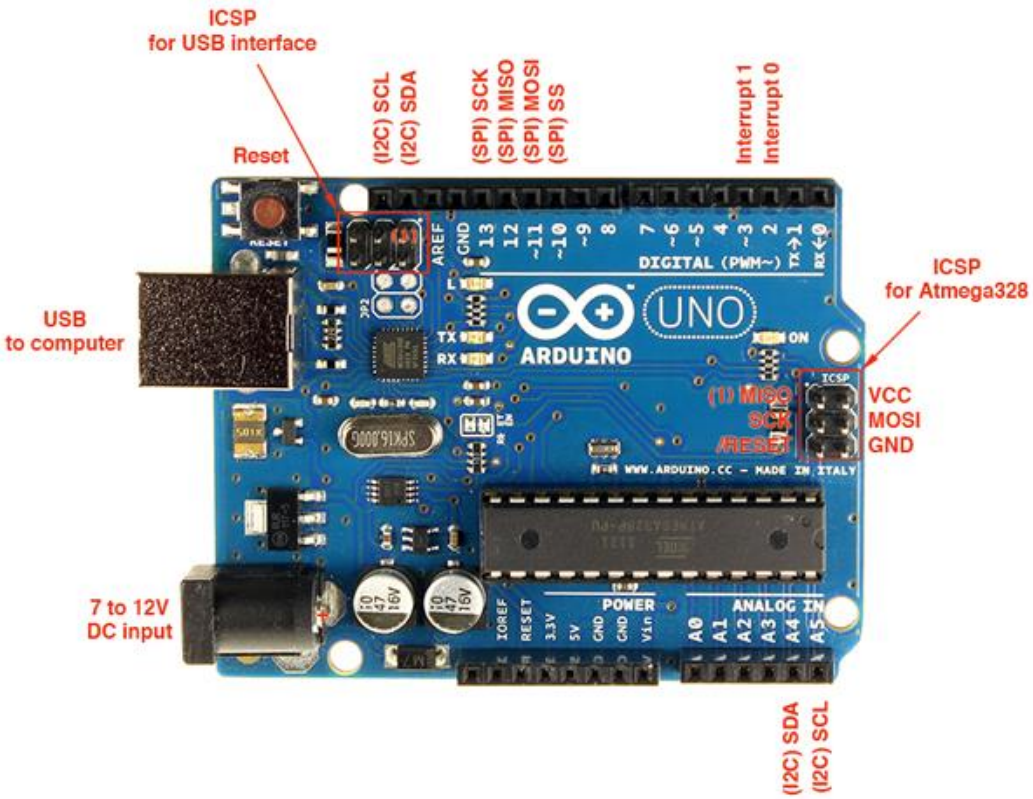


Figure 12 – Arduino Uno R3

1.1.1 Specifications of arduino uno r3

Basic Information	
Microcontroller	ATmeg328
Clock Speed	16MHz
Digital I/O Pins	14 Digital I/O pins – 6 of which can provides PWM outputs
Analog input	6
DC Current per I/O Pin	40mA
Flash Memory	32KB which 0.5 used for bootloader
SRAM	2KB
EEPROM	1KB

Table 3 – Arduino Uno R3 Specification

1.1.2 Functions.

- Communicates with all component.
- Processing data, obstacle avoidance, and path tracing algorithm.

1.1.3 Reason why we use this component

It has 14 digital input / output pins, 6 of them can be used as PWM outputs (For controlling DC Motor).

Arduino Uno R3 is the latest Uno at the moment, in this latest board, it has stronger reset circuit, dedicated reset button.

The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX) for Bluetooth module communication.

Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

1.2 Ultrasonic Sensor SRF-05

Figure 13 – Ultrasonic Sensor HY-SRF05

1.2.1 Specifications of ultrasonic sensor

Basic Information	
Operating voltage	DC 5V
Operating current	15mA
Longest range	4.5m
Recent range	2cm
Measurement of angle	15°
Input signal	TTL pulse of 10uS
Echo signal output	TTL level signal and the range in proportion
Dimension	45mm x 20mm x 15mm

Table 4 – Ultrasonic sensor HY-SRF05 Specification

1.2.2 Functions

- Convert energy into ultrasonic wave.
- Evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively.
- Use to detect & determine the distance to an object, by generate high frequency sound waves and evaluate the time when echo pulses are received back to the sensor.

1.2.3 Reason why we use this component.

- Has reasonable price.
- Ultrasonic wave is not effect by normal environment.
- Ultrasonic speed is fast.

- Revolution step over SRF-04, increase max distance to over 4 meters, one more pin for operating mode, enhance accuracy.
- Compatible with arduino and other boards.
- Compact size.

1.2.4 Connector

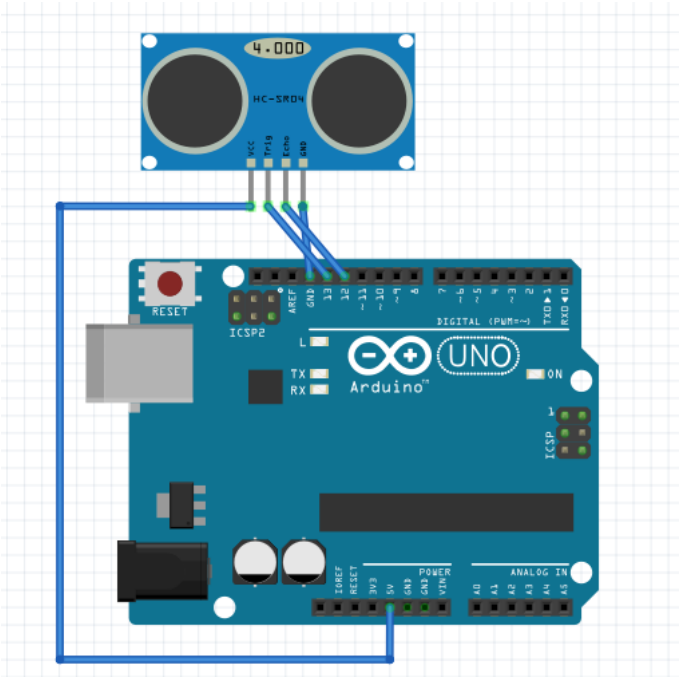


Figure 14 – Ultrasonic Sensor's Sketch

This sketch using SR04 module cause the SRF05 is not supported yet. We are using the same pins (VCC, Trig, Echo, and GND) in case of SRF05.

Pins of Ultrasonic Sensor SRF-05	*Pins of Arduino Uno R3*
VCC	5V
Trig	Pin 13
Echo	Pin 12
GND	GND

Table 5 – Assign pins of Ultrasonic sensor to Arduino Uno R3

1.3 Servo Motor SG-90

Figure 15 – Servo Motor – Tower Pro SG90

1.3.1 Specifications of servo motor sg90

Basic Information	
Torque	4.8V: 1.80kg – cm
Speed	4.8V: 0.10 sec/60^{o}
Weight	9.0g
Dimension	23.1mm x 12.2mm x 29mm
Motor type	3 poles.
Gear type	Plastic
Rotation/Support	Brushing
Rotation range	180^{o}
Pulse width	500-2400uS
Operating voltage	3V to 7.2V

Table 6 – Servo Motor TowerPro SG90

1.3.2 Functions

- In this project, we combine servo with ultrasonic sensor for better obstacles detection of our robot.
- Servo can just rotate 180 degree and it use to sweep in fixed angles.

1.3.3 Reason why we use this component.

- Enhance for obstacles detection.
- Price over performance is good, it can operate at a range of speeds without overheating.

1.3.4 Connector

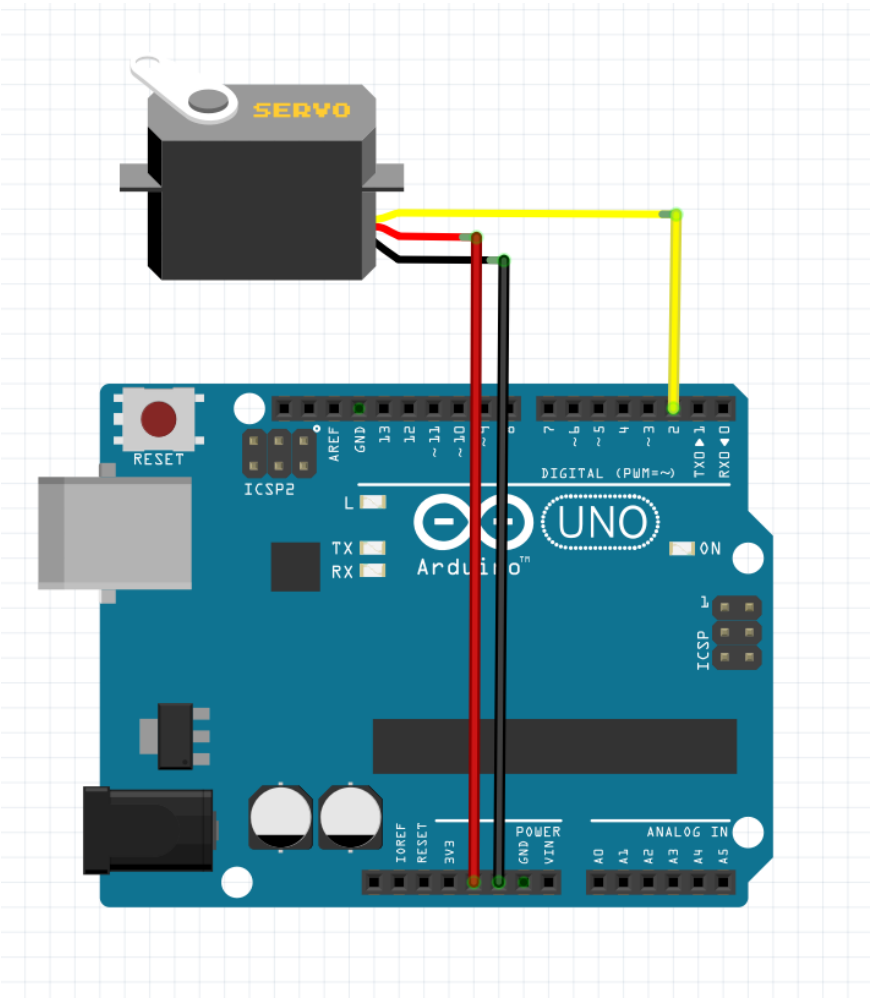


Figure 16 – Servo Motor's Sketch

1.4 DC Gear Motor.

Figure 17 – DC Motor with Gearbox

1.4.1 Specifications of dc motor

Basic Information	
Torque	6V: 45g x cm
Speed	6V – No load: 100RPM
Current	6V – No load: 60mA
Operating voltage	3V – 9V
Motor type	2 – poles
Gear type	Plastic
Rotational range	360^{o}
Dimension	70.5mm x 27mm x 23mm
Weight	40g

Table 7 – DC Motor Specification

1.4.2 Specifications of gear box

Basic Information	
Gear type	Plastic
Transmission ratio	1:48

Table 8 – Gear Box Specification

1.4.3 Functions

- Contain two parts, DC Motor and Gearbox.
    - DC motor: use to moving robot.
    - Gearbox: Decrease speed of dc motor and enhance torque in order to increase traction for motor (more torque, less speed). This gearbox has transmission ratio is 1:48 – This is the ratio of the rotation speed of primary cylinder to secondary cylinder in the gearbox.

1.4.4 Reason why we use this component.

- A good component for our robot to move and archive our goal.
- Gearbox increase the traction of dc motor, our robot car carries many components so we need enough traction for our robot car able to moves.

1.4.5 Connector

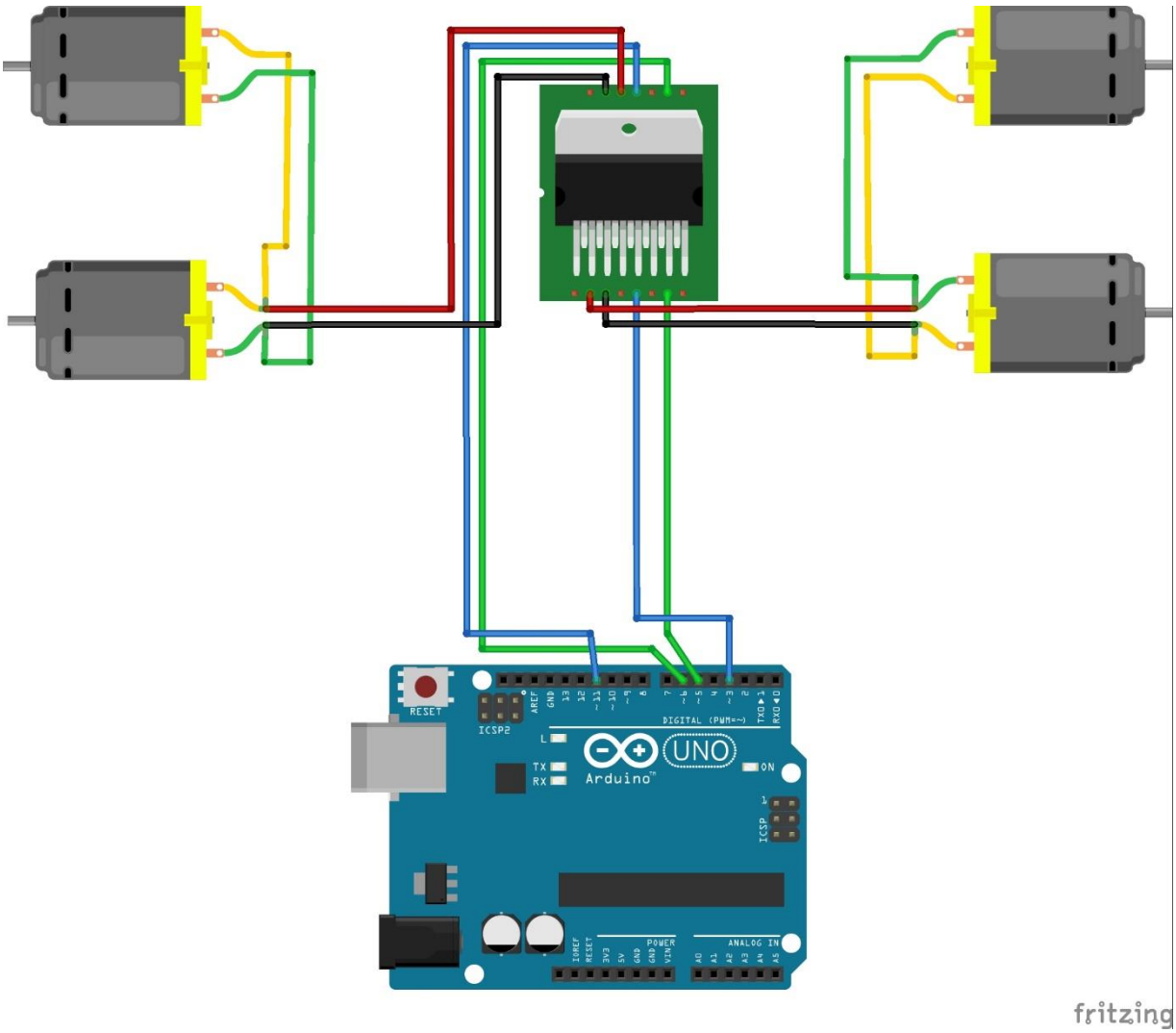


Figure 18 – DC Motor and H-bridge's Sketch

1.5 H-Bridge L298N

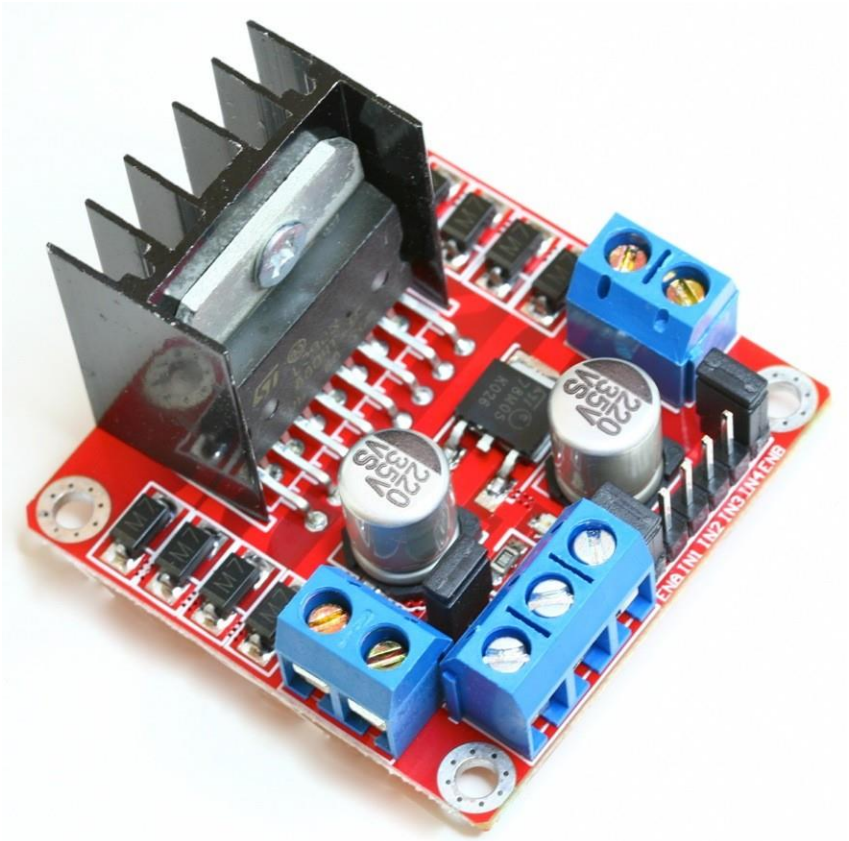


Figure 19 – H-bridge L298N

1.5.1 Specifications of l298n

Basic Information	
Microcontroller	ST L298N
Current	2A per channel or 3A max
Weight	35.0g
Dimensions	60mm x 55mm x 30mm
Operating Voltage	5V to 12V
Motor Controller	2 DC Motors or 1 Stepper motor
Operating Temperature	-25°C to 135°C

Table 9 – H-Bridge L298N Specification

1.5.2 Functions

- Drive motor in both directions, clockwise and anticlockwise.

- Arduino GPIO cannot drive motor with current that greater than 40mA, H-bridge does this job for Arduino GPIO.

1.5.3 Reason why we use this component.

- GPIO of Arduino cannot drive dc motor that has current over 40mA, so we choose this component.
- It has wide operating voltage, heat sink to enhance its durability, light weight and easy to setup and use with arduino.

1.5.4 Connector

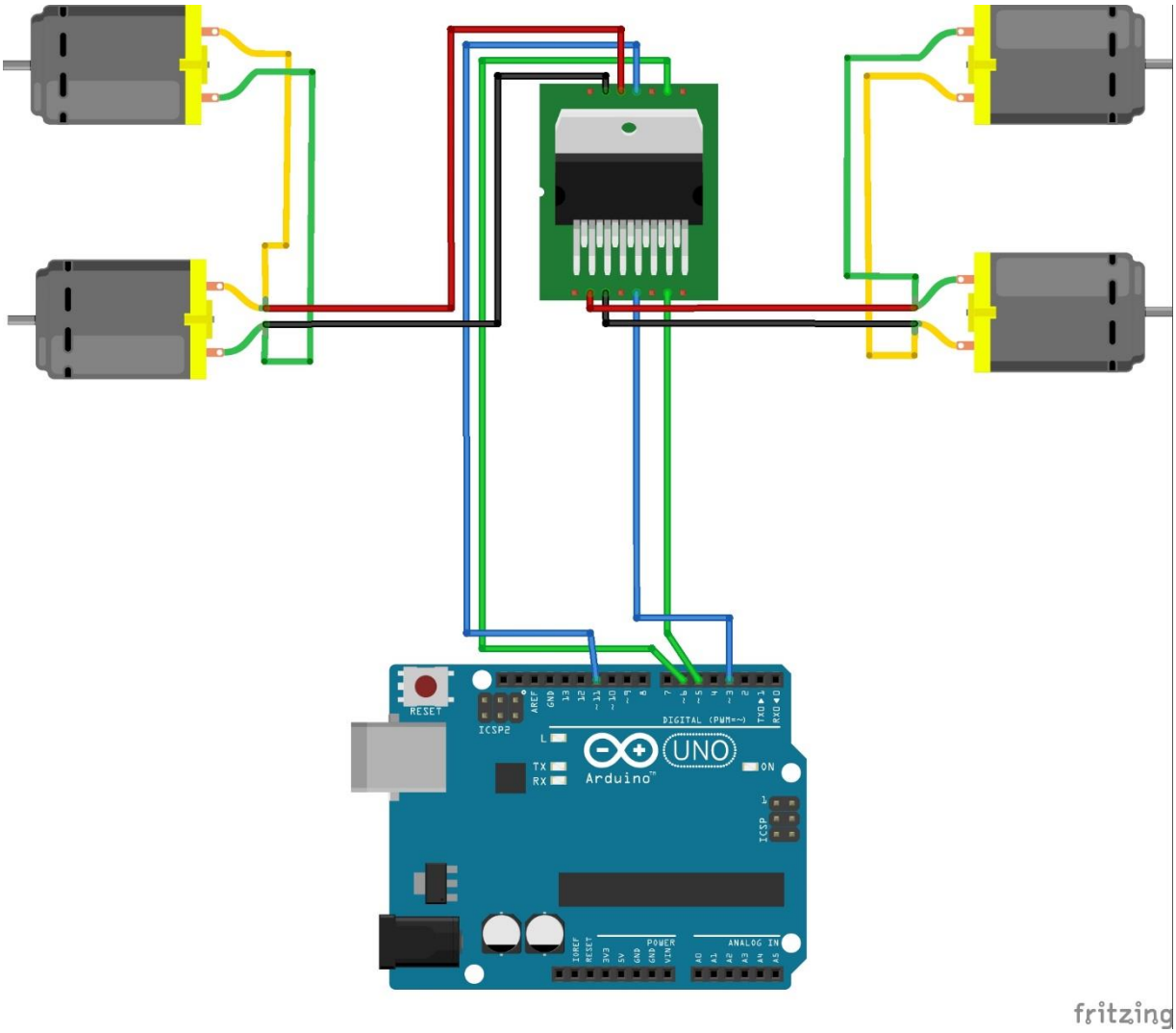


Figure 20 – H-bridge L298N's Sketch

1.6 Bluetooth HC-05

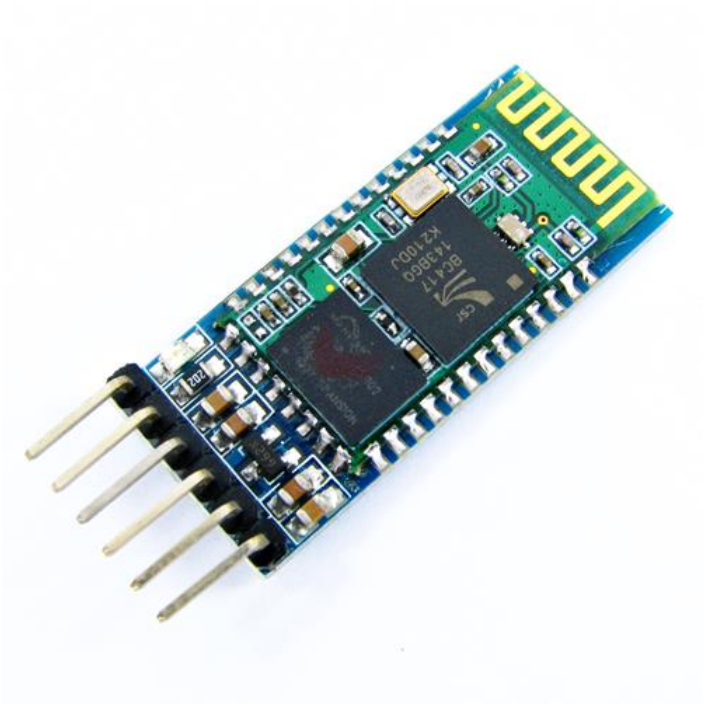


Figure 21 – Bluetooth HC-05

1.6.1 Specifications of hc-05

Basic Information	
Operating current	8mA
Bluetooth protocol	Bluetooth Specification v2.0+EDR
Baud rate	Selectable • 1200 • 2400 • 4800 • 9600 • 19200 • 38400 • 57600 • 115200
Operating voltage	3.3V – 5V
Paring code	1234
Range	More than 9m
Data transfer rate	3Mbit/s, practical 2.3Mbit/s
Dimensions	270mm x 130mm x 220mm

Table 10 – Bluetooth HC-05 Specification

1.6.2 Functions

- Wireless communication between android smartphones and Arduino Uno R3.

1.6.3 Reason why we use this component.

- Popularity.
- Wide range, up to 9 meter.
- Light weight, compact size.
- Fast data transfer rate.

1.6.4 Connector

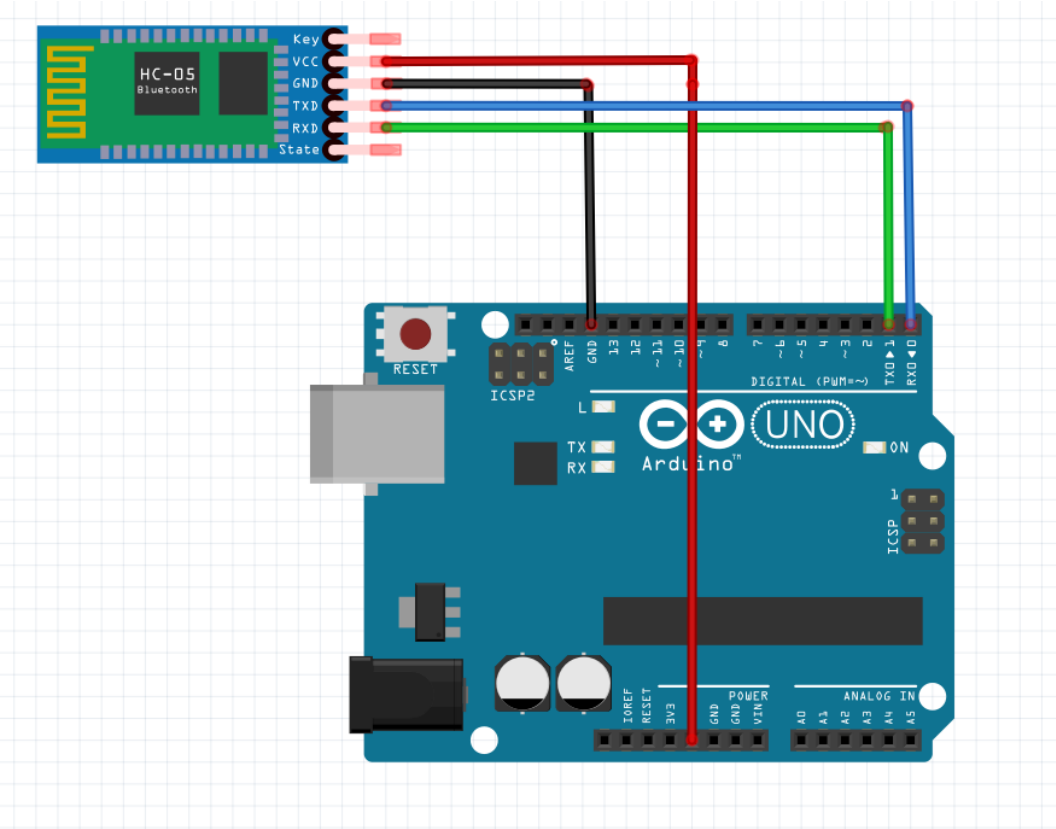


Figure 22 – Bluetooth HC-05's Sketch

1.7 Battery Wild Scorpion 11.1V – 1500mAh 35C

Figure 23 – Battery Wild Scorpion 11.1v 1500mAg

1.7.1 Specification of wild scorpion

Basic Information	
Voltage	11.1v
Capacity	1500mAh
Discharge rate	35C
Dimensions:	750mm x 220mm x 350mm
Weight	115g

Table 11 – Battery Wild Scorpion Specification

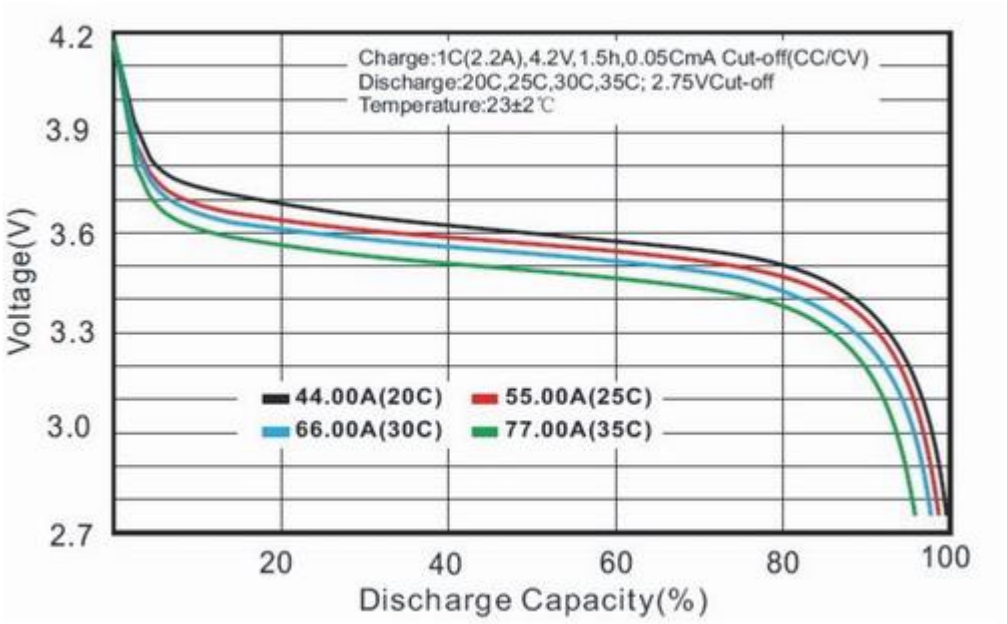


Figure 24 – Discharge Capacity

1.7.2 Functions

- Main power supply for whole robot.

1.7.3 Reason why we use this component.

- Popularity, well known in RC helicopters.
- Two 0.5 mm aluminum pieces as a heat sink.
- Light weight, compact size.
- Solid, light weight, has longer changing cycle than Li-ion battery.
- Less internal resistance for faster discharging and higher amperes, at up to 65C continuously, or 135C in bursts.

