Project Report on

NA - YATR

Submission to the THE ROBOTICS CLUB as a part of INDUCTION'19

TEAM 9

THE ROBOTICS CLUB
Integrating Knowledge....

THE ROBOTICS CLUB-SNIST

SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY

(AUTONOMOUS)

(Affiliated to JNT University, Hyderabad)

Yamnampet, Ghatkesar, Hyderabad – 501 301.

2019

CERTIFICATE

This is the project work titled **‘NA-YATR’** by ‘K.Anirudh’, ‘ESK Karthik’,’Harihara Viswakarma’,’RagaMohana’,’ShaikShoiab’,’AkashReddy’,’Sindhu’,’Varun’ under the mentorship of ‘Datta Lohith’, ‘Abhishek’ for the recruitment into **THE ROBOTICS CLUB-SNIST** and is a record of the project work carried out by them during the year 2018-2019 as part of **INDUCTION** under the guidance and supervision of

T.V. HARI KRISHNA	**MUHAMMAD TABISH**
Technical Head	**Technical Head**

Ms. V. Manasa
The President of THE ROBOTICS CLUB

Dr. A. Purushotham
Technical Advisor
Mechanical Department

DECLARATION

The project work reported in the present thesis titled **“NA-YATR”** is a record work done by Team “9” in **THE ROBOTICS CLUB** as a part of **INDUCTION-19.**

No part of the thesis is copied from books/ journals/ Internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the project work done entirely by TEAM “9” and not copied from any other source.

ACKNOWLEDGMENT

This project report is the outcome of the efforts of many people who have driven our passion to explore into implementation of **NA-YATR.** We have received great guidance, encouragement and support from them and have learned a lot because of their willingness to share their knowledge and experience.

Primarily, we would like to express our gratitude to our mentors, **‘Datta Lohith’** and **‘Abhishek’.** Their guidance has been of immense help in surmounting various hurdles along the path of our goal.

We thank our Technical Heads **T.V. Hari Krishn**a and **M. Tabish** for being with us till the end of the project completion.

We thank the **Executive Body and** the **Technical Advisory Board** of **The Robotics Club** for helping us with crucial parts of the project. We are deeply indebted to **Ms. V. Manasa -** the President, **Mr. R. Sai Sandeep -**the Vice-President and **Mr. Aditya Gupta** –the General Secretary of **THE ROBOTICS CLUB** respectively and also every other person who spared their valuable time without any hesitation whenever we wanted.

We also thank our Technical advisor **Dr. A. Purushotham**, Professor, Mechanical Department, who encouraged us during this project by rendering his help when needed.

Index

Contents

List of Figures

Abstract

Navigation is an indispensable part of a mobile robot, especially the ability to self-navigate is of immense importance to an autonomous car. For this purpose, it is necessary for the car to know its current position as well as an area of a map it has to navigate. GPS provides the required position data and a compass gives the current heading. This data in conjunction with a list of way-points obtained from the map is used to calculate the required correction in heading. This is then used to obtain the steering angle to navigate the robot. The concept of an autonomous bot has been around for a while, however the recent developments in various field now enables the technology to be transferred from laboratory conditions to practical scenarios. The thrust to deliver a working autonomous robot is attributed to the ever increasing incidents due to human error. Distractions during physical work cause the person to loose attention and may prove fatal as margin of error is a matter of few seconds. Prompt action and continuous focus can by far reduce the number of vandalisms .As humans are prone to distractions either due to fatigue or monotonicity of the task at hand, we look forward to computers more probably autonomous robots for solutions. The first requirement for complete physical autonomy is the ability for a robot to take care of itself. Many of the battery-powered robots on the market today can find and connect to a charging station. Environmental sensing and awareness, proportional motor control based upon environmental factors, digital speed decoding through programmable logic, differential drive control, and isolation of low and high power systems are the major advantages of autonomous bots. Through C compiler, it is possible to create programs and routines of control for the robot. The optoelectronic detection system gives to the robot an important grade of autonomy, it means, when the robot detects and recognizes the objects and obstacles during its trajectory, the robot can makes decisions about what to do with each of them. More advanced robots can analyze and adapt to unfamiliar environments, even to areas with rough terrain. These robots may associate certain terrain patterns with certain actions. A robot needs a power source to drive these actuators. Most robots either have a battery or they plug into the wall.

The autonomous system is further classified into four types:

1. Programmable.
2. Non-programmable.
3. Adaptive.
4. Intelligent.

Hence they are very essential in this modern world embracing of different AI techniques for human purpose and is considered to be one of the best rejoin in this present scenario.

