Autonomous Following Bot

By-
Avirat Varma - N17296397
Rishi Kavi - N15231052
Pavan Chowdary Cherukuri - N10938396
Vegesna Rishi Eswar Varma- N17479594

Table of Contents

INTRODUCTION

Advancements in mobile robotics, ranging from task specific applications to consumer applications such as the automotive sector, have heightened the combinational use of sensors and logic to equip mobile robots and vehicles with smart/intelligent functions that can increase safety, reduce human effort while driving or integrate additional features. The autonomous following bot (Figure1) is a scaled down prototype representing such a feature. In many applications, an autonomous following bot is suited to perform various functions. The general principle of operation for an autonomous following bot makes use of proximity/vision sensors to acquire data from its surrounding or target and to pass it on for further computation. In our case, 3 HC-SR04 ultrasonic sensors are statistically mounted on the custom designed sensor mast, which simultaneously acquire data and feed it to an Arduino UNO. In theory, the ultrasonic sensors transmit ultrasonic sound waves which bounce off the target and are received by the receiver. Since the speed of sound is known (343 meters/second), the elapsed time between transmission and reception is measured to give the distance of the target from the sensor (using: speed=(distance/time)). An Arduino UNO was used to perform computations and to control the peripherals of the bot. An Arduino is capable of controlling the number of I/O devices that the bot uses. Based on the simultaneous input of the three ultrasonic sensors, the Arduino, through its Behavioral Architecture (Figure2), tracks the target and executes a control decision by actuating the servo motors or trigger an audio signal (buzzer). For military applications, an autonomous following bot will be used to carry heavy loads for the soldier so that human fatigue is reduced. Or for civilian applications such as construction (applications explained further), the bot can carry construction equipment. For such large-scale applications, better sensors such as LiDAR, Radar, Camera can be used to track the target, this will make the tracking more robust and the motion more agile. A tracking algorithm (explained further) records the movements of the target and

stores it as a path for each trip. This feature helps sharing movement data with team members or for analysis/review later.

With the addition of an HC-05 Bluetooth module, the autonomous following bot has wireless communication properties with a mobile device for the user. Explained in further detail, the user interface on the mobile device allows the user to switch between modes of operation – *Following, Stop Following, Manual Mode*. In *Manual Mode*, the user can take over control and maneuver the bot as he/she wishes. This wireless capability is also used to review tracking data from the bot on the mobile device. A differential drive

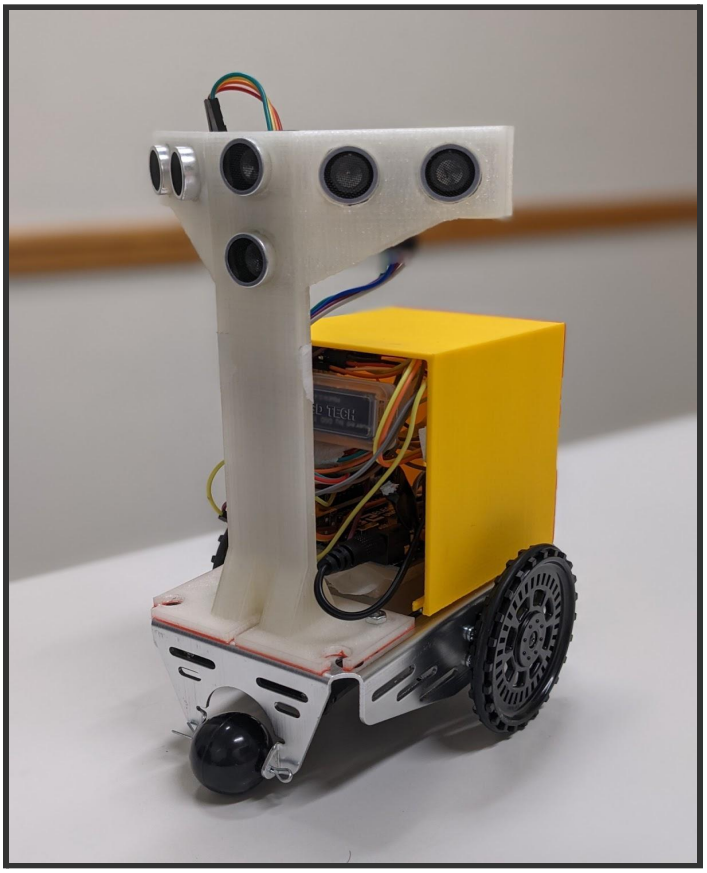


Figure 1: Autonomous Following Bot

configuration allows the bot to make point rotations for sharper maneuverability.

An Autonomous Following model such as this can further incorporate smarter sensors and heavier computation (AI,ML,CV) to increase functionality and robustness**.**

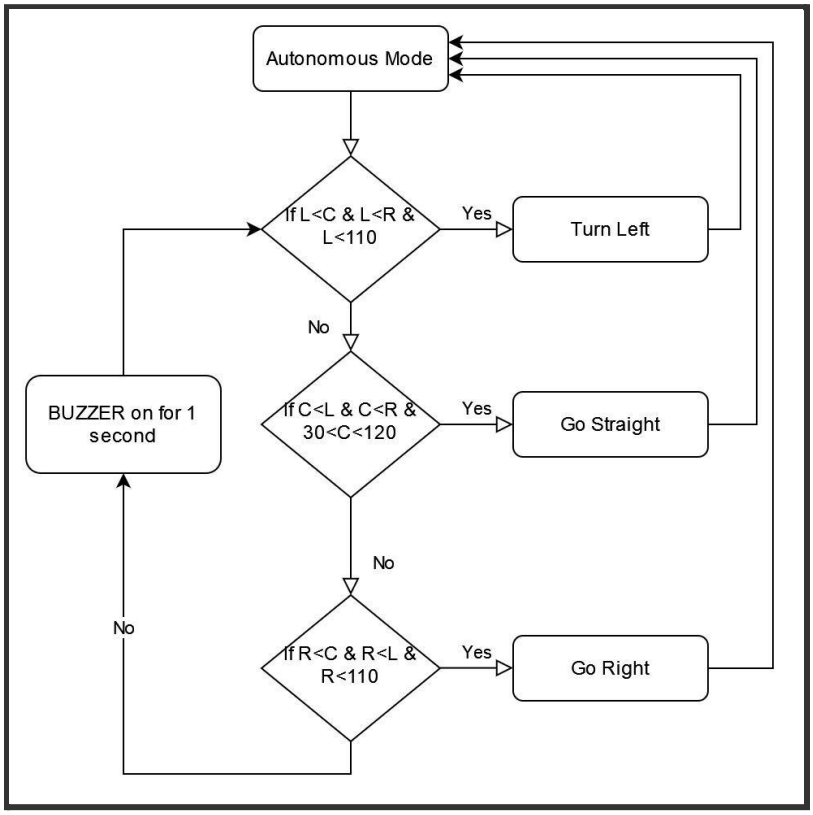


Figure 2: Behavioral Architecture

MECHANICAL DESIGN

The Mechanical Designs and all the drawings required for this project were made in AutoDesk Fusion 360 software. The designs were fabricated using 3D printed PLA (Polylactic Acid) at the NYU MakerSpace.