2. Development Environment Overview.

2.1 Arduino IDE 1.0.6 for C development.

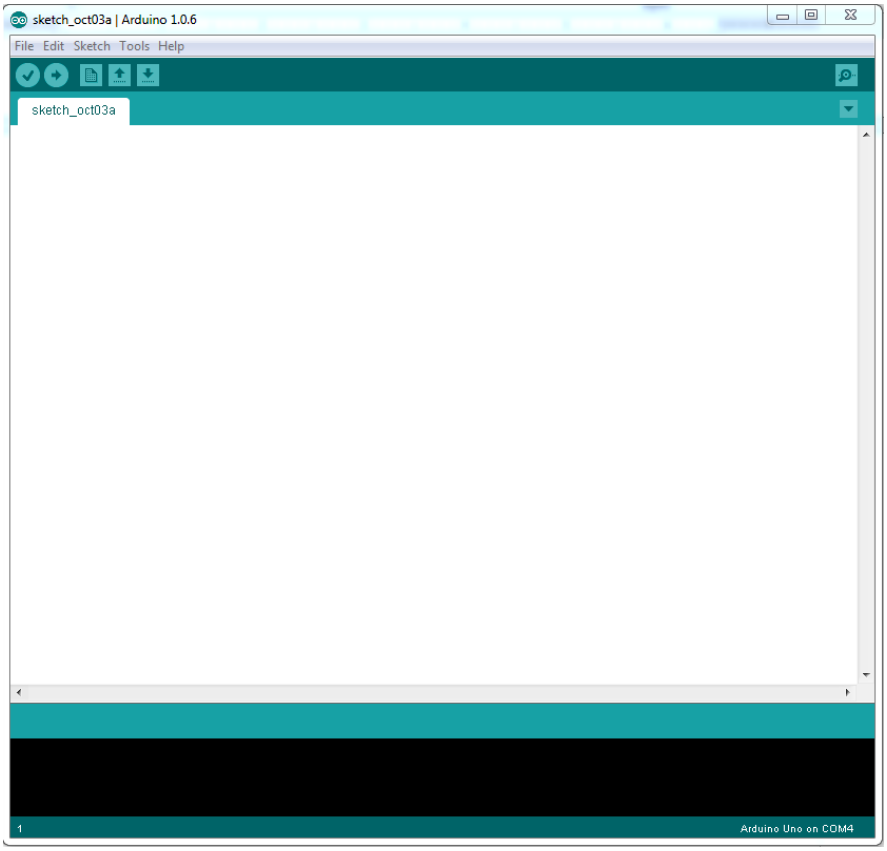


Figure 25 – Arduino IDE

2.2 Eclipse Android Development Toolkit – 20140702 version with JDK 8 update 20.

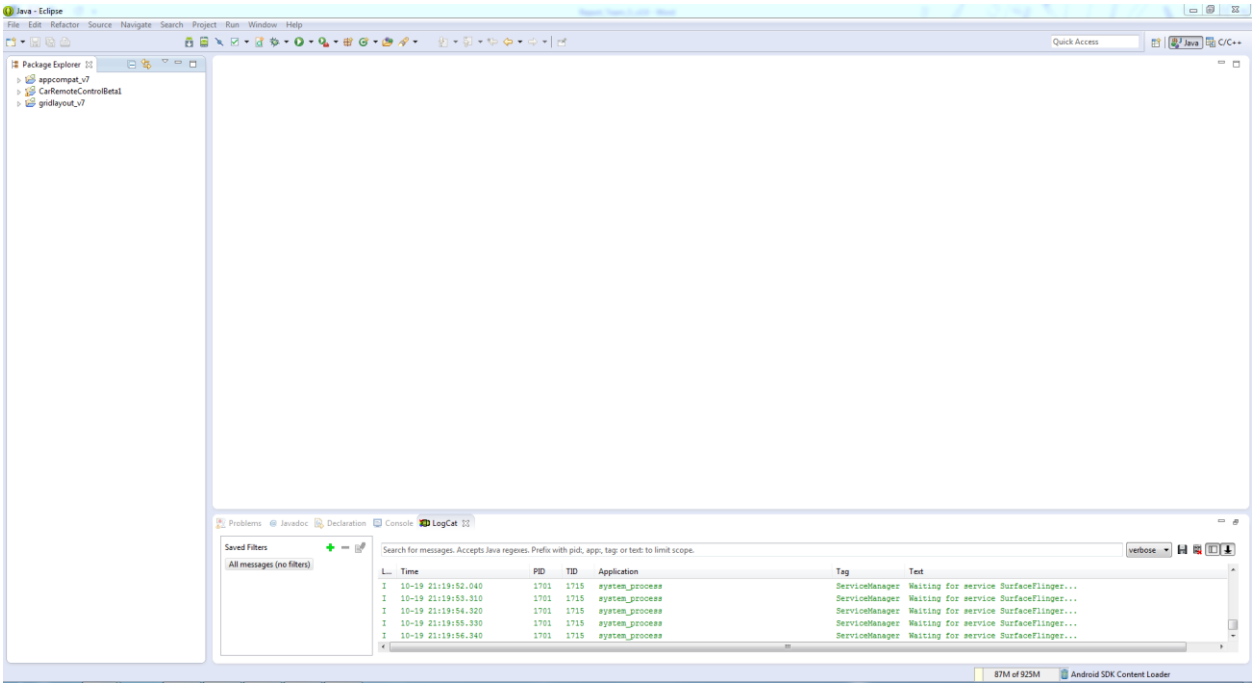


Figure 26 – Eclipse ADT

2.3 Fritzing.0.9.0b.64.pc

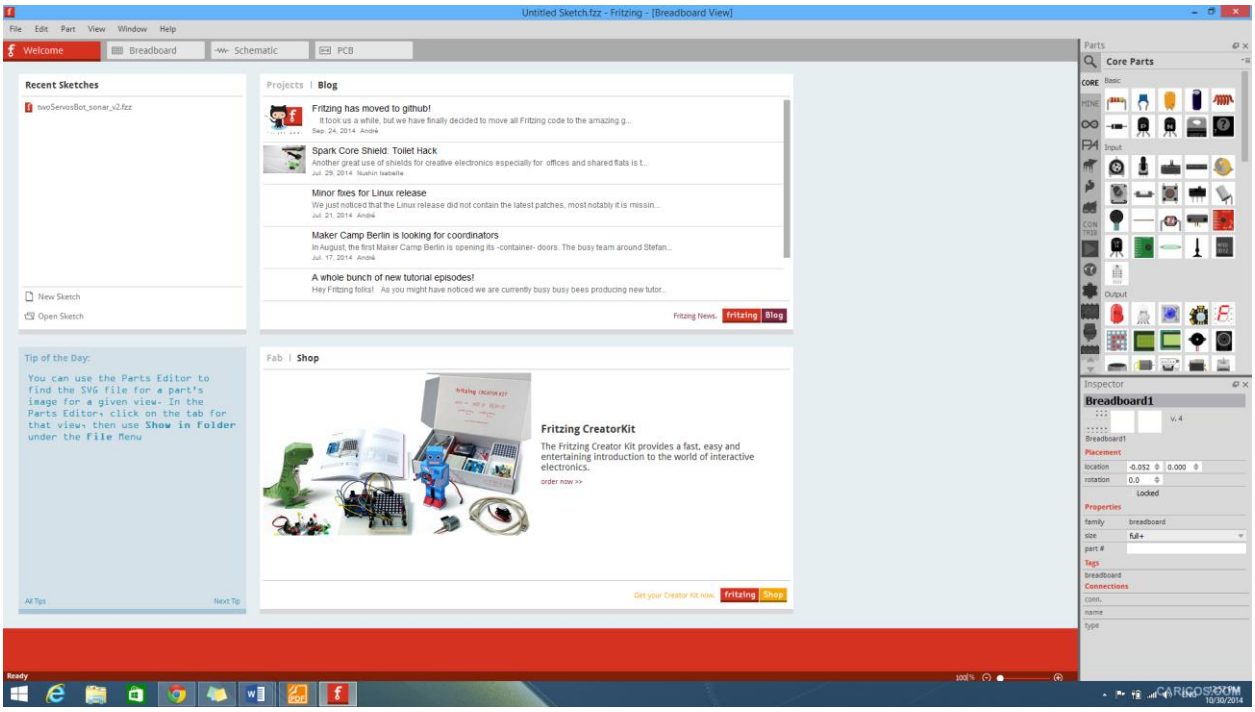


Figure 27 – Fritzing Sketch Drawing Tools

2.4 Operating System

- Windows 7 x64 – Professional.
- Windows 8.1 x64 – Professional.

VI. Team Introduction

Team's member infomation				
No	**Full name**	**Role**	**Position**	**Information**
1	Đỗ Đức Minh Quân	Supervisor	Supervisor	QuanDDM@fpt.edu.vn
2	Nguyễn Minh Nhật	Programmer	Team leader	NhatNMSE61103@fpt.edu.vn
3	Cao Đình Nguyên Khoa	Programmer	Team member	KhoaCDN60043@fpt.edu.vn

Table 12 – Team Introduction

Project Management Plan (PMP)

I. Problem Definition

1. Capstone Project Name

Capstone project name:

- English: Obstacle Avoidance Robot Car.
- Vietnamese: Xe tự động tránh vật cản.

Project code: OAR.

2. Problem Abstract

Our robot's goal is to auto navigating and obstacles avoidance and we can control it manually.

For obstacles avoidance, there are many methods. Use infrared radiation sensor, but its weakness is not accuracy, cannot be used outside in the sun, narrow beam width, the infrared sensor reading will be different for different surfaces , different colors, and different shades even if the range is the same. Most of the time this reading is not too off, so the robot can still function. On the other hand, camera, it is complicated, long processing time and it is not flexible.

To solve this problem, we use ultrasonic combines with servo motor. It is accurate, has wider beam width, no matter what color obstacles are and ultrasonic sensor can detect obstacles up to more than 4 meter far. To increase robot flexibility in detect obstacles, we mount ultrasonic with servo motor, so now the robot's ultrasonic sensor can sweep from left to right, robot will not have to turn left or right for ultrasonic sensor in order to detect obstacles.

About manual control for the robot, we choose Bluetooth technology instead of infrared radiation.

3. Project Overview

3.1 The Current System

Their weak spots are fixed hardware, not easy to extend new features or upgrade to stronger microcontroller, components are not a good choice for detect obstacles.

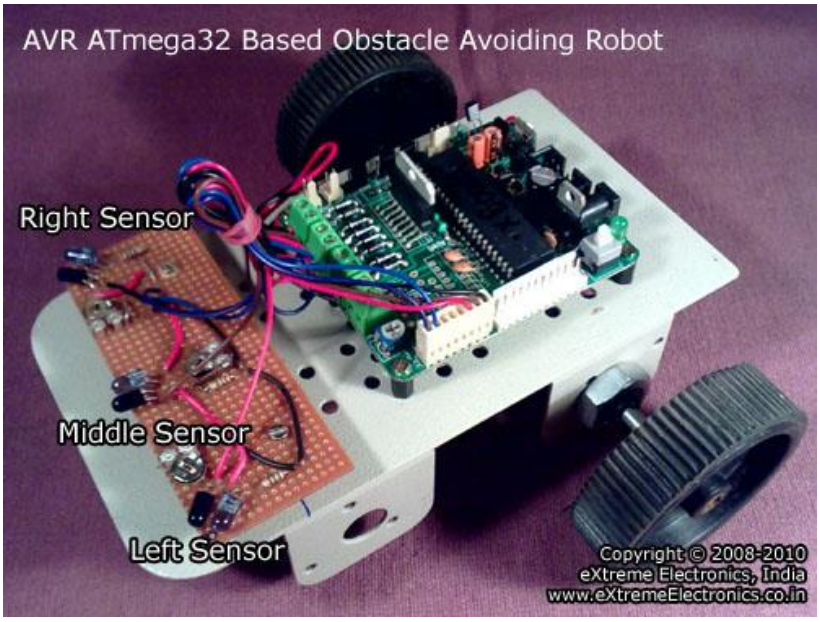


Figure 28 – Obstacle Avoidance Robot Using Infrared Radiation Sensor

Figure 29 – Obstacle Avoidance Robot Using Camera

3.2 Proposed System

Nowadays people need to create a robot that is flexible, stable, to archive their needs in exploring new areas or mines.

Our robot is 4 wheels drive, low profile, so it can able to work in different environments or surfaces.

To archive our goal, we have to focus on electro-mechanical mechanisms and control algorithms in order to make or robot works well in different environments. The main algorithm is mainly based on learning experience.

Our robot will use Arduino Uno R3 as a main brain, if we need stronger microcontroller with more GPIO pins, we can upgrade without any problem, we have great support from the manufacture, community so if there are any problems, we can able to fix it if it is not a critical damage.

All component we use are compatible with arduino board, and they are easy to replace if there are any damage during usage, just plug and play.

3.3 Boundary Of The System

Our robot system contains two main parts:

- **Obstacles Avoidance Robot:** Robot will received value from ultrasonic sensor and use obstacles avoidance algorithm to avoid obstacles.
- **Robot Car Remote Control:** An android application for user to monitor and control robot through Bluetooth connection.

Figure 30 – Boundary of the System

3.4 Development Environment

3.4.1 Hardware Environment

1. Arduino Uno R3.
2. Servo Motor SG90.
3. Ultrasonic Sensor SF-05.
4. DC Gear Motors.
5. H-bridge – L298N.
6. Laptop with appropriate configuration for embedded system development.

3.4.2 Software Environment

1. Arduino IDE.
2. Eclipse with ADT plugin.
3. Windows environment.

II. Project Organization

1. System Project Model

With this project, we devided to use waterfall system development model. At first we communicate to collect and understand system requirement, then we plan everything that we have to do to archive in this project, then we analyze requirement and design system overall after that we execute coding and testing phase, at last we deliver the final product.

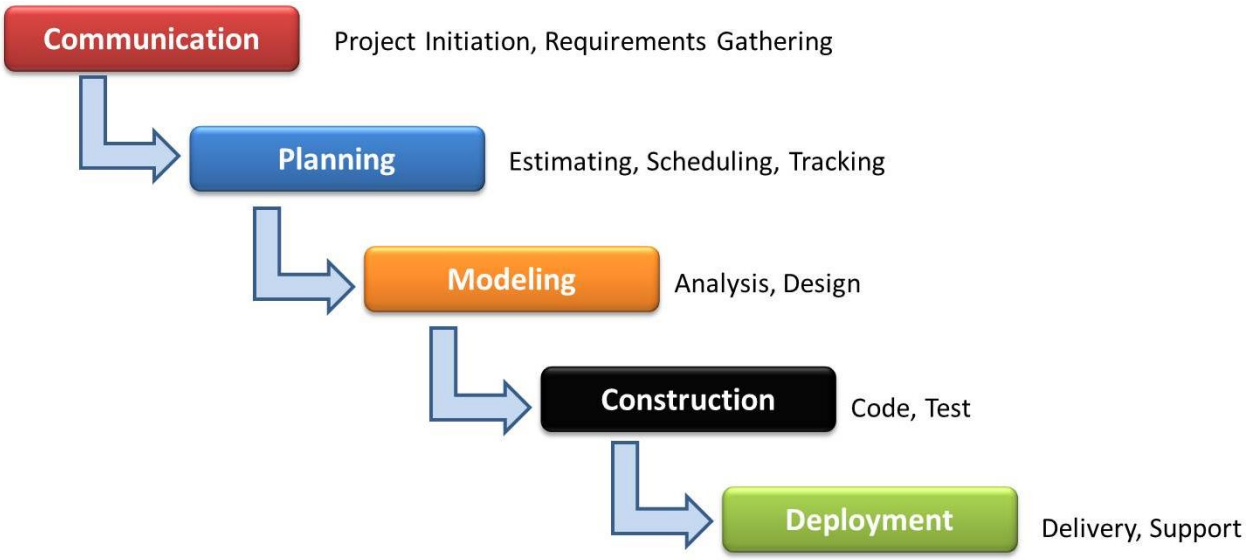


Figure 31 – Waterfall System Development Model

The reason why we choose this model is it runs sequentially, we move to next phase when previous phase is completed, it's to make sure that we will not miss or mess up anything during this project.

2. Roles and Responsibilities

No.	Full name	Role	Responsibilities
1	Do Duc Minh Quan	Supervisor	• Define business. • Support in technical issues.
2	Nguyen Minh Nhat	Programmer – Team Leader	• Outline works for members. • Assign works for members. • Answer question and deliver direction. • Support technique.

			• Buying hardware / components. • Encourage, maximize team's performance. • Manage project process. • Coding. • Prepare documents • GUI design.
2	Cao Dinh Nguyen Khoa	Programmer – Team Member	• Complete individual works on time. • Coding. • Prepare documents. • Support each other to complete team work. • Testing.

Table 13 – Roles and Responsibilities

3. Tools and Techniques

3.1 Development Environment

3.1.1 Hardware Environment

1. Arduino Uno R3.
2. Servo Motor SG90.
3. Ultrasonic Sensor SF-05.
4. DC Gear Motors.
5. H-bridge – L298N.
6. Laptop with appropriate configuration for embedded system development.

3.1.2 Software Environment

1. Arduino IDE: for C development.
2. Eclipse ADT: for android development.
3. StarUML: for create use classes, and diagrams.
4. Fritzing: For component sketch drawing.
5. Windows 8.1 x64 and Windows 7 x64.

3.2 Technique

- C, embedded programing, debugging in embedded system.
- Java, android programing.

3.3 Management Environment

- Google drive: source code, documents, reports.
- Microsoft SharePoint site:
    - Announcement.

- Task: assign task for member.
- Calendar: note about team meeting and teacher meeting.
- Document: storing documents, reports, source code.

3.4 Communication Environment

- Facebook group, Skype message group for online meeting.
- Newsfeed on Microsoft SharePoint.

III. Project Management Plan

1. Project Iteration

Phase / Iteration	Description	Deliverables	Resources Needed	Dependencies and Constraints
1. Initiating	- Project kick-off and introduction. - Requirement brainstorming. - Definition and solution.	- Determine hardware / components and tools	3 man days.	- N/A
2. Planning	- Scope management. - Time management. - Cost management. - Quality management. - Human resource management - Component management.	- Project plan.	4 man days.	- Finish initiating phase.
3. Study component's datasheets	- Arduino Uno R3 - DC Gear Motor - Servo Motor - Ultrasonic sensor - H Bridge L298N - Bluetooth HC-05	- Understand each component functionality and specification	10 man days.	- Finish planning phase

4. Develop servo control with Arduino Uno R3	- Implement code that allow Arduino Uno R3 can communicate and control servo	- Arduino Uno R3 successfully communicate and control servo.	10 man days	- Finish study component's datasheet phase
5. Develop ultrasonic sensor control with Arduino Uno R3	- Implement code that allow Arduino Uno R3 can communicate, control ultrasonic sensor to send and receive signal.	- Arduino Uno R3 successfully communication and control ultrasonic sensor.		- Finish study component's datasheet phase
6. Develop DC Gear Motor control with Arduino Uno R3	- Implement code that allow Arduino Uno R3 can control DC gear motor to go forward, go back, turn left and turn right.	- Successfully control DC gear motor to go forward, go back, turn left, and turn right.		- Finish study component's datasheet phase
7. Develop a combination of servo and ultrasonic sensor with Arduino Uno R3	- Implement code that allow servo and ultrasonic sensor works together as user required.	- Servo and ultrasonic sensor works together, archive user requirement.	3 man days	- Finish study component's datasheet phase
8. Develop Bluetooth communication with arduino	- Implement code that allow arduino read and send data from Bluetooth HC-05.	- Send and receive data successfully with arduino.	3 man days	Finish study component's datasheet phase
8. Assemble OAR	- Assemble OAR + Mount Ultrasonic sensor + Mount Servo motor + Mount H-bridge + Mount Arduino Uno + Mount battery case and battery - Assign pins for all components	- Successfully assembled OAR.	5 man day	Finish study component's datasheet phase
9. Develop avoid obstacle algorithm	- Develop avoid obstacle algorithm. - Implement avoid obstacle code without Servo motor. - Implement avoid obstacle code with combination of Servomotor and ultrasonic sensor	- OAR can avoid obstacle mechanism with Servo and Ultrasonic sensor.	10 man days	- Finish study component's datasheet phase
10. Develop path remembering algorithm	- Develop path-remembering algorithm.	- OAR can remember and tracing the old path.	10 man days	- Finish study component's datasheet phase
11. Develop Bluetooth control	- Develop android application that allow user to control OAR manually.	- Successfully control OAR manually	10 man days	-Finished obstacles avoidance and path tracing

	- Develop program that allow arduino to receive and process data that sent from android application			algorithm
10. Develop main program for OAR navigation and bug fixing	- Complete all individual code. - Combine all code. - Develop main program for OAR navigation. - Fixing bugs.	- OAR, auto avoid obstacle	60 man days	Finished obstacles avoidance and path tracing algorithm and Bluetooth control
11. Integration, acceptance and system test for OAR	- Create test plan. - Create test case. - Expected result.	- Test case. - Test plan. - Test result	10 man days	- Finish Develop main program for OAR and bug fixing
12. Monitoring and controlling	- Make sure project is going on right scope / direction.	- Project is going on right scope		- Finish planning phase

Table 14 – Project Iteration

2. Task Sheet

No	Product Deliverables	Task	NhatNM	KhoaCDN
1	Study Component's datasheet	**Arduino Uno R3.**	**O**	**O**
		+ Pins, connectors.		
		+ Operating voltage.		
		+ Integrated development environment (IDE).		
		+RAM, ROM, Flash Memory.		
		+ Input, output.		
		+ Communication.		
		+ Programming.		
		DC Gear Motor.	**O**	
		+ Functionality.		
		+ Pins.		
		+ Operating voltage.		
		+ Communication.		
		Servo Motor SG90.	**O**	
		+ Functionality.		
		+ Pins.		

		+ Operating voltage.		
		+ Communication.		
		Ultrasonic sensor SF-05.		**O**
		+ Functionality.		
		+ Pins.		
		+ Operating voltage.		
		+ Communication.		
		H bridge L298N.	**O**	
		+ Functionality.		
		+ Pins.		
		+ Operating voltage.		
		Bluetooth HC-05		**O**
		+ Functionality		
		+ Pins.		
		+ Operating voltage.		
		+ Communication.		
2	**Connect Servo motor, Ultrasonic sensor, DC Gear Motor, H – Bridge L298N to Arduino and collect raw value.**	**Servo Motor SG90.**	**O**	
		+ Connect to Arduino.		
		+ Control the motor to spin.		
		Ultrasonic sensor SF-05.		**O**
		+ Connect to Arduino.		
		+ Send signal.		
		+ Receive signal.		
		+ Process received signal and print out through Serial Monitor in Arduino IDE.		
		DC Gear Motor	**O**	
		+ Connect to H-bridge, Arduino.		
		+ Control through analog output.		
		+ Control through digital output		
3	**Assemble K-robot**	**Assemble robot car**	**O**	
		+ Mount Ultrasonic sensor.		
		+ Mount Servo motor.		
		+ Mount H-bridge.		
		+ Mount Arduino Uno.		
		+ Mount battery case and battery.		
		+ Connect pins for all components.		
4	**Test individual code on assembled robot**	**Test DC Motor**	**O**	
		+ Control DC Gear Motor by analog / digital output		

		Test Servo Motor	**O**	
		+ Control Servo motor to spin.		
		Test Ultrasonic Sensor		**O**
		+ Send / Receive signal from Ultrasonic sensor.		
5	**Develop obstacles avoidance algorithm**	**Develop obstacles avoidance algorithm**	**O**	
		+ Develop obstacles avoidance without servo motor		
		+ Develop obstacles avoidance with servo motor		
6	**Develop back-tracking algorithm**	**Develop back-tracking algorithm**		**O**
7	**Test and bug fixing**	**Test and bug fixing**	**O**	**O**
		+ Obstacles avoidance algorithm		
		+ Back-tracking algorithm		
8	**Bluetooth remote control**	**Smartphone side – Android application**	**O**	
		+ Develop android application that allow user can control OAR manually		
		Arduino side – Bluetooth control	**O**	**O**
		+ Receive data send from android application		
		+ Processing received data		

Table 15 – Task Sheet

IV. Coding Convention.

Reference C/C++ Coding Convention document can be found at:

http://msdn.microsoft.com/en-us/library/vstudio/ff926074.aspx

Reference Java Coding Convention document can be found at:

http://www.oracle.com/technetwork/java/codeconvtoc-136057.html

System Requirement Specification (SRS)

I. Software

1. Software Requirement

An android application using Bluetooth technology to send command or receive message to/from robot's Bluetooth module.

User can use application to control robot, switch from automatic mode to manual control mode.

In automatic mode, user cannot do anything, robot will running itself.

In manual mode, user is allowed to control robot manually. User can drive robot manually like:

- Drive OAR forward.
- Drive OAR backward.
- Drive OAR to the left.
- Drive OAR to the right.

2. GUI Requirement

GUI design must satisfied the following requirement:

- User friendly.
- Dedicated buttons to avoid confusing during normal use.
- Meet all main function requirements.
- Buttons:
    - Connect: Connect to OAR Bluetooth.
    - Disconnect: Disconnect from OAR Bluetooth.
    - Auto Mode: Activate automatic avoid obstacles mode.
    - Manual: Activate remote control mode.
    - Manual control button:
        - Forward: Drive OAR forward.
        - Backward: Drive OAR backward.
        - Left: Drive OAR left.
        - Right: Drive OAR right.

- Stop: Stop OAR.

Manual control buttons are image button, represent for five directions.

II. Hardware

Based on project requirement we have choose following hardware components.

1. Hardware Requirement

The hardware interface must satisfied the following requirement.

- Easy to replace.
- Low-cost module.
- Easy to implement

1.1 Arduino Uno R3.

To communicate with all other hardware components and processing value, we have to have a microcontroller, there are many kinds of microcontroller in the market nowadays. After evaluate project's requirement, we decide to choose Arduino Uno R3, it is microcontroller board based on the ATMega328 microcontroller.

It has 14 digital input / output pins, 6 of them can be used as PWM outputs (For controlling DC Motor).

Arduino Uno R3 is the latest Uno at the moment, in this latest board, it has:

- Stronger reset circuit.
- Dedicated reset button.
- ATmega 16U2 replace the 8U2.

The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX) for Bluetooth module communication.

Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

1.2 Ultrasonic Sensor HY SRF-05.

For detecting obstacle, we choose ultrasonic sensor instead of infrared radiation sensor because IR sensor has many weaknesses, they are not accuracy, cannot be used outside in the sun, narrow beam width, the infrared sensor reading will be different for different surfaces , different colors, and different shades even if the range is the same.

With ultrasonic sensor, it is accurate, has wider beam width, no matter what color obstacles are and ultrasonic sensor can detect obstacles up to more than 4 meter far.

Therefore, we will choose Ultrasonic Sensor HY SRF-05 because:

- It has reasonable price.
- Revolution step over SRF-04, increase max distance to over 4 meters, one more pin for operating mode, enhance accuracy.
- Compatible with arduino and other boards.
- Compact size.

1.3 Servo Motor SG90.

To increase robot flexibility in detect obstacles, we mount ultrasonic with servo motor, so now the robot's ultrasonic sensor can sweep from left to right, robot will not have to turn left or right for ultrasonic sensor in order to detect obstacles.

For that abstract, we will choose Servo Motor SG90 because:

- Enhance for obstacles detection.
- Component price over performance ratio is good, it can operate at a range of speeds without overheating.

1.4 DC Gear Motor.

With the robot, we have planned to implement many components, we choose dc gear motor instead of dc motor because:

- Gearbox increases the traction of dc motor, our robot car carries a lot of components so we need enough traction for our robot car able to moves and maintain stability either.

1.5 H-Bridge L298N.

GPIO of Arduino cannot drive dc motor that has current over 40mA, so we choose H Bridge L298N because:

- It is a full bridge driver, allow us to control two motor separately.
- Each channel handle load currents up to 3A.
- It has wide operating voltage, heat sink to enhance its durability, light weight and easy to setup and use with arduino.

1.6 Bluetooth HC-05.

For manual control and wireless data transfer, we choose module Bluetooth HC-05 because:

- Popularity.
- Wide range, up to 9 meter.
- Light weight, compact size.

- Fast data transfer rate.

1.7 Battery Wild Scorpion 11.1V – 1500mAh 35C

We choose Battery Wild Scorpion 11.1V - 1500mAh.

- To ensure our robot can operate for long time, and supply enough power for all hardware components.
- It's a Lithium Polymer battery:
    - Solid, light weight, has longer changing cycle than Li-ion battery.
    - Less internal resistance for faster discharging and higher amperes, at up to 65C continuously, or 135C in bursts.

2. Hardware Interface Requirement

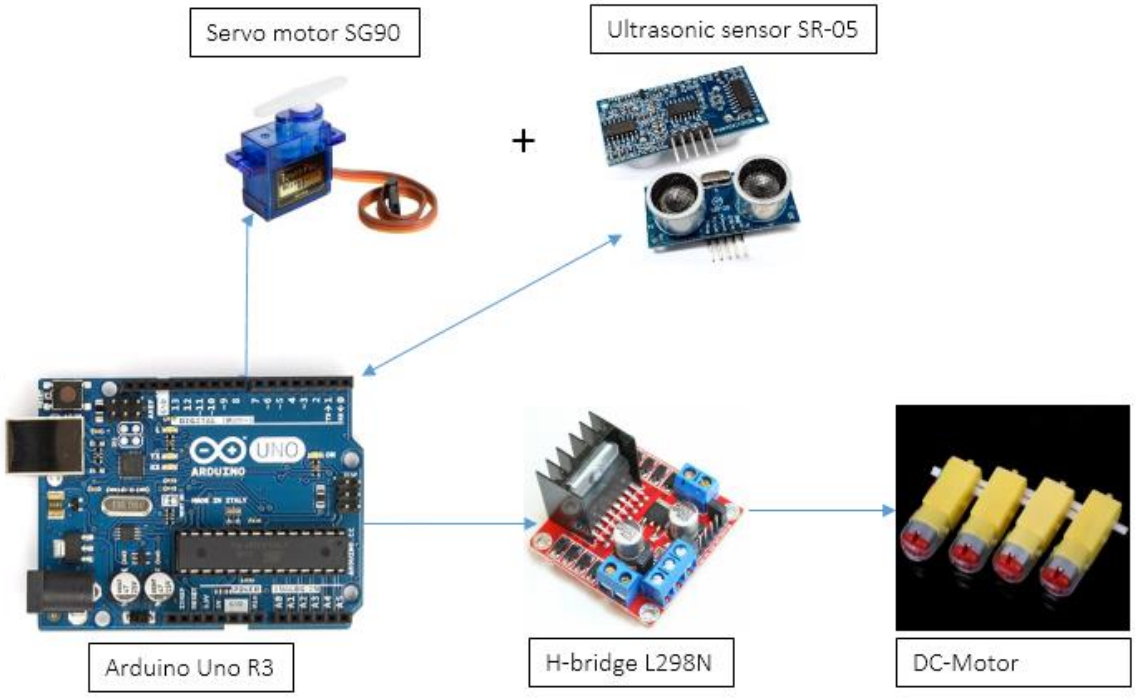


Figure 32 – Overall Hardware Interface

2.1 Hardware Interface

2.1.1 Ultrasonic Sensor Interface

Figure 33 – HY-SRF05 Interfaces

HY-SRF05		
Pin name	Type	Description
VCC	Input	Power supply pin
Trig	Output	TTL pulse
Echo	Input	Input TTL level signal and the range in proportion
Out	N/A	Connect to HIGH state, transceiver will acts as both transceiver and receiver.
GND	N/A	Ground

Table 16 – HY-SRF05 Interfaces

2.1.2 Servo Motor Interface

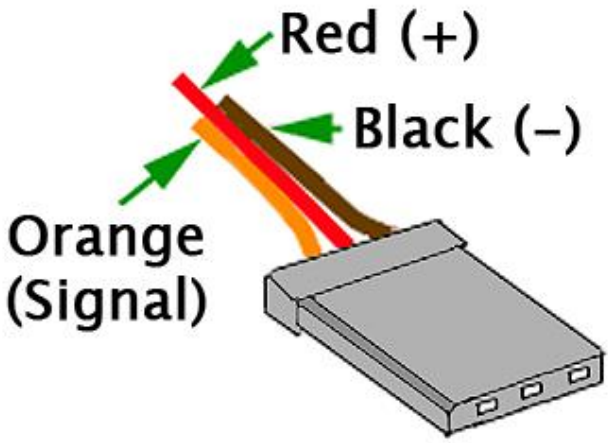


Figure 34 – SG90 Interfaces

Servo motor SG90		
Pin name	Type	Description
Red	Input	Power supply
Black	N/A	Ground
Orange	Output	Signal

Table 17 – Servo Motor SG90 Interfaces

2.1.3 Bluetooth HC-05

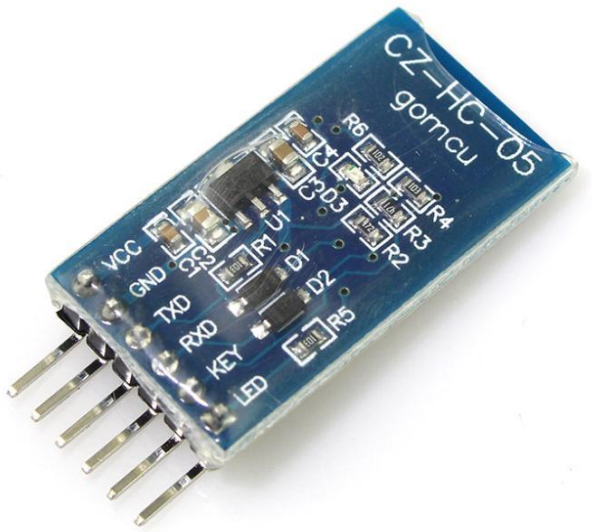


Figure 35 – Bluetooth HC-05 Interfaces

Bluetooth HC-05		
Pin name	Type	Description
VCC	Input	
GND	N/A	Ground
TXD	Output	Transmitter data line
RXD	Input	Receiver data line
KEY	Input	Change status to AT mode
LED	Input	Show current status

Table 18 – Bluetooth HC-05 Interfaces

2.1.4 H-Bridge L298N

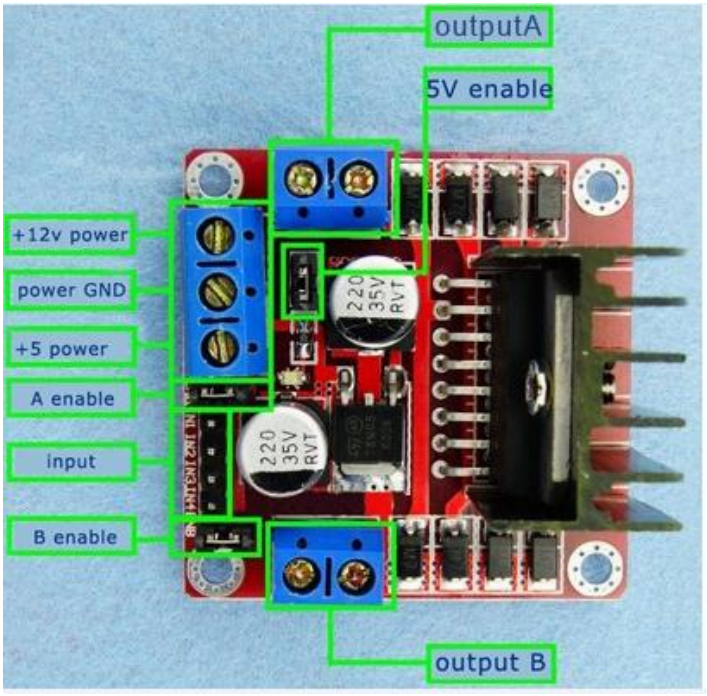


Figure 36 – H-Bridge L298N Interfaces

H-Bridge L298N		
Pin name	Type	Description
ENA	Input	TTL Compatible Enable Input of Bridge A
IN1	Input	TTL Compatible Input of Bridge A
IN2	Input	TTL Compatible Input of Bridge A
OUTPUT A	Output	Outputs of Bridge A
ENB	Input	TTL Compatible Enable Input of Bridge B
IN3	Input	TTL Compatible Input of Bridge B
IN4	Input	TTL Compatible Input of Bridge B
OUTPUT B	Output	Outputs of Bridge B
12V	Input	Power supply input
GND	N/A	Ground
5V	Output	Power supply output

Table 19 – H-Bridge L298N Interfaces

2.2 Hardware Communication Protocol

We communication between hardware component and board through GPIO pins.