CHAPTER 1
INTRODUCTION

1.1 PROBLEM STATEMENT:
To make an autonomous self operating bot where the user will mark the location (longitude and latitude) and will be sent to the bot via serial communication after that the bot will enter a loop moving around ,avoiding any obstacle in its path, until it reach the final point

1.2 INTRODUCTION TO PROJECT:
The present technology used in this bot is the use of ultrasonic sensors which uses high frequency waves of sound and this sound wave will be bounced back to calculate the amount of distance. The gps technology which detects the positioning of the car by triangulating using satellites .The Servo motor is used additionally to reduce the number of ultrasonic sensors to be used and rotate when obstacles are detected where the cost of the bot is reduced

1.3 LITERATURE SURVEY:
Basically the source of idea is obtained from the autonomous moving or the driverless car As we know the autonomous driving will be the next big innovation in future ,it supposed to have a massive impacts in society, like reducing accidents will increase travel time reliability and reduce congestion as we know that the human errors are responsible for almost 90 % of accidents and people with mobility restriction i.e who does not no drive and is elderly i.e not a license holder or a physically challenged person can use these time of driverless cars

1.4 ORGANISATION OF THE PROJECT:

1. Chapter 1 describes about the aim and the introduction part of the project
2. Chapter 2 gives the description of the components in detail used in project.
3. Chapter 3 describes implementation of the project with block diagram and the circuit description.
4. Chapter 4 shows the results and discussions what had been done is explained in detail

CHAPTER 2
ARCHITECTURE

2.1 COMPONENTS USED

2.2.1 Hardware

- SIDE SHAFT MOTORS
- ARDUINO MEGA Microcontroller
- Neo-6m GPS
- HMC5883L DIGITAL COMPASS

2.1.2 Software

- GUI using MATLAB
- ARDUINO IDE

2.2 COMPONENTS DESCRIPTION

2.2.1 Hardware

DC MOTORS:

- A **DC motor** is any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields.

Common types of motors:

- Centre shaft Motors
- Side shaft Motors.
- Side shaft motors are generally used for higher torque and centre shaft motors are used for high speed.

- SIDE SHAFT MOTORS:

Side Shaft motors are generally used for high torque. The unique design lets the motor to have good torque as it is placed at some distance from the centre creating that required torque and efficiency of the motor.

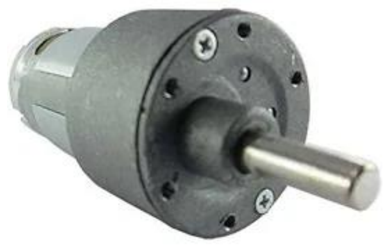


Fig 1

- ARDUINO MEGA Microcontroller:

Arduino Mega known after the company Arduino is a whole assorted form of printed circuit board consisting of microcontroller, analog pins, digital pins, power supply, ground pins, heat sinking pins, pulse width modulation pins etc. Arduino is generally used to dump the code which intern is used for the interaction of hardware and software of a respective bot.

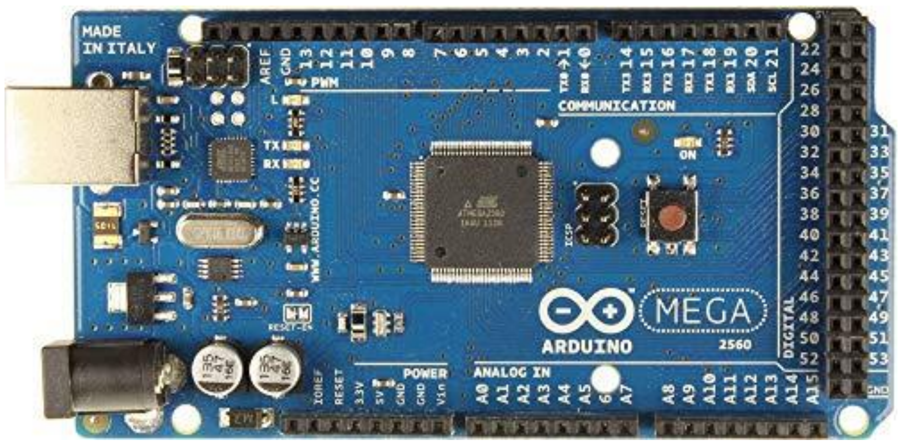


Fig 2

- Neo-6m GPS:

GPS is generally referred as GLOBAL POSITIONING SYSTEM is used to detect the location of the object. This is mainly possible through satellite navigation.the time to time location of the bot or any automobile using GPS is taken and sent to satellites. Three satellites form a triangle which are nearby to the position of the bot and coincide downward side and give the exact and accurate location of the respective object moving with respect to time and speed.thus time to time location is obtained using appropriate GPS.