SENSOR MAST

A Sensor Mast was custom designed and 3-D printed, whose primary function is to hold the three HC-SR04 ultrasonic sensors. For this, the Sensor Mast incorporates two flanks as shown in Figure 3. The flanks of the Mast were made symmetrical for unbiased

performance and easier computations. The Sensor Mast consists of 6 grooves which not only supports the sensors, but also facilitates the task of demounting the sensors.

A 15 cm Sensor Mast height and 30 degree Flank angles(1) were considered which were calculated through an iterative process. These calculations will be discussed further in the Detailed analysis and optimisations section.

Figure 3: Sensor Mast

The thickness of the Sensor Mast was considered to be 5 mm, for a better strength and low weight. Sensor Mast has a base of thickness 2mm, consisting of appropriate grooves in which screws can be driven easily to fix the battery holder and the mast simultaneously to the boe-bot chassis. Furthermore, additional supports and fillets were incorporated accordingly to support the structure.

HOUSING DESIGN

Along with the Sensor Mast, a housing was designed and 3D printed to cover the electronic circuits of the bot and give an aesthetic appeal. The required grooves were included in the design for easy assembling purposes. The housing was left with an open face to provide a flexible circuit accessibility for the user. The thickness of the box was taken to be 2mm. Figure 7 depicts the assembled CAD Model of the Sensor Mast, Housing, and Boe- Bot Chassis.

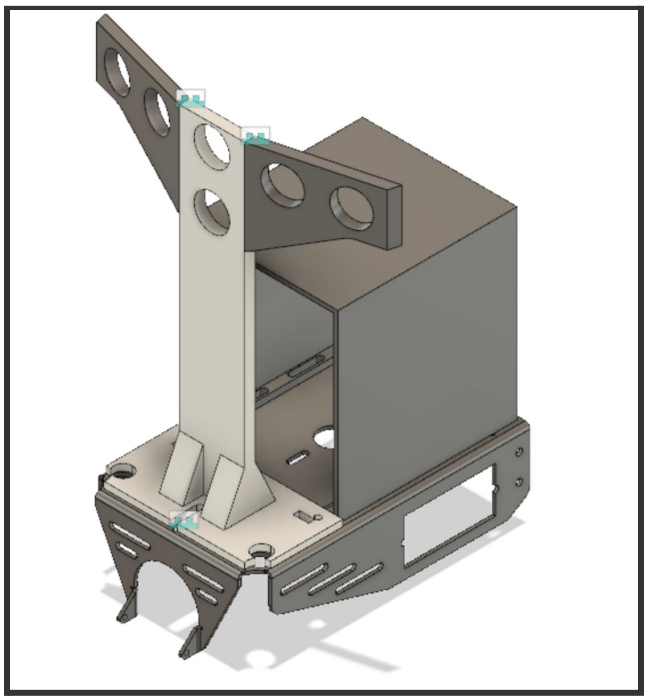


Figure 4: 3-D CAD model of Sensor Mast and Housing mounted on the Boe-Bot Chassis

ELECTRONIC DESIGN

List of components are as follows:

1. 3 HC-SR04 Ultrasonic Sensors
2. Servo motors

3. Bluetooth module HC-05
4. Arduino UNO R3
5. Bread Board
6. 6V Battery pack
7. Active Buzzer

ULTRASONIC SENSOR

Three ultrasonic sensors are used in the project. Ultrasonic sensor is an electronic device which is used to measure the proximity of the target from the bot. It works by transmission and reception of ultrasonic sound waves which is used to calculate the target distance by using time taken for transmission and receiving the waves. The ultrasonic sensor has 4 pins vcc, ground, and trigger, echo. These pins are connected to suitable pins on the microcontroller board. The ultrasonic module used in the project is HC-SR04. This module was selected because of the following reasons:

1. Quiescent current is less than 2mA
2. working voltage is 5v
3. Effectual angle ~ 15^0
4. ranging distance 2cm-400cm
5. Resolution 0.3cm
6. Measuring angle 30^0

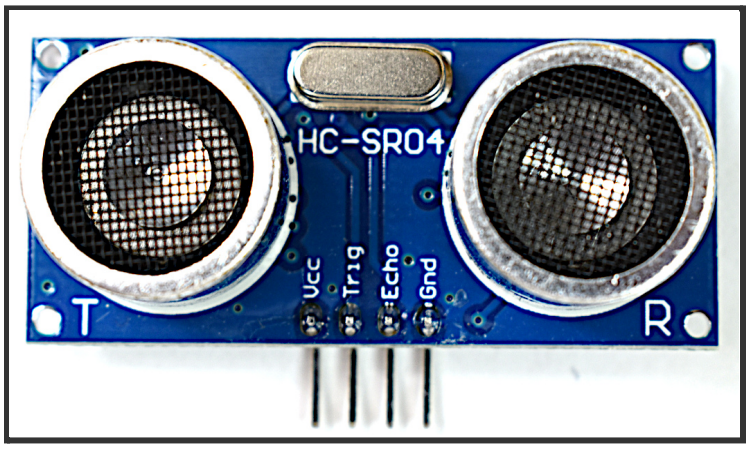


Figure 5: HC-SR04 Ultrasonic Sensor

SERVO MOTORS

Two servo motors are used in the project to move the bot. A servo motor is a rotary actuator. The servo motor consists of three pins: power, ground, and signal pins. The signal pin is connected to the microcontroller board which sends speed control based pulses respective to which the motor rotates. The motors used in the project are parallax continuous servo motors. These particular motors are used in the project because they are readily available and have the following features 0 to 50 RPM, with linear response to PWM for easy ramping, can be easily mounted on the chassis of a boe-bot, and are easy to interface with any PWM capable device.

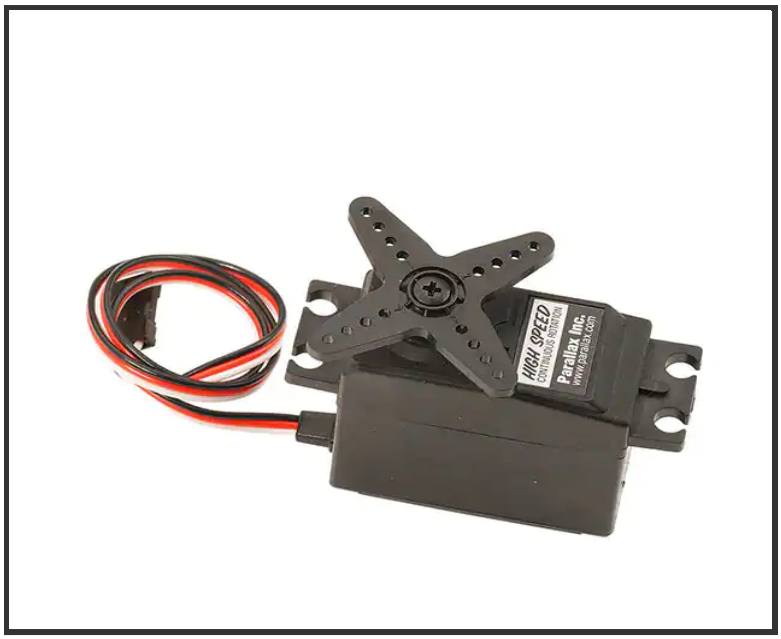


Figure 6: Servo Motor

BLUETOOTH MODULE

A Bluetooth module was used to control the bot using a smartphone. Other feature sof the Bluetooth module are the option to change the mode of operation of the bot and to receive path data from the bot remotely. The Bluetooth module has 6 pins out of which 4

pins were used. The 4 pins are ground, Vcc, Tx, and Rx. The Tx and Rx pins are connected to the Arduino board to send and receive data from smartphones. The Bluetooth module used is HiLetgo HC-05 Wireless Bluetooth RF transceiver master slave integrated Bluetooth module. The particular Bluetooth module was selected because it has features such as effective transmission rate of 10 meters, Quiescent current less than 10mA, default baud rate of 9600, and can be easily interfaced with Arduino board.