General-purpose input/output (GPIO) is generic pin on an integrated circuit whose behavior, including whether it is an output pin or input pin, can be controlled by user at runtime.

2.2.1 Ultrasonic Sensor

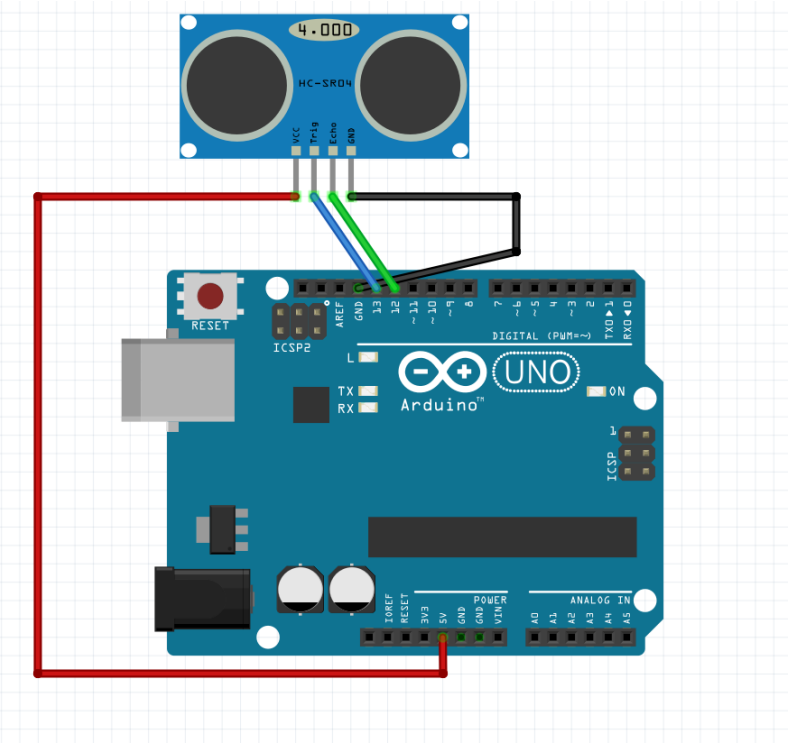


Figure 37 – HY-SRF05's Sketch

Note: This sketch using SRC04 module causes the SRF05 is not supported yet. We are using the same pins (VCC, Trig, Echo, and GND) in case of SRF05.

Pins of Ultrasonic Sensor SRF-05	Pins of Arduino Uno R3
VCC	5V
Trig	Pin 13
Echo	Pin 12
GND	GND

Table 20 – HY-SRF05 Connects to Arduino

2.2.2 Servo Motor

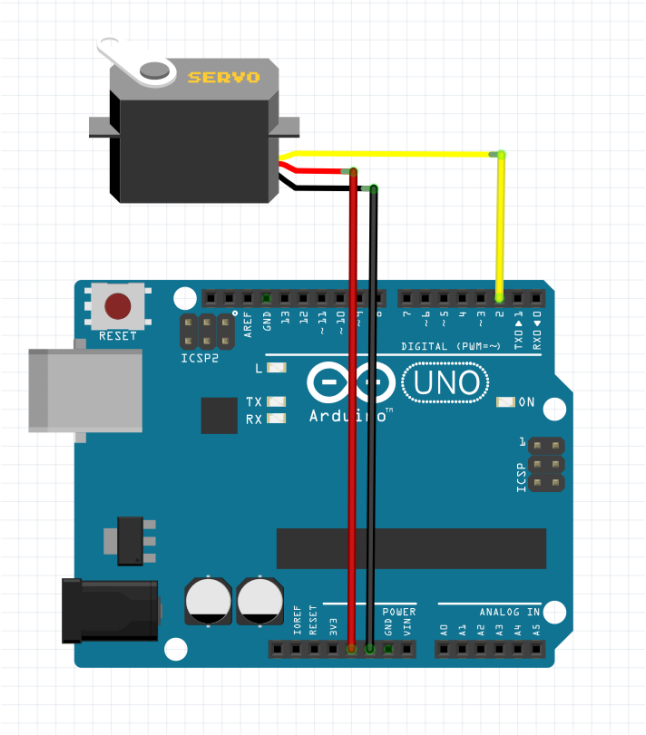


Figure 38 – Servo Motor SG90's Sketch

Pins of Servo Motor SG90	Pins of Arduino Uno R3
Yellow	Pin 2
Red	5V
Black	GND

Table 21 – Servo SG90 Connects to Arduino

Servo Motor SG90 has one pin as a signal pin.

- Signal pin: Controlling servo by PPM pulse, so we need to configure GPIO as output pin, which has ability of providing PPM pulse.

2.2.3 Bluetooth HC-05

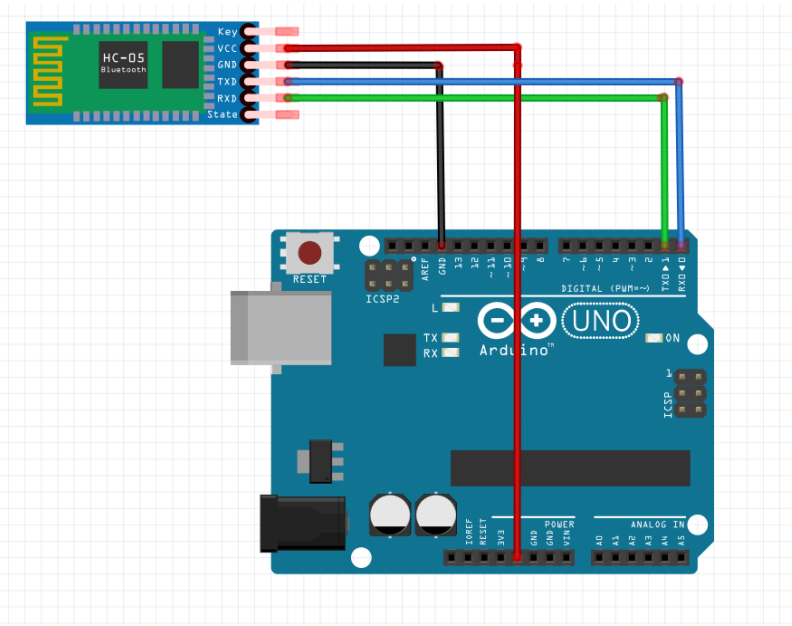


Figure 39 – Bluetooth HC-05's Sketch

Pins of Bluetooth HC-05	Pins of Arduino Uno R3
VCC – Red	5V
GND – Black	GND
TXD – Blue	RXD
RXD – Green	TXD

Table 22 – Bluetooth HC-05 Connects to Arduino

Bluetooth HC-05 has TX and RX pin for transmit and receive data. Therefore, we will need to configure our GPIO as UART communication.

- TXD: Transmitter data line, this pin is connect to GPIO on board which acts as RX pin.
- RXD: Receiver data line, this pin to connect to GPIO which acts as TX pin.

2.2.4 H-Bridge L298N

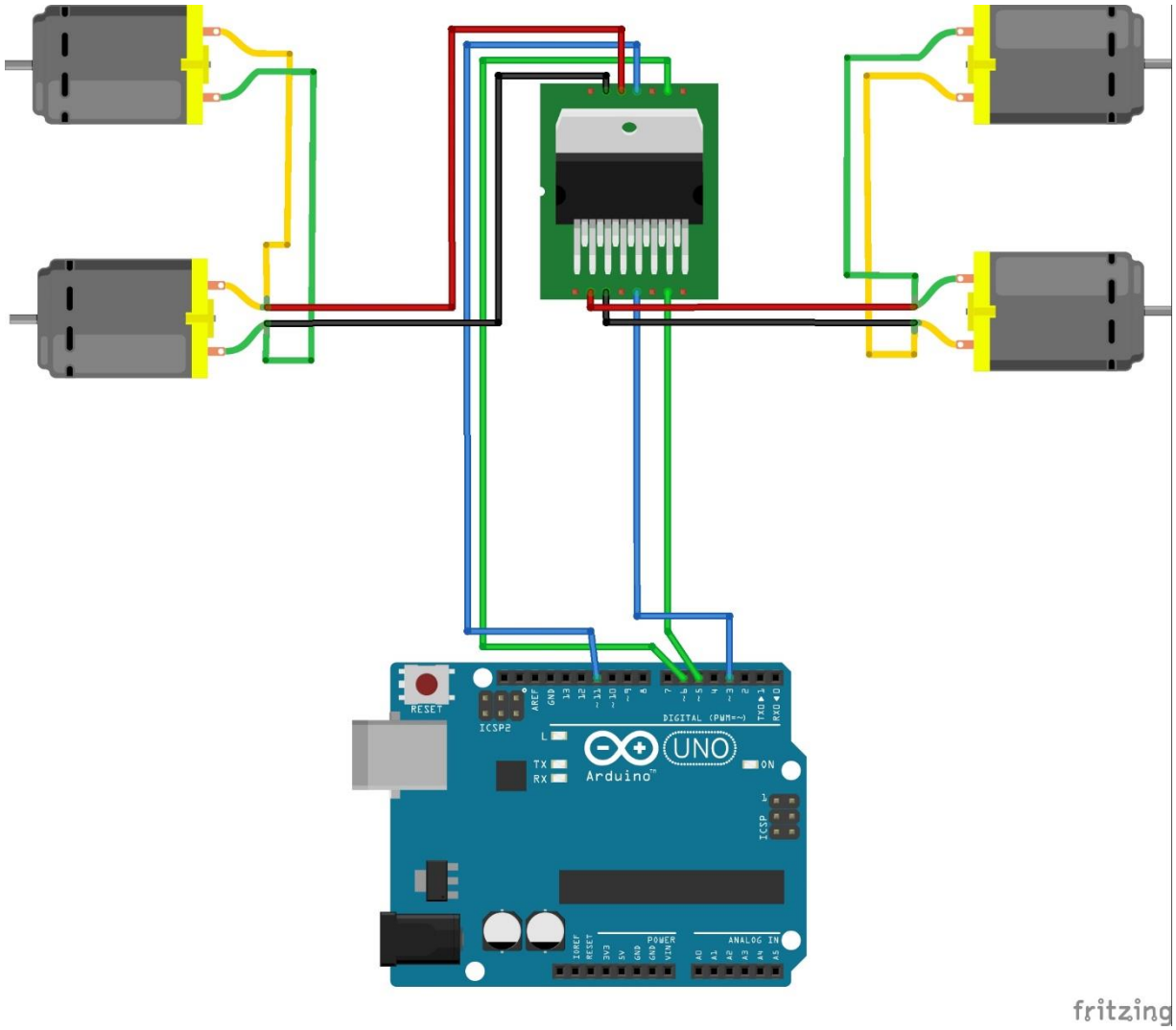


Figure 40 – H-Bridge L298N's Sketch

H-Bridge L298N	Arduino Uno R3
IN1	Pin 3
IN2	Pin 5
IN3	Pin 6
IN5	Pin 11

Table 23 – H-Bridge L298N Connects to Arduino

IN1: Motor A direction 1 PIN.
IN2: Motor A direction 2 PIN.
IN3: Motor B direction 1 PIN.
IN4: Motor B direction 2 PIN.

III. Controller

1. Controller Requirement

1.1 Automatic Obstacles Avoidance Mode.

With this mode, robot has ability to navigate autonomously while avoiding obstacles. Obstacles would be everything, every shapes, and every materials. The goal is that robot will not hit the obstacles on its path.

After a specific time, our robot is able to reroute, go back to the place where it started, it will have to go right on the previous path, which is the path that our robot has gone.

1.2 Manual Control Mode.

With this mode, user can control robot to go forward, go backward, turn left, and turn right manually using android application and connect to OAR via Bluetooth technology.

2. System Overall Use Case

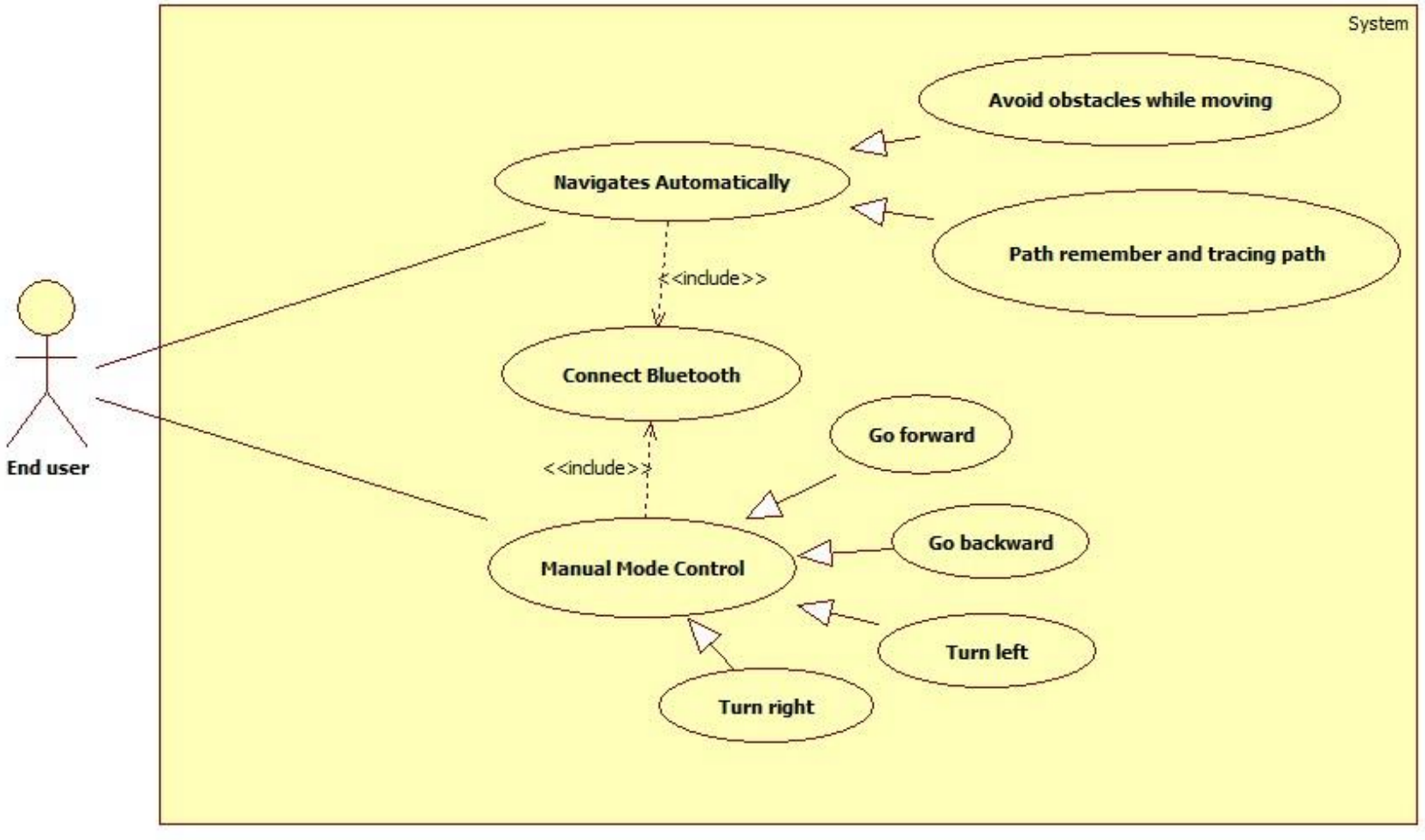


Figure 41 – System Overall Use Case

3. List of Use Cases

3.1 Connect Bluetooth

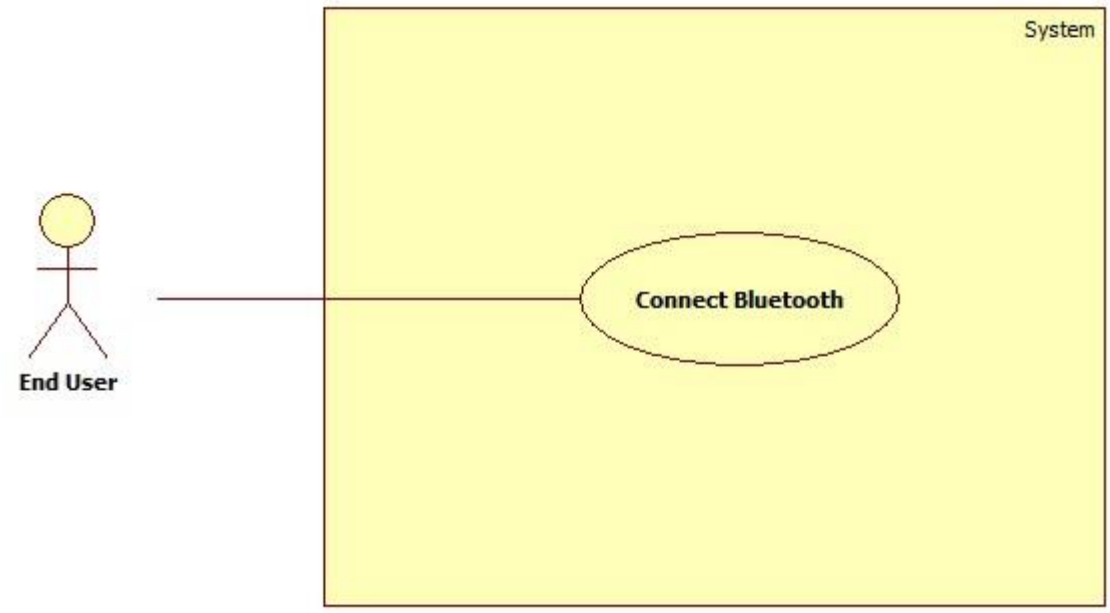


Figure 42 – Connect Bluetooth Use Case

3.1.1 Connect Bluetooth use case specification

USE CASE – UC001 – Connect Bluetooth			
Use Case No.	UC	**Use Case Version**	1.0
Use Case Name	Connect Bluetooth		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **User**

Summary:
This user case shows how to connect to OAR through Bluetooth.

Goal:
Android application successfully connect to OAR's Bluetooth.

Triggers:

- **User press** Connect **button.**

Preconditions:

- **Robot is power on.**
- **Bluetooth on Android smartphone is enabled.**
- **Paired with HC-05.**

Post Conditions:

- Success: **Android application is successfully connect to OAR's Bluetooth.**
- Fail: **Android application cannot connect to OAR's Bluetooth.**

Main Success Scenario:

Step	Actor Action	System Response
1	Go to phone setting.	Phone setting screen is shown.
2	Scan and pair HC-05 with paring code "1234".	Connected with HC-05.
3	Open Android Application.	Application is shown.
4	User click Connect button.	Connected message is shown. Led on HC-5 will blink every 2 second.

Alternative Scenario:

N/A

Exceptions:

No	Actor Action	System Response
1	Open Android Application.	Application is shown.
2	Power off OAR User click Connect button.	Green led on Arduino is off. "Connected" is not show.

Relationships: **Manual Mode Control, Navigates Automatically.**

Business Rules: **N/A**

Table 24 – Connect Bluetooth Use Case

3.2 Choose Manual Control Mode

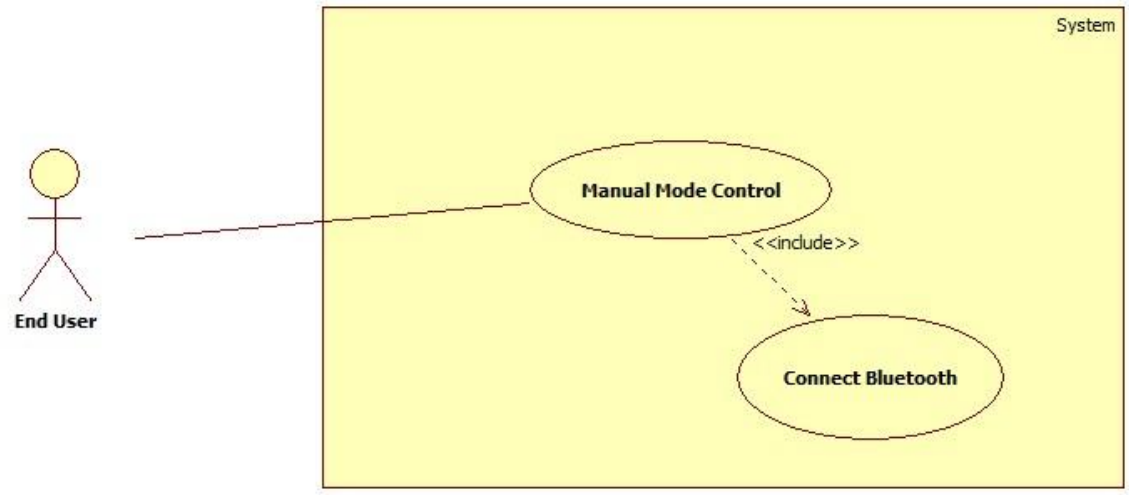


Figure 43 – Manual Mode Control Use Case

3.2.1 Choose manual control mode use case specification

USE CASE – UC002 – Choose Manual Control Mode			
Use Case No.	UC	**Use Case Version**	1.5
Use Case Name	Choose manual control mode		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **End user**

Summary:
This user case shows how to choose OAR manual control mode.

Goal:
OAR is switched to manual control mode.

Triggers:

- **User press** Manual Mode **button.**

Preconditions:

- **Robot is power on.**
- **Bluetooth is connected.**

Post Conditions:

- Success: **Robot successfully switched to manual control mode.**
- Fail: **Robot cannot switch to manual control mode.**

Main Success Scenario:

Step	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	User click Connect button.	Connected message is shown. Led on HC-5 will blink every 2 second.
3	User click Manual Mode button	"Manual Control activated" message is shown.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	Power off OAR. Click Connect button	Green led on Arduino is off. "Connected" is not show.

Relationships: **Connect Bluetooth, Go Forward, Go Backward, Turn Left, Turn Right**

Business Rules: **N/A**

Table 25 – Choose Manual Mode Control Use Case

3.3 Navigates Automatically

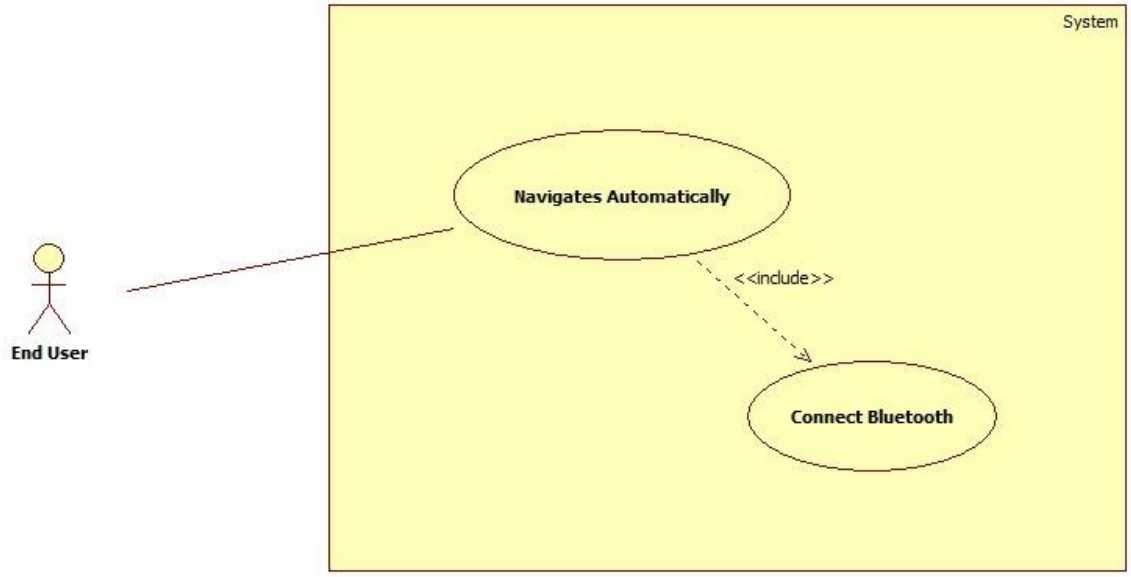


Figure 44 – Navigates Automatically Use Case

3.3.1 Navigates automatically use case specification

USE CASE – UC003 – Choose Navigates Automatically			
Use Case No.	UC	**Use Case Version**	1.5
Use Case Name	Choose manual control mode		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **End user**

Summary:
This user case shows how to choose OAR automatic navigation mode.

Goal:
OAR is switched to automatic navigation mode.

Triggers:

- **User press** Auto Mode **button.**

Preconditions:

- **Robot is power on.**
- **Bluetooth is connected.**

Post Conditions:

- Success: **Robot successfully switched to automatic navigation mode.**
- Fail: **Robot cannot switch to automatic navigation mode.**

Main Success Scenario:

Step	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	User click Connect button.	Connected message is shown. Led on HC-5 will blink every 2 second.
3	User click Auto Mode button	"Auto Mode activated" message is shown.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	Power off OAR. Click Connect button	Green led on Arduino is off. "Connected" is not show.

Relationships: **Connect Bluetooth**

Business Rules: **N/A**

Table 26 – Navigates Automatically Use Case

3.4 Go Forward

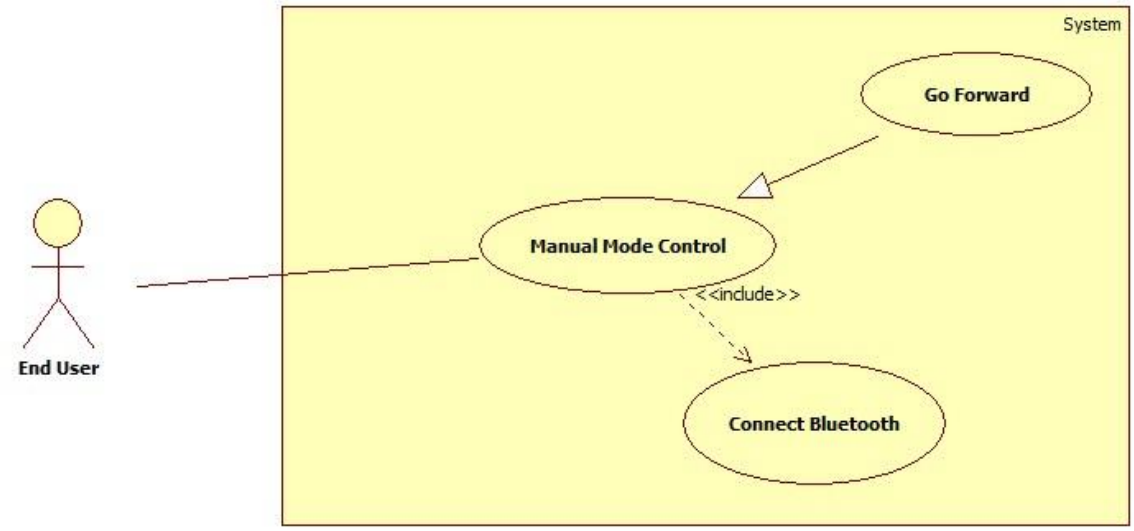


Figure 45 – Go Forward Use Case

3.4.1 Go forward use case specification

USE CASE – UC004 – Go Forward			
Use Case No.	UC	**Use Case Version**	1.5
Use Case Name	Go forward		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **System**

Summary:
This user case shows how OAR will go forward.

Goal:
Robot is moving forward

Triggers:

- **User press**  **button.**

Preconditions:

- **Robot is power on.**
- **Bluetooth is connected.**
- **Manual mode is chosen.**

Post Conditions:

- Success: **Robot successfully moving forward.**

- Fail: **Robot cannot move forward.**

Main Success Scenario:

Step	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	User click Connect button.	Connected message is shown. Led on HC-5 will blink every 2 second.
3	User click Manual Mode button	"Manual Control activated" message is shown.
4	User click  button.	Robot is moving forward. "Forward" message on application is shown.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	Power off OAR. User click Connect button.	Green led on Arduino is off. "Connected" is not show.

Relationships: **Connect Bluetooth, Manual Mode Control**

Business Rules: **N/A**

Table 27 – Go Forward Use Case

3.5 Go Backward

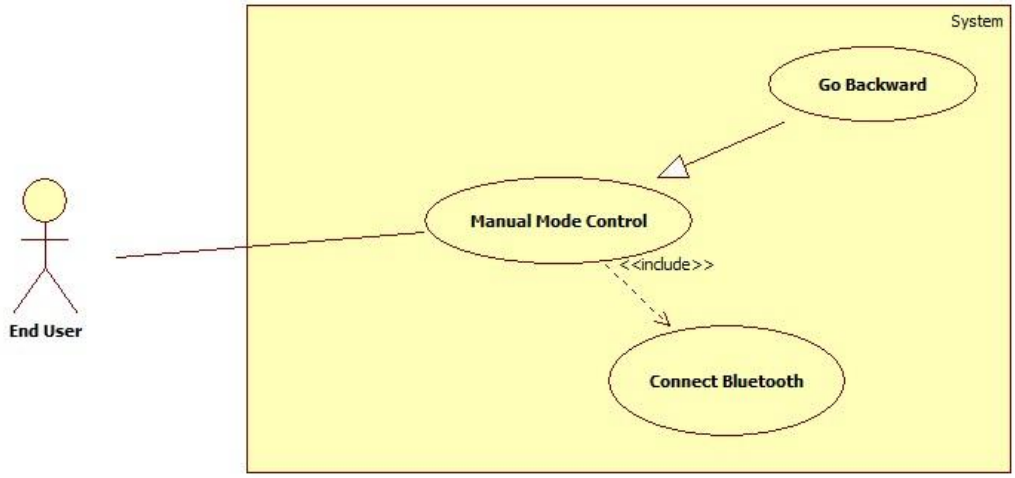


Figure 46 – Go Backward Use Case

3.5.1 Go backward use case specification

USE CASE – UC005 – Go Backward			
Use Case No.	UC	**Use Case Version**	1.5
Use Case Name	Go backward		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **System**

Summary:
This user case shows how Robot will go backward.

Goal:
Robot is moving backward

Triggers:

- **User press**  **button.**

Preconditions:

- **Robot is power on.**
- **Bluetooth is connected.**
- **Manual mode is chosen**

Post Conditions:

- Success: **Robot successfully moving backward.**

- Fail: **Robot cannot move backward.**

Main Success Scenario:

Step	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	User click Connect button.	Connected message is shown. Led on HC-5 will blink every 2 second.
3	User click Manual Mode button	"Manual Control activated" message is shown.
4	User click  button.	Robot is moving forward. "Backward" message on application is shown.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	Power off OAR. User click Connect button.	Green led on Arduino is off. "Connected" is not show.

Relationships: **Connect Bluetooth, Manual Mode Control**

Business Rules: **N/A**

Table 28 – Go Backward Use Case

3.6 Turn Left

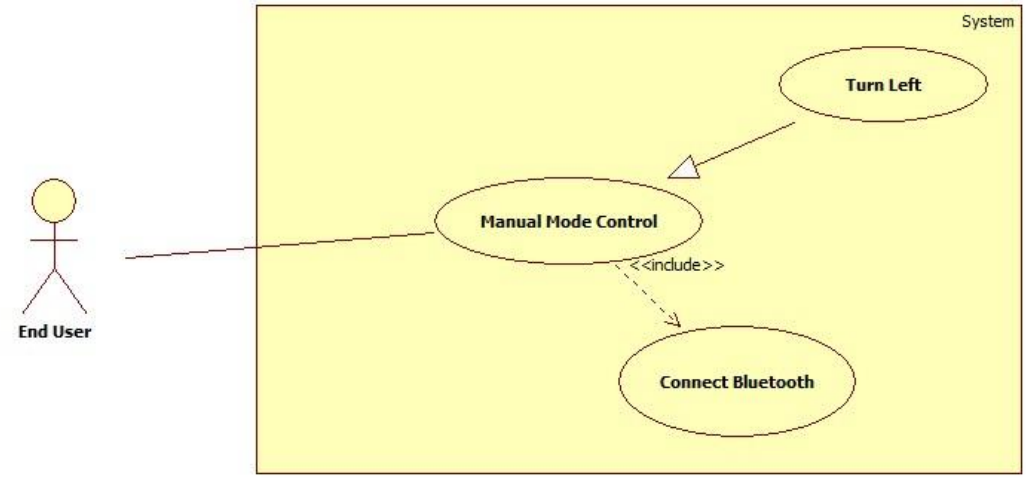


Figure 47 – Turn Left Use Case

3.6.1 - Turn left use case specification

USE CASE – UC006 – Turn Left			
Use Case No.	UC	**Use Case Version**	1.5
Use Case Name	Turn left		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **System**

Summary:
This user case shows how Robot will turn left.

Goal:
Robot will turn left.

Triggers:

- **User press**  **button.**

Preconditions:

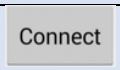
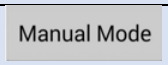
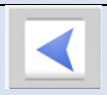
- **Robot is power on.**
- **Bluetooth is connected.**
- **Manual mode is chosen**

Post Conditions:

- Success: **Robot successfully turning left.**

- Fail: **Robot cannot turn left.**

Main Success Scenario:

Step	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	User click  button.	Connected message is shown. Led on HC-5 will blink every 2 second.
3	User click  button	"Manual Control activated" message is shown.
4	User click  button.	Robot is moving forward. "Left" message on application is shown.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	Power off OAR. User click  button.	Green led on Arduino is off. "Connected" is not show.

