Autonomous Radio Controlled Car

Tanu Nanda Prabhu, *Student, University of Regina*

***Abstract*—This project summarizes suggested modifications made to the existing project. The modifications made to the existing project includes inculcating sensors, actuators, tweaking the existing hardware design and finally documenting all the hardware and software implementations within the given time constraint. In this project, the old radio-controlled car will be made autonomous by adding an ultrasonic sensor to it. This ultrasonic sensor helps in avoiding obstacles in front of the car by emitting ultrasonic sound waves. This autonomous car detects the obstacle and avoids collision by changing its direction of movement. The ultrasonic sensor sends the data to the Arduino micro-controller and then the micro-controller with the help of a motor-shield controls the movements of the wheels via DC motors. Further, the ultrasonic sensor signals the DC motors what action to perform, based on that the DC motor operates by going forward, reverse, left or right. The motor-shield acts as an H-bridge by controlling the DC motors simultaneously.**

***Index Terms*—project, report, computer science, CS 807**

I. Introduction

THE autonomous car is one of the greatest innovations of this era. There has been a large amount of research that has gone into making it convenient enough for customers to rely on. The autonomous car is a car which runs automatically by sensing the environment around it and capable of moving on its own with no one controlling it. Although this project might be a perfect example to understand the autonomous functionality and its outcome. The prime feature of this autonomous car in this project is avoiding obstacles on its way and moving on. This feature is very helpful, and it is the base for many large projects such as collision detection, parking aid system [PAS] and robotics [1]. The main aim of this project is to develop a car that will move according to the assigned code but finds a free space, navigating from any obstacle on its way. There are many ways to implement this autonomous functionality such as by using computer vision, radars, sonars, GPS and many more but one of the best approaches is using ultrasonic sensors. The reason for choosing ultrasonic sensors is that they are low cost and has high ranging capabilities and easy to use. This autonomous car gets the information from the surrounding area through the ultrasonic sensor mounted in front of the car. This autonomous concept although is applied on a radio-controlled toy car which has a huge chassis and two DC motors attached to it. The DC motor attached at the front will make the car either to move left or right, the one on the back helps to move either forward or backward. Now this chassis must be revamped by hooking sensors to it, adding tail lights, and controlling it via Arduino Uno with motor-shield attached to it.

II. Problem Definition

The problem definition is such that building an autonomous radio-controlled car. Likewise, the building process of the car comprises an ultrasonic sensor to detect the obstacles and avoid the collision, motor-shield (dual H-bridge) to control two DC motors simultaneously and adding cool lighting effects to the car. To summarize, this project transforms a manually operated radio-controlled car into an autonomous car.

III. Background

The main aim of this project is to recreate the functionality of an existing framework for a radio-controlled car by making it an autonomous car. There were a couple of projects that were taken as a reference while building this project such as Marcelo Jos Rovai on instructables.com shows the dismantling of one of the Radio-controlled cars, racking its fundamental parts and supplanting the original embedded hardware by an Arduino controlled remotely through an Android device [2]. This project is used as a reference for hacking the radio-controlled car and making it an Arduino controlled car. There are many tutorials and projects that can be found online such as Arduino Self-driving Cars from instructables.com [3] in which they use an ultrasonic sensor to detect an object to avoid a collision. The major drawbacks of these projects are that they use four DC motors to control the movement of the car and no differential steering concept was used to control either left or right movement. Differential steering is the means to steer the vehicle either left or right. However, in this project, the RC car has a differential steering, and it was hard to control the wheels at the front. In most of the online projects, they built the car from scratch or assembled the parts from the DIY kit, and so they built a smaller car, but in this case, the chassis is already built and thus must be reused in such a way that it fits for this project. This is how this project differs from the existing projects available online. The obstacle avoiding robot from create.arduino.cc is a similar do-it-yourself project whose focus was to build obstacle avoiding robot. It was also taken as a reference for building this current project. The major advantage of this project was that it had a smaller chassis and contained only 2 small DC motor with no differential steering feature. The current project is a combination of all the other projects.

IV. Design Description

A. Overview

This project requires an ultrasonic sensor to detect objects around it, Arduino UNO for logic processing, triggering motors, LED, an Adafruit motor-shield is used to control two motors simultaneously and LEDs are used for brake lights.