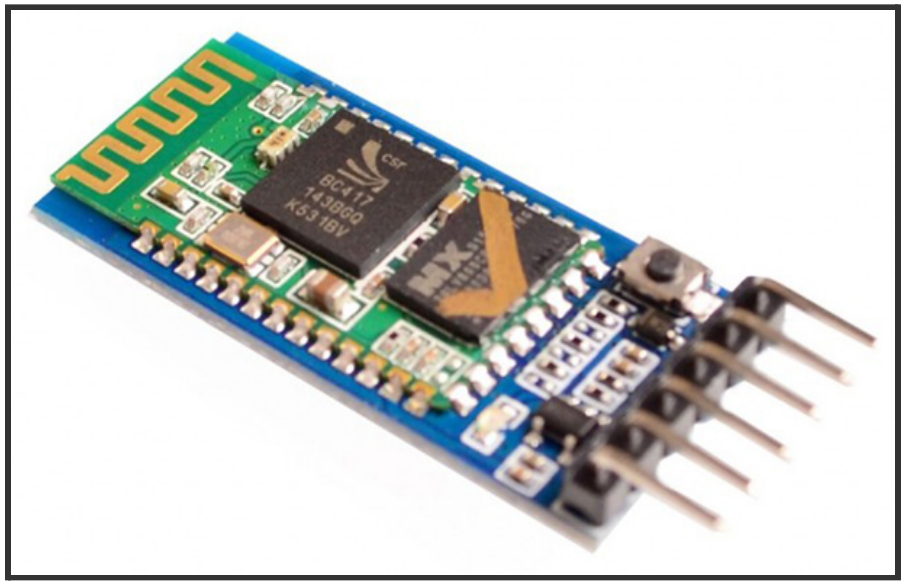
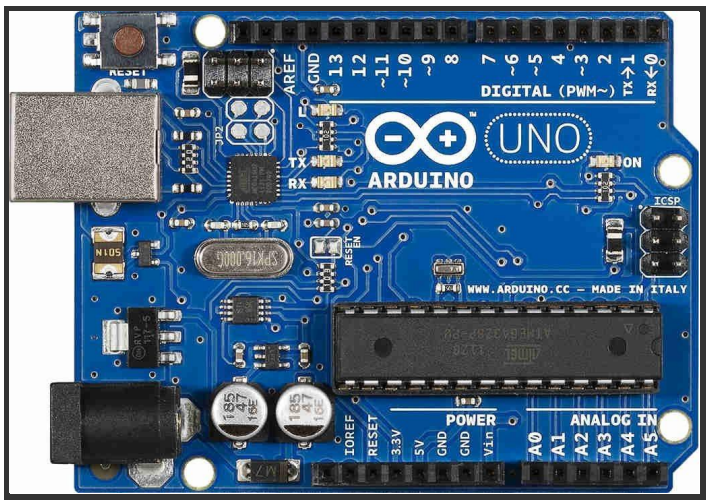


Figure 7: HC-05 Bluetooth Module

ARDUINO UNO

An Arduino Uno R3 board was implemented in the project to control all the electronic components of the system. The Arduino board used consists of 14 digital pins, has 32k flash memory and 16 Mhz clock speed. Arduino Uno was preferred over Basic stamp 2 because this project requires a high processing rate to continuously validate data from 3 ultrasonic sensors, Bluetooth module, and servo motors.

For this application a multi threaded processor is needed but basic stamp 2 is not a multi thread processor so an Arduino uno R3 is used which is also not a multi thread processor



but is fast enough to act as a pseudo multi thread processor. The Arduino can be connected to the computer using a USB cable to transfer the code and it can be powered by connecting it to a battery pack using a barrel pin.

BATTERY PACK

Two battery packs are used in the project. Each battery pack consists of 4 AA 1.5V batteries and gives an output voltage of 6V. Initially one battery pack is used to power the entire project but since the power distribution is not sufficient among the electronic components two battery packs are used. One battery pack to power up the two servo motors and the other battery pack to power up the Arduino, Bluetooth module, and ultrasonic sensors. The battery pack which powers servo motors is connected to them through the breadboard. This battery pack is completely isolated from the remaining components of the projects. This is achieved by using a breadboard without using any isolators.

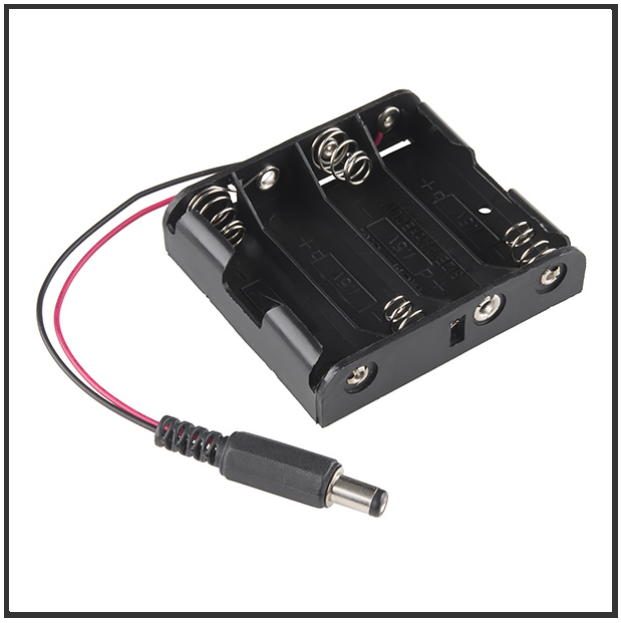


Figure 9: 6V Battery Pack

BUZZER

A buzzer is used in the project to signal the user if the target is too close or too far from the bot. A buzzer is an audio signaling device. An active buzzer was used. The buzzer used is from elegoo uno r3 kit and is readily available.

Figure 10: Active Buzzer

CONNECTIONS

Figure 11 represents the connections between ultrasonic sensors, Bluetooth module, servo motors, battery pack, buzzer. All the ground pins from ultrasonic sensors(Gnd), Bluetooth module(Gnd), servo motors(black wire), buzzer(shorter pin) are connected to the common ground and these connections are represented by black lines. All the Vcc pins from ultrasonic sensors(Vcc), Bluetooth module(Vcc), servo motors(red wire) are connected to the common Vcc of 5V and these connections are represented by red lines from 5V to each part. All input signal pins from Arduino are connected to respective components and are represented by violet lines from Arduino board to respective components. All output signal pins from each electronic component to Arduino are represented by green lines originating from each component and connecting to the Arduino board. Apart from this the Vcc Pins of servo motors connected to 5V are connected to a separated 5V supply to keep up with the power requirement of the motors.