Relationships: **Connect Bluetooth, Manual Mode Control**

Business Rules: **N/A**

Table 29 – Turn Left Use Case

3.7 Turn right

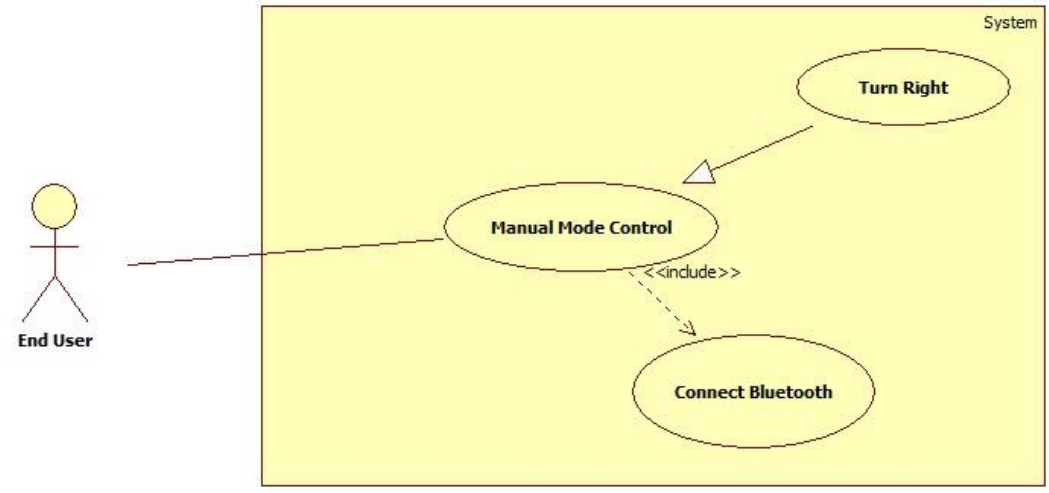


Figure 48 – Turn Right Use Case

3.7.1 Turn right use case specification

USE CASE – UC007 – Turn Right			
Use Case No.	UC	**Use Case Version**	1.5
Use Case Name	Turn right		
Author	Nguyen Minh Nhat		
Date	03/10/2014	**Priority**	Normal

Actor: **System**

Summary:
This user case shows how Robot will turn right.

Goal:
Robot will turn right.

Triggers:

- **User press**  **button.**

Preconditions:

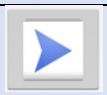
- **Robot is power on.**
- **Bluetooth is connected.**
- **Manual mode is chosen.**

Post Conditions:

- Success: **Robot successfully turning right.**

- Fail: **Robot cannot turn right.**

Main Success Scenario:

Step	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	User click Connect button.	Connected message is shown. Led on HC-5 will blink every 2 second.
3	User click Manual Mode button	"Manual Control activated" message is shown.
4	User click  button.	Robot is moving forward. "Right" message on application is shown.

Alternative Scenario: N/A

Exceptions:

No	Actor Action	System Response
1	User open the application.	Application is appeared. Main layout is shown.
2	Power off OAR. User click Connect button.	Green led on Arduino is off. "Connected" is not show.

Relationships: **Connect Bluetooth, Manual Mode Control**

Business Rules: **N/A**

Table 30 – Turn Right Use Case

4. Nonfunctional Requirement

4.1. Reliability.

4.1.1 Arduino Uno R3

- Latest revision, fixed almost issue from previous version.
- USB overcurrent protection, if more than 500 mA is applied to USB port, the fuse will automatically break the connection until the short of overloaded is removed.
- Automatic software reset, one of the hardware flow control lines (DTR) of theATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100-

nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment.

- Several of I/O ports.
- Made in Italy.
- RoHS compliant.

4.1.2 Ultrasonic Sensor SR-05

- Fast boot up time 500mS after power on or reset.
- Osc. Accuracy 1% factory calibrated.
- Range accuracy + / - 1 inch or + / - 10 microseconds.

4.1.3 H-bridge L298N

- Over temperature protection.
- Heat sink to enhance its durability, to make sure component can work for long time.

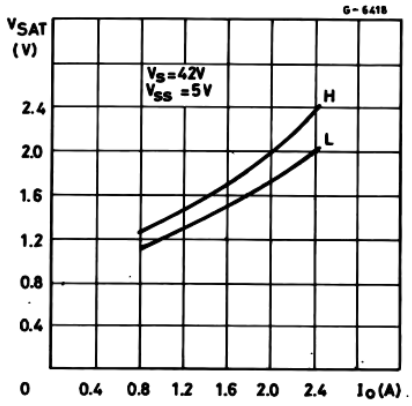


Figure 49 – Typical Saturation Voltage vs Output Current.

4.1.4 Servo Motor SG90

- Light weight.
- Budget component.

4.1.5 DC Gear Motor

- N/A

4.2 Availability.

- All component can be found at local hardware store and easy to buy.
- All component can be replace by ones, which have similar function.

4.3 Security

N/A

4.4. Maintainability.

- Easy to replace one component by another when it is broken.
- With Arduino, we can easily maintain and extend more features in the future or we can switch to another powerful Arduino board without any problem.

4.5 Portability.

N/A

4.6 Performance.

4.6.1 Arduino Uno R3

- ATmega328 ARM based with 16 MHz clock speed.
- 14 digital I/O pins – 6 provides PWM output.
- 6 analog I/O pins.
- 32KB Flash memory – 0.5KB for bootloader.
- 2KB SRAM.
- 1KB EEPROM.

4.6.2 Ultrasonic Sensor SR-05

- Fast boot up time 500mS after power on or reset.
- Osc. Accuracy 1% factory calibrated.
- Range accuracy + / - 1 inch or + / - 10 microseconds.

4.6.3 H-bridge L298N

- Heat sink to enhance its durability, to make sure component can work for long time.
- Has two channel, control up to 2 motor separately and up to 2A per channel.
- Driver L298 with two H-bridges.

4.6.4 Servo Motor SG90

- Operating Speed at 4.8 volts (no load) = 0.12 sec/ 60 degrees
- Stall Torque at 4.8 volts = 16.7 (1.2 kg/cm)

4.6.5 DC Gear Motor

- Motor speed: 100 rpm / min.
- Torque: 4500g / CM.

System Design Description (SDD)

This document describes a general view of system, which includes:

- System overall architecture.
- Hardware component diagram.
- The detailed description of components mechanism and the relationship between each component.
- Sequence diagram to know how to the system run.
- User Interface design which describes system's screens and how users interact to systems. Each screen includes functions, types of input/ output and event handling.
- How to control and communicate with each component.
- Main algorithm including obstacles avoidance algorithm and backtracking algorithm.

I. System Architecture Design

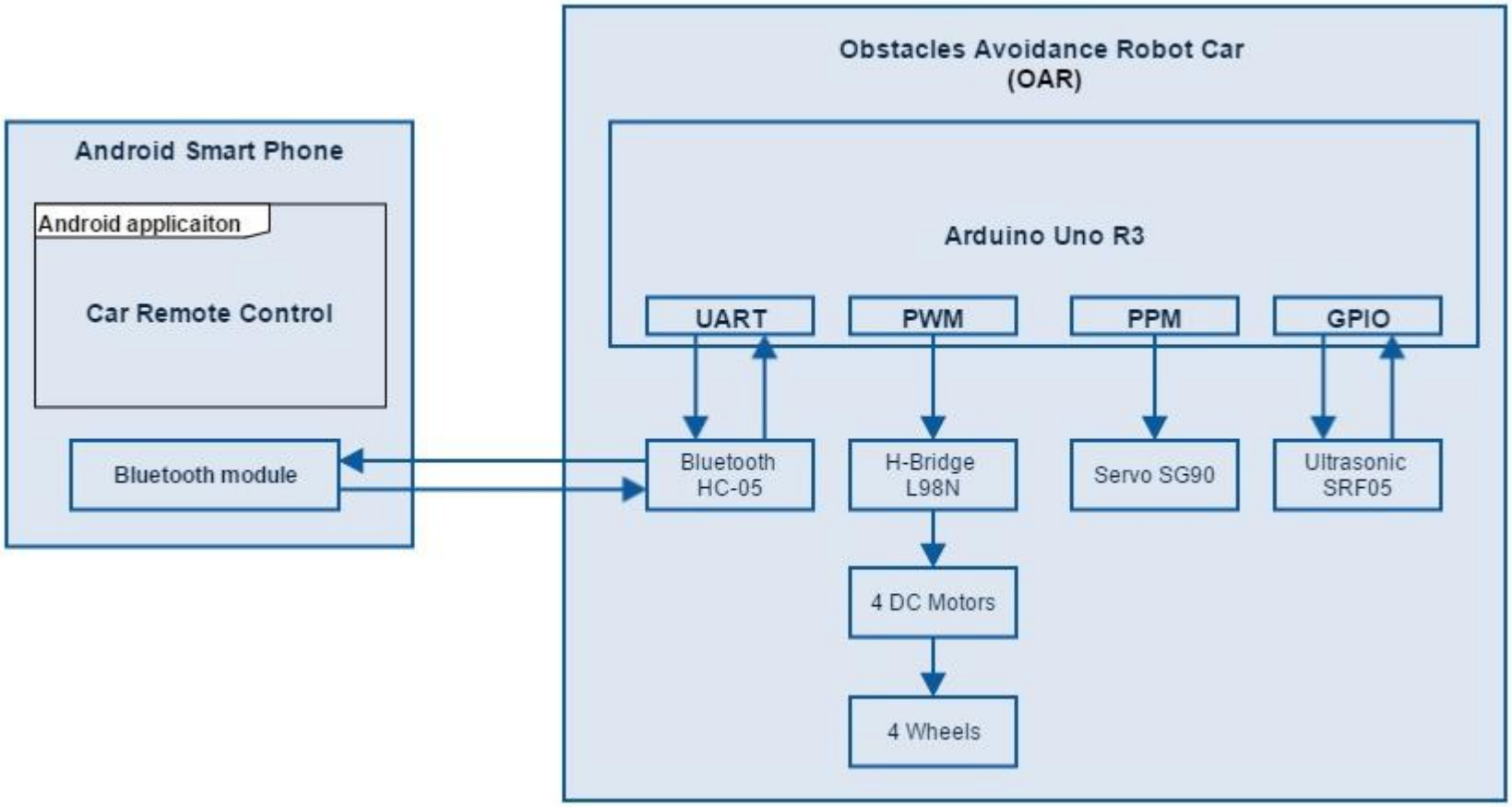


Figure 50 – System Architecture Design

II. System Component Diagram

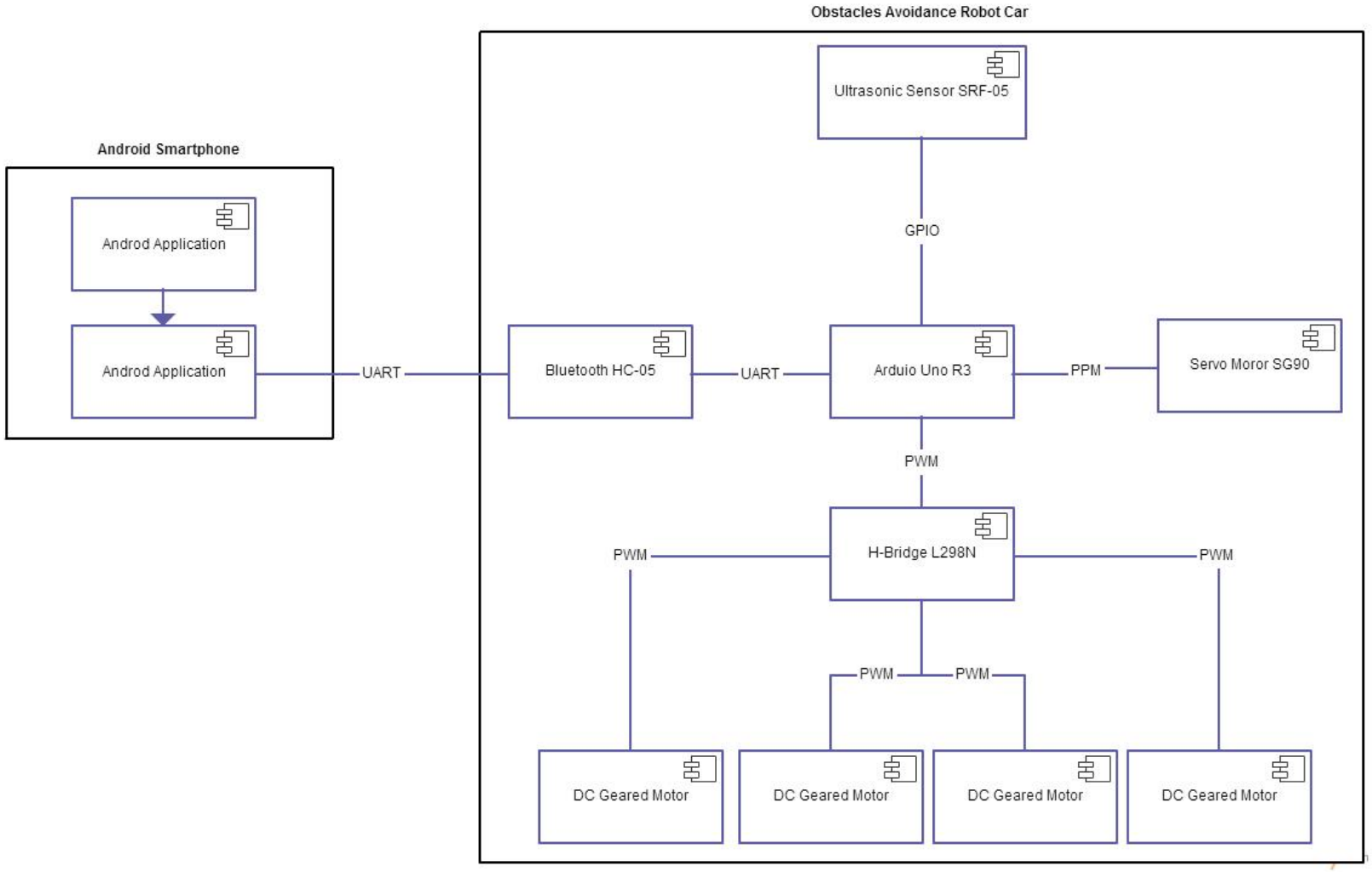


Figure 51 – Component Diagram

III. Software

1. User Interface Design

Main application screen.

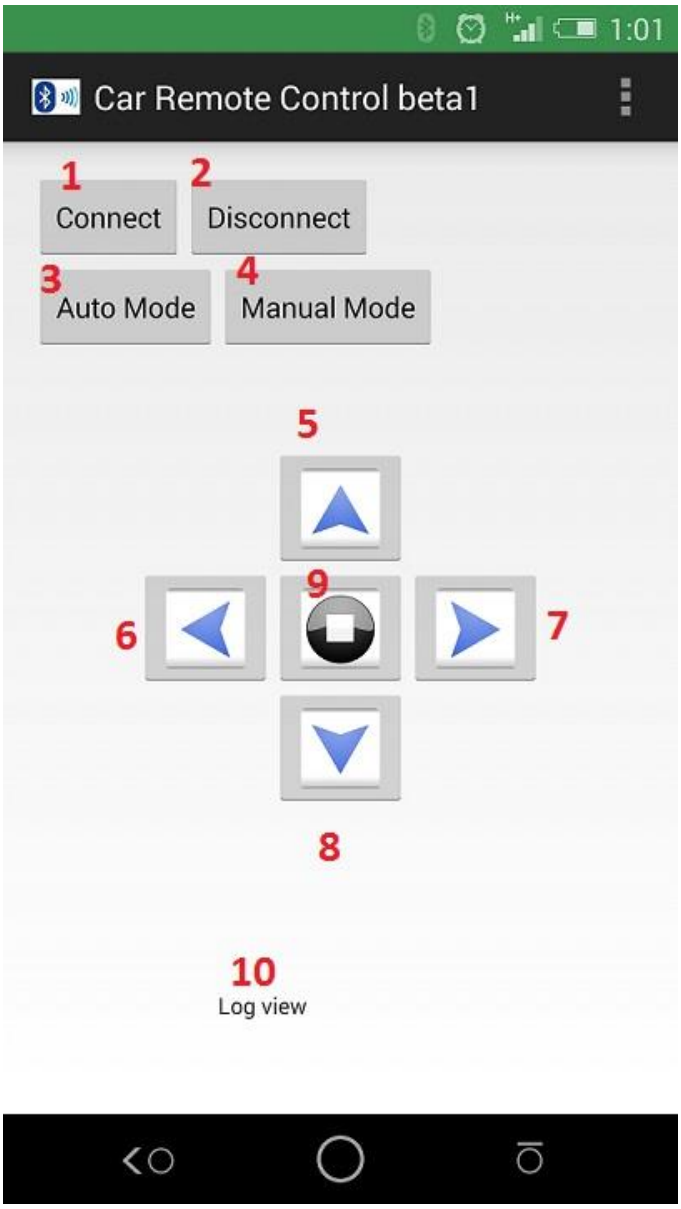


Figure 52 – Android Application

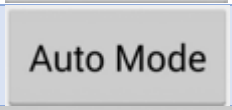
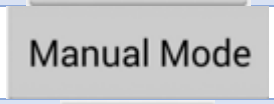
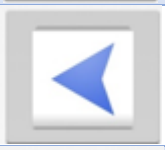
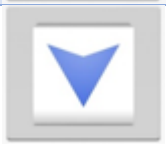
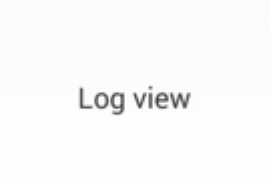
No.	Button	Function
1		Touch to connect to Arduino through Bluetooth module
2		Touch to disconnect from Arduino
3		Touch to switch Robot Car to auto navigation mode
4		Touch to switch Robot Car to manual control mode
5		Touch to control Robot Car to goes forward
6		Touch to control Robot Car to goes left
7		Touch to control Robot Car to goes right
8		Touch to control Robot to goes backward
9		Touch to stop Robot Car
10		Display value that send back from Arduino

Table 31 – Android Application Description

IV. Hardware

1. Hardware Component Diagram

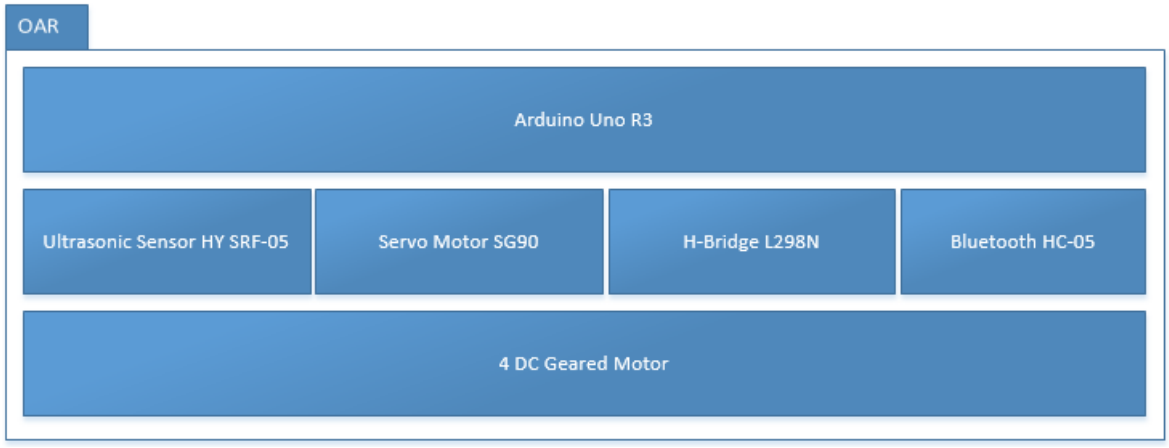


Figure 53 – Hardware Component Diagram

2. Hardware Details

2.1 Ultrasonic Sensor Working Principle

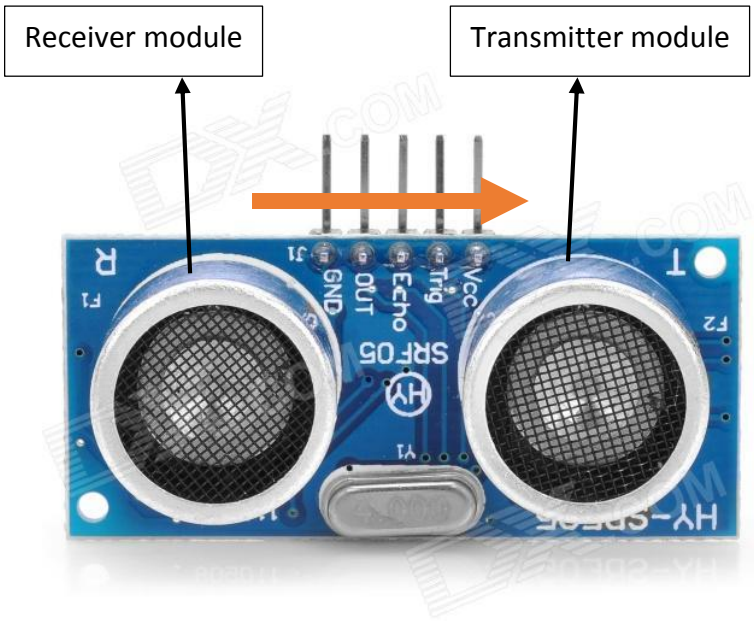


Figure 54 – Ultrasonic Sensor HY-SRF05

From left to right we have:

- GND – Ground pin.
- OUT – We have not used this pin.
- Echo – Echo pin.
- Trig – Trigger pin.
- Vcc – Power supply pin.

For ultrasonic sensor to work, first, generate a short 10 uS (microsecond) pulse to the Trigger Pin to start the sensor and it will beam out an 8 cycle burst of ultrasound at 40Khz, after 10 uS, the sensor automatic raise its echo line to HIGH.

Then it will listen for an echo pulse. As soon as it detects one, it lowers the echo line to LOW. If nothing is detected, the module will lower its echo line anyway after about 30 mS (millisecond).

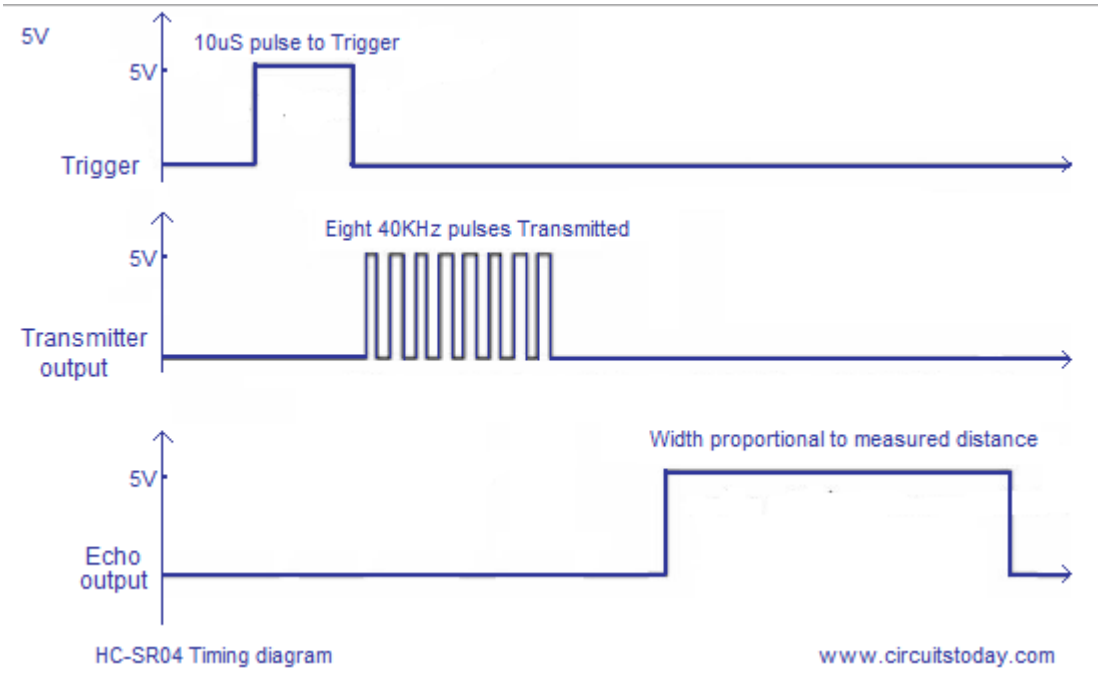


Figure 55 – HC-SR04 Timing Diagram

With SRF-05, to measure distance trigger pin will send a beam with eight 40 kHz ultrasonic-wave out in conical form, echo pin will detect reflected wave, so we will have the time wave travel from sensor to object, by that, we can calculate the distance.

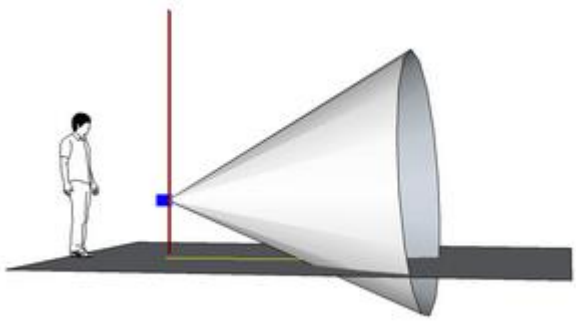


Figure 56 – Ultrasonic Pulse – Conical Form

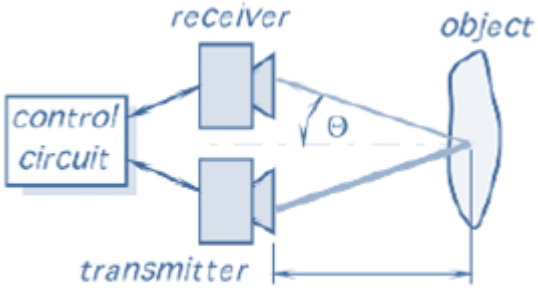


Figure 57 – Reflected Wave When First Ultrasonic Wave Hits the Obstacles

2.2 Servo Motor Working Principle

Servo mainly consist of 4 basis components:

- DC Motor.
- Gears.
- Potentiometer.
- A circuit.

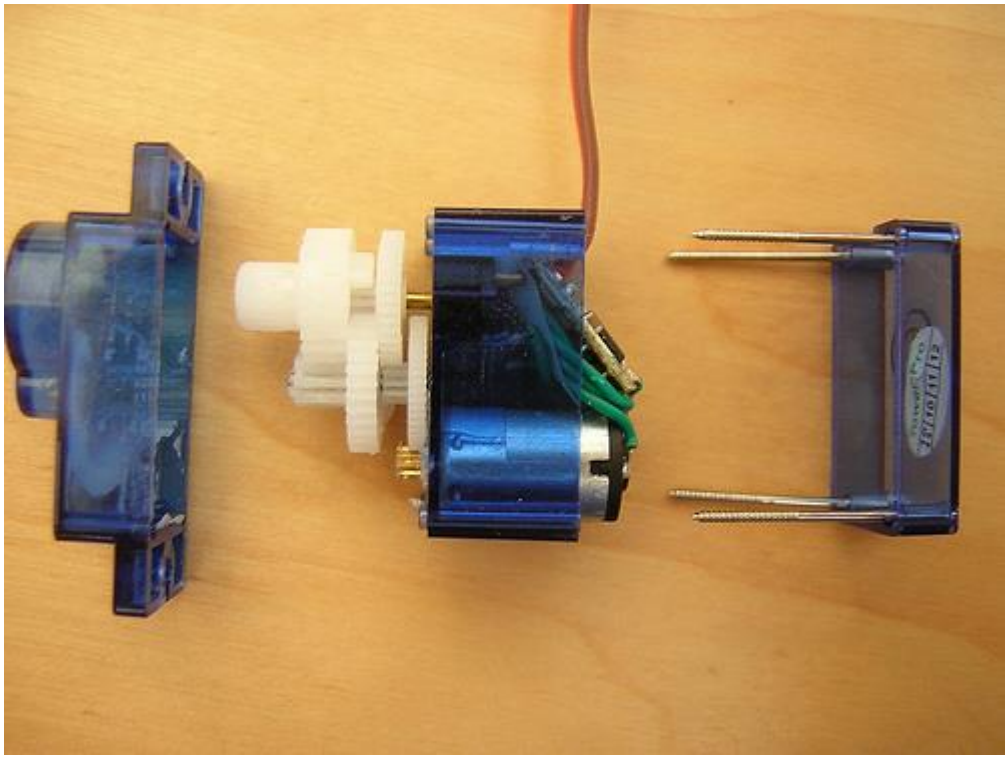


Figure 58 – Servo Motor SG90 Disassembly

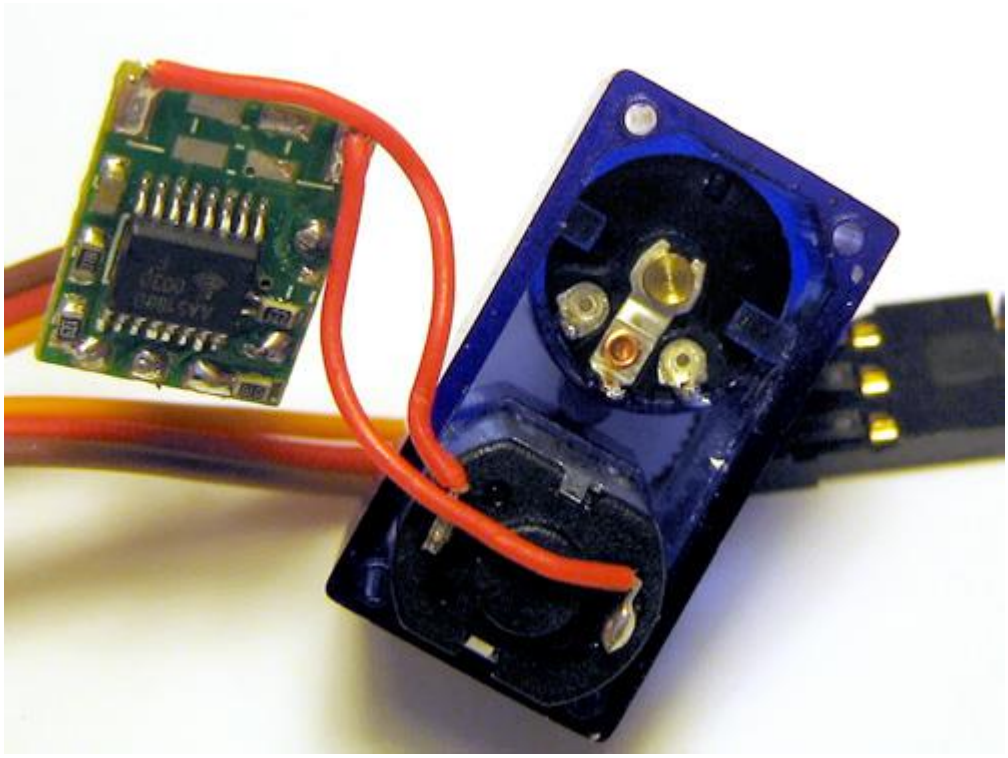


Figure 59 – Servo Motor SG90's Main Components

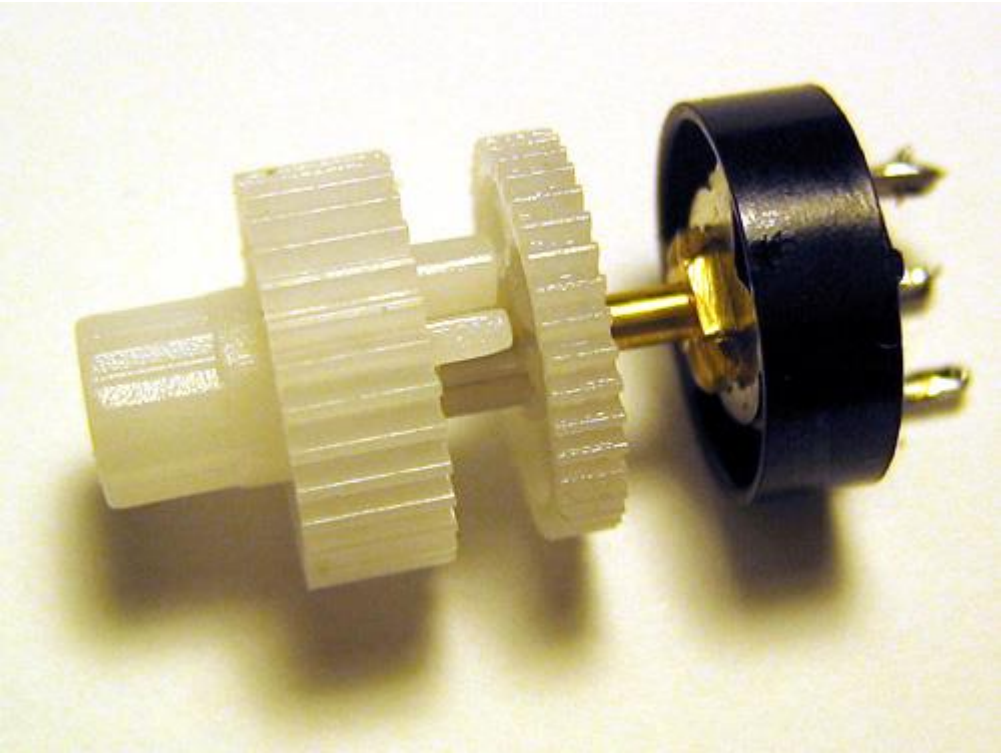


Figure 60 – Servo Motor SG90's Cog Wheel

A servo motor is basically a DC motor along with some other special purpose components that make a DC motor a servo. In a servo unit, it has a small DC motor, a potentiometer, gear arrangement and an intelligent circuitry. The intelligent circuitry along with the potentiometer makes the servo to rotate according to our wishes.