Fig 3

- HMC5883L DIGITAL COMPASS:

HMC5883L is a 3-axis digital compass used for two general purposes: to measure the magnetization of a magnetic material like a ferromagnet, or to measure the strength and, in some cases, the direction of the magnetic field at a point in space. These types of compasses are very essential for proper chanelling and direction of the bot.

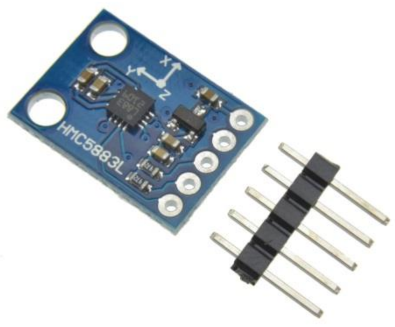
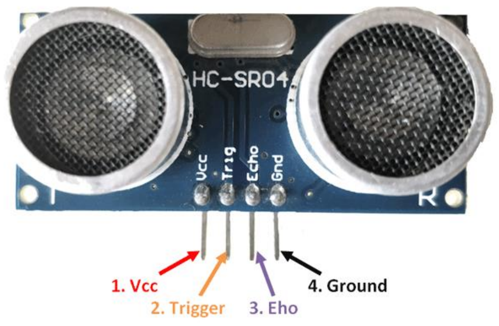


Fig 4

- HC-SR04 ULTRASONIC SENSOR:

  HC-SR04 Ultrasonic (US) sensor is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module.



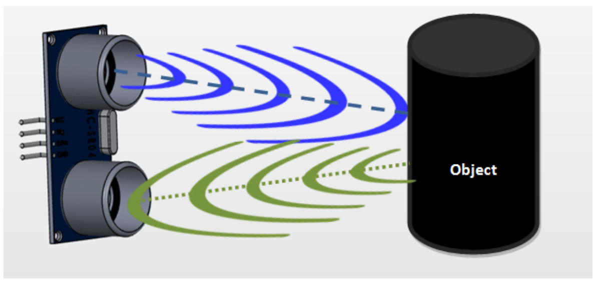


Fig 5a and 5b

- L298N MOTOR DRIVERS:

  L298N H-bridge Dual Motor Controller Module 2A with Arduino. This allows you to control the speed and direction of two DC motors, or control one bipolar stepper motor with ease. The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC. There is also an onboard 5V regulator, so if your supply voltage is up to 12V you can also source 5V from the board.

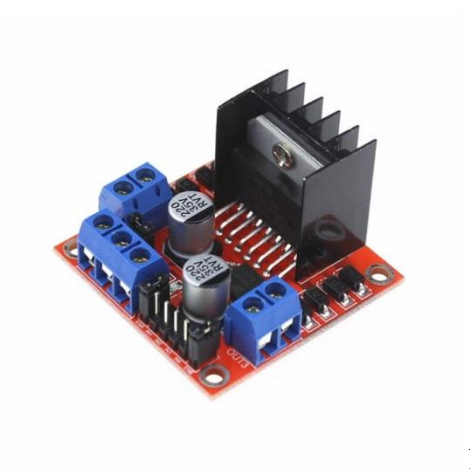
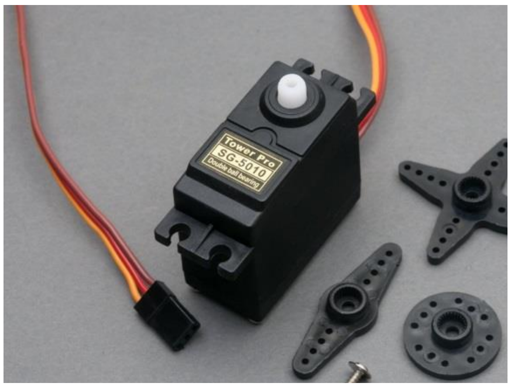


Fig 6

- SERVO MOTOR:

  A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. A servomotor is a closed-loop that uses position feedback to control its motion and final position. The input to its control is a signal representing the position commanded for the output shaft.