B. Design Process

The design process of this project comprises the components such as ultrasonic sensor, LED's, motor-shield, DC motors, and the power source are connected to the Arduino board, whereas the motor shield is stacked on top of the Arduino and two DC motors and a power source is connected to the motor shield. Fig 1 in the appendix section represents the block diagram of the project which gives a clear understanding of how the components are connected.

The schematics and the circuit diagram are represented as Fig 2 and Fig 3, they are both given in the appendix section. The breadboard is only used to connect the ultrasonic sensor and the LEDs. The ultrasonic sensor and the LEDs are connected to the digital pins on the Arduino board. In the breadboard (schematics) view, the Arduino board is below it and on top of it is the motor-shield which is connected to the Arduino. The motor-shield connection is trivial because it comprises 2 motor ports (M1, M2, and M3, M4) and two servo ports along with VCC and GND. In this project, only M1 and M2 ports of the motor-shield are used. The reason for choosing M1 and M2 ports is that both ports operate at 64 kHz and 8 kHz, whereas the M3 and M4 are 2khz and 1khz. The higher the kHz the higher the speed (rotations) of the motor can be. When it comes to the connection of the components the ultrasonic sensor is connected to the digital pin 3, 4, 5 and 6 of the Arduino board. There was no need to solder any wire or component. The Motor-shield comprises of screws which need to be tightened after connecting the wires. But we must be careful that all the pins of the motor-shield coincide with the Arduino pins, or else the motor-shield would be burnt or wont function properly. Finally, we must connect the battery source to the VCC and GND ports of the motor shield. Here's one thing that should be taken care of, since this radio-controlled car is huge both the Arduino and the motor shield must be supplied by an external power supply i.e (Arduino must be provided 5V and Motor-shield with 10V) as indicated below in the schematics as well as the circuit diagrams.

C. Build Process

The build process of this autonomous car was a bit challenging due to the size of the car. First, when all the components were connected the rear DC motors rotated rapidly. After decreasing the speed of the motor from 255 to 100, this issue was solved. Because of this, the motor-shield turned red-hot, the reason for this could have been because of the high-power supply that the motor shield was drawing from the battery or due to reducing the speed of the motor. If the power supply was decreased, the DC motor would then stop rotating. This was a trade-off that was taken into consideration between more and less power supply to DC Motors. This was the setback of the project but eventually, this problem was solved by removing the batteries and maintaining 10V overall. Second, the differential steering problem was solved by tweaking the code which can be seen in the appendix section. To control the motor movements, the Adafruit motor shield library was used. The overall code of the project is simple enough to understand most of it, which is also included in the appendix section. For further details about the project such as the build process, design process, libraries, and code refers to Github repository which can be found in the references section down below [4].

D. Use

The working of this project is shown below. First, the user must insert the batteries into the battery holder and place the car on the ground. Once the car is powered up, it moves forward unless it finds an obstacle, if it does, the car then stops and goes backward and eventually turns left. If it finds no obstacle, it moves to the right. There are two ways that the user should power up the car.

1) The Arduino UNO must be supplied with 5V of battery, but this instead cannot move the car. 2) Hence, we must provide an extra 10V battery supply, this is done by connecting the batteries to the motor shield.

Below are the steps that gives a clear understanding of how the device works, for example:

1) Upload the code to the Arduino.
2) Power both the Arduino and the Motor-shield via batteries.
3) Then the car moves to the front automatically, it then stops if it finds an obstacle, and then goes back fora bit, and then turns to the left and again checks for the obstacle, otherwise, it turns to the right this time and further continues to go in the same direction.
4) This would be cool to watch because it automatically does all these things by itself.
5) Last, don't forget to remove the batteries from the battery holder, sometimes both the Arduino and the Motor-Shield can become hot after an immense usage.

V. Evaluation

A. Overview

The method that was used to test the autonomous radio-controlled car was: Testing the ultrasonic sensor by blocking it via bare hands, just to make sure that the car stopped, and eventually changes its direction. This approach was used because if it was directly tested on the ground then there are high chances that the car might bump against the wall and get destroyed because of inaccurate values from the sensor. Initially, the car could not move because of the cheap and low powered 10V battery that came with the kit. Then after updating the batteries (using 10V AA), the car started to function as expected.