A DC motor will rotate with high speed but the torque generated by its rotation will not be enough to move even a light load. This is where the gear system inside a servomechanism comes into picture. The gear mechanism will take high input speed of the motor (fast) and at the output, we will get an output speed which is slower than original input speed but more practical and widely applicable.

Say at initial position of servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. This output port of the potentiometer is connected with one of the input terminals of the error detector amplifier. An electrical signal is given to another input terminal of the error detector amplifier. Now difference between these two signals, one comes from potentiometer and another comes from external source, will be amplified in the error detector amplifier and feeds the DC motor. This amplified error signal acts as the input power of the dc motor and the motor starts rotating in desired direction.

As the motor shaft progresses the potentiometer knob also rotates as it is coupled with motor shaft with help of gear arrangement. As the position of the potentiometer knob

changes there will be an electrical signal produced at the potentiometer port. As the angular position of the potentiometer knob progresses the output or feedback signal increases.

After desired angular position of motor shaft the potentiometer knob is reaches at such position, the electrical signal generated in the potentiometer becomes same as of external electrical signal given to amplifier. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer. As the input signal to the motor is nil at that position, the motor stops rotating. This is how a simple conceptual servo motor works.

Three wires control servo motor SG90 module: ground (black), power (red) and pulse (orange).

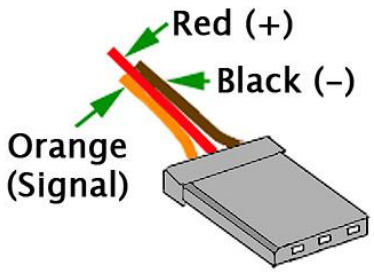


Figure 61 – Three Wires of Servo Motor

2.3 DC Geared Motor Working Principle

A DC motor will rotate with high speed but the torque generated by its rotation will not be enough to move even a light load. Therefore, we will need something to decrease its speed and enhance the traction.

Figure 62 – DC Gear Motor and Wheel

This is called DC Geared Motor, which means DC motor with gearbox, the yellow part is the gear box. The transmission ratio of this gearbox is 1:48.

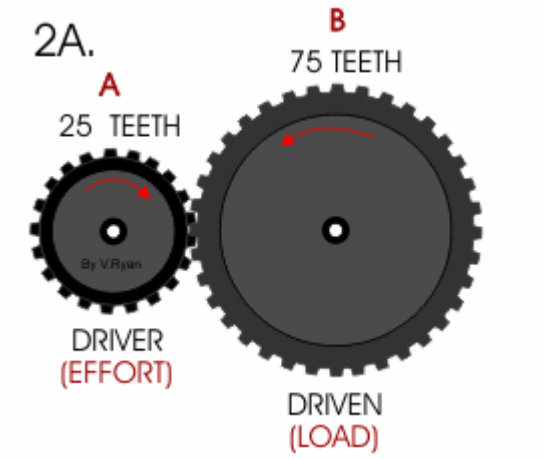


Figure 63 – Gear Box Mechanism

The transmission ratio: example we have two wheels, the A gear has 25 cogwheels, the B gear has 75 cog-wheels.

When we attach two wheels together and spin, after gear A spin 3 rounds, gear B has just spin one round.

So we say the transmission rate of first wheel and second wheel is 3:1, at this time traction of first wheel will be greater 3 times than second wheel. Conclusion, transmission ratio is the ratio of the rotation speed of primary cylinder to secondary cylinder in the gearbox.

An object effected by a force that make it spin on its axis, at this time, torque is generated.

Torque is a quantitative represent for its traction and acceleration. Our robot carries many components, so we need a motor and gearbox that generate enough torque in order to make the robot moving.

2.4 H-Bridge L298N Working Principle

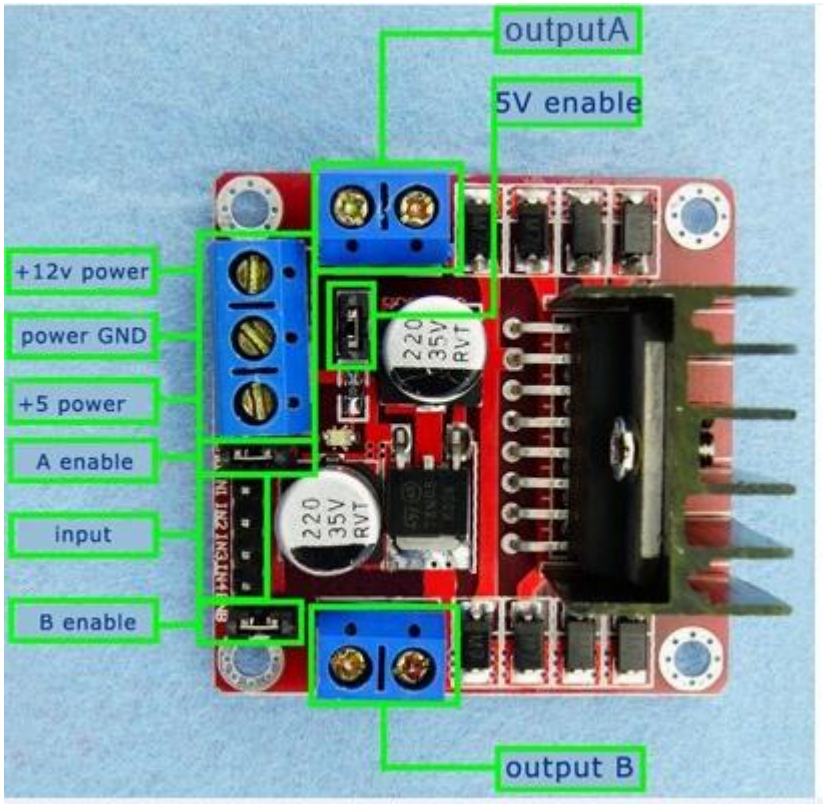


Figure 64 – H-bridge L298N Usage

Pin name	Type	Description	
ENA	Input	TTL Compatible Enable Input of Bridge A	- ENA connected to HIGH state will enable OUTPUT A. - IN1 to 5V - IN2 to GND => Motor A will move clockwise - IN1 to GND - IN2 to 5V => Motor A will move anti-clockwise
IN1	Input	TTL Compatible Input of Bridge A	
IN2	Input	TTL Compatible Input of Bridge A	
OUTPUT A	Output	Outputs of Bridge A	

ENB	Input	TTL Compatible Enable Input of Bridge B	- ENB connected to HIGH state will enable OUTPUT B. - IN3 to 5V - IN4 to GND => Motor B will move clockwise - IN3 to GND - IN4 to 5V => Motor B will move anti-clockwise
IN3	Input	TTL Compatible Input of Bridge B	
IN4	Input	TTL Compatible Input of Bridge B	
OUTPUT B	Output	Outputs of Bridge B	
12V	Input	Power supply input	N/A
GND	N/A		N/A
5V	Output	Power supply output	Supply 5V for other components.

Table 32 – H-bridge L298N Usage

It has two bridges, one on the left side and one on the right of the chip, its purpose is to control 2 motors individually.

L298 is a dual full bridge driver that has a wide operating voltage range and can handle load currents up to 3A. The IC also features low saturation voltage and over temperature protection.

H-Bridge is a circuit has 4 switches are suffering under the H-shaped.

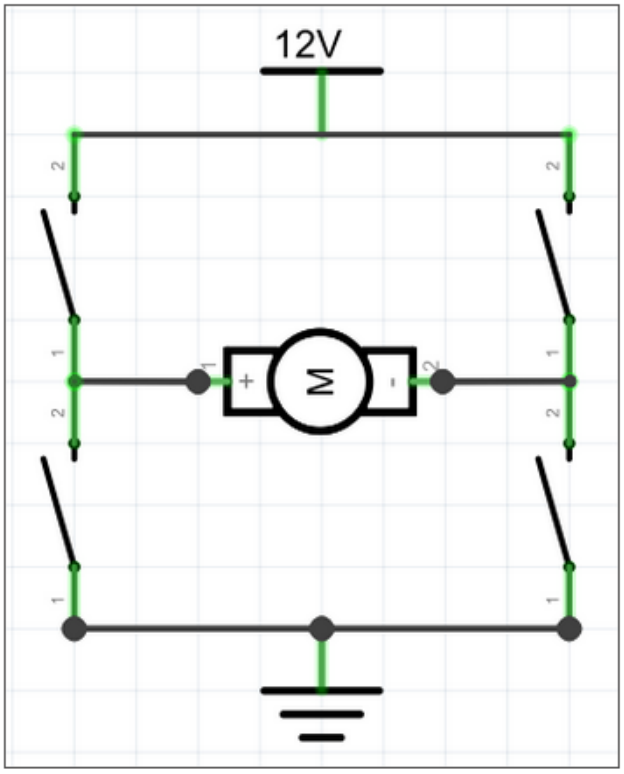


Figure 65 – H-Bridge

By controlling these switches to open or close, we can control current flow over motor.

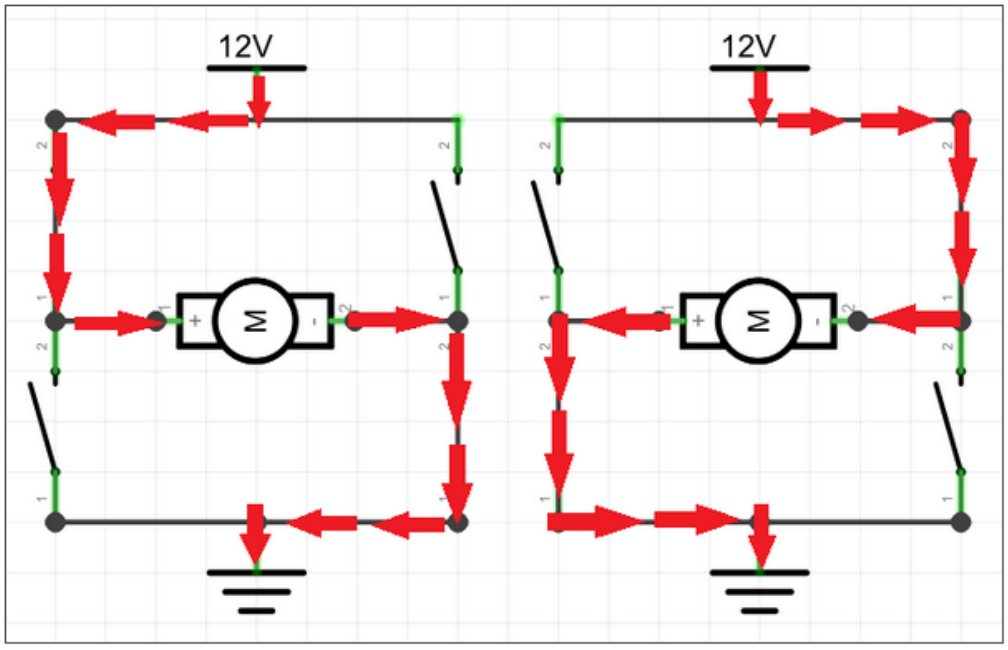


Figure 66 – Current Flow of H-Bridge

4 of these switches usually are bipolar junction transistor (BJT), MOSFET or relay. Depending on the requirements of different driver that people choose different kind of the “switch”.

H-Bridge using transistor BJT is quite commonly used in control low capacity motor. The reason if transistor BJT is easy to find and cheaper than MOSFET or relay. L298N is an H-Bridge using BJT.

The following figure is general H-Bridge circuit using bipolar junction transistor (BJT). BJTs are current-controlled transistor that allow for current amplification. It contains:

- 4 switches: Q1, Q2, Q3, Q4.
- PNP and NPN are bipolar junction transistor. They are the same in their function, they provide amplification and/or switching capability.
    - NPN: turn on by a HIGH signal (current).
    - PNP: turn on by a LOW signal (ground).

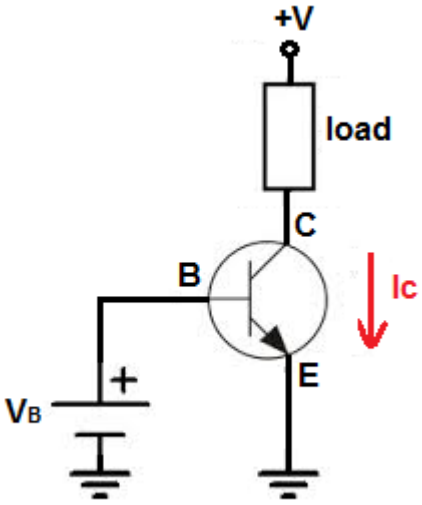


Figure 67 – NPN Transistor

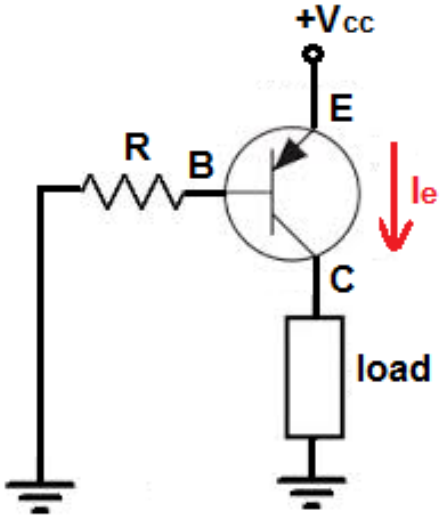


Figure 68 – PNP Transistor

In a NPN transistor, positive voltage is given to the collector terminal and the current flow from the collector to the emitter.

In a PNP transistor, positive voltage is given to the emitter terminal and current flows from the emitter to the collector.

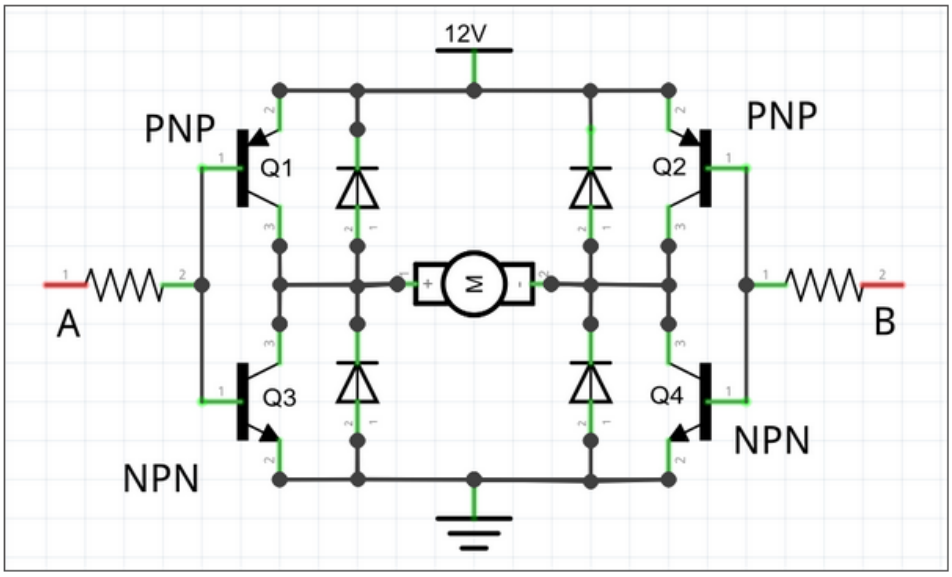


Figure 69 – General H-Bridge Circuit Using BJT Transistor

A, B is controller pole, 4 diodes, its purpose is to eliminate induced currents generated during engine work, if not, induced currents will damage the transistor.

We control A, B pole by sending HIGH and LOW signal with appropriate voltage 5v or 0v.

With HIGH/LOW signal in pole A and B, we have 4 different cases:

1 – A is HIGH and B is LOW

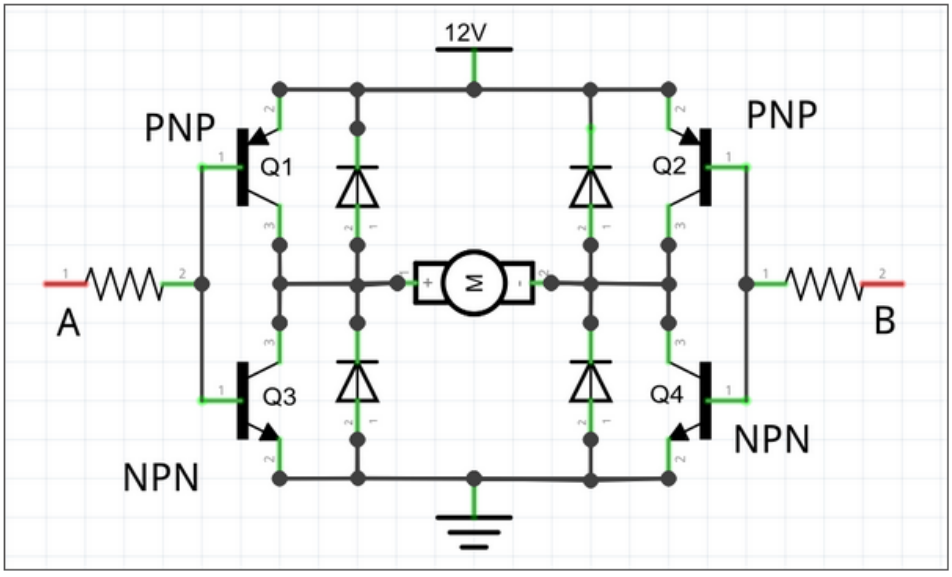


Figure 70 – General H-bridge Circuit Using BJT Transistor

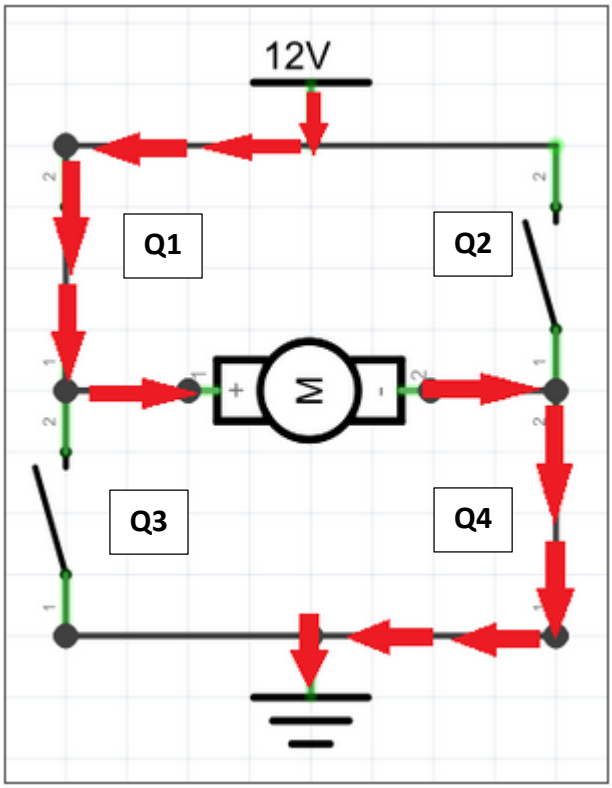


Figure 71 – Switch Q1, Q4 close, Q2, Q3 open.

On A side, transistor Q1 close, Q3 open. On B side, transistor Q2 open, Q4 close. So power current will flow from power supply through Q1, motor, Q4 to ground (GND).

2 – A is LOW and B is HIGH

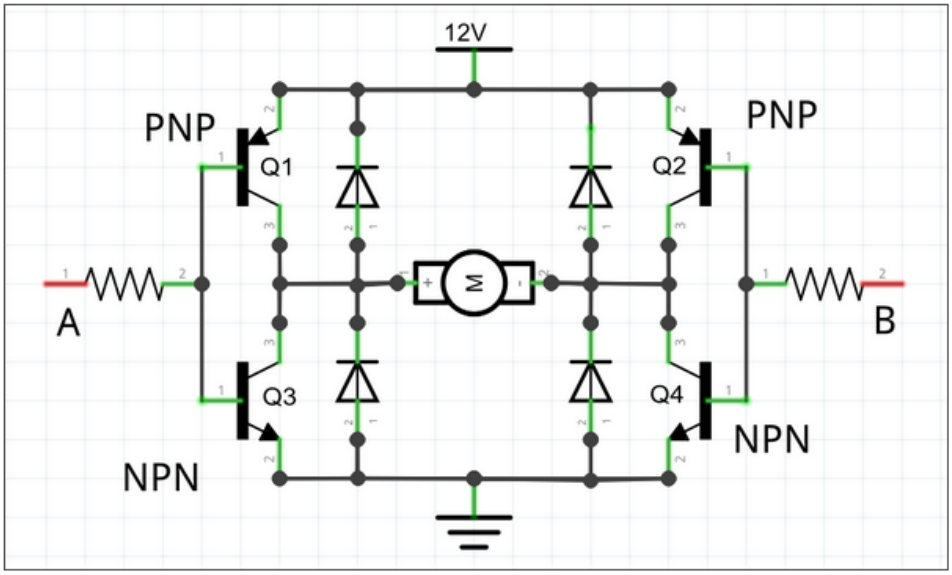


Figure 72 – General H-bridge Circuit Using BJT Transistor

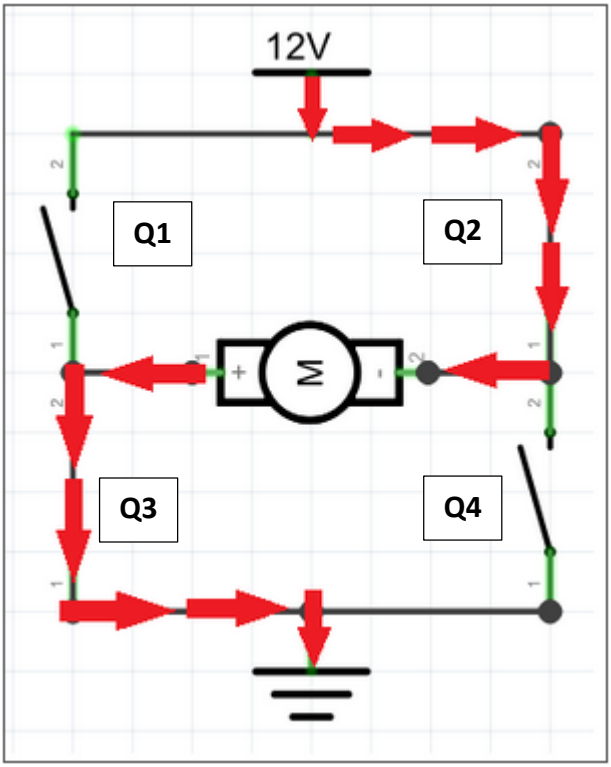


Figure 73 – Switch Q1, Q4 open, Q2, Q3 close.

On A side transistor A1 open, Q3 close. On B side, transistor Q2 close. Q4 open. So power current will flow from power supply through Q2, motor, Q3 to ground (GND).

3 – A, B is all LOW.

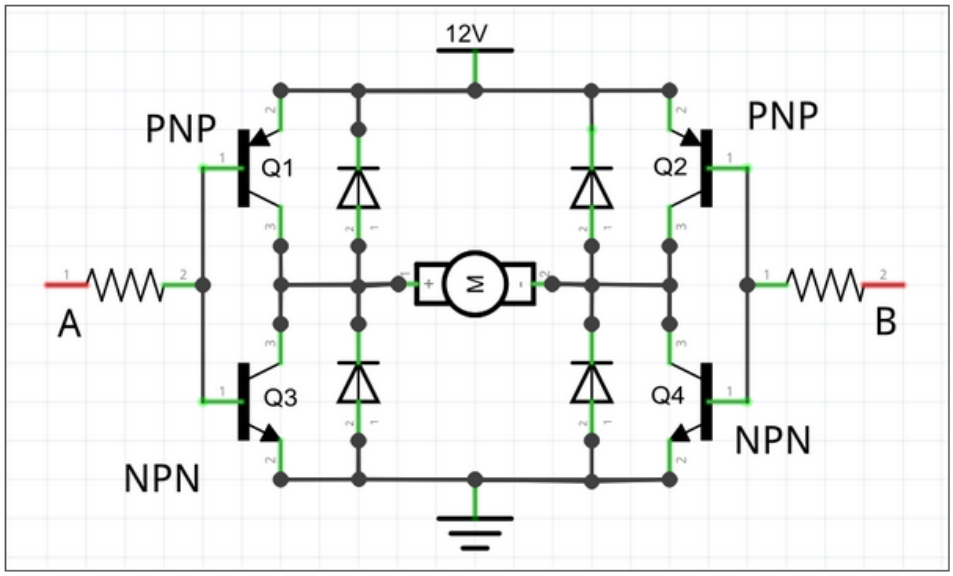


Figure 74 – General H-bridge Circuit Using BJT Transistor

Transistor Q1 and Q2 open, Q3 and Q4 close. Power current cannot go to GND, so that motor cannot work.

4 – A, B is all HIGH.

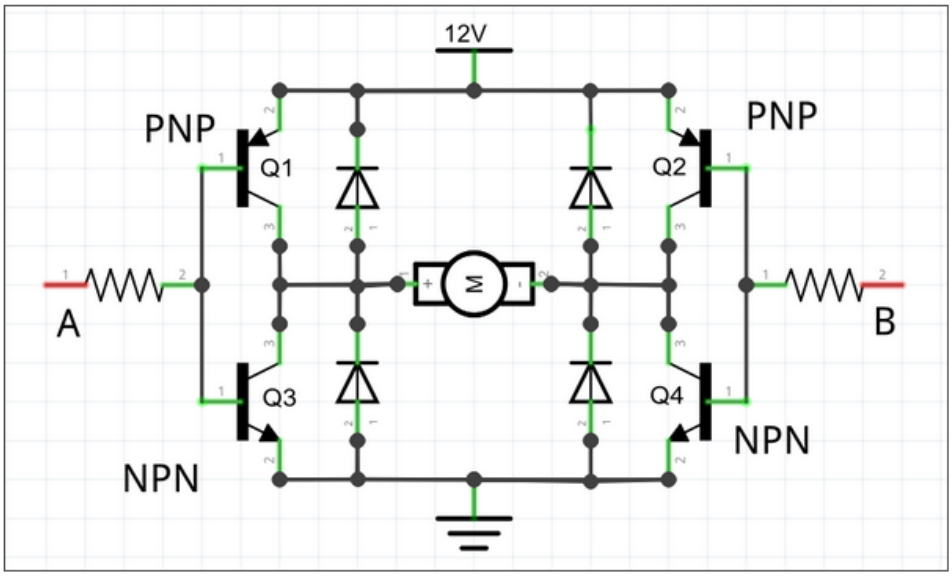


Figure 75 – General H-bridge Circuit Using BJT Transistor

Transistor Q1 and Q2 close, Q3 and Q4 open. Power current cannot come out power supply, so that motor cannot work.

2.5 Bluetooth Working Principle

2.5.1 Bluetooth radio transmission

Mobile phones, FM radio and television all use radio waves to send information wirelessly.

To avoid interfering with other systems, Bluetooth devices is designed to send out very weak radio signals (1- 100 mW).

Radios and TV broadcasts over many miles or kilometers, Bluetooth technology sends information at distances up to 100 meters.

Bluetooth uses a radio technology called frequency-hopping spread spectrum (FHSS). The transmitted data are divided into packets and each packet is transmitted on one of the 79 designated Bluetooth channels randomly. It usually performs 1600 hops per second.

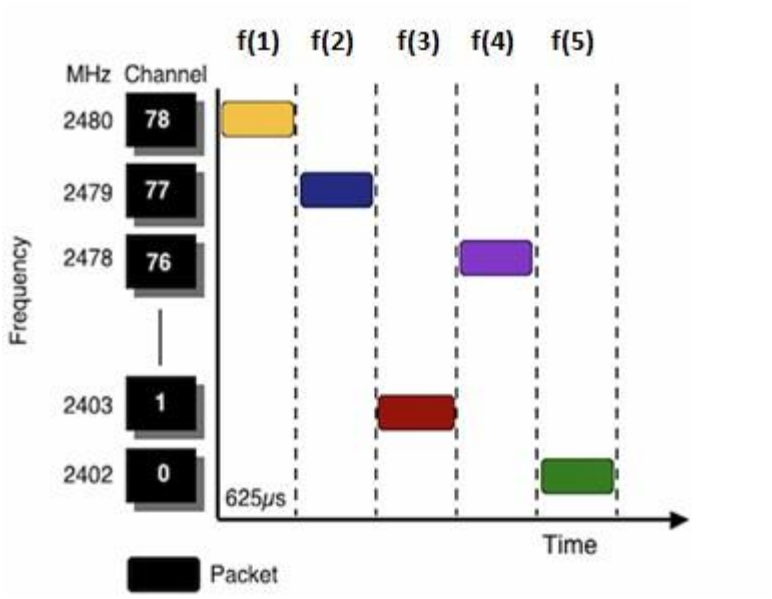


Figure 76 – Bluetooth Radio Transmission

Each channel has a bandwidth of 1 MHz Bluetooth 4.0 uses 2 MHz spacing that allows for 40 channels.

In the newest Bluetooth version (4.0), data transmission rate is 24 Mbit/s.

2.5.2 Bluetooth network

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 slave devices. Any slave device in the piconet can only be connected to a single master.

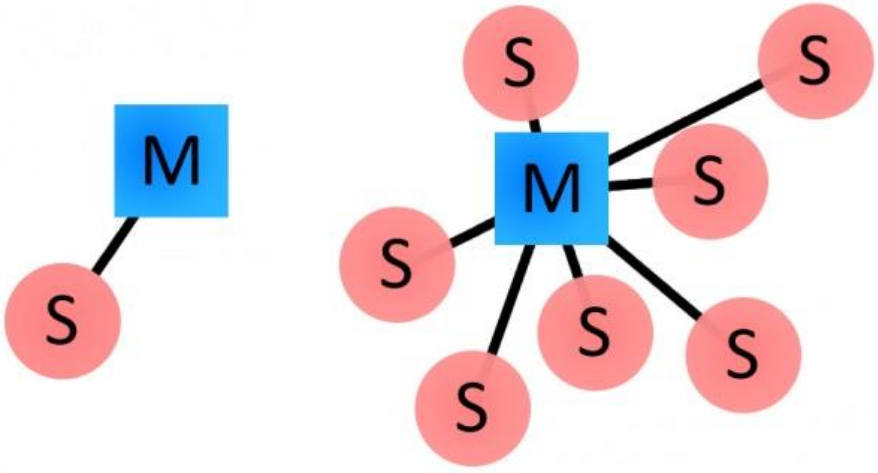


Figure 77 – Bluetooth network

The master coordinates communication throughout the piconet. It can send data to any of its slaves and request data from them as well.

2.5.3 Bluetooth device

Every single Bluetooth device has a unique 48-bit address, commonly abbreviated BD_ADDR.

This will usually be presented in the form of a 12-digit hexadecimal value.

The most-significant half (24 bits) of the address is an organization unique identifier (OUI), which identifies the manufacturer.

The lower 24-bits are more unique part of the address. This address should be visible on most Bluetooth devices.

Figure 78 – Bluetooth device

Bluetooth devices can also have user-friendly names given to them. These are usually presented to the user, in place of the address, to help identify which device it is.

2.5.4 Bluetooth connect process

Creating a Bluetooth connection between two devices is a multi-step process involving three progressive states:

1. Inquiry – If two Bluetooth devices know absolutely nothing about each other, one must run an inquiry to try to discover the other. One device sends out the inquiry request, and any device listening for such a request will respond with its address, and possibly its name and other information.
2. Paging (Connecting) – Paging is the process of forming a connection between two Bluetooth devices. Paging involves in synchronizing the clock and frequency of the two devices
3. Connection – After a device has completed the paging process, it enters the connection state. The devices now randomly hop frequencies in unison so they can start transmitting data.

Sometimes, devices may include a security mechanism called pairing, which restricts access to authorized users only, in order to give the piconets a certain measure of protection. Pairing is done with an encryption key commonly known as a "PIN" (*Personal Information Number*)

To do so, slave device sends a pairing request to the master device. User need to enter the access point's PIN. If the PIN received is correct, the connection is made.