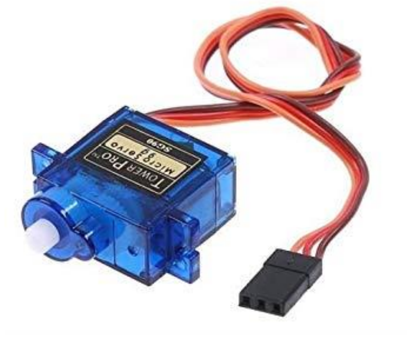


Fig 7

- WHEELS:

  Robot wheels are used for locomotive purpose or movement of the robot accordingly. Wheels are mechanical part of the robot which are basic step for chassis essential for the transportation of the bot in respective field.

Fig 8

- Xbee S2C:

  This Xbee module is an RF module used for wireless communication. Its transmission frequency is between 2.4GHz to 2.5GHz. Its range is 200ft indoors and upto 4000ft outdoors. Its supply voltage is from 2.1V to 3.6V. It is useful for home automation and medium range wireless communications.

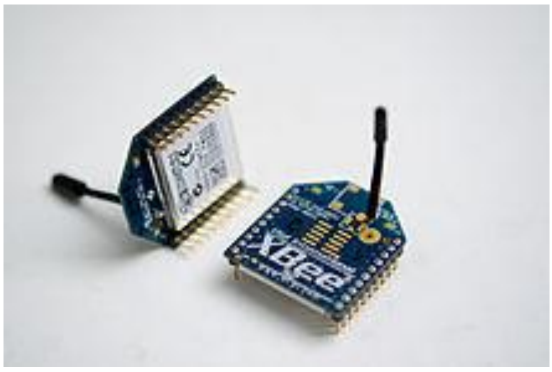


Fig 9

2.2.2 Software

GUI using MATLAB:

GUIs (graphical user interfaces) provide point-and-click control of software applications, eliminating the need to learn a language or type commands in order to run the application. MATLAB apps are self-contained MATLAB programs with GUI front ends that automate a task or calculation. The GUI typically contains controls such as menus, toolbars, buttons, and sliders. Many MATLAB products, such as Curve Fitting Toolbox, Signal Processing Toolbox, and Control System Toolbox include apps with custom user interfaces. You can also create your own custom apps, including their corresponding UIs, for others to use.

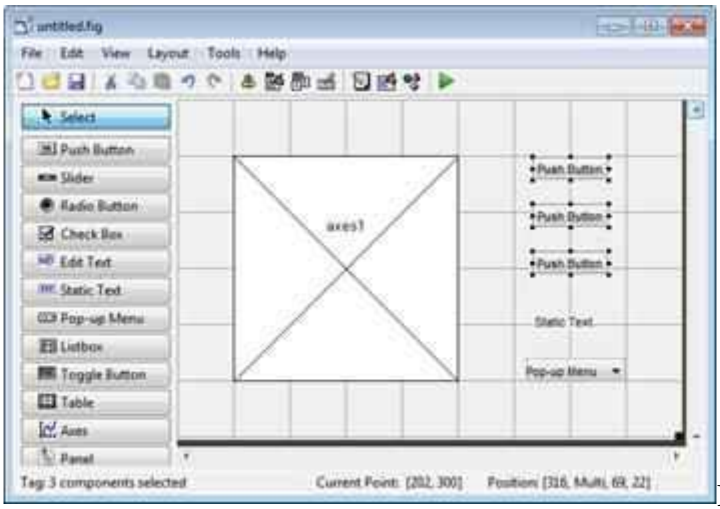


Fig 10

- ARDUINO IDE:

The Arduino integrated development environment (IDE) is a cross-platform application for Windows, mac OS, Linux that is written in the programming language Java. It is used to write and upload programs to Arduino board. The Arduino IDE supports the languages C and C++ using special rules of code structuring.

AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE

LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

Fig 11

CHAPTER 3

IMPLIMENTATION AND WORKING

3.1.1 BLOCK DIAGRAM

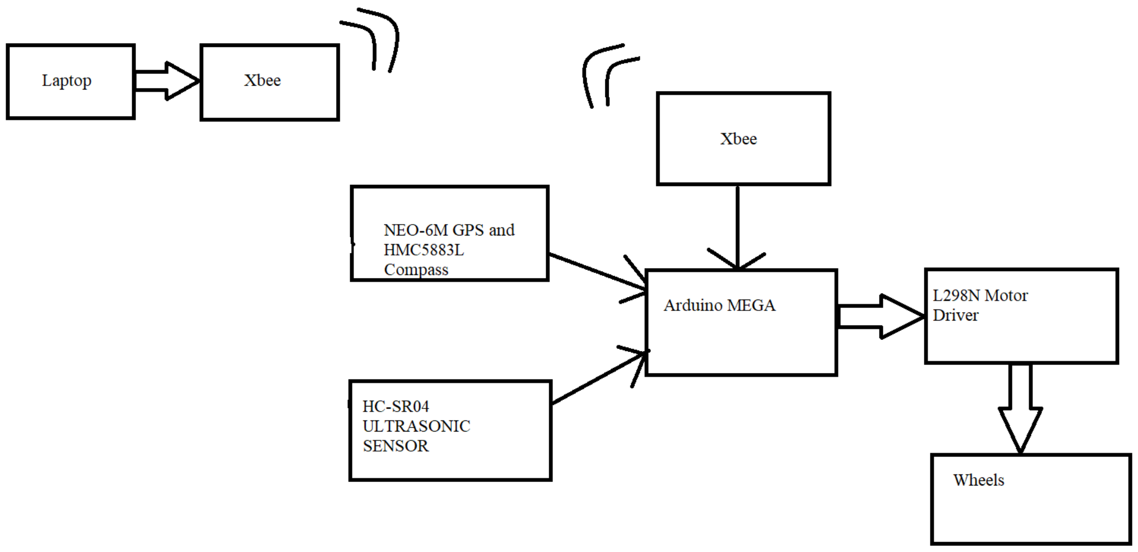


Fig 12

3.1.2 CIRCUIT DIAGRAM

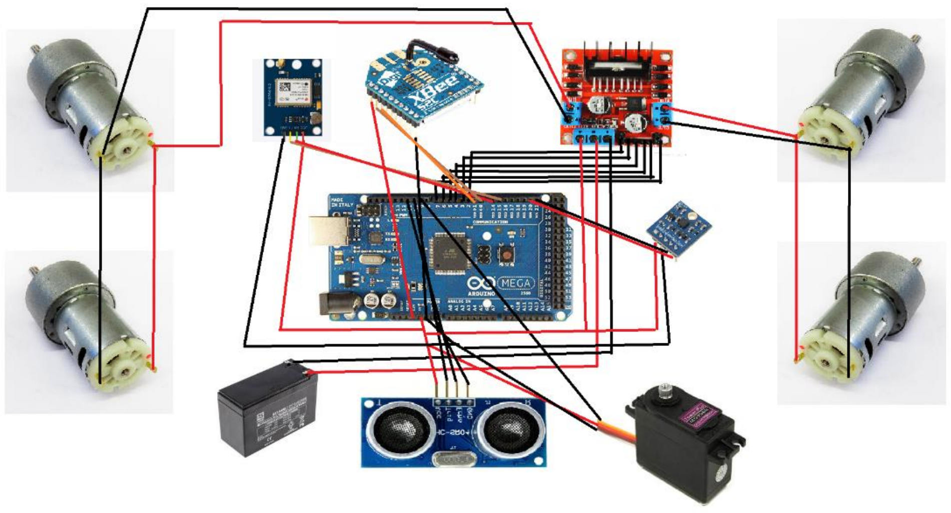


Fig 13

3.2 WORKING

A-star method:

A-star method is an computer algorithm which used for pathfinding. This method converts the plane into multiple modules and find the shortest path avoiding the obstacles. A-star method is most efficient than any other method. This method uses heuristics to give maximum accuracy and performance.

Anti-collision system:

Anti-collision system can be used to avoid obstacles and damage of the bot. For the application of anti-collision system we use ultrasonic radar by attaching a ultrasonic sensor on the top of servo motor. The software developed and uploaded into the microcontroller works in a way that the ultrasonic sensors detect an obstacle less than a particular distance the bot stops. Then the bot looks for an alternate way to move ahead which has an obstacle at maximum distance. This process continues until the bot reaches its final destination.

3.3 ALGORITHM

Step1:The person should give a location on the map.

Step2: The computer software extracts only the route and obstacles in between. Excludes all other information available from the map.

Step3: Using A-star method computer finds a shortest route for the bot to reach the destination.

Step4: This information is transferred from the computer to the microcontroller using Xbee transmission devices.

Step5: The information is analyzed by the microcontroller and using GPS & compass the bot makes it’s move.

Step6: Any obstacles are identified in the way the ultrasonic sensors detects it and changes the direction.

Step7: The followings continue in a loop like A-star method, GPS tracking, anti-collision system etc… until the destination is arrived.

