# All-Terrain Robot for Controlling Wildfires (Term project)

Yadukrishna BG
*MAE Department*
*NYU Tandon School of Engineering*
ybg211@nyu.edu

Deep Trivedi
*MAE Department*
*NYU Tandon School of Engineering*
dht258@nyu.edu

Shivam Joshi
*MAE Department*
*NYU Tandon School of Engineering*
shivam.joshi@nyu.edu

***Abstract*—A prototype of all-terrain robot has been fabricated as first module of project in developing a robot for assistance in wildland firefighting. Aim is to follow closely and inculcate most of the potential functions of a ground robot as discussed in report by Texas Engineering Extension Service [1]. Complete list of these functions is as follows:**

**1. Hualing**
**2. Direct Fire Suppression**
**3. Mobile Weather Station**
**4. Reconnoiter**
**5. Hot Spot Identification**
**6. Investigate Fire Hazard Zone.**

**Through a series of mini-projects we have implemented the first four functionalities. In this part we will add following functions to existing robot prototype:**

- **Hot Spot Identification**
- **Investigate Fire Areas.**

**This will be done using a raspberry Pi camera, communication between Pi and arduino and some web-services from Google.**

***Index Terms*—unmanned ground robots, rescue robots, wildfire, firefighting, rocker-bogie, Arduino**

## INTRODUCTION

This report presents the design considerations, fabrication methods, underlying arduino sketch, Raspberry Pi recipe, mechatronics design, code documentation, results, and conclusions for the Integrated term project. As we have been focusing, wildfires have become a recent concern due to increased frequency through out the world. Wildland firefighting is not a novel engineering domain though it is indeed a developing application in the field of robotics. We tried to abate the gap in desired functions and current implementations adopted by smoke jumpers - the group of most elite firefighting squads in USA.

As mentioned in the report [1] there are fewer attempts in scientific literature regarding ways to tackle requirements of wildland firefighters and hence there are limited implementations like the work being done under this project.

## OBJECTIVE

The main purpose of the robot is to provide assistance during wildfires by providing insightful data, perform remote tasks and creating firelines. In this Firelines or anchor points in terms of wildfire fighting are basically natural or manually created gaps in the vegetation beyond which wildfires do not proceed due to absence of fuel. An example of firelines is shown in the fig. 1. In order to meet these requirements, the robot is quipped with capabilities of remote operation, remote surveillance and remote manipulation.

## MECHANICAL DESIGN

**Mobile Base:** The robot uses rocker-bogie mechanism (shown in figure 2) as it allows robot to move over obstacles as large as twice the size of robot wheel while keeping all six wheels on ground during the maneuver, example includes passing through pile of fallen trees in our case. This is achieved virtue of absence of springs and stub axles for wheels. Springs and other suspension systems undesirably limit the tilt stability by the height of centre of gravity and tend to tip easily as the loaded side yields. Moreover, the mechanism minimizes

Fig. 1. Representation of a fireline

dynamic shocks and potentials damages to the robot when obstacles are encountered.

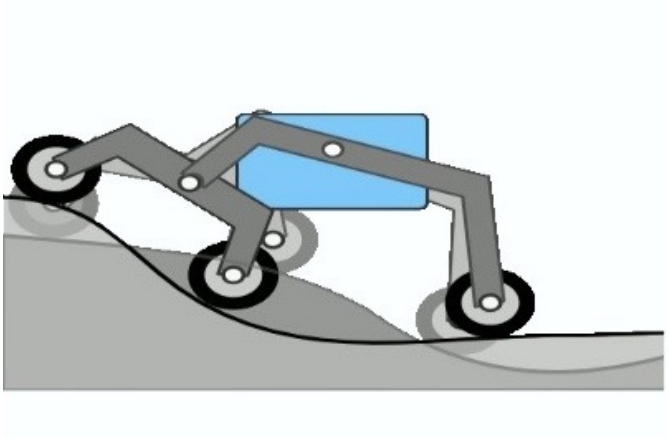


Fig. 2. Typical rocker-bogie mechanism

Because of the above mentioned features, The Rocker-Bogie system has been the suspension arrangement used in the Mars rovers. It is currently NASA's favored design for rovers and multiple rovers including Spirit, Opportunity, and Curiosity.

**Manipulator:** The Manipulator Arm has a simple mechanical design. It is a 3-Degree of Freedom robotic arm having a gripper as it's end-effector.The manipulability and the dexterity of any robotic manipulators depend upon its degree of the redundancy. Serial robotic arm is very popular in industrial applications because of its simplistic designs. Serial robotic manipulators are also designed for the joint fault tolerance. The design of 3-Degree of Freedom serial robotic arm has been presented in figure.2. Its mechanical structure has been developed using the CAD software.

In robotics terminology, an end-effector is the device at the end of a robotic arm, designed to interact with the environment. In the strict definition,the end effector means the last link of the robot. The exact nature of this device depends on the application of the robot. Here, we have a Gripper as our end-effector as it is required to grip the tree firmly in order to let the cutter mechanism function without any discrepancy.