V. Controller

1. Sequence Diagram

2.1 Connect Bluetooth Sequence Diagram

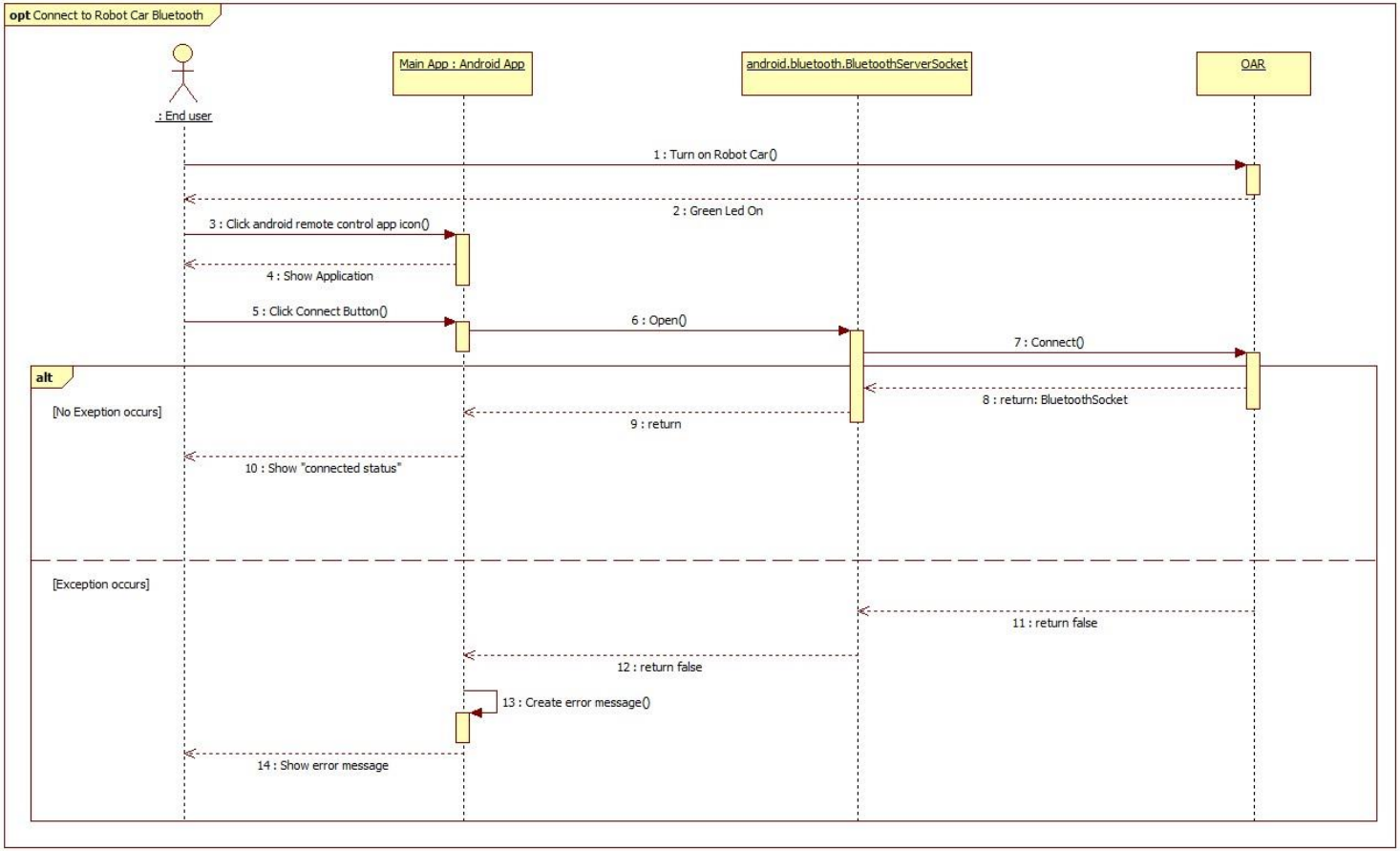


Figure 79 – Connect Bluetooth Sequence Diagram

2.2 Choose Operating Mode Sequence Diagram

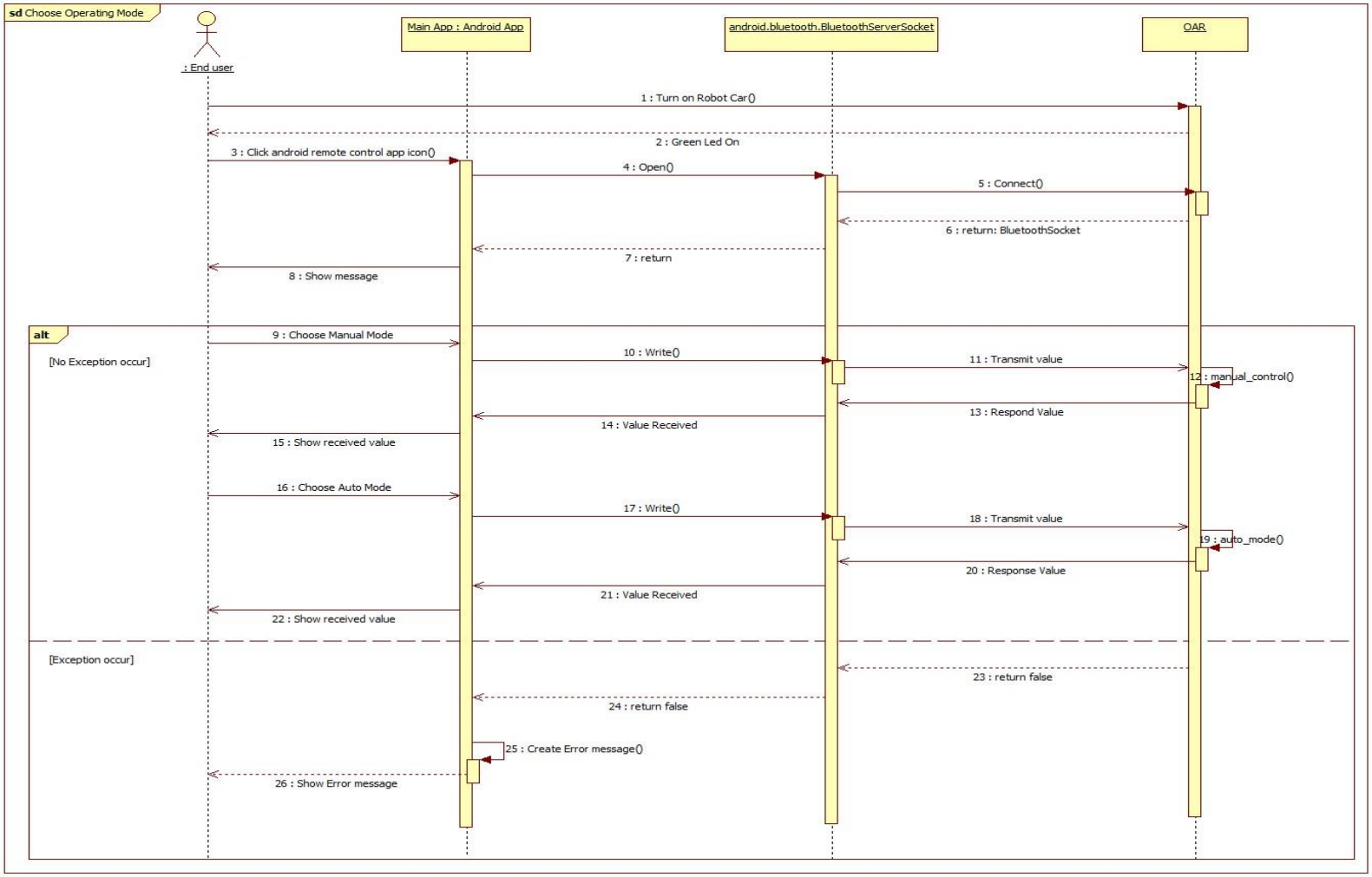


Figure 80 – Choose Operating Mode Sequence Diagram

2.3 Manual Control Sequence Diagram

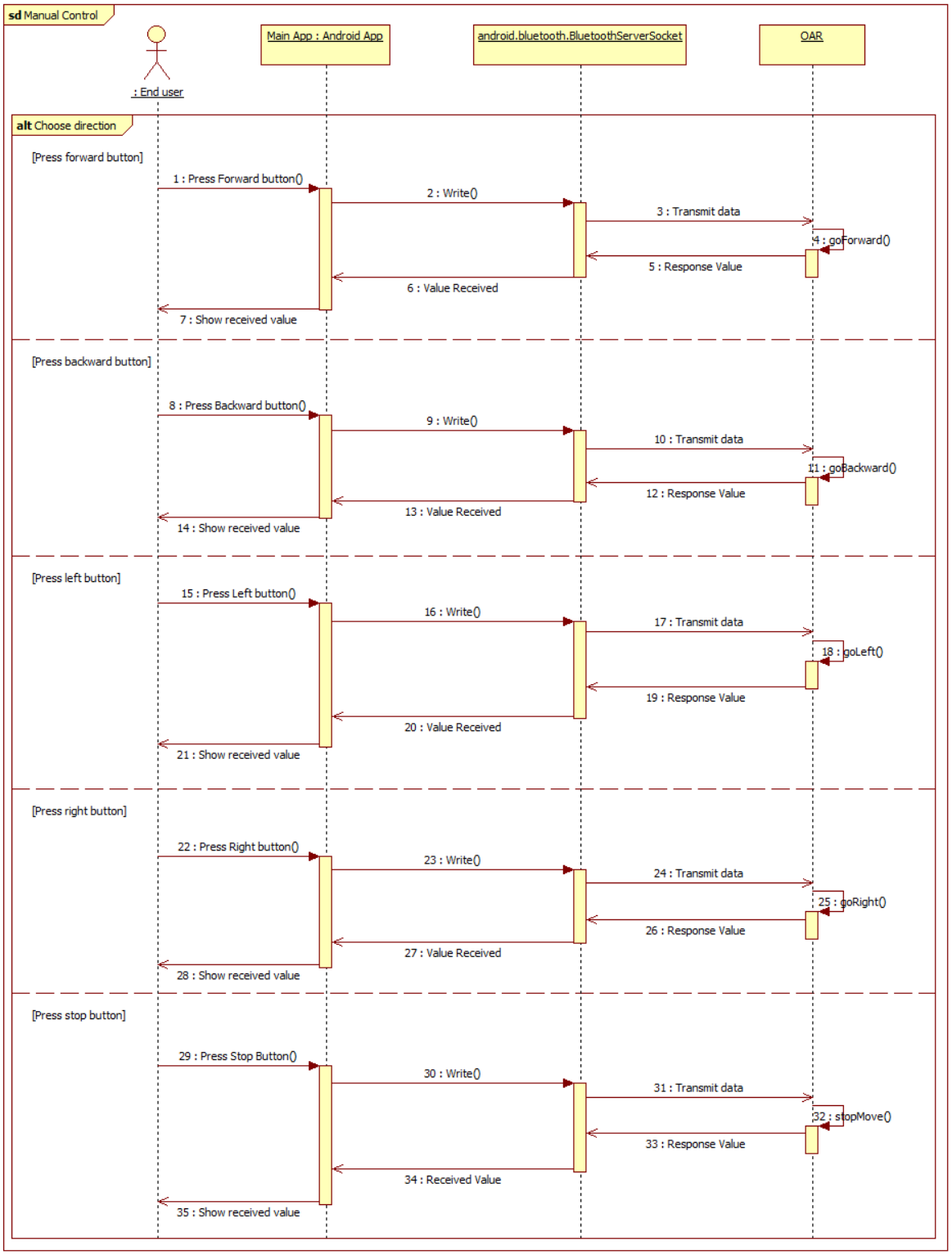


Figure 81 – Manual Control Mode Sequence Diagram

2. Controlling Components and Algorithm

2.1 Controlling Ultrasonic Sensor

2.1.1 Get Distance from Ultrasonic Sensor

First we supply power for ultrasonic sensor module to start operating by function digitalWrite(pin, value):

- pin: trigger pin of ultrasonic sensor
- value (HIGH or LOW): we need to supply power for module, so we choose HIGH.

In module datasheet, we just have to trigger the module for 10uS (microsecond), after this time, we have to quit supply power to the module. After that module will automatically raise it echo line to HIGH state.

To read data from echo pin, we use pulseIn(pin, value, timeout) function. This function will read a pulse (either HIGH or LOW) on a pin. If the value is HIGH, pulseIn() wait for the pin to goes HIGH, starts timing, then wait for the pulse to go LOW and stop timing. Return the length of the pulse in microsecond. Return 0 if no pulse starts within specific time out, default is 1 second.

2.1.2 Calculate Distance from Sensor to an Object

We had the time of the pulse that reflected from an obstacle after it came out from the sensor.

We measure distance from ultrasonic sensor to an object by calculating the duration between send and receive reflect signal with the principles Time of Flight (TOF).

$D = \frac{s*t}{2}$.

- D: distance from sensor to object.
- S: speed of ultrasonic, which is 343 m/s or
- T: travel time of ultrasonic.

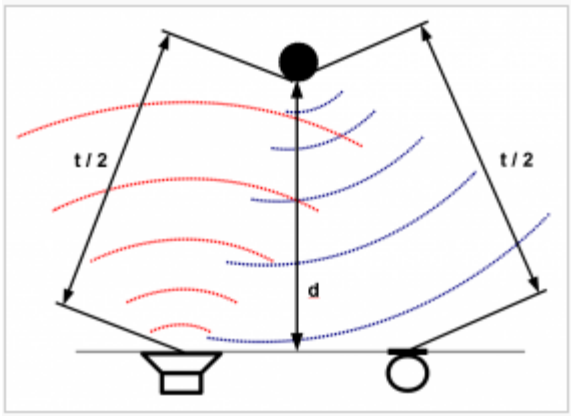


Figure 82 – Measure Distance from Sensor to an Obstacle

2.2 Controlling Servo Motor

Servos are controlled by sending an electrical pulse of variable width, or **pulse position modulation** (PPM), through the pulse wire, it is the same as PWM and the different is PPM has 50Hz frequency. There is a minimum pulse, a maximum pulse and a repetition rate. A servo motor can usually only turn 90° in either direction for a total of 180° movement.

The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse sent via the control wire the rotor will turn to the desired position.

However servo position is not defined by the PWM duty cycle (i.e., ON vs OFF time) but only by the length of the pulse and the length of the pulse will determine how far the motor turns

The servo control pulse is repeated every 20 milliseconds. In essence, every 20 milliseconds you are telling the servo, “go here”. If the servo is ON, and you try to move position of its original, you’re simply cannot.

The amount of time that a pulse is on during a 2ms period dictates the servos position. This period, or sequence, repeats around 50 times per second. If that pulse is high for 1 millisecond, then the servo angle will be zero, if it is 1.5 milliseconds, then it will be at its center position and if it is 2 milliseconds it will be at 180 degrees.

We can give any pulse width between 1ms and 2ms. Servo’s position.

To control servo in arduino, we use “Servo library” with servo.write(angle) function. This function will set the angle of the shaft (in degree), moving the shaft to that orientation.

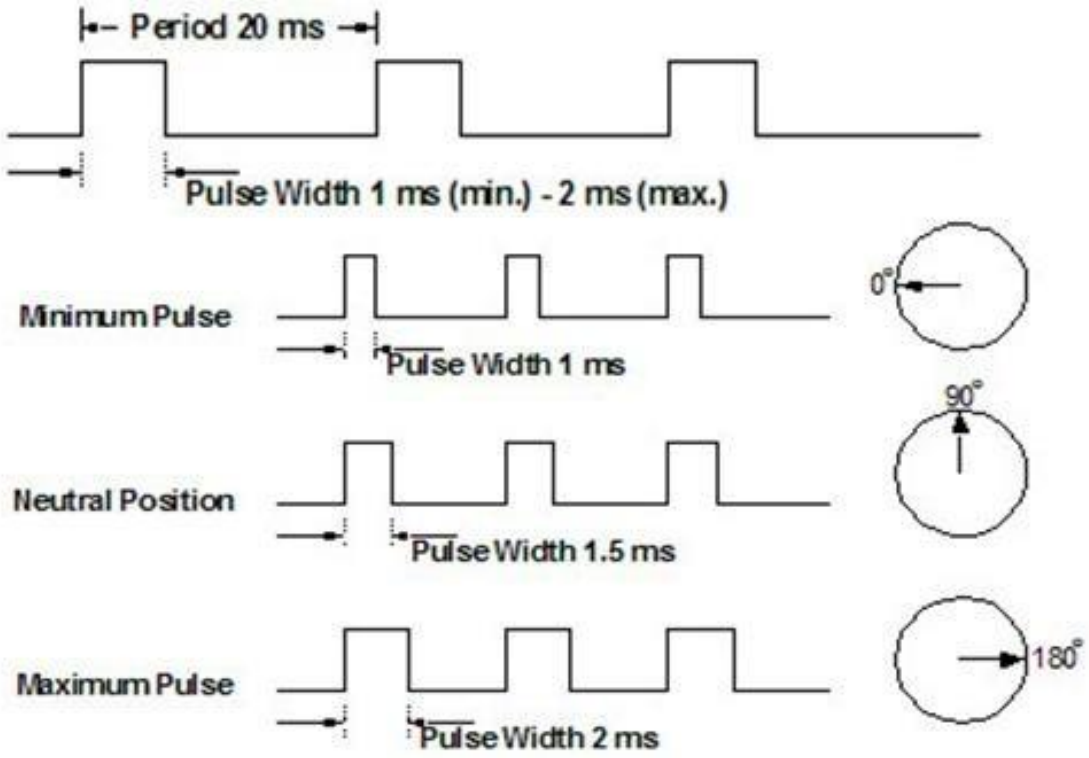


Figure 83 – Servo SG90 Pulse Position Modulation Graph

2.3 Controlling DC Gear Motor by PWM

With DC motors, we can control it by most popular way is to use Pulse Width Modulation (PWM). PWM techniques, as well as how to use the PWM registers directly for more control over the duty cycle and frequency, or in simpler way PWM is a technique allow us to control the width of the pulse, formally the pulse duration. We can understand that PWM help us to control output tension, from low to high. It is providing us variable speed control for motors.

Arduino provides function analogWrite(pin, dutyCycle), where dutyCycle is a value from 0 to 255, and pin is one of the PWM pins (3, 5, 6, 9, 10, or 11). The analogWrite function provides a simple interface to the hardware PWM, but does not provide any control over frequency.

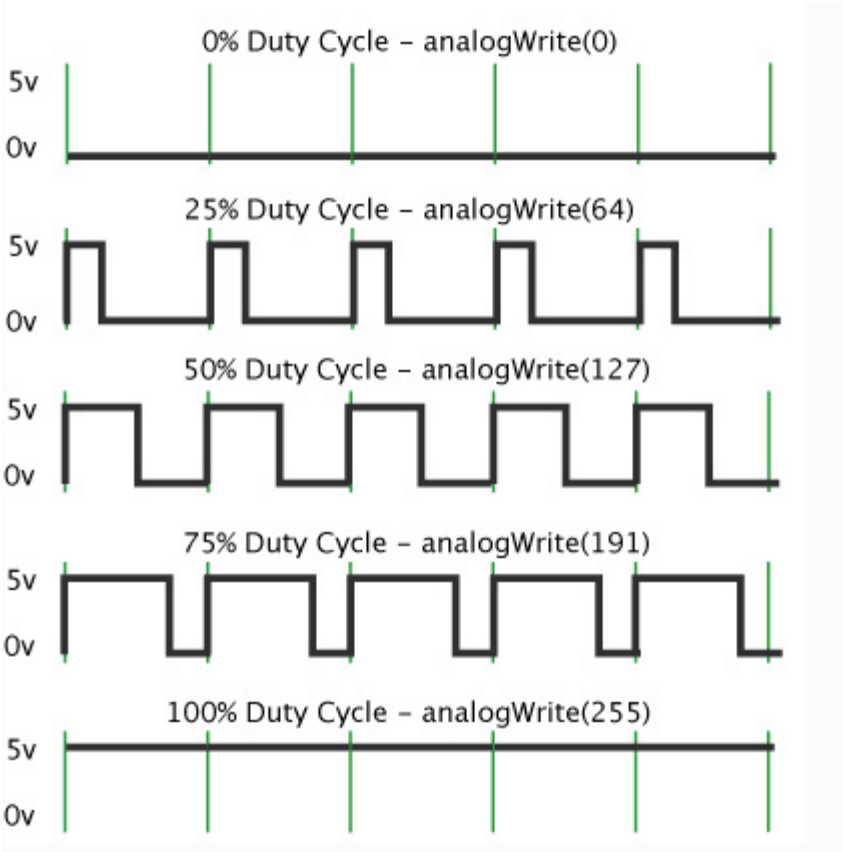


Figure 84 – PWM Duty Cycle

analogWrite	Ratio	Duty cycle
analogWrite(0)	0/255	0%
analogWrite(64)	64/255	25%
analogWrite(127)	127/255	50%
analogWrite(191)	191/255	75%
analogWrite(255)	255/255	100%

Table 33 – analogWrite Function in Arduino

Arduino Uno R3 (ATmega 328 microcontroller) has 6 digital input / output pin can be used as PWM outputs with 8-bits resolution. The resolution will be $2^8 - 1 = 255$. We can control DC motor speed from 0 (motor stop) to 255 (highest motor speed).

2.4 Controlling Bluetooth HC-05

Arduino Uno R3 communicates with HC-05 through Universal Asynchronous Receiver/Transmitter. Those are RX and TX pin – or pin 0 and pin 1 on Uno R3.

UART is a piece of computer hardware that translates date between parallel and serial form. Some case UART do exist as stand-alone ICs, but they commonly found inside micro controller. In this case is ATmega328, which has a single UART.

UART are responsible for both sending and receiving serial data. On the transmit side, UART must create a data packet – appending and party bits – and send that packet out the TX line with precise timing (according to set the baud rate). On the receiver end, UART has to sample the RX line at rates according to the expected baud rate, pick out the sync bits and spits out the data.

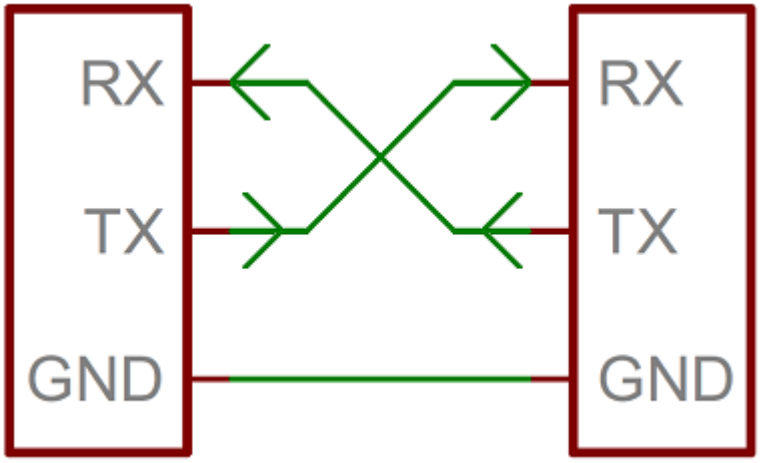


Figure 85 – UART Wiring

Baud rate specify how fast data is send over a serial line. It is usually express in units of bits-per-second (bps). If we invert the baud rate, we can find out just how long it takes to transmit a single bit. This value determines how long the transmitter holds a serial line high/low or at what period the receiving device samples its line.

Baud rates can be just about any value within reason. The only requirement is that both devices operate at the same rate. One of the more common baud rates, especially for simple stuff where speed is not critical, is 9600 bps. Other “standard” baud are 1200, 2400, 4800, 19200, 38400, 57600, and 115200.

The higher a baud rate goes, the faster data is sent/received, but there are limits to how fast data can be transferred. We usually will not see speeds exceeding 115200 - that is fast for most microcontrollers. Get too high, and we will begin to see errors on the receiving end, as clocks and sampling periods just cannot keep up.

More advanced UART may throw their received data into a buffer, where it can stay until the microcontroller comes to get it. UART usually release their buffered data on a first in first out (FIFO) basis. Buffers can be as small as few bits, or as large as thousands of bytes.

In Arduino, to read data from serial port we use Serial.read() function. It reads incoming serial data, return the first byte of it and return -1 if there is not data available.

To send data to serial port, we use Serial.print() or Serial.println() function, but first we have to use Serial.begin(baudrate) function to set its baud rate before using. It will print to serial

port as human-readable ASCII text followed by a carriage return character (ASCII 13 or '\') and a new line character.

2.4 Steering Mechanism

2.4.1 Normal Car Steering Mechanism.

Out OAR does not have steering wheel, so our OAR steering mechanism is not the same as normal car.

Figure 86 – Normal Car Steering Mechanism

In normal 4-wheeled car, they used rack and pinion steering mechanism, when the steering wheel turns the pinion gear, the pinion will move the rack, which is a linear gear that messes with the pinion, converting circular motion into linear motion along the transvers axis of the car (side-to-side motion). The motion applies steering torque to the swivel pinball joints that replaced previous used kingpins of the stub axle of the steered wheels via tie rods and a short lever arm called the steering arm.

2.4.2 OAR Steering Mechanism

Instead of using rack and pinion steering mechanism, our OAR steering mechanism looks like tank steering mechanism.

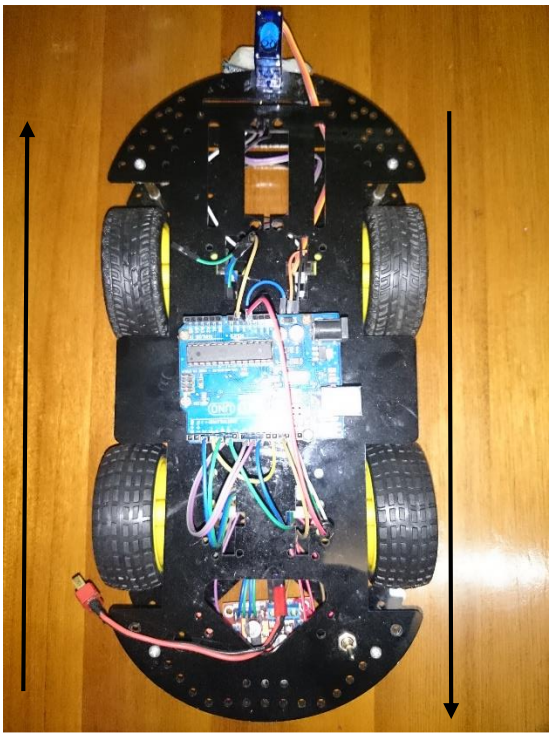


Figure 87 – OAR Turn Right Mechanism

With OAR, we have 4 wheels, two wheels at each side. If OAR need to steer right, left wheels will rotate forward and right wheels will rotate backward.

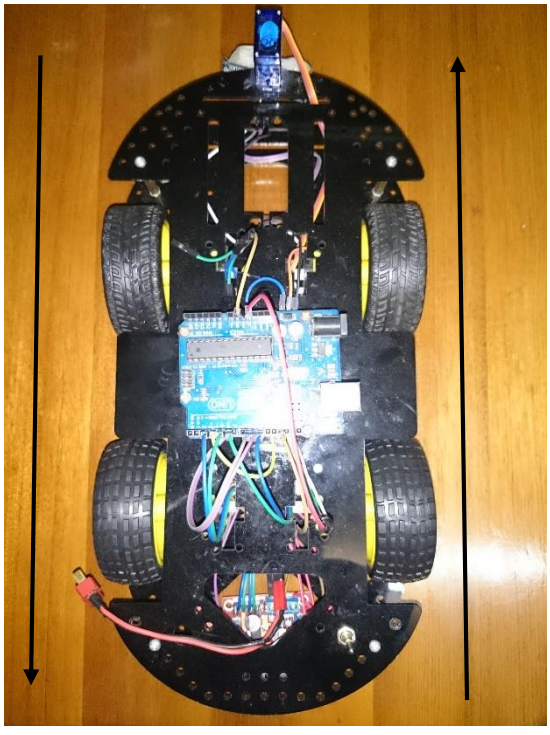


Figure 88 – OAR Steer Left Mechanism

If OAR need to steer left, left wheels will rotate backward, right wheels will rotate forward.

2.5 Obstacles Avoidance Algorithm

After tested and field experiment, at last we decide to implement this algorithm.

With OAR moving speed of 60cm per second and because of inertial force and friction force, we have set distance limit from ultrasonic sensor to obstacles to 30cm to avoid OAR hits the obstacles. We do not have breaking force of our OAR so we cannot give out the precise number from when OAR see the obstacles to it totally stop. 30cm is the number that we have tested in real field, that distance is suitable for our robot and its flexibility or safety.

We set distance limit from ultrasonic sensor to obstacles to 30cm. When OAR reaches this distance limit. First OAR will stop, second servo will rotate five specific angles with rotate speed about 0.10 sec/60^{o} (fixed speed). This solution is better than sweeping from left to right because it takes a lots of time, if we make faster sweeping time, ultrasonic will miss reflected wave so OAR cannot make right decisions.

So we decided for ultrasonic sensor scan for five specific angles. These are left (10 degree), left sidelong (50 degree), center (90 degree), and right sidelong (140 degree) and right (180 degree). At each angle, we use ultrasonic sensor to get distance (from the sensor to obstacles) and store it in a queue, with 5 angles, we will have 5 different distances. After that we compare those 5 distances, choose that largest one. OAR will turn to that angle. If new angle does not have any obstacles ahead, OAR will move forward, if yes, OAR will scan again to find new direction.

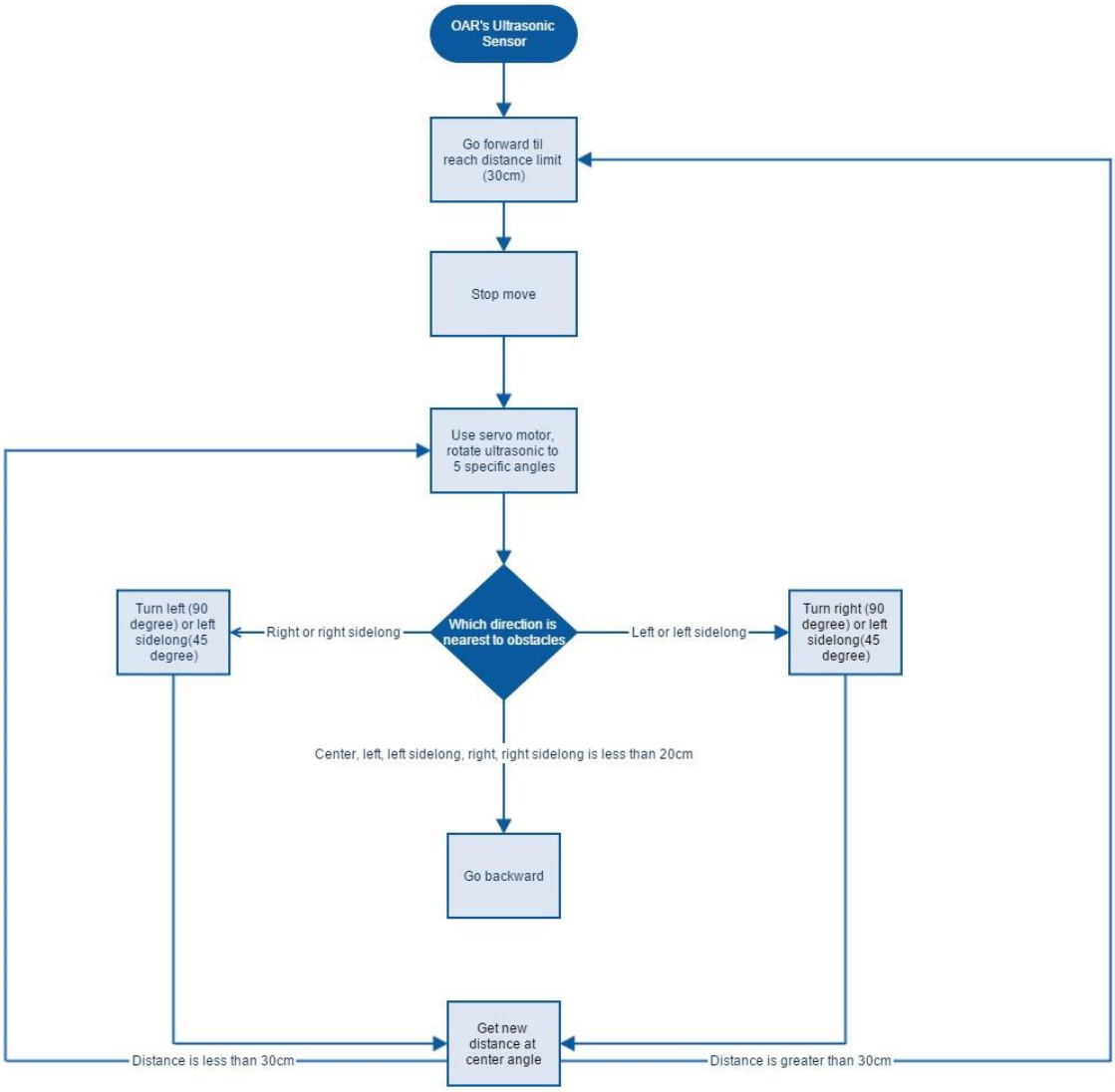


Figure 89 – Obstacles Avoidance Algorithm Flow Chart

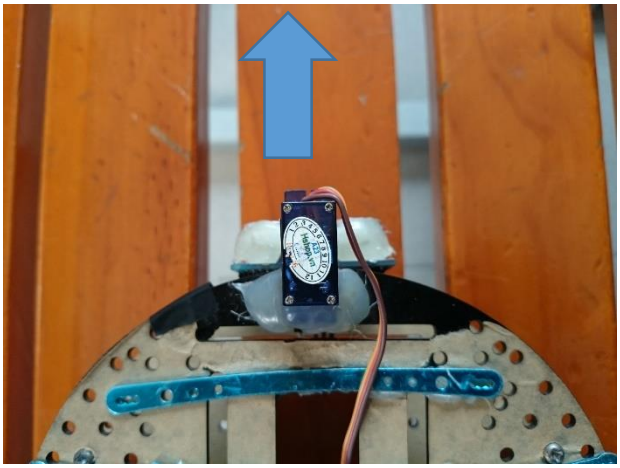


Figure 90 – Ultrasonic Sensor at Center Angle – 90 degree

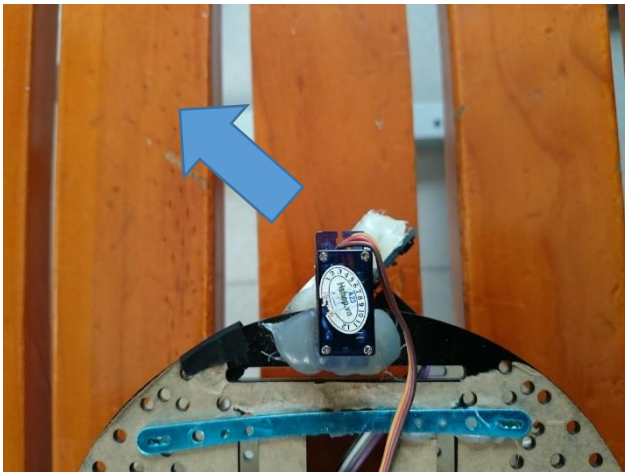


Figure 91 – Ultrasonic Sensor at Left Sidelong Angle – 50 degree

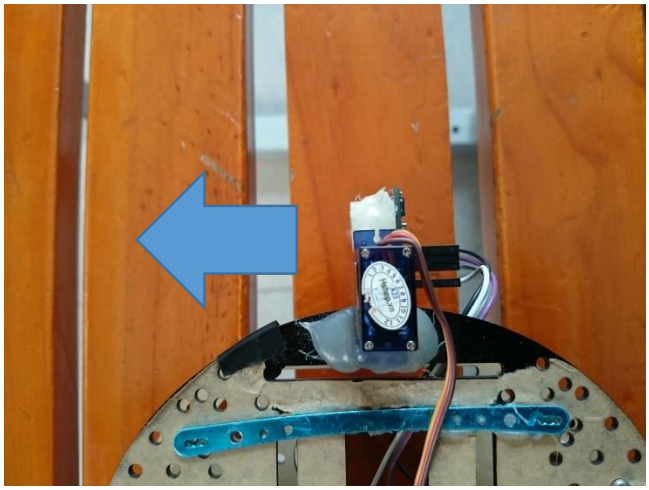


Figure 92 – Ultrasonic Sensor at Left Angle – 10 degree

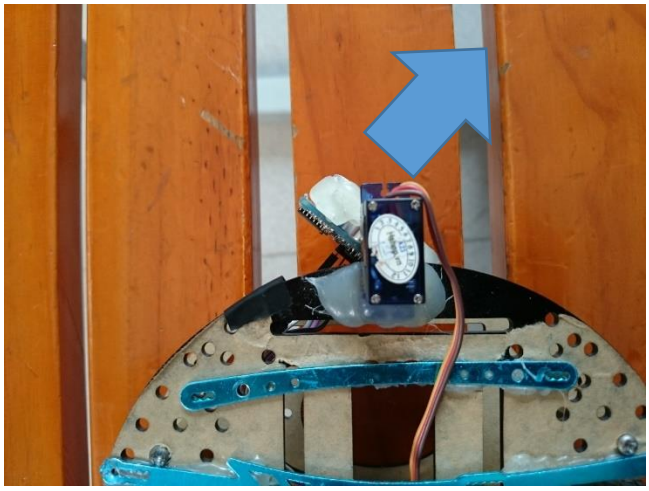


Figure 93 – Ultrasonic at Right Sidelong Angle – 140 degree

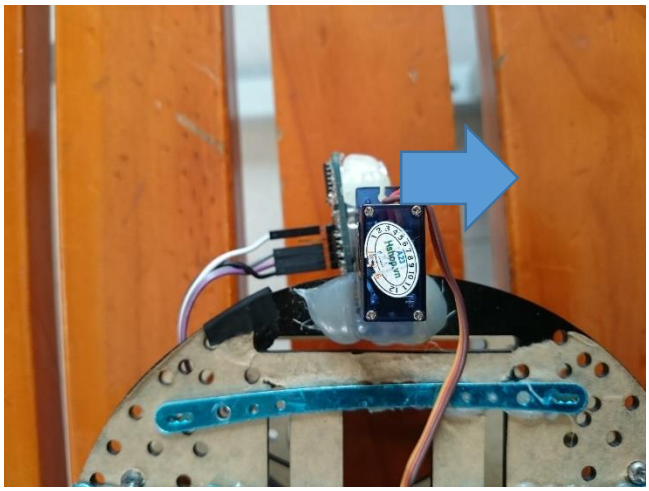


Figure 94 – Ultrasonic Sensor at Right Angle – 190 degree

After tested obstacles avoidance algorithm above in real field. We have seen that OAR could not find a way out after enter some special corner.