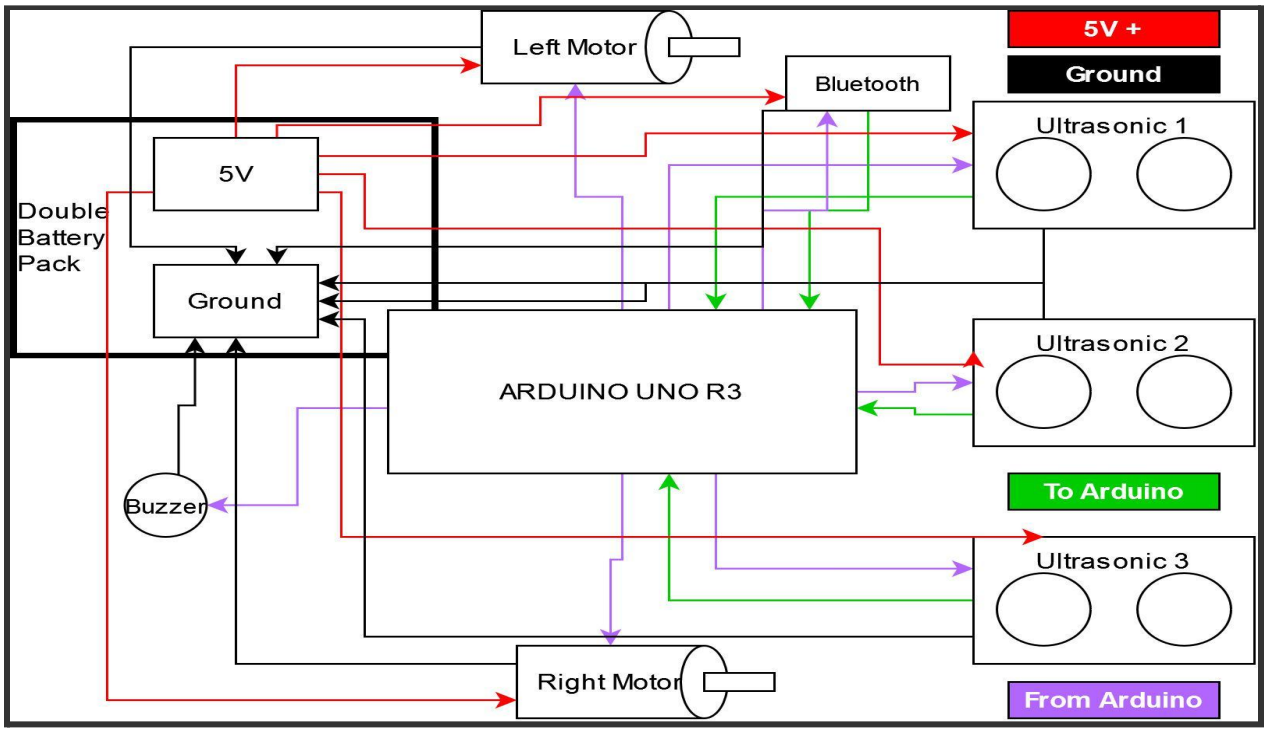


Figure 11: Circuit Schematic

USER INTERFACE

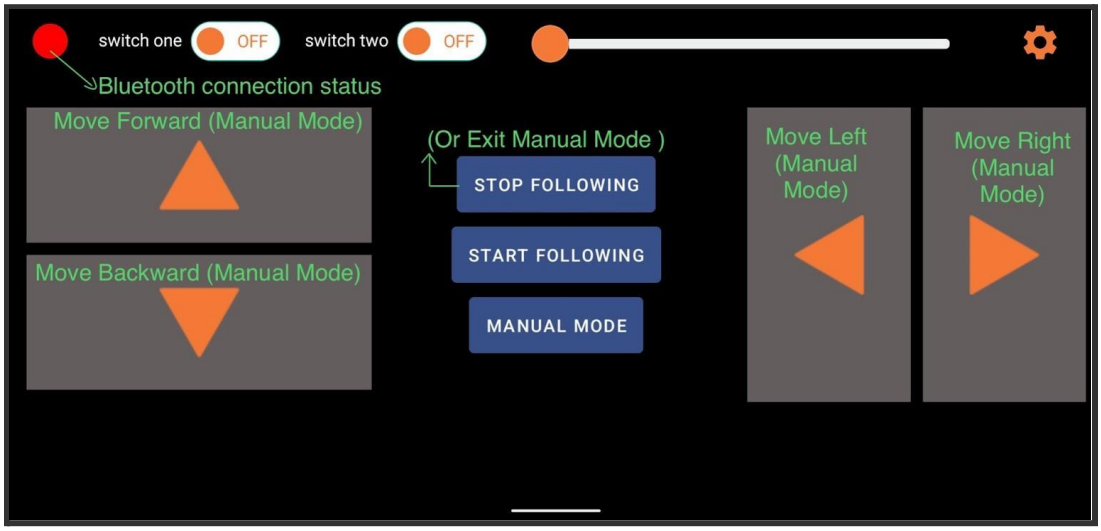


Figure 12: User Interface

With an agenda of making a bot that can be easily communicated by all types of users, a wireless communication through mobile was opted. To achieve this, an app called 'Arduino Serial Monitor & Plotter' was used, which is available for free on Google Play Store. The control options were customised accordingly and made an easily accessible User Interface as shown in Figure 12. As labeled in Figure 12, the *'start following'* button allows the bot to enter autonomous following mode and the *'manual mode'* button makes the bot enter manual mode- in which the user can control the bot manually. In the manual mode, the forward movement can be controlled by '↑' button, the backward movement by '↓' button, left and right movements by '←' and '→' buttons. To exit the manual mode or to exit the autonomous following mode, the *'stop following'* button can be used. In this layout, the movement keys in the manual mode are tactile buttons, whereas all the others are toggle buttons. The red circle on the left top corner indicates the bluetooth connection status, red implies that the device is not connected to the bot, green indicates a flawless connection status.

The ‘Arduino Serial Monitor and Plotter’ app can plot the recorded tracking data of the target by its ‘Serial Plotter’ function. Serial communication was being used between bluetooth and arduino such that all the data that arduino computes, will be sent serially to the bluetooth module, which in turn will be sent to a mobile device via bluetooth communication to the mobile device. At each iteration of the code loop, the Serial Plotter recognises the trigger string present in the code and plots the data. The Serial Monitor also displays the raw data. Figure 13 depicts the Serial Plotter and Serial Monitor respectively.