B. Prototype

The prominent feature of the project was that the car moves on its own with no one controlling it. Although there are many radio-controlled cars available in the market, this promising feature outperforms the others. Initially, the plan was to implement the path-finding algorithm for the car, but because of the time constraint, this was unable to implement. This is one of the ideas that differ from the final solution.

C. Testing and Results

Before building the final instrument, there was an attempt made to use the L298 Dual H-bridge instead of L293D. But the L298 dual H-bridge used to get hotter in less time when compared to the L293d. The reason for this is unknown. Next is that L293d motor-shield could be easily stacked on top of the Arduino Board with less or no wiring. The next approach was testing the ultrasonic sensors so that the DC motors responded to its values. Initially, the values of the ultrasonic sensors were not accurate, the code was designed in a way that when the car finds an obstacle; it goes back and then goes left. Many a time the car just bumped into the wall because of the inaccurate values from the sensor, due to this the ultrasonic sensors were replaced by a new one. The last approach was to try all combinations of the batteries needed for the car. This was because the more batteries that were added the heavier the car became.

D. Assessment

During the test, everything was up to the expectations. There is one drawback of this project, the car miserably fails to move on the ground. This may be because of less torque generated by the DC motors, or because of the heavyweight of the batteries. The car performs as expected when its not on the ground. A couple of things were burnt during the building of this project, the motor-shield were burnt twice due to the higher power supply, the batteries were shorted, both the DC motors in the car were burnt due to the massive power and excessive usage. The strength of this car is the autonomous functionality, the weakness of this car is that it fails to move on the ground.

E. Next Steps

There are a few modifications that can be done to take this car to the next level that will likely to be used in the future. These include completely redesigning the power management system without negatively impacting the micro-controller and the motor shield. Also, a new chassis can be deployed which is lighter compared to the existing one. Adding a better ultrasonic sensor to the car can give more accurate results. Also implementing path-finding algorithms so that the car knows where its going. This can be a handy device especially for the blind, as it could help blind people to navigate against obstacles, this can be done by attaching one end of the belt to the car and the other end can be held by the blind person. Also, this can be used in military applications. This can be done by sending these cars or robots to detect land mines, this could save many soldiers' lives.

VI. MILESTONES

Overall 6 milestones were required to build this prototype. They are listed below:

1) Milestone 1: Understanding everything about an RC car, gathering all the materials. This milestone was met successfully, as my professor gave me the chassis of a huge RC car to experiment on. The materials which were gathered to build this autonomous car were Arduino Uno, Adafruit Motor-shield L239D, Ultrasonic sensors, LEDs, batteries, and some jumper wires.

2) Milestone 2: Hacking electronics. This milestone was completed. The only electronic that was found on this car was the electronic module and this was replaced by an Arduino Uno. This is responsible for logic processing, triggering the motors via motor-shield, LED and an ultrasonic sensor. This car needs 10V of power wherein we need to feed it at least 8-9 AA batteries, as the DC motors in this car need a lot of power. The rear DC motor needs a lot of power when compared to the one in the front because it takes the entire load of the car and moves it either forward or backward. For this task, the Arduino Uno 5V can be used and a battery or a power bank can power it up. Initially, hacking the electronics was a challenging task, but with the help of Hacking the RC can with Arduino tutorial helped me to solve this task.

3) Milestone 3: Initial Assembly but no coding involved. This milestone was met successfully. The initial assembly involved connecting the Arduino Uno to the car, stacking a motor shield on top of the Arduino, connecting all the wires from the two DC motors to the motor shield. Here M1 and M2 ports were used to connect the DC motors. Then the ultrasonic sensor was attached to the front of the car and its pins were connected to the Arduino board. In the same way, the LEDs were attached to the rear end of the car to act as the brake lights. Finally, the battery holder was also attached at the bottom of the car and was directly connected to the motor shield. No coding was involved in this milestone.

4) Milestone 4: Sensors and Actuators are being calibrated and positioned in the car for functioning. This milestone was completed successfully. In this prototype sensors and actuators are categorized as Ultrasonic sensors and DC motors. Calibrating the ultrasonic sensors was a daunting task because the values returned by the ultrasonic sensor were fluctuating and were not accurate, because these sensors were cheaply built and were already present in the kit, so the values were not uniform. The ultrasonic sensor was positioned in the front of the car. Secondly, the DC motors were calibrated, and their results were acceptable, so the rear DC motor was easy to handle because of its forward and backward motion. However, the front DC motors were very hard to control because of its left and right movement (differential steering). When it turns to the left it was unable to align to the center, the same thing was happening with the right turn as well. This RC car did not have any spring in its steering system which helps the wheels to align themselves to the center. Later this issue was solved with the help of small code snippets.