Case 1:

- Example: OAR turns left into a corner, go forward, when OAR detects obstacles it will stop and start scanning. At this time both center, left and right side is less than 20cm, OAR cannot turn left or right, OAR will decide to go backward with backward time equal to the time it went forward after turned left into a corner. While OAR is going backward if there is an obstacles on the path (We use second ultrasonic sensor placed at the back of robot) OAR will stop and blink a led light, signal that there is no way to go. If not OAR will decide to turn right. If there is nothing ahead, robot will move forward, if there is OAR will scan and choose suitable direction to move.

Case 2:

- Example: OAR runs into an area. After turns left/left sidelong or right/right sidelong 3 times and cannot find a way out. OAR will get back to the point which where lead OAR to that area choose another direction and keep going. If there is an obstacle on the old path, OAR will blink a led signal that it stuck in dead area and there is no way to get out.

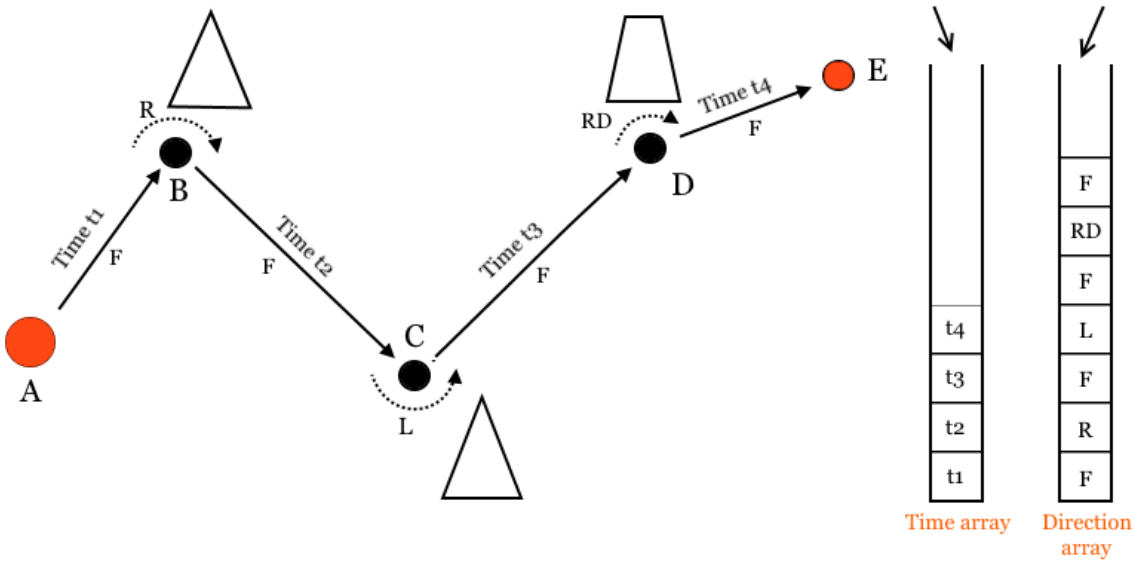
2.6 Back-tracking Algorithm

After a lot of effort, we finally get OAR to run outside. After a few minutes of testing, the OAR moved so far from the start point. What if the OAR run for some dozen of minutes, and we ended up losing track of it? On the other hand, if the OAR ran out of power and end up stuck in some small corners, it will be hard to retrieve it back.

To avoid those problems, we create a backtracking algorithm for the OAR. With this algorithm, the OAR can run freely for many minutes. In addition, when we need to retrieve it back, it can go back to the starting point with no problems.

The Back Tracking Algorithm contains two major parts:

2.6.1 Path Saving



In this example, the OAR start running at A point. At these B-C-D points, OAR detect obstacles and decide to turn to another direction. Finally, it stopped at E point and wait to go back.

To do backtracking, first the OAR need to remember the path it ran.

To go from Start point (A) to End point (E), we have the following OAR sequence of movements:

- From A to B: Move forward.
- From B to C: Turn Right, Move Forward.
- From C to D: Turn Left, Move Forward.
- From D to E: Turn Right Diagonal, Move Forward.

Each time OAR want to move, the system will call the right function to move the wheels.

After a duration, it will stop the wheels, so the movement is finished.

In our program, every turn movement has a fixed duration :

- Turn Left: cost 400 milliseconds.
- Turn Right: cost 400 milliseconds.
- Turn Left Diagonal: cost 200 milliseconds.
- Turn Right Diagonal: cost 200 milliseconds.

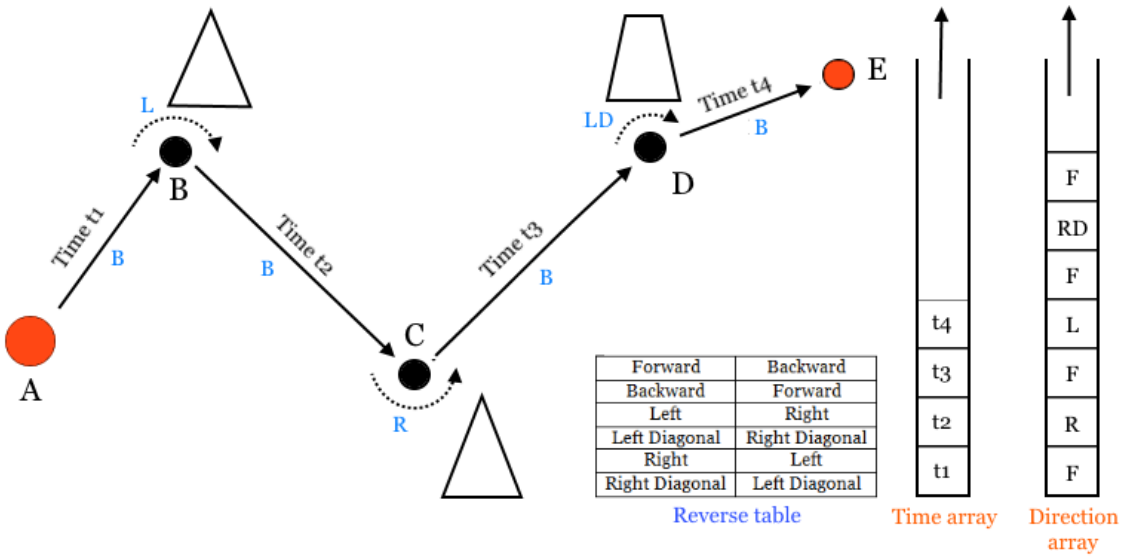
Only in a two movement : “Forward” and “Backward”, the duration don’t have a fixed value and change everytime the OAR move. We need to calculate them.

After we get all the movement directions and movement durations, we need to find some places to store them. The solution is create two different arrays:

- Direction array: store all the direction values. The values is store from the oldest to the latest.

Time array: store all the duration values. The values is also store from the oldest to the latest.

2.6.2 Back Tracking



After we finish store all the need values in the first part. We can now do backtracking.

First, we retrieve the values from Direction array, using LIFO method.

Then we change the directions to the opposite one, using Reverse table :

Original direction	Reverse direction
Forward	Backward
Backward	Forward
Left	Right
Left Diagonal	Right Diagonal
Right	Left
Right Diagonal	Left Diagonal

Table 34 – Reserve Table

The final backtracking sequence of movements will be like this :

- Backward.
- Right Diagonal.
- Backward.
- Right.
- Backward.
- Left.
- Backward.

We order the OAR to move backtracking with this sequence movements, step by step. For each “Backward” values, we retrive a value back from Time array (using LIFO method). This value will be set at the duration for “Backward” function.

After the OAR finished the whole sequence of movements, it will be back at the Start Point. But the problem is not end yet.

7.3 Memory Management

During backtracking algorithm, we need to store many values. The new problems are:

7.3.1 Where do we store the values?

In this project, we are using the Arduino Uno R3 microcontroller board, which using ATmega328 chip. There are 3 pools of memory in Arduino board:

- Flash memory (program space): is where the Arduino sketch is stored. Can store 32K bytes
- SRAM (static random access memory): is where the sketch creates and manipulates variables when it runs. Can store 2k bytes.

- EEPROM: is memory space that programmers can use to store long-term information. Can store 1k bytes.

After some researches, we decided to store the values in EEPROM memory.

- The first half of EEPROM (from Byte #0 to #511): is used to store the Time values.
- The second half (from Byte #512 to #1023): is used to store the Direction values.

7.3.2 How do we store data in EEPROM?

To store data in EEPROM memory, we use the EEPROM library function:

- EEPROM.read()
- EEPROM.write()

Because Time values usually exceed thousands (millisecond), we cannot store it in bytes type (unsigned byte = 255 max). We need to store it in integer type (unsigned int = 65.535) Therefore, we need to save the Time values in two continuous addresses, the first address store the first 8-bit data while the second address store the last 8-bit data. In the end, two new functions are added to read / write Time values.

7.3.3 How much values can be store?

OAR can store at most: 512 Directions values and 256 Time values.

In real tests, OAR will be run out of EEPROM memory after running about 20 minutes.

In backtracking mode, when the memory is full, OAR will automatically move back to the start point.

Software Implementation and Testing (SIT)

I. Implementation

Reference to section "2. Controlling Component and Algorithm" at page 118.

II. Testing

1. Testing Overview

I this section, there has all necessary information about test plan, test cases, test result, the environment for testing, test pass/fail criteria, risks estimations and a checklist to check when test this system.

2. Test Approach

To make sure whole system works fine, as less bugs as good, we apply following testing method.

White box testing: Developer will do the test of all functions, which have coded by them. We applied unit testing and integration testing.

Black box testing: We have many hardware components, before test whole system, we have tested for each hardware components to make sure it works fine. After tested all hardware components, we did system test to ensure whole system is going well.

Goal: Detect as many bugs as possible, after identify them, we will fix it and do the test again. Once the system is stable, we will verify to ensure that all functions meet requirement. If everything is done, we will have whole completed system.

III. Test Plan

1. Features To Be Tested.

Test item	Description
Hardware	Test Arduino Uno R3
	Test Ultrasonic Sensor SRF-05

	Test DC motor connected with L298N
	Test Servo motor SG90
	Test Bluetooth HC-05
OAR	Move forward
	Move backward
	Turn left
	Turn right
	Stop move
	Detects and avoid obstacles with specific situation
Android Application	Connect to HC-05
	Send message to HC-05
	Receive message to HC-05

Table 35 – Features to Be Tested

2. Testing Environment.

2.1 Operating System:

- Laptop with Windows 7 or Windows 8.1.
- Smart phone with Android 4.0.3 or greater.

2.2 Tools:

- "Serial monitor" in Arduino IDE.

IV. Test Case

1 Test Hardware Components

No.	Function			Pre-condition	Steps of testing	Expected result	Create by	Executed by	Result
	Large Function	Medium Function	Small Function						
1	OAR hardware devices connected with micro controller Arduino Uno R3	SRF05 Ultrasonic Sensor	Sending signals	None	Turn on Trigger Pin. After 10uS, turn it off.	SRF05 sent out signals to detect obstacles	KhoaCDN	KhoaCDN	Pass
2				None	Turn on Trigger Pin. After 5uS, turn it off.	SRF05 don't send any signals	KhoaCDN	KhoaCDN	Pass
3				None	Turn on Trigger Pin. After 10mS, turn it off.	SRF05 sent out signals to detect obstacles	KhoaCDN	KhoaCDN	Pass
4			Detect obstacle within the range (3 cm – 4m)	No obstacle in front of sensor	Checking Echo Pin status while counting time. If echo pin turn off, return the time	After 30mS, return max distance. Echo pin automatic turned off	KhoaCDN	KhoaCDN	Pass
5				One obstacle in front of sensor	Checking Echo Pin status while counting time. If echo pin turn off, return the time	Detected obstacle. Signal traveling time is returned.	KhoaCDN	KhoaCDN	Pass
6				Two obstacle in	Checking Echo Pin status while	Nearest obstacle is detected.	KhoaCDN	KhoaCDN	Pass

				front of sensor	counting time. If echo pin turn off, return the time	Signal traveling time is returned.			
7			Calculate distance in centimeters	None	Using Time of Flight formula, convert the Time (uS) values to Distance (cm). D = T / 58	Convert successfully. Return distance of obstacle back in cm	KhoaCDN	KhoaCDN	Pass
8			Calculate distance in inches	None	Using Time of Flight formula, convert the Time (uS) values to Distance (inch). D = T / 148	Convert successfully. Return distance of obstacle back in inch	KhoaCDN	KhoaCDN	Pass
9			Performance test	Face directly to the wall. Turn 30 degrees to the left side	Sending signals, try to detect the wall	Wall detected successfully.	KhoaCDN	KhoaCDN	Pass
10				Face directly to the wall. Turn 60 degrees to the left side	Sending signals, try to detect the wall	Failed to detect the wall. Return timeout value	KhoaCDN	KhoaCDN	Pass
11				None	Prepare a shining obstacle: Glass. Sending signals, try to detect the glass	Failed to detect the glass Return timeout value	KhoaCDN	KhoaCDN	Pass
12				None	Prepare a soft obstacle: fluffy	Failed to detect the toy	KhoaCDN	KhoaCDN	Pass

					toy. Sending signals, try to detect the toy	Return timeout value			
13				None	Prepare an obstacle Put it 1 cm far from sensor. Sending signals, try to detect the obstacle	Failed to detect the obstacle. Return timeout value	KhoaCDN	KhoaCDN	Pass
14				None	Prepare an obstacle Put it 5 m far from sensor. Sending signals, try to detect the obstacle	Failed to detect the obstacle. Return timeout value	KhoaCDN	KhoaCDN	Pass
15				None	Calculate the time duration to received back the result	The time duration is around 30mS	KhoaCDN	KhoaCDN	Pass
16		Servo Motor SG90	Turn directions	Set the motor at default degree (0^o)	Turn 45 degrees to the left	Motor turn left exactly at 45 degrees	KhoaCDN	KhoaCDN	Pass
17				Set the motor at default degree (0^o)	Turn 100 degrees to the left	Motor turn leff and reach the limit at 90 degrees	KhoaCDN	KhoaCDN	Pass
18				Set the motor at default degree (0^o)	Turn 45 degrees to the right	Motor turn right exactly at 45 degrees	KhoaCDN	KhoaCDN	Pass
19				Set the	Turn 100 degrees	Motor turn right	KhoaCDN	KhoaCDN	Pass

				motor at default degree (0°)	to the right	and reach the limit at 90 degrees			
20				Set the motor to the leftmost (90 degrees left from default)	Turn the motor to the rightmost side	Motor turn right for 180 degrees, reach the right limit	KhoaCDN	KhoaCDN	Pass
21			Performance test	None	Turn the motor 45degrees to left / right side. Calculate the time duration	The time duration is around 0.075 second	KhoaCDN	KhoaCDN	Pass
22				None	Turn the motor 90 degrees to left / right side. Calculate the time duration	The time duration is around 0.15 second	KhoaCDN	KhoaCDN	Pass
23				Set the motor to the leftmost (90 degrees left from default)	Turn the motor to the rightmost side. Calculate the time duration	The time duration is around 0.3 second	KhoaCDN	KhoaCDN	Pass
24				Set the Wall block the left side of motor	Turn the motor to the left. Motor hit the wall	Motor won't stop turn to the left Can be break	KhoaCDN	KhoaCDN	Pass
25		SRF05 combine with SG90	Obstacle detection test	Obstacle is 45 degrees to the left	Turn servo 45 degrees to the left. Sending ultrasonic signal to detect	Obstacle detected Obstacle distance is calculated and received back	KhoaCDN	KhoaCDN	Pass

					obstacles				
26				Obstacle is 45 degrees to the right	Turn servo 45 degrees to the right. Sending ultrasonic signal to detect obstacles	Obstacle detected Obstacle distance is calculated and received back	KhoaCDN	KhoaCDN	Pass
27				Obstacle is 90 degrees to the left	Turn servo 90 degrees to the left. Sending ultrasonic signal to detect obstacles	Obstacle detected Obstacle distance is calculated and received back	KhoaCDN	KhoaCDN	Pass
28				Obstacle is 90 degrees to the right	Turn servo 90 degrees to the right. Sending ultrasonic signal to detect obstacles	Obstacle detected Obstacle distance is calculated and received back	KhoaCDN	KhoaCDN	Pass
29				Obstacle is 110 degrees to the right	Turn servo 110 degrees to the right. Sending ultrasonic signal to detect obstacles	Servo react its limit and stop at 90 degrees to the right. Obstacles is not detected	KhoaCDN	KhoaCDN	Pass
30		DC Gear Motor attached to Car wheels Combine	Adjust wheels speed	None	Set analog pulse = 150 to Motor 1	Wheel is moving forward, rising its speed and then stop after it react analog pulse value.	KhoaCDN	KhoaCDN	Pass

31		with H-bridge L298N		None	Set analog pulse = 255 to Motor 1	Wheel is moving forward, rising its speed to maximum	KhoaCDN	KhoaCDN	Pass
32				Wheel is moving	Set analog pulse = 0 to Motor 1	Wheel stop moving. Its speed dropping to zero	KhoaCDN	KhoaCDN	Pass
33				None	Set analog pulse = 0 to Motor 1	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
34				None	Set analog pulse = - 100 to Motor 1	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
35				None	Set analog pulse = 150 to Motor 2	Wheel is moving forward, rising its speed and then stop after it react analog pulse value.	KhoaCDN	KhoaCDN	Pass
36				None	Set analog pulse = 255 to Motor 2	Wheel is moving forward, rising its speed to maximum	KhoaCDN	KhoaCDN	Pass
37				Wheel is moving	Set analog pulse = 0 to Motor 2	Wheel stop moving. Its speed dropping to zero	KhoaCDN	KhoaCDN	Pass
38				None	Set analog pulse = 0 to Motor 2	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
39				None	Set analog pulse = - 100 to Motor 2	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
40				None	Set analog pulse	Wheel is moving	KhoaCDN	KhoaCDN	Pass

					= 150 to Motor 3	forward, rising its speed and then stop after it react analog pulse value.			
41				None	Set analog pulse = 255 to Motor 3	Wheel is moving forward, rising its speed to maximum	KhoaCDN	KhoaCDN	Pass
42				Wheel is moving	Set analog pulse = 0 to Motor 3	Wheel stop moving. Its speed dropping to zero	KhoaCDN	KhoaCDN	Pass
43				None	Set analog pulse = 0 to Motor 3	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
44				None	Set analog pulse = - 100 to Motor3	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
45				None	Set analog pulse = 150 to Motor 4	Wheel is moving forward, rising its speed and then stop after it react analog pulse value.	KhoaCDN	KhoaCDN	Pass
46				None	Set analog pulse = 255 to Motor 4	Wheel is moving forward, rising its speed to maximum	KhoaCDN	KhoaCDN	Pass
47				Wheel is moving	Set analog pulse = 0 to Motor 4	Wheel stop moving. Its speed dropping to zero	KhoaCDN	KhoaCDN	Pass
48				None	Set analog pulse = 0 to Motor 4	Wheel stand stills and do	KhoaCDN	KhoaCDN	Pass

						nothing			
49				None	Set analog pulse = - 100 to Motor 4	Wheel stand stills and do nothing	KhoaCDN	KhoaCDN	Pass
50			Movement	Attach all 4 motors to OAR wheels	Set analog pulse = 100 to Motor 1 Set analog pulse = 100 to Motor 2 Set analog pulse = 0 to Motor 3 Set analog pulse = 0 to Motor 4	OAR successfully move forward	KhoaCDN	KhoaCDN	Pass
51				Attach all 4 motors to OAR wheels	Set analog pulse = 0 to Motor 1 Set analog pulse = 100 to Motor 2 Set analog pulse = 100 to Motor 3 Set analog pulse = 0 to Motor 4	OAR successfully turn left	KhoaCDN	KhoaCDN	Pass
52				Attach all 4 motors to OAR wheels	Set analog pulse = 100 to Motor 1 Set analog pulse = 0 to Motor 2 Set analog pulse = 0 to Motor 3 Set analog pulse = 100 to Motor 4	OAR successfully turn right	KhoaCDN	KhoaCDN	Pass
53				Attach all 4 motors to OAR wheels Car is moving	Set analog pulse = 0 to Motor 1 Set analog pulse = 0 to Motor 2 Set analog pulse = 0 to Motor 3 Set analog pulse	OAR successfully stop	KhoaCDN	KhoaCDN	Pass

					= 0 to Motor 4				
54				Attach all 4 motors to OAR wheels	Set analog pulse = 0 to Motor 1 Set analog pulse = 0 to Motor 2 Set analog pulse = 100 to Motor 3 Set analog pulse = 100 to Motor 4	OAR successfully move backwards	KhoaCDN	KhoaCDN	Pass
55			Performance test	None	Set analog pulse = 100 Move forward. Calculate the average speed	The average forward speed is around 0.2 m/s	KhoaCDN	KhoaCDN	Pass
56				None	Set analog pulse = 200 Move forward. Calculate the average speed	The average forward speed is around 0.4 m/s	KhoaCDN	KhoaCDN	Pass
57				None	Set analog pulse = 255 Move forward. Calculate the average speed	The average forward speed is around 0.5 m/s	KhoaCDN	KhoaCDN	Pass
58				None	Turn a 45 degrees to the left. Calculate the turn time	Turn time is about 0.1 second	KhoaCDN	KhoaCDN	Pass
59				None	Turn a 45 degrees to the right. Calculate the turn time	Turn time is about 0.1 second	KhoaCDN	KhoaCDN	Pass
60				None	Turn a 90	Turn time is	KhoaCDN	KhoaCDN	Pass

					degrees to the left. Calculate the turn time	about 0.25 second			
61				None	Turn a 90 degrees to the right. Calculate the turn time	Turn time is about 0.25 second	KhoaCDN	KhoaCDN	Pass
62				None	Turn a 180 degrees to the left. Calculate the turn time	Turn time is about 0.5 second	KhoaCDN	KhoaCDN	Pass
63				None	Turn a 180 degrees to the right. Calculate the turn time	Turn time is about 0.5 second	KhoaCDN	KhoaCDN	Pass
64				None	Set analog pulse = 100 Move backward Calculate the average speed	The average backward speed is around 0.2 m/s	KhoaCDN	KhoaCDN	Pass
65				None	Set analog pulse = 200 Move backward. Calculate the average speed	The average backward speed is around 0.4 m/s	KhoaCDN	KhoaCDN	Pass
66				None	Set analog pulse = 255 Move backward. Calculate the average speed	The average backward speed is around 0.5 m/s	KhoaCDN	KhoaCDN	Pass

67		Bluetooth HC-05 connect with Arduino Uno R3	Connection	HC-O5 is off	Smartphone send request to nearby Bluetooth devices	HC-05 is not detected	KhoaCDN	KhoaCDN	Pass
68				HC-O5 is on	Smartphone send request to nearby Bluetooth devices	HC-05 is detected	KhoaCDN	KhoaCDN	Pass
69				Smartphone detected HC-05 device	Smartphone request to connect	HC-05 reply back, request PIN code from smartphone	KhoaCDN	KhoaCDN	Pass
70				Smartphone detected HC-05 device	Smartphone send wrong PIN code	HC-05 denied connect request	KhoaCDN	KhoaCDN	Pass
71				Smartphone detected HC-05 device	Smartphone send correct PIN code	HC-05 accepted connect request Piconet created	KhoaCDN	KhoaCDN	Pass
72			Send / receive messages	Connected with smartphone	Smartphone send data	Received all the data successfully	KhoaCDN	KhoaCDN	Pass
73				Connected with smartphone	HC-05 send data	Send all the data successfully	KhoaCDN	KhoaCDN	Pass
74				Connected with smartphone	Arduino read data received from HC-05	Arduino read data successfully	KhoaCDN	KhoaCDN	Pass

75			Performance Test	Connected with smartphone	Smartphone send a byte data. Calculate the time duration for receiving data	The time duration is around 0.05 second	KhoaCDN	KhoaCDN	Pass
76				Connected with smartphone	HC-05 sends a byte data. Calculate the time duration for sending data	The time duration is around 0.05 second	KhoaCDN	KhoaCDN	Pass
77				Connected with smartphone	Smartphone send data to oppose directions of HC-05	Data received successfully	KhoaCDN	KhoaCDN	Pass
78				Connected with smartphone	Smartphone is in different room with HC-05 Smartphone send data through the room wall	Data received successfully	KhoaCDN	KhoaCDN	Pass
79				Connected with smartphone	Moving HC-05 device far away from Smartphone. Calculate the working range.	HC-05 lost connection when the range > 50m	KhoaCDN	KhoaCDN	Pass
80				Connected with smartphone	Moving HC-05 device back to Smartphone direction	HC-05 connected again when the range < 50m	KhoaCDN	KhoaCDN	Pass

Table 36 – Hardware Components Test Cases

2. Test Obstacles Avoidance Algorithm

No.	Function			Pre-condition	Steps of testing	Expected result	Create by	Executed by	Result
	Large Function	Medium Function	Small Function						
1	Movement in reality	Move forward	Move forward	No obstacle	Checking obstacle Checking the distance Decide move	OAR move forward	KhoaCDN	KhoaCDN	Pass
2		Move forward	Move forward	Obstacle's distance larger than DistanceLimit	Checking obstacle Checking the distance Decide move	OAR move forward	KhoaCDN	KhoaCDN	Pass
3		Move forward	Move forward	Obstacle's distance smaller than DistanceLimit	Checking obstacle Checking the distance Decide move	OAR isn't move forward	KhoaCDN	KhoaCDN	Pass
4		Stop	Stop	Obstacle's distance smaller than DistanceLimit	Checking obstacle Checking the distance	OAR stop	KhoaCDN	KhoaCDN	Pass

					Decide move				
5		Turn directions after Stop	Turn Left	After OAR stop Left distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Left	KhoaCDN	KhoaCDN	Pass
6			Turn Left Diagonal	After OAR stop Left Diagonal distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Left Diagonal	KhoaCDN	KhoaCDN	Pass
7			Turn Right	After OAR stop Right distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Right	KhoaCDN	KhoaCDN	Pass
8			Turn Right Diagonal	After OAR stop Right Diagonal distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Right Diagonal	KhoaCDN	KhoaCDN	Pass
9			No turn	After OAR stop No direction distance is larger than DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR isn't turn	KhoaCDN	KhoaCDN	Pass
10		Move after	Move	After OAR can't	Checking recent	OAR go	KhoaCDN	KhoaCDN	Pass

		no turns	backward	turn. No distance in 4 directions larger than DistanceLimit	move Checking 4 others directions Decide move	backward			
11			Move after a backward movement	After OAR go backward Left Diagonal distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Left	KhoaCDN	KhoaCDN	Pass
12				After OAR go backward Left distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Left Diagonal	KhoaCDN	KhoaCDN	Pass
13				After OAR go backward Right Diagonal distance is larger than other directions and DistanceLimit	Checking recent move Checking 4 others directions Decide move	OAR turn Right Diagonal	KhoaCDN	KhoaCDN	Pass
14				After OAR go backward Right distance is larger than other directions	Checking recent move Checking 4 others directions	OAR turn Right	KhoaCDN	KhoaCDN	Pass

				and DistanceLimit	Decide move				
15		Decide move after turn directions	Move forward	Front distance is larger than DistanceLimit	Checking recent move Checking front direction.	OAR move forward	KhoaCDN	KhoaCDN	Pass
16			Turn Left Diagonal	After OAR turn direction Left Diagonal distance is larger than Left distance and DistanceLimit	Checking recent move Checking two directions (Left / Left Diagonal)	OAR turn Left Diagonal	KhoaCDN	KhoaCDN	Pass
17			Turn Left	After OAR turn direction Left distance is larger than Left distance and DistanceLimit	Checking recent move Checking two directions (Left / Left Diagonal)	OAR turn Left Diagonal	KhoaCDN	KhoaCDN	Pass
18			Turn Right	After OAR turn direction Right distance is largest in 3 directions and larger than DistanceLimit	Checking recent move Checking 2 Right & Front distance Decide move	OAR turn Right	KhoaCDN	KhoaCDN	Pass
19			Turn Right Diagonal	After OAR turn direction Right distance is largest in 3 directions and	Checking recent move Checking 2 Right & Front distance	OAR turn Right Diagonal	KhoaCDN	KhoaCDN	Pass

				larger than DistanceLimit	Decide move				
20	OAR stress test	Performance test	Power test	OAR is powered by AAA cell.	Set OAR run automatically l in 1 minutes	OAR ran normally.	KhoaCDN	KhoaCDN	Pass
21				OAR is powered by AAA cell	Repeat #21 test 50 times	OAR ran normally.	KhoaCDN	KhoaCDN	Pass
22				OAR is powered by AAA cell	Set OAR run automatically l in 30 minutes	OAR ran normally.	KhoaCDN	KhoaCDN	Pass
23				OAR is powered by AAA cell	Set OAR run automatically l in 1 hour	AAA cell ran out of power OAR stop moving	KhoaCDN	KhoaCDN	Pass
24			Environment test	Running in a big room	Set OAR run automatically l in 10 minutes	OAR run normally. No mistakes is found	KhoaCDN	KhoaCDN	Pass
25				Put OAR under the hill	Order OAR to run up the hill	OAR speed decrease. OAR turn degrees is decrease	KhoaCDN	KhoaCDN	Pass
26				Put OAR on the hill	Order OAR to run down the hill	OAR speed increase. OAR turn degrees is decrease.	KhoaCDN	KhoaCDN	Pass