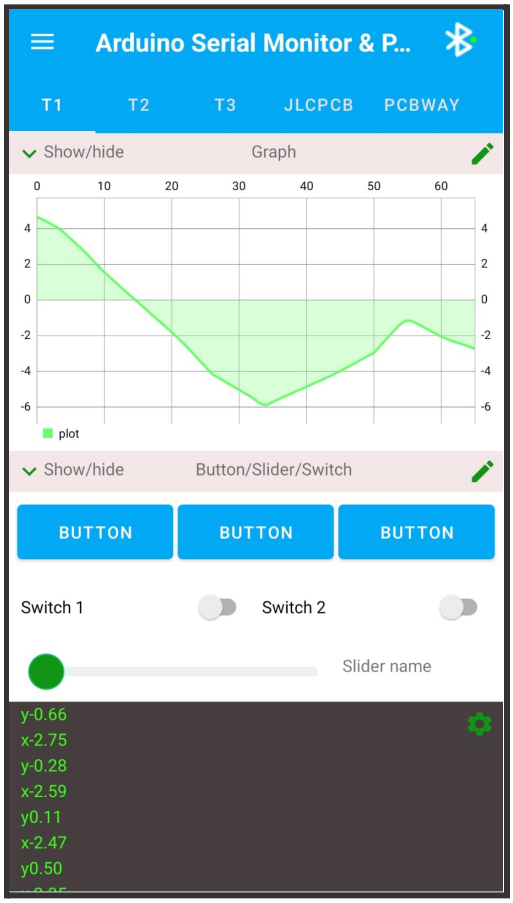


Figure 13: Arduino Serial Plotter showing the serial data and a plot

BILL OF MATERIALS

Component	Cost/Unit	Quantity	Cost
Boe-Bot Chassis	$25	1	$25
Continuous Servo Motor	$15	2	$30
Arduino UNO R3	$20	1	$20
HC-SR04 Ultrasonic Sensor	$2	3	$6
HC-05 Bluetooth Module	$10	1	$10
Custom Designed Sensor Mast	No Cost	1	-
Custom Designed Housing	No Cost	1	-
Battery Pack	$7	2	$14
Buzzer	$0.5	1	$0.5
Switch	No Cost	1	-
Total Cost			**$105.5**

COST ANALYSIS FOR MASS PRODUCTION

As discussed earlier the wide spectrum of applications for an autonomous following bot opens the doors to make different types of products in High Level Assembly Manufacturing. An analysis of the cost required to mass produce all these products will be an uphill task, hence only the electrical components that stay constant throughout all

the products that this bot can be mutated into were considered. These components were Arduino UNO R3, HC-SR04 Ultrasonic Sensors, and HC-05 Bluetooth Module, Buzzer.

DETAILED ANALYSIS AND OPTIMIZATION

The ultimate goal of the prototype was to exhibit an ultrasonic sensor based follower robot which can be upgraded with features relevant to whichever application one chooses to design it for. During prototyping, many factors were considered, analysed and optimised. A detailed description of these analyses is discussed below.

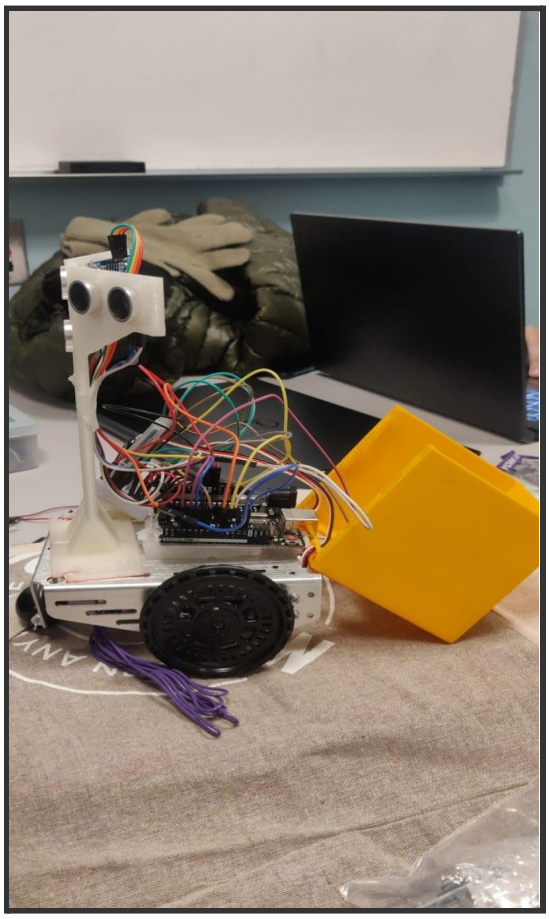


Figure 14: An exploded view of the bot

HEIGHT CONSIDERATIONS

This bot involves a Sensor holder or Mast to mount the ultrasonic sensors in the required orientation. Design considerations and optimisations made while designing the Mast are discussed in this section. The HC-SR04 Ultrasonic Sensor emits a sound wave(ping) at a 15 degree conical angle which can be depicted clearly as shown in Figure 15. To avoid any interference between the ping and the ground, i.e., to avoid the detection of ground by the ultrasonic sensor as an object/target, sensors are required to be placed at a minimum height from the ground.

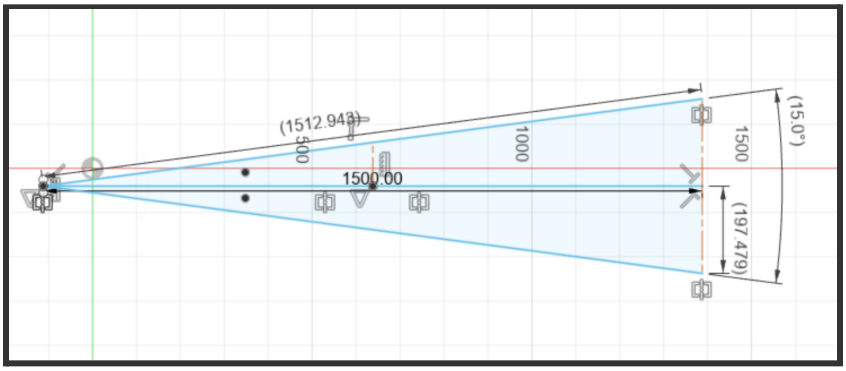


Figure 15: Field of View and Height Calculations (Side View)

Figure 15 depicts the side view for the Field of View(1) and Height Calculations for a single HCR04 Ultrasonic Sensor. The working range of the bot was considered to be 30 cm to 100 cm. Considering some factor of safety, the upper limit was increased and the design is optimized for 30-150 cm of working range. So for the bot to work efficiently even at the upper limit of the working range, there should be no interference from the ground at this distance. From Figure 15, it can be deduced that, for the sensors to not detect the ground at 150 cm, the center of the transmitter for the ultrasonic sensor should be placed at a minimum height of 197.479 mm or 19.7 cm above the ground level. The height of the main plate of the chassis with wheels of diameter 8cm, is 7cm from ground level. Hence, the height of the sensor mast should be a minimum of (19.7-7)cm = 12.7

cm. Taking this result into consideration, the designed sensor mast was chosen to be at a height of 15 cm.

FLANK ANGLE OPTIMIZATION

The main objective of the bot is to follow the nearest target it gets locked into, so there is a constraint that the Robot's Field of View(3) should not exceed 180 degrees. For proper working of the bot, the three sensors should be oriented properly i.e. The Flank Angle should be optimized. To determine the required flank angle, the Field of view of the sensors and the deadband(4) should be optimal.