Step8: When the destination is arrived it stops.

Step9: End.

CHAPTER 4
EXPERIMENTAL RESULTS

4.1 RESULTS

The robot is performing motions in these directions: forward, backward, side-wards and also rotating around its own axis. The robot is efficient and precise in obstacle detection and changing its path accordingly. The robot moves in the path defined to it whilst avoiding any obstacles without any ambiguity.

4.2 FUTURE ENHANCEMENTS

We are planning to improvise this robot by adding the following features:

1. By adding image processing technology to it.
2. By adding a light sensor after which it can be used as a fire detecting and fighting robot in future.
3. By adding an IR sensor.
4. By using Lidar for 3D mapping and developing a indoor route map.
5. By adding a camera.

4.3 CONCLUSION

The overall conclusion of the project is that we have successfully constructed a robot that can be useful to work on any terrain given with some satisfactory gaits. This robot is able to communicate with all of its components with utmost efficiency. This robot with little variations can be used in various fields from Military to Agriculture. The robot has wide range of application scaling from domestic to industries. The features of this robot shows how human capital can be reduced and how human safety is not compromised while using it in mines and military operations.

SOURCE CODES

```
/*Connections:
1=tx
2=rx

a4=sda
a5=scl

13=trig
12=echo

11=enA
8=in1
7=in2

10=enB
6=in3
5=in4

4=servo
*/

#include <TinyGPS.h>
#include <HMC5883L.h>
#include <I2Cdev.h>
#include <Wire.h>
#include <SoftwareSerial.h>
#include <Servo.h>
int f1=0;
long cm1;
int i=0;
```

```
float heading;
TinyGPS gps;
SoftwareSerial ss(1,2);
float x2lon = radians(00000000), x2lat = radians(000000000); //enter final location here as
lon and lat
HMC5883L comp;
int16_t mx, my, mz;
float head, distance = 0.0;
#define trigPin 13
#define echoPin 12

//Right motor
int enA = 11;
int in1 = 8;
int in2 = 7;

//Left Motor
int enB = 10;
int in3 = 6;
int in4 = 5;

const int danger = 50; //danger distance in cm
int leftDistance, rightDistance;

Servo uservo;
long dur;
long dist;
void setup() {
  // put your setup code here, to run once:
  uservo.attach(4);
  uservo.write(90);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
Wire.begin();
  comp.initialize();
  ss.begin(9600);
  Serial.begin(9600);
  Serial.println("Complete");
  Serial.println("Ready!");
}

void loop() {
  // put your main code here, to run repeatedly:
 if (f1 != 0)
    Forward();
  cm1 = ping();
  delay(50);
  cm1 = ping();
  Serial.print("cm");
  Serial.print(cm1);
  if (cm1>danger){
    comp.getHeading(&mx, &my, &mz);
    heading = atan2(my, mx);
    if (heading < 0)
      heading += 2 * M_PI;
    heading = heading * 180 / M_PI;
    if ((i % 2) == 0)
    { head = heading + 100;
      if (head > 360)
        head = head - 360;
      Left();
    }
    else
    { head = heading - 100;
      if (head < 0)
        head = head + 360;
      Right();

    }
    while ((heading > head + 8) || (heading < head - 8)) // this loop turns the bot till its facing
(head)degrees east of north
    {
      turn();
      delay(5);
      comp.getHeading(&mx, &my, &mz);
      heading = atan2(my, mx);
      if (heading < 0)
        heading += 2 * M_PI;
      heading = heading * 180 / M_PI;
    }
    Stop();
    delay(100);
  }
  else {

    Forward();



    delay(500);
    Stop();


    do {
      gpshead();
    } while (distance == 0.0);
    if (distance < 15)
```

```
      while (1)
        Stop();
    if (f1 == 0){
      comp.getHeading(&mx, &my, &mz);
      heading = atan2(my, mx);
      if (heading < 0)
        heading += 2 * M_PI;
      heading = heading * 180 / M_PI;



      while ((heading > head + 8) || (heading < head - 8))
      {
        turn();
        delay(5);
        comp.getHeading(&mx, &my, &mz);
        heading = atan2(my, mx);
        if (heading < 0)
          heading += 2 * M_PI;
        heading = heading * 180 / M_PI;
      }
      f1 = 4;
    }
    f1--;
  }
}
  void turn(){
  float tur = heading - head;
if (tur < 0.0)
  { if (tur > -180.0)
      Right();
    else
      Left();
  }
  else
  { if (tur < 180.0)
      Left();
    else Right();
  }
}
void gpshead()
{
  bool newData = false;
  for (unsigned long start = millis(); millis() - start < 1000;)
  {
    while (ss.available())