5) Milestone 5: A complete working demo of the prototype is ready. This milestone was completed successfully. The working of the demo was complete, the ultrasonic sensor and the DC motors performed well in all conditions. Except there was a major drawback in this demo. The car performed extremely well when it was not on the ground (when it was held in the air), when the car was released on the ground it was struggling to move forward, giving a gentle push also did not help the car, but there was a weird noise that the DC motor was making, and sometimes the motor shield was burning hot. Maybe this was because of high battery power or even a lack of torque from the DC motor.

6) Milestone 6: Making the car know where it's going and what direction it's moving, avoiding random direction using some active logic such as pathfinding algorithms. This

milestone could not be met successfully because of the time constraint. Initially, it took over 3-4 weeks just to match the battery configuration of the car because the more the battery added the more the load the car had to carry. The second task was to fix the steering issue of the car, this took two weeks more and on top of that, the steering of the car was not responsive. Hence, the stretch goal could not be met, but the car goes in a random direction right now. But in the future, this can be further enhanced such that the car goes in a specific direction.

VII. Team Roles

This project is done only by me (Tanu N Prabhu) and no teammates were involved in this project. All the project work including project selection, research, hardware design, prototype testing, GitHub documentation/writing the paper and programming all in whole wasdone solely by me in order to perform smooth functioning of the prototype.

VIII. Conclusion

The main motivation for choosing this project was, the increase in autonomous self-driven cars today in the real world, and it was a relevant area of interest to work with radio-controlled cars. Everything today is on the verge of being automated, keeping this in mind along with the existing project helped in arriving at a conclusion of building an autonomous radio-controlled car. Also, this project would be interesting to see a radio-controlled car automatically runs with no one controlling it. The proposed project when completed turns out to be a more sophisticated and easier to operate with minimum requirements and making it an autonomous fully functioning car. Likewise, the goal of this project was to make the RC car fully autonomous such that making the car to run on its own and avoiding obstacles on its way, this concept can be further implemented in real time cars if enhanced.

Appendix A
Diagrams and Pictures

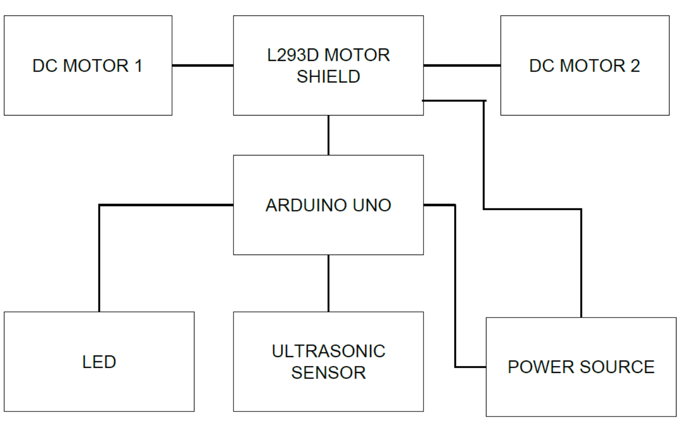

Fig. 1. The block diagram which shows the essential components used in this project