For this, three cases for the flank angles were considered: 15 degrees, 30 degrees, 45 degrees. All the calculations were made considering the working range to be 100 cm.

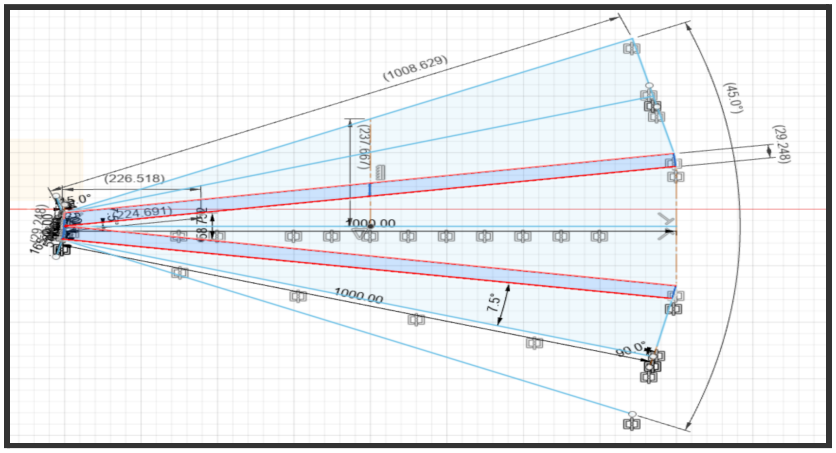


Figure 16: Field of View and Dead Band Calculations (Top View) for 15 degree flank angle

Figure 16 represents the Top View for the Robot's Field of View and Dead Band Calculations for three HCR04 sensors performed for 15 degree flank angle.

The region highlighted in red lines represents the dead band, and the Field of View comes out to be 45 degrees. Even though the dead band turns out to be very less for this case,

the Robot's Field Of View comes out to be low, and there is a possibility of interference (due to slight convergence observed in the dead band) of sensor echoes with each other if the working range is increased beyond 100 cm.

Some definitions of terms is given below:

- Field of View: The conical angle within which a single sensor can detect an object
- Robot's Field of View: The conical angle within which a Robot can detect an object
- Flank Angle: The angle between the vertical column and either flank of the Sensor Mast from the top view.
- Deadband: Dead Band is defined to be the area or zone within the Robot's Field of View, within which an object cannot be detected.

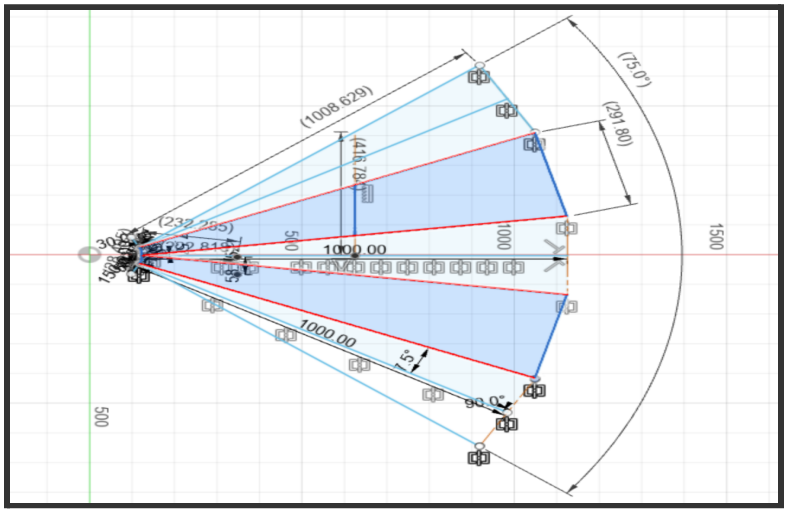


Figure 17: Field of View and Dead Band Calculations (Top View) for 15 degree flank angle

Figure 17 represents the Top View for the Robot's Field of View and Dead Band Calculations for three HCR04 sensors performed for 30 degree flank angle.

The region enclosed in red lines represents the dead band, and the Robot's Field of View comes out to be 75 degrees. In this case, the width of the dead band is diverging and it comes out to be 291.8mm or 29.18 cm.

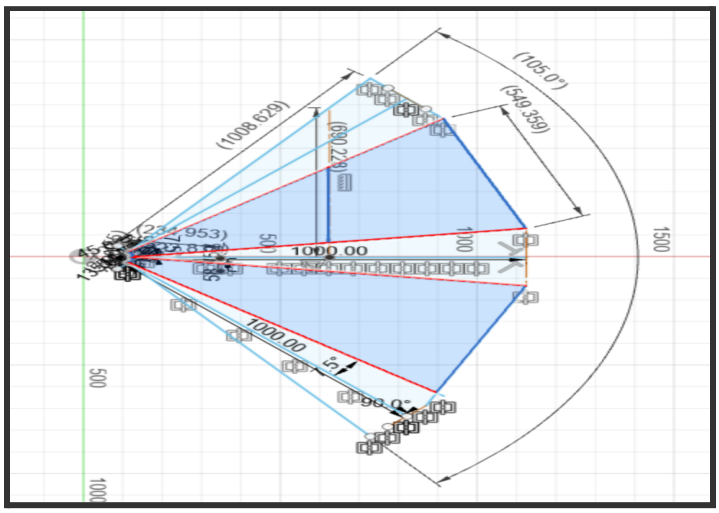


Figure 18: Field of View and Dead Band Calculations (Top View) for 45 degree flank angle

Figure 18 depicts the Top View for the Robot's Field of View and Dead Band Calculations for three HCR04 sensors performed for 45 degree flank angle.
The region enclosed in red lines represents the dead band, and in this case, the Robot's Field of View comes out to be 105 degrees. In this case, the width of the dead band is 549.359 mm or 54.93 cm. Though there is an increased Field of View, the deadband is more in this case. This might lead to increasing situations where the object goes undetected for all the three sensors. Hence, the case of 45 degree Flank angle is disregarded.

Without relying only on the mechanical design drawings and theoretical interpretations, Sensor Masts with both 15 degree and 30 degree Flank Angles were 3-D printed. After

running numerous iterations and performing various tasks with both the Masts, it was concluded that the bot runs efficiently with the '30 degree Flank angle Sensor Mast' thereby declaring this design to be optimal.

Sensor Orientation:

The ultrasonic sensors were mounted on the mast in such a way that the ping transmitters of the sensors were closer to the center of the mast and the echo receivers were towards the end of the flank. Keeping the transmitters closer to the center provided a narrower deadband and keeping the receivers as further from the center as possible avoided the possibility of interference between the echoes of two adjacent sensors.

Range Of Operation:

Concluding from multiple tests and trials it was realised that the optimum position for object detection by the sensor was 75 cm (this is specific to the mast with 30 degree flank angles) from the sensor mast. Hence the range of operation was set between 30cms and 100 cms. The robot obviously would be functional even beyond these limits but won't perform as well.