    {
      char c = ss.read();
       Serial.write(c);
      if (gps.encode(c))
        newData = true;
    }
  }
  if (newData)
  {
    float flat1, flon1;
    unsigned long age;
    gps.f_get_position(&flat1, &flon1, &age);
    flon1 = radians(flon1);  //also must be done in radians
    flat1 = radians(flat1);  //also must be done in radians
    head = atan2(sin(x2lon - flon1) * cos(x2lat), cos(flat1) * sin(x2lat) - sin(flat1) * cos(x2lat)
* cos(x2lon - flon1));
    head = head * 180 / 3.1415926535; // convert from radians to degrees
    float dist_calc = 0;
    float diflat = 0;
    float diflon = 0;
    diflat = x2lat - flat1; //notice it must be done in radians
    diflon = (x2lon) - (flon1); //subtract and convert longitudes to radians
    distance = (sin(diflat / 2.0) * sin(diflat / 2.0));
    dist_calc = cos(flat1);
    dist_calc *= cos(x2lat);
    dist_calc *= sin(diflon / 2.0);
    dist_calc *= sin(diflon / 2.0);
    distance += dist_calc;
    distance = (2 * atan2(sqrt(distance), sqrt(1.0 - distance)));
    distance *= 6371000.0; //Converting to meters
    if (head < 0) {
      head += 360; //if the heading is negative then add 360 to make it positive
    }
  }
  else
  {
    head = 0.0;
    distance = 0.0;
  }
}
void movement()
{
  int distanceFwd = ping();
  if (distanceFwd>danger) //if path is clear
  {
    Forward(); //move forward
  }
  else //if path is blocked
  {
    Stop();
    uservo.write(0);
    delay(500);
    rightDistance = ping(); //scan to the right
    delay(500);
    uservo.write(180);
    delay(700);
    leftDistance = ping(); //scan to the left
    delay(500);
    uservo.write(90); //return to center
    delay(100);
    compareDistance();
  }
}
void compareDistance()
{
  if (leftDistance>rightDistance) //if left is less obstructed
  {
    Left(); //turn left
    delay(500);
  }
  else if (rightDistance>leftDistance) //if right is less obstructed
  {
    Right(); //turn right
    delay(500);
  }
   else //if they are equally obstructed
  {
    Backward(); //turn 180 degrees
    delay(1000);
  }
}

long ping()
{
 //Trigger the sensor to send out a ping
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 dur = pulseIn(echoPin, HIGH);
 dist = 0.034*dur/2;
 Serial.println(+distance);
 return dist;
}

void Forward() //This function tells the robot to go forward
{
  //for (int i=0; i <= 50; i++){
  Serial.println("");
  Serial.println("Moving forward");
  // turn on left motor
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  // set speed out of possible range 0~255
  analogWrite(enA, 155);
  // turn on right motor
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  // set speed out of possible range 0~255
  analogWrite(enB, 155);
  delay(100);
  //}
  //Stop();
}

void Backward() //This function tells the robot to move backward
{
  //for (int i=0; i <= 50; i++){
  Serial.println("");
  Serial.println("Moving backward");
  // turn on left motor
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  // set speed out of possible range 0~255
  analogWrite(enA, 155);
  // turn on right motor
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  // set speed out of possible range 0~255
  analogWrite(enB, 155);
  delay(100);
  //}
  //Stop();
}

void Left() //This function tells the robot to turn left
{
  //for (int i=0; i <= 50; i++){
  Serial.println("");
  Serial.println("Moving left");
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  //  set speed out of possible range 0~255
  analogWrite(enA, 140);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  //set speed out of possible range 0~255
  analogWrite(enB, 155);
  delay(100);
  //   }
  //   Stop();
}

void Right() //This function tells the robot to turn right
{
//   for (int i=0; i <= 50; i++){
   Serial.println("");
   Serial.println("Moving right");
   digitalWrite(in1, LOW);
   digitalWrite(in2, HIGH);
```

```
//  set speed out of possible range 0~255
   analogWrite(enA, 135);
   digitalWrite(in3, LOW); //Was High
   digitalWrite(in4, HIGH);
   analogWrite(enB, 155);
   delay(100);
//    }
//     Stop();
}

void Stop() //This function tells the robot to stop moving
{
  Serial.println("");
  Serial.println("Stopping");
// now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  //analogWrite(enA, 0);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  //analogWrite(enB, 0);
}
```