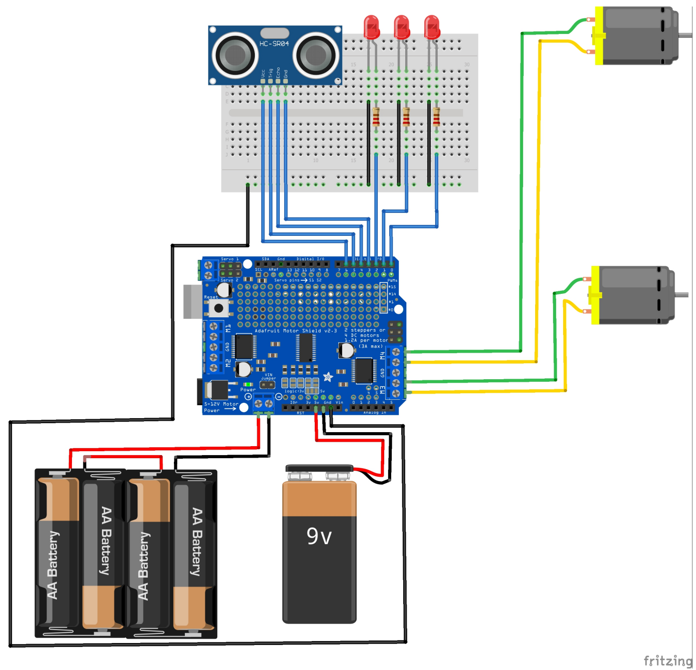

Fig. 2. The schematics of the autonomous radio controlled car

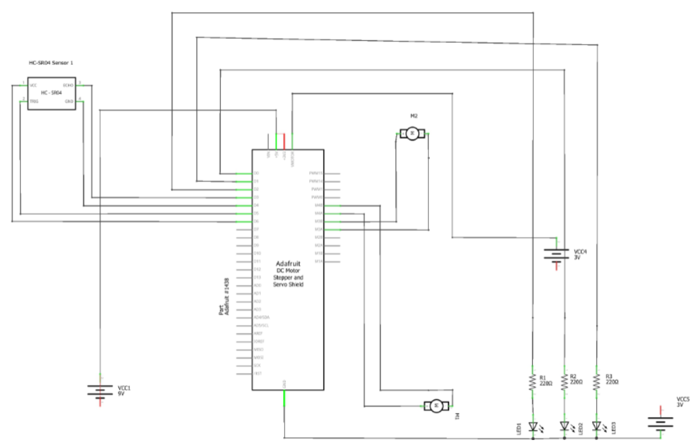

Fig. 3. The circuit diagram of the autonomous radio controlled car

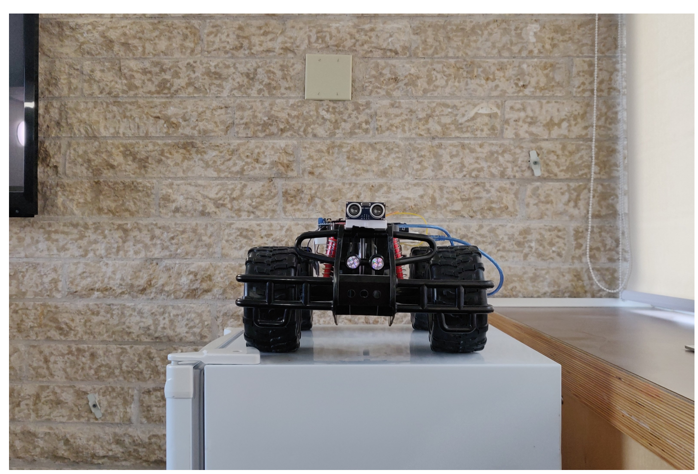

Fig. 4. The front view of the car

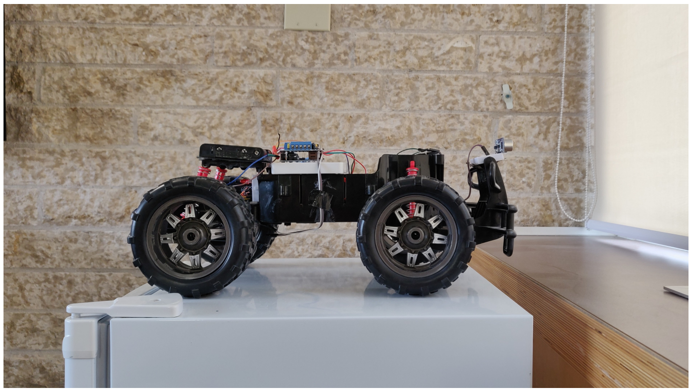

Fig. 5. The side view of the car

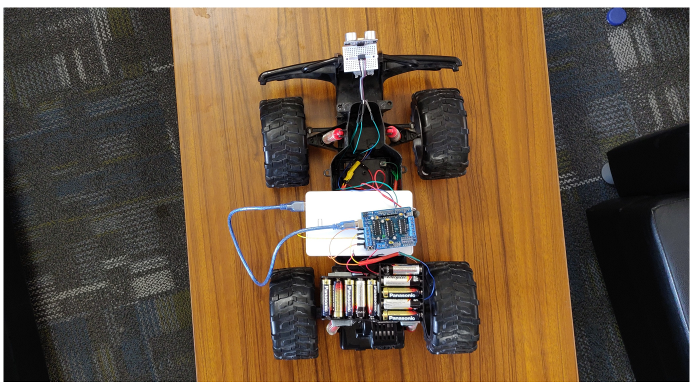

Fig. 6. The top view of the car

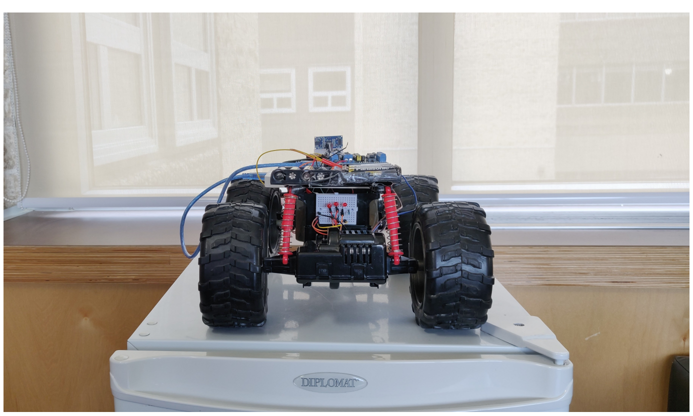

Fig. 7. The rear view of the car

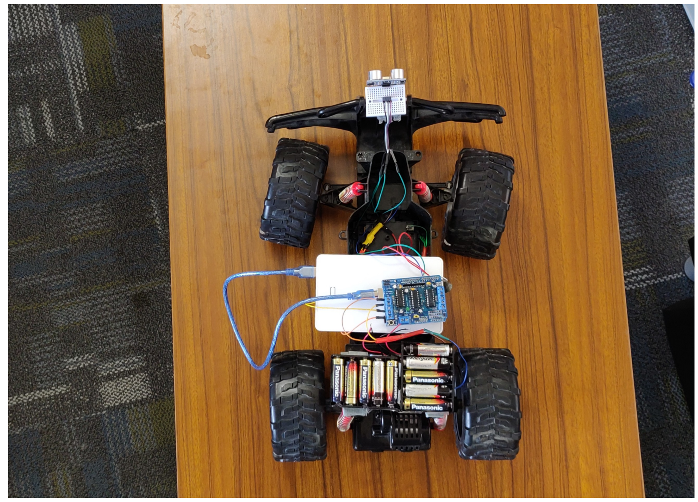

Fig. 8. The differential steering of the car in left direction

Assembly List

Label	Part Type	Properties
HC-SR04 Sensor 1	HC-SR04 Ultrasonic Distance Sensor	chip LM324; variant variant 1
LED1	Red (633nm) LED	package 1206 [SMD]; color Red (633nm)
LED2	Red (633nm) LED	package 1206 [SMD]; color Red (633nm)
LED3	Red (633nm) LED	package 1206 [SMD]; color Red (633nm)
M1	DC Motor	
M2	DC Motor	
Part	Adafruit Motor/Stepper/Servo Shield	variant variant 1; part # Adafruit #1438