Coding Considerations:

Since this is a real-time robot system, it requires a multi-threaded processor to parallely process multiple tasks at a time. But since resources were limited to use of a single thread single core processor, the code had to be designed and optimized to take almost negligible time between two tasks. Some of the key tricks were:

- Code without the use of any functions. If functions are used, the function call statement causes the program flow to move to another location, execute the lines of the function, and then return to the previous sequence. This includes passing and returning parameters through variables to and from the function.
- Use if-else loops instead of just if loops or switch cases.
- Evaluate multiple conditions in the same line.

- No redundant computations
- Programmed all tasks with least amount of machine cycles

MATHEMATICAL BACKGROUND

Mathematical Computations and Geometry analyses were needed to optimize the Sensor Mast Design as well as design the Path Tracing Algorithm. Consequently an optimum range was selected for the functioning and positioning of the robot behind the target.

- The sensor mast geometric optimization has been explained earlier in the Detailed Analysis part.
- Formula for calculation of distance of the target from the ultrasonic sensor:

  $$Distance = Velocity * Time$$

  Velocity= Velocity of sound in free air= 343m/s

  Time= Time measured by the sensor between the transmission of the sound wave from the sensor and the reception of the reflected (echoed) sound wave by the sensor. This quantity was measured each execution cycle of the robot code.

- Path Tracing Algorithm - Dead Reckoning
  - Xt=Yt=angle=i=0; Path array has a dimension of 2x4000
  - Value for rotation per loop = d deg/loop
  - Value of translation per loop = dist cm/loop
  - When moving left:
    - angle= angle + d
  - When moving right:
    - angle= angle - d
  - When moving straight:
    - Xt=Xt - dist*sin(angle)
    - Yt=Yt + dist*cos(angle)

- Path[0,i]= Xt
- Path[1,i]= Yt
- i++

Variable ‘d’ stores a predefined value of degrees rotated when the left or right rotation code is accessed each time.

Variable ‘dist’ stores a predefined value of the distance traveled when the translational motion code is accessed each time.

NOTE: the values of ‘d’ and ‘dist’ were derived from the testing results of the robot.

Here the variable “angle” stores and updates the angle of the robot head in a 360 degree space with respect to the initial robot head direction(starts at zero degrees).

“Xt” and “Yt” store the current, constantly updated ‘x’ and ‘y’ coordinates of the robot’s position.

“Path” array is used to store the value of Xt and Yt every time the robot translates. “i” variable is used to index the “Path” array columns by iterating every cycle.

ADVANTAGES

- Autonomous operation
- Optional manual driving mode
- Remote Controllability
- Easily implementable for numerous applications
- Dead-Reckoning based path tracking hence data cannot be intercepted.
- Low cost design
- Physical safety switch directly between the system and power source.

LIMITATIONS

- Deadband of 15 degrees: Between the sound transmission angle of each ultrasonic

sensor. ~25 cm deadband at 100 cm distance.

- Pulsating motor behavior - since only one pulse of motion can be sent per loop of code.
- The detection is great for only smooth surfaced objects. With higher computation power, more complex algorithms can be implemented to perceive complex objects better.
- Will lock on to the nearest object as its target.

FUTURE SCOPE & UPGRADES

Though the project has completed, some additional features can be added to bring in some upgrades to the bot in the future which help in expanding the efficiency and applications of the bot. some of the future scope and upgrades can be as follows

1. Addition of GPS sensor

   Currently the bot is storing the locations and doing path tracking with respect to the bot's local coordinates. But if a Gps sensor can be added to the bot it can store the location of the target and its current position with respect to the global frame. This helps to track the bot's current location in a global frame and also an additional feature can be added in which when the user enters the location coordinates and the bot navigates itself by using a GPS sensor on board.

2. Addition of Inertial Measurement Unit(IMU)

   An inertial measurement unit is an electronic device that is used to calculate and report a body's linear acceleration, orientation, and position. The IMU works on the principle of coupling various sensors like accelerometer, and gyroscope. To furthermore improve the accuracy of localisation of the bot and to reduce the error during path tracking an Inertial Measurement Unit (IMU) can be added.

3. Using LiDAR

   Although ultrasonic sensors are easily available, low on cost and easily available they have limitations. Ultrasonic sensors cannot differentiate between object types. For example, when the bot is following a target and encounters a stationary object in its field of vision it misinterprets the stationary object as target and may stop

following the target. So in order to avoid this problem a higher resolution ultrasonic sensor or some advanced sensors like LiDAR can be used to avoid the inaccuracies in the bot while following a target. Though adding these advanced sensors like LiDAR increases the price of the bot it can be useful in military applications where precision is more important than cost.

4. Using Multi-core Processor

   The main reason for using Arduino instead of basic stamp 2 in the project is because of high processor speed and its pseudo multi-processor in Arduino. The bot has to compute the input from three ultrasonic sensors and send, read data from the Bluetooth module and control the servo motors. All this process is to be done in parallel. In this case the Arduino is used and the code is optimized in such a way that the time taken by each task is negligible. But for faster movement and tracking of the bot a multi-core processor can be used.

5. Using faster Motors

   The servo motors that are used in the project are parallax servo motors which have a speed of 0-50 Rpm. So even though the bot is moving it might not be able to keep up with fast moving targets so faster motors with better speed control can be used for example BLDC motors. And also motors with embedded PID controllers can be used so that the bot can maintain a particular distance behind the target and more optimized turning action.

APPLICATIONS

Some of the applications the project have are as follows

1. Load carrying military

   The bot can be used in the military to carry the military ruck. This military ruck weighs at least 45 pounds and can cause tiredness among military personnel when carrying for longer durations so the bot can be used to carry the bag and follow the personnel through the terrain.

2. Swarm Robots

Swarm robot is the coordination between multiple robots which work as a system to perform a specific task. In swarm robotics often one robot should follow a robot or be able to read the state of another robot. So one of the main tasks of this is to make a robot follow another robot and to help it in performing various tasks. The bot can be made to follow a robot.

3. Shopping carts

Whenever one goes to supermarkets or large stores they often take a trolley and drag it wherever they go in that place. This bot can be made as a shopping cart which follows the customer in the store without the need of him pushing the cart. This may make the shopping experience for a customer hassle free

4. Medical applications

This bot can be used in hospitals to follow the patients and carry the IV fluids the patients need. And also it can be used by the nurses as a trolley that follows them automatically while they are visiting the patients to give the necessary medications.

5. Light assistance robot

With added mobile phone control in the bot one can use the bot with lights on its heads to lead the path so that the user won't stumble or hit any objects in the dark. But many may think that one can use a smartphone for it. But when the user has luggage to carry and wants to walk in the dark, the user can use the bot to carry the luggage and lead the path at the same time.

6. Tracking Vehicles

One can see road trains and Long Combination Vehicles (LCV) which are used to move road freight. Some of the disadvantages of these LCV's is that they cannot turn easily because of trailers connected through a link but this can modify this bot in a way that each trailer follows the leading trailer without any linkages. And also a convoy of trucks carrying freight over long distances cost a lot. One can use

this bot in a way that the trucks follow the leading truck without the need of a driver which in turn reduces the cost by eliminating the use of additional drivers to transport the freight.