APPENDIX

Global Positioning System(GPS) - Gives the location of the bot

A*star Path Finding Algorithm - Gives the path of the bot

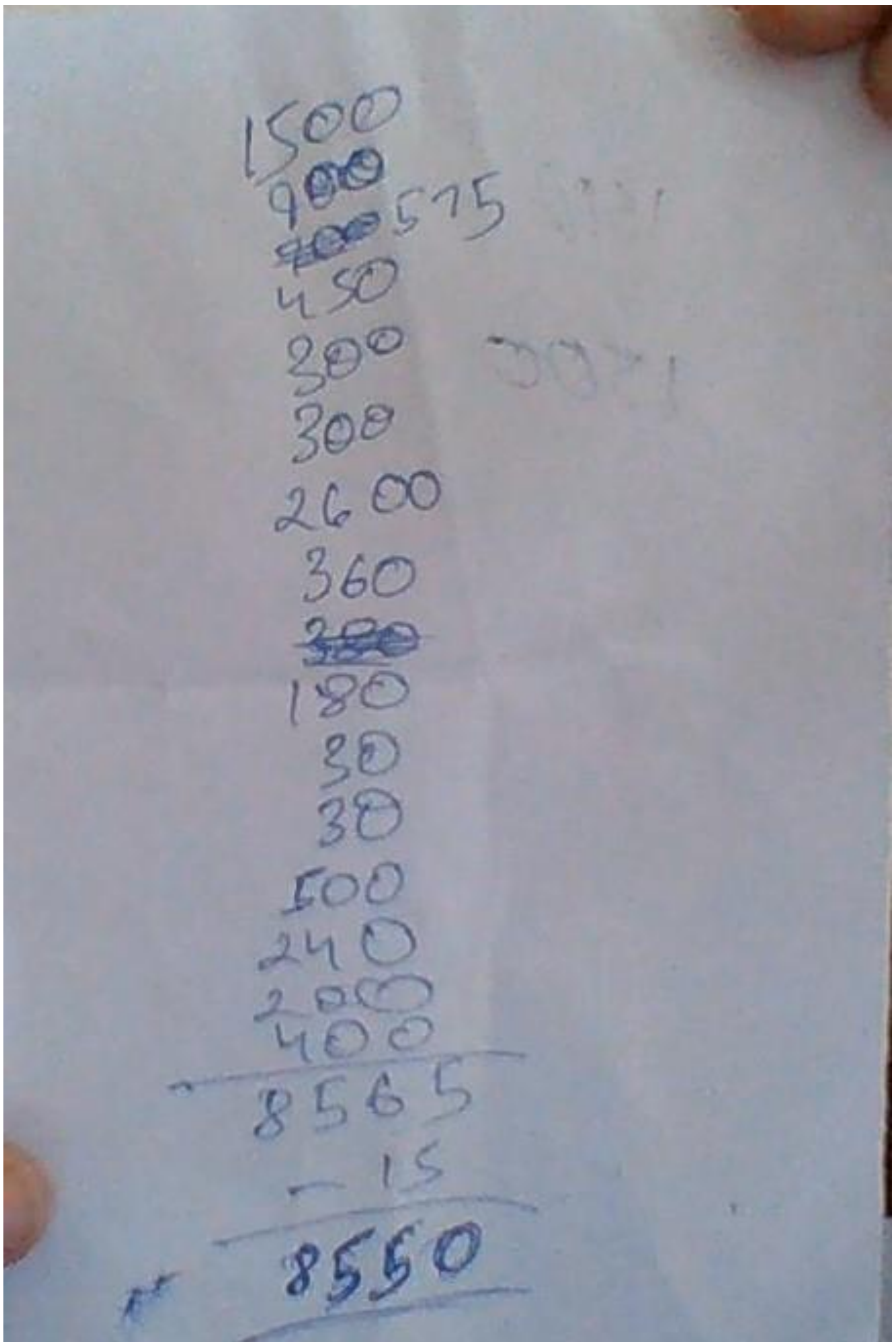
L298n - Controls the motors

Xbee - For wireless communication

Arduino Mega – Micro controller to process the data

Compass – Gives the heading of the bot

RECORD OF EXPENSES



1500
900
~~700~~ 575
450
200
300
2600
360
~~300~~
180
30
30
500
240
200
400

8565
− 15

8550