Part1	Arduino Uno (Rev3) - ICSP	type Arduino UNO (Rev3) - ICSP (w/o icsp2)
R1	220Ω Resistor	pin spacing 400 mil; tolerance ±5%; resistance 220Ω; bands 4; package THT
R2	220Ω Resistor	pin spacing 400 mil; tolerance ±5%; resistance 220Ω; bands 4; package THT
R3	220Ω Resistor	pin spacing 400 mil; tolerance ±5%; resistance 220Ω; bands 4; package THT
VCC1	Battery block 9V	voltage 9V
VCC2	Battery	voltage 3V
VCC3	Battery	voltage 3V

Fig. 9. The assembly list of materials required during building the prototype

Shopping List

Amount	Part Type	Properties
1	HC-SR04 Ultrasonic Distance Sensor	chip LM324; variant variant 1
3	Red (633nm) LED	package 1206 [SMD]; color Red (633nm)
2	DC Motor	
1	Adafruit Motor/Stepper/Servo Shield	variant variant 1; part # Adafruit #1438
1	Arduino Uno (Rev3) - ICSP	type Arduino UNO (Rev3) - ICSP (w/o icsp2)
3	220Ω Resistor	pin spacing 400 mil; tolerance ±5%; resistance 220Ω; bands 4; package THT
1	Battery block 9V	voltage 9V
2	Battery	voltage 3V