CODE

```
//no functions were created in the code, to increase computation speed (no referencing and function calling)
#define ult1 5  //left ultrasonic trigger
#define ule1 4  //left ultrasonic echo
#define ult2 6  // center ultrasonic tigger
#define ule2 7  //center ultrasonic echo
#define ult3 12  //right ultrasonic trigger
#define ule3 13  //right ultrasonic echo
#define mleft 10 //left motor signal line
#define mright 9 //right motor signal line
#define hcrx 0   //bluetooth receive
#define hctx 1   //bluetooth transmit
#define buzz 3   //buzzer
int dr = 0;
int dl = 0;
int dc = 0;
int right=0;
int left=0;
int straight=0;
float angle=0;
int i=0;
float deg=3.3;    //in degrees
float dist=0.41;  //in cms
float X=0;
float Y=0;
float XT=0;
float YT=0;
float path[2][4000]; //array to store the path taken
int r;
int c;
int l ;
char junk;
String inputString="";
String inputString1="";
```

```
char flag;
//-------------------------------------------------------------
void setup() {
  //initialization
  pinMode(ult1, OUTPUT);
  pinMode(ult2, OUTPUT);
  pinMode(ult3, OUTPUT);
  pinMode(ule1, INPUT);
  pinMode(ule2, INPUT);
  pinMode(ule3, INPUT);
  pinMode(mleft, OUTPUT);
  pinMode(mright, OUTPUT);
  pinMode(hctx, OUTPUT);
  pinMode(hcrx, INPUT);
  pinMode(buzz, OUTPUT);
  Serial.begin(9600);
}
//---------------------------------------------------------------

void loop() {
  flag='z';
  inputString="";
     //receive data from bluetooth
     if(Serial.available()){
  while(Serial.available())
    {
     char inChar = (char)Serial.read(); //read the input
     inputString += inChar;        //make a string of the characters coming on serial
    }
    //Serial.println(inputString);
    while (Serial.available() > 0)
    { junk = Serial.read() ; }      // clear the serial buffer
    if(inputString == "f"){        //in case of 'f' start the autonomous following mode

  //-------------------------------------------------------------------
 while(1)
 {
  //all sensor readings are taken at the same instant so all the further distance comparisons are
more relevant
  //read distance from central ultrasonic sensor
```

```
  digitalWrite(ult2, LOW);
  delayMicroseconds(2);
  digitalWrite(ult2, HIGH);
  delayMicroseconds(10);
  digitalWrite(ult2, LOW);
  int duration1 = pulseIn(ule2, HIGH);
  dc = (duration1 * .0343) / 2;
  //---------------------------------------------------------------
  //read distance from central ultrasonic sensor
  digitalWrite(ult1, LOW);
  delayMicroseconds(2);
  digitalWrite(ult1, HIGH);
  delayMicroseconds(10);
  digitalWrite(ult1, LOW);
  int duration2 = pulseIn(ule1, HIGH);
  dl = (duration2 * .0343) / 2;
  //---------------------------------------------------------------
  //read distance from central ultrasonic sensor
  digitalWrite(ult3, LOW);
  delayMicroseconds(2);
  digitalWrite(ult3, HIGH);
  delayMicroseconds(10);
  digitalWrite(ult3, LOW);
   int duration3 = pulseIn(ule3, HIGH);
  dr = (duration3 * .0343) / 2;
 //---------------------------------------------------------------
r= abs(dr); //eliminating negative values that might arise due to noise
c= abs(dc);
l= abs(dl);
if (c<r && c<l && c > 30 && c<120)  //if true, robot moves straight
{

     digitalWrite(mright, HIGH);
     delay(0.6);
     digitalWrite(mright, LOW);
     digitalWrite(mleft, HIGH);
     delay(2.4);
     digitalWrite(mleft, LOW);
     delay(3);
     //the path tracing algorithm is executed
```

```
    XT= XT - (dist*sin(angle*(3.14/180)));
    YT= YT + (dist*cos(angle*(3.14/180)));
    path [0][i]= XT;
    path[1][i]= YT;
    i++;
    //end of path tracing
    //plotting on smartphone
    Serial.print("x");
    Serial.println(XT);
    Serial.print("y");
    Serial.println(YT);
    //end of plotting

}
else if (l<c && l<r && l<110) //if true, robot turns left
{
    digitalWrite(mright, HIGH);
    delay(0.6);
    digitalWrite(mright, LOW);
    digitalWrite(mleft, HIGH);
    delay(0.6);
    digitalWrite(mleft, LOW);
    delay(3);
    angle= angle + deg; //updating heading angle of bot

}
else if (r<c && r<l && r<110) //if true, robot turns right
{
    digitalWrite(mright, HIGH);
    delay(2.4);
    digitalWrite(mright, LOW);
    digitalWrite(mleft, HIGH);
    delay(2.4);
    digitalWrite(mleft, LOW);
    delay(3);
    angle= angle - deg; //updating heading angle of bot
}
else
{
  if(c>120 && l>110 && r>110) //buzz if target is lost or unavailable
```

```
  {
  digitalWrite(buzz, HIGH);
  delay(1000);
  digitalWrite(buzz, LOW);
  }
}
if(Serial.available()) //checking if stop button is pressed
{
flag=(char)Serial.read();
junk=Serial.read();
if(flag == 'x'){ //x corresponds to the stop button
 return;
}
}
}
}
else if(inputString == "m"){ //m corresponds to manual mode operation
  while(1)
  {
  flag='z';
  if(Serial.available())
  flag=(char)Serial.read();
  if(flag=='w')  //w corresponds to move straight in manual mode
  {   digitalWrite(mright, HIGH);
      delay(0.6);
      digitalWrite(mright, LOW);
      digitalWrite(mleft, HIGH);
      delay(2.4);
      digitalWrite(mleft, LOW);
      delay(3);
  }
  else if(flag=='a') //a corresponds to turn left in manual mode
  {
      digitalWrite(mright, HIGH);
      delay(0.6);
      digitalWrite(mright, LOW);
      digitalWrite(mleft, HIGH);
      delay(0.6);
      digitalWrite(mleft, LOW);
      delay(3);
```

```
  }
   else if(flag=='d') //d corresponds to turn right in manual mode
   {
     digitalWrite(mright, HIGH);
     delay(2.4);
     digitalWrite(mright, LOW);
     digitalWrite(mleft, HIGH);
     delay(2.4);
     digitalWrite(mleft, LOW);
     delay(3);
   }
   else if(flag=='s') //s corresponds to move backwards in manual mode
   {
     digitalWrite(mright, HIGH);
     delay(2.4);
     digitalWrite(mright, LOW);
     digitalWrite(mleft, HIGH);
     delay(0.6);
     digitalWrite(mleft, LOW);
     delay(3);
   }
   else if(flag=='x') //to stop manual mode
   {
    return;
   }
   }
   }
  }
}
```

REFERENCES

1. https://forum.arduino.cc/
2. https://doi.org/10.1088/1757-899x/705/1/012045 , A Study of Ultrasonic Sensor Capability in Human Following Robot System
3. https://doi.org/10.1016/j.procs.2015.12.310. , Development of Human Following Mobile Robot System Using Laser Range Scanner.