Table 37 – Obstacles Avoidance Test Case

3. Test on Android application

No.	Function			Pre-condition	Steps of testing	Expected result	Create by	Executed by	Result
	Large Function	Medium Function	Small Function						
1	OAR android controller	Connection	Turn on	None	Turn on smartphone Turn on Bluetooth service Open OAR controller	Controller is ON	KhoaCDN	KhoaCDN	Pass
			Turn off	Controller is on	Click on Turn off button	Controller is OFF	KhoaCDN	KhoaCDN	Pass
2			Searching for devices	Controller program is on OAR is off	Click on “Search” button Search for OAR device on the list	No OAR device show on the list	KhoaCDN	KhoaCDN	Pass
3				Controller program is on OAR is on	Click on “Search” button Search for OAR device on the list	OAR device show on the list	KhoaCDN	KhoaCDN	Pass
4			Request connection	Controller program is on OAR is on Detected OAR on the search list	Sending connection request to OAR	OAR received request message	KhoaCDN	KhoaCDN	Pass
5				Sending connection request	OAR reply back, request PIN code for connection Enter PIN code : 1111	Wrong PIN code. Request denied. OAR request another PIN	KhoaCDN	KhoaCDN	Pass

						code			
6				Sending connection request	OAR reply back, request PIN code for connection Enter PIN code : 1234	Correct PIN Code. Piconet created	KhoaCDN	KhoaCDN	Pass
7		Manual control	Select mode	OAR connected	Select controller mode on the list. Click “Manual control”	Controller switch to Manual control mode	KhoaCDN	KhoaCDN	Pass
				OAR connected	Select controller mode on the list. Click “Automatic mode”	Controller switch to Automatic control mode UI switch to Automatic tab	KhoaCDN	KhoaCDN	Pass
				OAR connected	Select controller mode on the list. Click “Back-Tracing control”	Controller switch to Back-tracing control UI switch to Back-tracing tab	KhoaCDN	KhoaCDN	Pass
				OAR connected Mode selected	Click “Turn off”button	Controller is OFF	KhoaCDN	KhoaCDN	Pass
8			Movement control	OAR connected Manuel control mode	Click “↑” button Immediately release button.	OAR do nothings	KhoaCDN	KhoaCDN	Pass
9				OAR connected Manuel control mode	Click “↑” button Hold button Release the button	OAR go forward when the button is hold. Stop when	KhoaCDN	KhoaCDN	Pass

						the button is released			
10				OAR connected Manuel control mode	Click "←" button Immediately release button	OAR do nothings	KhoaCDN	KhoaCDN	Pass
11				OAR connected Manuel control mode	Click "←" button Hold button Release the button	OAR turn left when the button is hold. Stop when the button is released	KhoaCDN	KhoaCDN	Pass
12				OAR connected Manuel control mode	Click "→" button Immediately release button	OAR do nothings	KhoaCDN	KhoaCDN	Pass
13				OAR connected Manuel control mode	Click "→" button Hold button Release the button	OAR turn right when the button is hold. Stop when the button is released	KhoaCDN	KhoaCDN	Pass
14				OAR connected Manuel control mode	Click "↓" button Immediately release button	OAR do nothings	KhoaCDN	KhoaCDN	Pass
15				OAR connected Manuel control mode	Click "↓" button Hold button Release the button	OAR go backward when the button is hold. Stop when the button is released seconds	KhoaCDN	KhoaCDN	Pass
16				OAR connected	Click "↓" button	OAR go	KhoaCDN	KhoaCDN	Pass

				Manuel control mode	Hold button OAR's power is off	backward. Immediately stop when the power is off			
17				OAR connected Manuel control mode Bluetooth service on Smartphone suddenly stopped	Click "↓" button Hold button	Message appear : "Connection lost. Please re-connect" OAR do nothing	KhoaCDN	KhoaCDN	Pass
18				OAR connected Manuel control mode Bluetooth service on OAR suddenly stopped	Click "↓" button Hold button	Message appear : "Connection lost. Please re-connect" OAR do nothing	KhoaCDN	KhoaCDN	Pass
19				OAR connected Manuel control mode OAR is too far from Smartphone	Click "↓" button Hold button	Message appear : "Connection lost. Please re-connect" OAR do nothing	KhoaCDN	KhoaCDN	Pass
20		Automatic control	Select mode	OAR connected	Select controller mode on the list. Click "Manual control"	Controller switch to Manual control mode	KhoaCDN	KhoaCDN	Pass
				OAR connected	Select controller mode on the list. Click "Automatic mode"	Controller switch to Automatic control mode UI switch to	KhoaCDN	KhoaCDN	Pass

						Automatic tab			
				OAR connected	Select controller mode on the list. Click “Back-Tracing control”	Controller switch to Back-tracing control UI switch to Back-tracing tab	KhoaCDN	KhoaCDN	Pass
				OAR connected Mode selected	Click “Turn off “button	Controller is OFF	KhoaCDN	KhoaCDN	Pass
21			Movement control	OAR connected Automatic control mode	Not click on the any button	OAR run completely automatic	KhoaCDN	KhoaCDN	Pass
22				OAR connected Automatic control mode	Click on the “Stop” button	OAR stop moving	KhoaCDN	KhoaCDN	Pass
23				OAR connected Automatic control mode Bluetooth service on Smartphone suddenly stopped	Click on the “Stop” button	Message appear : “Connection lost. Please re-connect” OAR still run automatic	KhoaCDN	KhoaCDN	Pass
24				OAR connected Automatic control mode Bluetooth service on OAR suddenly stopped	Click on the “Stop” button	Message appear : “Connection lost. Please re-connect” OAR still run automatic	KhoaCDN	KhoaCDN	Pass
25		Back-tracing control	Select mode	OAR connected	Select controller mode on the list. Click “Manual control”	Controller switch to Manual control mode	KhoaCDN	KhoaCDN	Pass

				OAR connected	Select controller mode on the list. Click “Automatic mode”	Controller switch to Automatic control mode UI switch to Automatic tab	KhoaCDN	KhoaCDN	Pass
				OAR connected	Select controller mode on the list. Click “Back-Tracing control”	Controller switch to Back-tracing control UI switch to Back-tracing tab	KhoaCDN	KhoaCDN	Pass
				OAR connected Mode selected	Click “Turn off “button	Controller is OFF	KhoaCDN	KhoaCDN	Pass
26			Recording movement	OAR connected Back-tracing control mode	Not click on the any button	OAR run automatic Directions and time durations are saving after each move	KhoaCDN	KhoaCDN	Pass
27				OAR connected Back-tracing control mode OAR is already running.	Click the “Back tracing button”	OAR stop moving OAR turn 180 degrees to the right OAR begin tracing back	KhoaCDN	KhoaCDN	Pass
28				OAR connected Back-tracing control mode OAR is not running	Click the “Back tracing button”	OAR do nothing	KhoaCDN	KhoaCDN	Pass

29				OAR connected Back-tracing control mode OAR is already running Bluetooth service on Smartphone suddenly stopped	Click the “Back tracing button”	Message appear : “Connection lost. Please re-connect” OAR still run automatic	KhoaCDN	KhoaCDN	Pass
30				OAR connected Back-tracing control mode OAR is not running Bluetooth service on Smartphone suddenly stopped	Click the “Back tracing button”	Message appear : “Connection lost. Please re-connect” OAR do nothing	KhoaCDN	KhoaCDN	Pass
31				OAR connected Back-tracing control mode OAR is already running Bluetooth service on OAR suddenly stopped	Click the “Back tracing button”	Message appear : “Connection lost. Please re-connect” OAR still run automatic	KhoaCDN	KhoaCDN	Pass
32				OAR connected Back-tracing control mode OAR is not running Bluetooth service on OAR suddenly	Click the “Back tracing button”	Message appear : “Connection lost. Please re-connect” OAR do nothing	KhoaCDN	KhoaCDN	Pass

				stopped					
33				OAR connected Back-tracing control mode OAR is running EEPROM memory is running low	Not click on the any button	Message appear : “Low memory” Memory left < 100 Address slot	KhoaCDN	KhoaCDN	Pass
34				OAR connected Back-tracing control mode OAR is running EEPROM memory is full	Not click on the any button	Message appear : “Full memory” EEPROM is full OAR stop moving and automatically tracing back	KhoaCDN	KhoaCDN	Pass

Table 38 – Android Application Test Cases

4. Test Back-Tracing Algorithm Application

No.	Function			Pre-condition	Steps of testing	Expected result	Create by	Executed by	Result
	Large Function	Medium Function	Small Function						
1	Back-Tracing function	Path recording	Movement Direction saving	None	OAR move forward. Then stop. Save the direction	“Forward” direction save successfully	KhoaCDN	KhoaCDN	Pass

2				None	OAR move backward. Then stop. Save the direction	“Backward” direction save successfully	KhoaCDN	KhoaCDN	Pass
3				None	OAR turn right Then stop. Save the direction	“Right” direction save successfully	KhoaCDN	KhoaCDN	Pass
4				None	OAR turn left Then stop. Save the direction	“Left” direction save successfully	KhoaCDN	KhoaCDN	Pass
5				None	OAR turn right diagonal. Then stop. Save the direction	“Right Diagonal” direction save successfully	KhoaCDN	KhoaCDN	Pass
6				None	OAR turn left diagonal. Then stop. Save the direction	“Left Diagonal” direction save successfully	KhoaCDN	KhoaCDN	Pass
7			Movement duration saving	None	OAR move forward. Then stop. Save the duration.	“Forward” duration save successfully	KhoaCDN	KhoaCDN	Pass
8				None	OAR move backward. Then stop. Save the duration.	“Backward” duration save successfully	KhoaCDN	KhoaCDN	Pass
9				None	OAR turn right Then stop.	“Right” duration save	KhoaCDN	KhoaCDN	Pass

					Save the duration.	successfully			
10				None	OAR turn left Then stop. Save the duration.	“Left” duration save successfully	KhoaCDN	KhoaCDN	Pass
11				None	OAR turn right diagonal. Then stop. Save the duration.	“Right Diagonal” duration save successfully	KhoaCDN	KhoaCDN	Pass
12				None	OAR turn left diagonal. Then stop. Save the duration.	“Left Diagonal” duration save successfully	KhoaCDN	KhoaCDN	Pass
			Path tracing	OAR move forward for 5 seconds. Then stop. Direction and duration are saved	Get the saving values Direction = Forward Duration = 5 Order OAR to move Forward Then stop after 5 seconds	The tracing run corrected	KhoaCDN	KhoaCDN	Pass
13				OAR turn Left. Then stop. Direction and duration are saved	Get the saving values Direction = Left Duration = 0.25 Order OAR to turn Left Then stop after 0.25 seconds	The tracing run corrected	KhoaCDN	KhoaCDN	Pass
14				OAR turn Right.	Get the saving values	The tracing run corrected	KhoaCDN	KhoaCDN	Pass

				Then stop. Direction and duration are saved	Direction = Right Duration = 0.25 Order OAR to turn Left Then stop after 0.25 seconds				
15				OAR turn Left Diagonal. Then stop. Direction and duration are saved	Get the saving values Direction = Left Diagonal Duration = 0.1 Order OAR to turn Left Then stop after 0.1 seconds	The tracing run corrected	KhoaCDN	KhoaCDN	Pass
16				OAR turn Right Diagonal. Then stop. Direction and duration are saved	Get the saving values Direction = Right Diagonal Duration = 0.1 Order OAR to turn Left Then stop after 0.1 seconds	The tracing run corrected	KhoaCDN	KhoaCDN	Pass
17				OAR move backward for 5 seconds. Then stop. Direction and duration saved	Get the saving values Direction = Backward Duration = 5 Order OAR to move Forward Then stop after 5 seconds	The tracing run corrected	KhoaCDN	KhoaCDN	Pass
18				OAR move and turn	Get the saving values from the	The tracing run corrected	KhoaCDN	KhoaCDN	Pass

				around more than 5 times. Directions and durations are saved	latest to the newest. Then order OAR to move.				
19		Memory control	Memory management	None	Store 50 directions in the EEPROM.	Data store successfully	KhoaCDN	KhoaCDN	Pass
20				TC ??	Read the EEPROM data	All datas from TC?? Are saved from address 0 to address 49	KhoaCDN	KhoaCDN	Pass
21				None	Store 512 directions in EEPROM	Data store successfully	KhoaCDN	KhoaCDN	Pass
22				TC ??	Read the EEPROM data	All datas from TC?? Are saved from address 0 to address 511	KhoaCDN	KhoaCDN	Pass
23				None	Store 513 directions in EEPROM	The last direction (#513) is not saved Message Box appear : “Out of memory”	KhoaCDN	KhoaCDN	Pass
24				None	Store 50 time durations in the EEPROM.	Data store successfully	KhoaCDN	KhoaCDN	Pass
25				TC ??	Read the EEPROM data	All datas from TC?? Are saved from address	KhoaCDN	KhoaCDN	Pass

						512 to address 561			
26				None	Store 512 time durations in EEPROM	Data store successfully	KhoaCDN	KhoaCDN	Pass
27				TC ??	Read the EEPROM data	All datas from TC?? Are saved from address 512 to address 1023	KhoaCDN	KhoaCDN	Pass
28				None	Store 513 time durations in EEPROM	The last time durations (#513) is not saved Message Box appear : “Out of memory”	KhoaCDN	KhoaCDN	Pass
29				Address #0 already store a data	Save new data in Address 0 of EEPROM	Old data is replaced with the new one	KhoaCDN	KhoaCDN	Pass
			Memory performance test	None	Save data in EEPROM Calculate the duration for writing data on EEPROM	The duration is around 33mS for each byte	KhoaCDN	KhoaCDN	Pass
30				None	Clear EEPROM Calculate the duration for clearing EEPROM	The duration is around 4 seconds	KhoaCDN	KhoaCDN	Pass
31				None	Save data in EEPROM Turn off OAR Turn on OAR again	Data still remains safety in EEPROM	KhoaCDN	KhoaCDN	Pass

					Read EEPROM's data				
32				Address #0 already store a data	Save new data in Address 0 of EEPROM Calculate the duration for writing data on EEPROM	The duration is around 4 seconds	KhoaCDN	KhoaCDN	Pass

Table 39 – Back-Tracing Algorithm Test Cases

System User's Manual (SUM)

I. System Requirement

Android smartphone, which support:

- Bluetooth 2.0 or higher.
- Android 4.0 or higher.

II. Install/Uninstall Program

1. Install

1 – Go to setting menu in your phone.

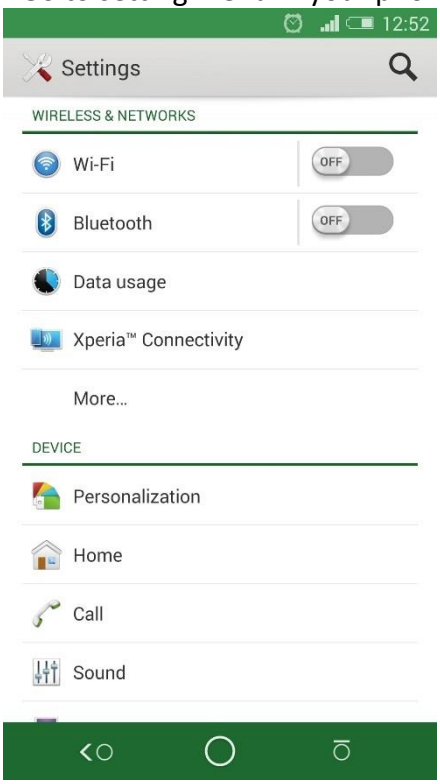


Figure 95 – Phone Setting Menu

2 – Scroll down to "Security" menu.

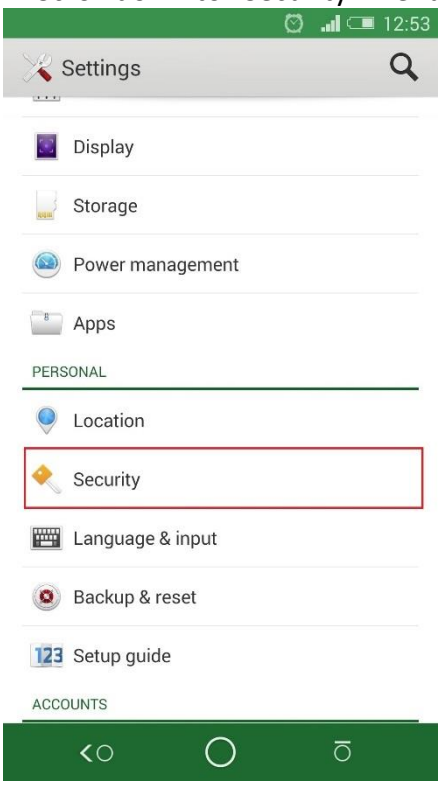


Figure 96 – Phone Security Menu

3 – Inside "Security" menu, scroll down and tick "Unknown Source".

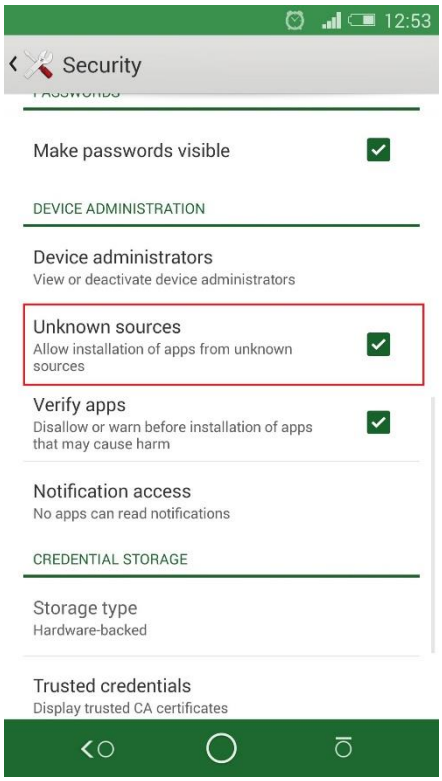


Figure 97 – Unknown Source Option

4 – After that, using system explorer navigate to APK file "CarRemoteControlBeta1".

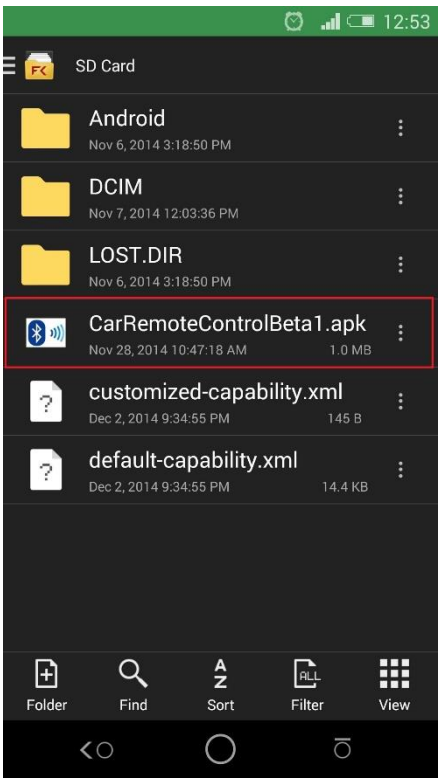


Figure 98 – Installation File

5 – Tap it and choose install, after that, click finish to complete installation process.

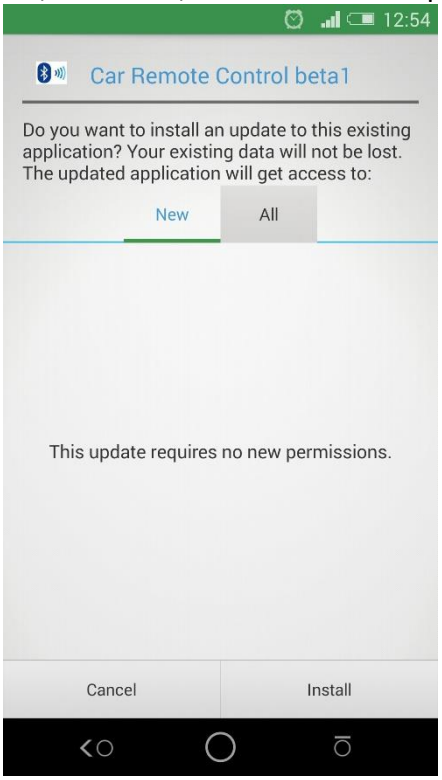


Figure 99 – Installation Process

2. Uninstall

1 – Go to setting menu and choose "App".

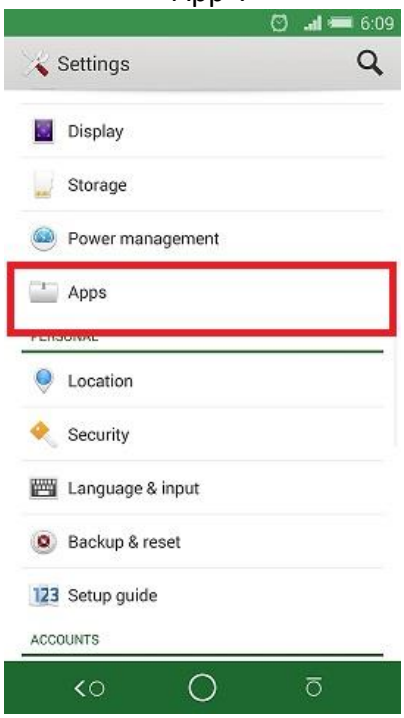


Figure 100 – App Menu

2 – Scroll and find "Car Remote Control beta1".

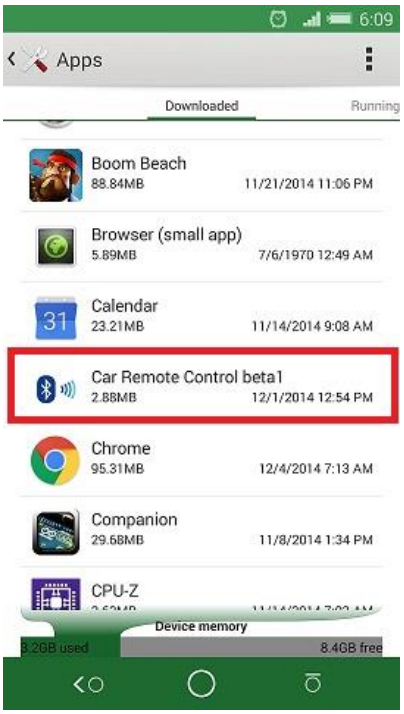


Figure 101 – Installed Application

3 – Choose "uninstall".

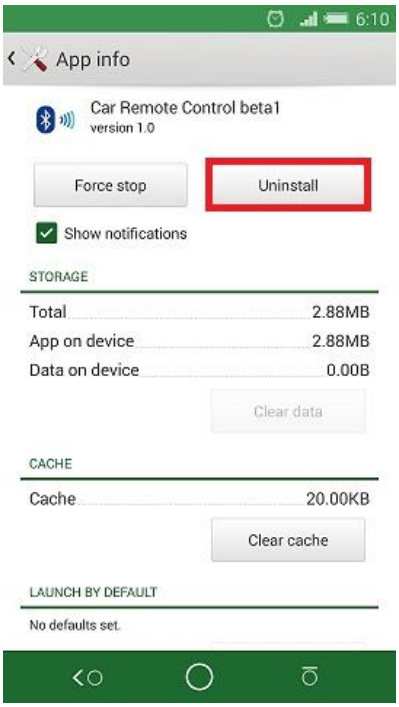


Figure 102 – Uninstall Application

3. Configuration

3.1 Paring with Bluetooth HC-05

1 – Go to setting menu in your phone.

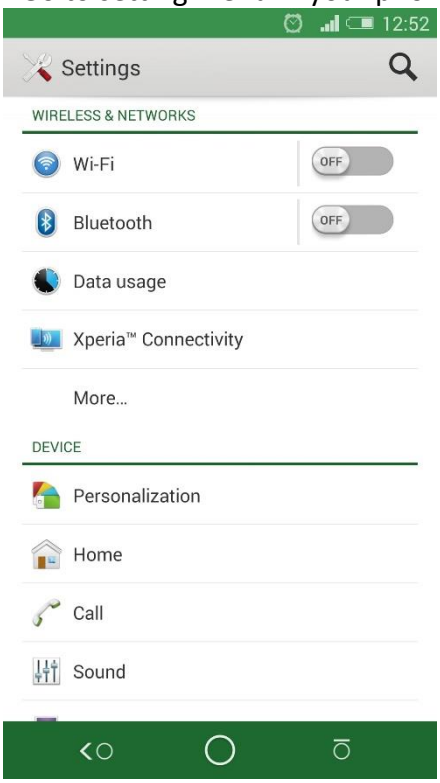
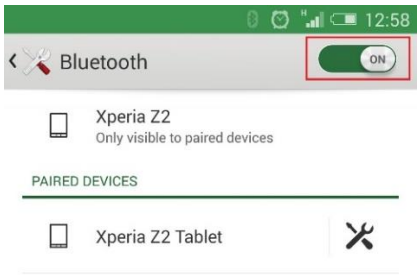


Figure 103 – Phone Setting Menu

2 – Choose "Bluetooth" and turn it on afterward.



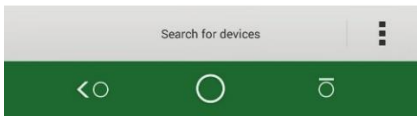


Figure 104 – Turn on Bluetooth

3 – Press "Scan for devices".

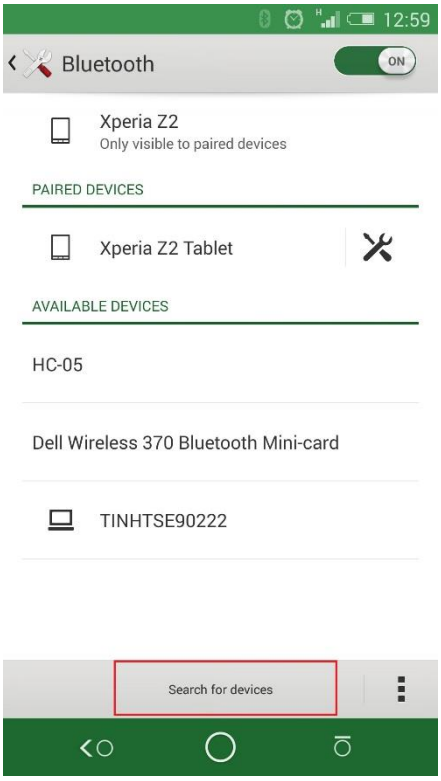


Figure 105 – Scan New Device

4 – "HC-05" will appear, tap it.

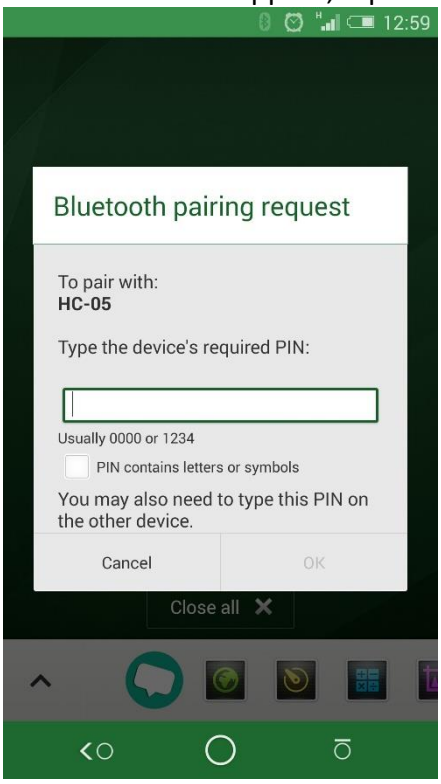


Figure 106 – Paring Request

5 – System will ask you to input paring code for HC-05. Default paring code is "1234". Enter paring code and tap "OK".

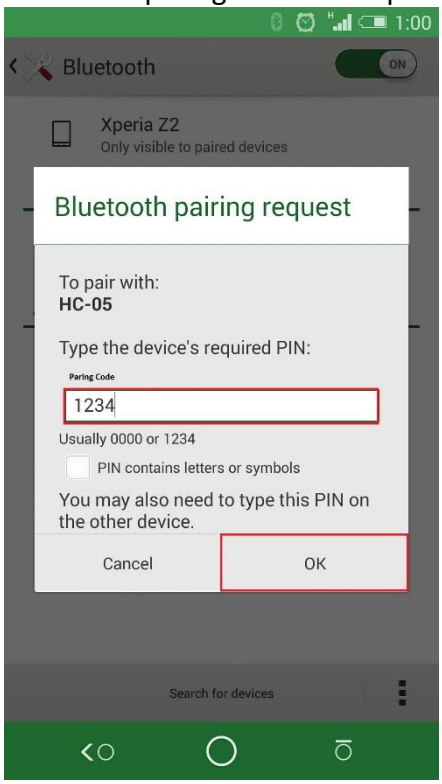
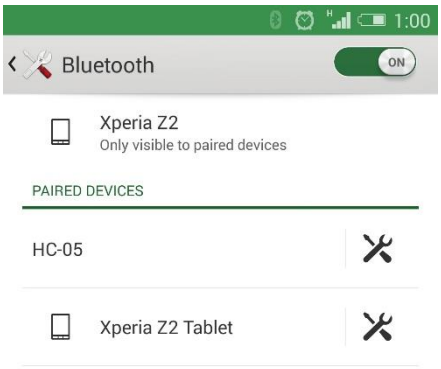


Figure 107 – Paring Code

6 – After that HC-05 will appear in "Paired device" section. You do not have to pair between your device and HC-05 if you have completed paring process above.



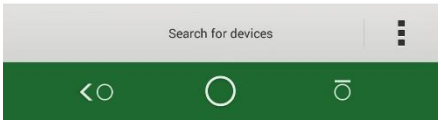


Figure 108 – Paired Device Menu

4. How To Use Application

1 – Go to your phone application drawer, choose "Car Remote Control Beta1".

Figure 109 – Application

2 – Main application shown.

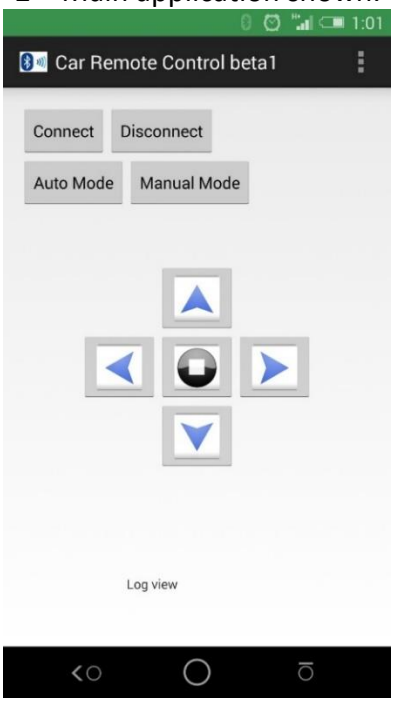


Figure 110 – Main Application Layout

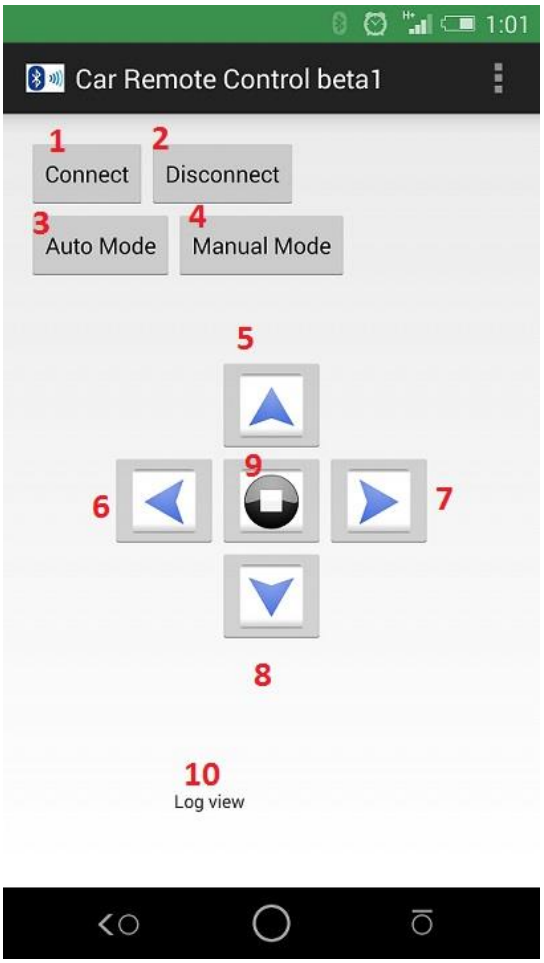


Figure 111 – Application Layout Details

No.	Button	Function
1	Connect	Touch to connect to Arduino through Bluetooth module
2	Disconnect	Touch to disconnect from Arduino
3	Auto Mode	Touch to switch Robot Car to auto navigation mode
4	Manual Mode	Touch to switch Robot Car to manual control mode
5		Touch to control Robot Car to goes forward

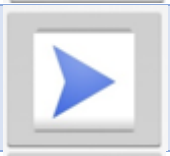
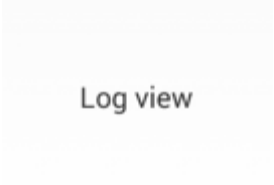
6		Touch to control Robot Car to goes left
7		Touch to control Robot Car to goes right
8		Touch to control Robot to goes backward
9		Touch to stop Robot Car
10		Display value that send back from Arduino

Table 40 – Android application buttons and function

References

- Arduino reference page, URL: http://arduino.cc/en/Reference/HomePage
- Arduino product information page, URL: http://arduino.cc/en/Main/Products
- Arduino forum, URL: http://forum.arduino.cc/
- Android Bluetooth development, URL
http://developer.android.com/guide/topics/connectivity/bluetooth.html