Fig. 10. The shopping list of materials required during building the prototype

APPENDIX B
SOURCE CODE

```
// program: Autonomous_Car
// author: Tanu. N. Prabhu
// course: CS 807
// date: 17/04/2019
// Project Report
/* description: This programs is used to give
    an autonomous functionality to the
    manually controlled car, with the help of
    ultrasonic sensors. Below is the code
    which explains each and every line.*/
//--------------------------------------------

#include <AFMotor.h>

AF_DCMotor motor1(1);// create motor #1,
    64KHz pw
AF_DCMotor motor2(2);// create motor #2, 8KHz
    pw

int vcc = 3;  //attach pin 2 to vcc
int trig = 4; //attach pin 3 to Trig
int echo = 5; //attach pin 4 to Echo
int gnd = 6;  //attach pin 5 to GND
int led1 = 0; //led1 red to arduino pin 0
int led2 = 1; //led2 red to arduino pin 1
int led3 = 2; //led3 red to arduino pin 2

void setup()
{
 // Set led1, led2 and led3 - pin 0, 1, 2 as
     an output
 pinMode(led1, OUTPUT);
 pinMode(led2, OUTPUT);
 pinMode(led3, OUTPUT);

 // Set the Vcc and GNC - pin 3,4 as output
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);

 // set the speed to 200/255
 motor1.setSpeed(80);

 // stopped
 motor1.run(RELEASE);

 // set the speed to 200/255
 motor2.setSpeed(255);

 // stopped
 motor2.run(RELEASE);

 //initialize serial communication:
 Serial.begin (9600);

}

void loop()
{
 // Set Motor1 to move forward
 motor1.run(FORWARD);
 int duration, distance;
 digitalWrite(trigPin, HIGH);
 delay(2000);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = (duration/2) / 29;

 if (distance < = 50 )
 {
  motor1.run(RELEASE);
  motor2.setSpeed(255);
  motor2.run(FORWARD);
  // This is for the steering to align to
      the center.
  motor2.setSpeed(90);
  motor2.run(BACKWARD);
  // Tell the car to go back
  motor1.run(BACKWARD);
  delay (1000);
  motor2.run(RELEASE);
  motor1.run(RELEASE);
  delay (2000);
  if(distance < = 20 )
  {
   motor1.run(RELEASE);
   motor1.run(BACKWARD);
   brakeLights();
   delay (500);
   motor2.run(RELEASE);
   motor1.run(RELEASE);

  }
  motor2.setSpeed(255);
  motor2.run(FORWARD);
  // This is for the steering to align to
      the center.
  motor2.setSpeed(90);
  motor2.run(FORWARD);
  motor1.run(FORWARD);
  delay (1000);
  motor2.run(RELEASE);
 }
}

void brakeLights()
{
 digitalWrite(led1, HIGH);
 digitalWrite(led2, HIGH);
 digitalWrite(led3, HIGH);
 delay(80);

 digitalWrite(led1, LOW);
 digitalWrite(led2, LOW);
 digitalWrite(led3, LOW);
 delay(80);
}
```

REFERENCES

[1] My-cardictionary.com. (n.d.). Parking aid — Driver Assistance Systems — My car dictionary. [online] Available at: https://www.my-cardictionary.com/driver-assistance-systems/parking-aid.html [Accessed 17 Apr. 2019].

[2] Jos Rovai, M. (2015). Hacking a RC Car With Arduino and Android. [online] Instructables. Available at: https://www.instructables.com/id/Hacking-a-RC-Car-With-Arduino-and-Android/ [Accessed 17 Mar.2019].

[3] T, Aditya. (2017). [online] Instructables.com. Available at: https://www.instructables.com/id/Arduino-Self-Driving-Car/ [Accessed 17 Apr. 2019].

[4] Nanda Prabhu, T. (2019). Tanu-N-Prabhu/Autonomous-RC-Car. [online] GitHub. Available at: https://github.com/Tanu-N-Prabhu/Autonomous-RC-Car [Accessed 17 Apr. 2019].