## FABRICATION METHODS

The structure of the robot combines a chassis, a platform for microcontrollers , motor drivers, and batteries, motor mountings at the base, six DC motors, some additional accessories for joining the links and clamping motors, and wide thumper wheels for higher traction. In chassis, the links of the mechanism are made using easily accessible u-PVC pipes as they serve the purpose of providing sturdiness to the whole mechanism and keeping the total weight low at the same time. The platform for the robot is 3D printed using PLA material. The platform is used to mount Arduino Mega, motor drivers, batteries ultrasonic and weather sensors. Same PLA material is used for 3D printing motor mountings too which serve as stable housing to sustain high torque motors. In figure 2 we have shown the robot chassis mounted with motors, mo-

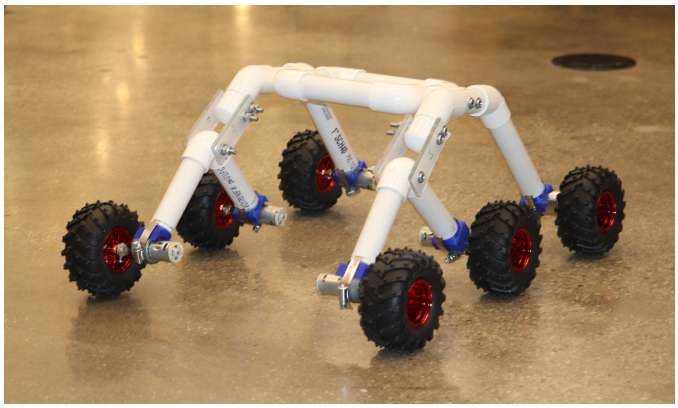


Fig. 3. Chassis with motor, wheels mounted

tor mountings and thumper wheels mounted. Metal

clamps are used to further fasten the motors rigidly to the chassis.

For the manipulator, we have used the 3D printed platform from the first phase to mount the manipulator base. The manipulator arm has three linkages and a gripper. The linkages are basically iron clamps used to hold the whole structure in place.The fabrication of linkages is done keeping in mind the dimension of motor and shaft so that motors easily fit in appropriately.The actual representation is seen in fig 4.

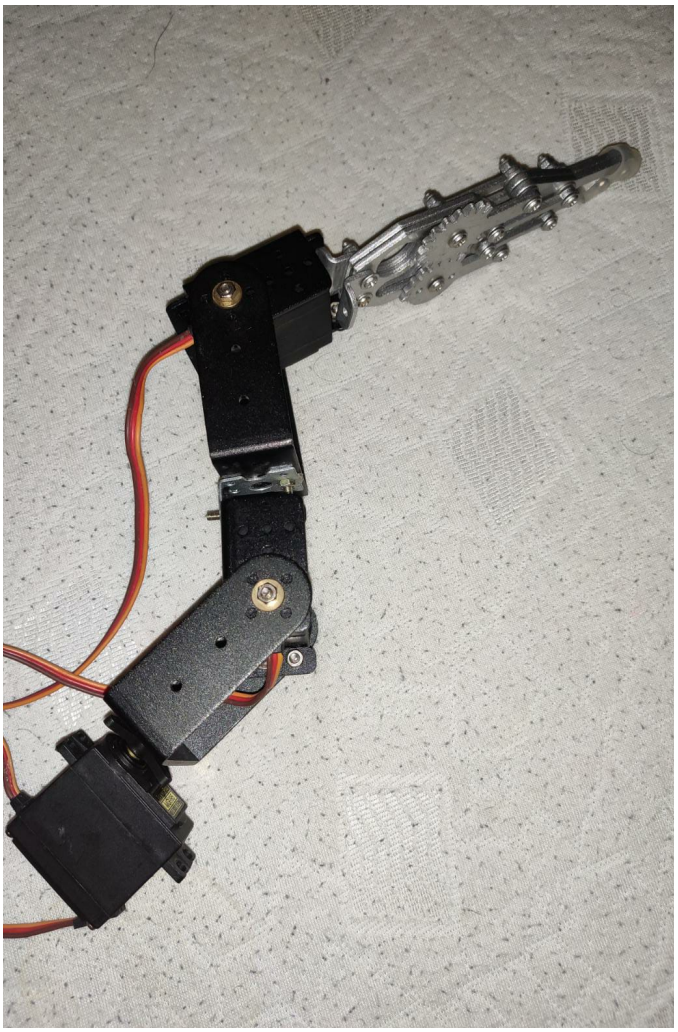


Fig. 4. Actual Manipulator Arm

## MECHATRONICS COMPONENTS AND CIRCUIT DESIGN

Mechatronics components used in this phase of the project include:

1. Ardiuno Mega microcontroller (ATmega2560)
2. Bluetooth Module (HC-05 BSM)
3. Motor drivers (L298N)
4. Li-ion Battery (7.4V 1500mAh)
5. Alkaline Battery (9V)
6. Humidity and Temperature Sensor (DHT 22)
7. Ultrasonic Sensor (HC - SR04)
8. DC Motors
9. Raspberry Pi
10. Raspberry Pi Camera

Arduino Mega was used over the more popular design Uno because of the availability of 15 PWM output pins for future requirements. Bluetooth connection is established between the smartphone and Mega microcontroller using a HC-05 module and android application. The commands sent via smartphone are read by the microcontroller and then it directs corresponding commands to the motor drivers in three L298N motor driver which then control motors. The motor drivers employ a 7.4V - 1500mAh Li-ion Battery which solely drives the motors. We have used another power source to power the microcontroller circuitry using a small 9v alkaline battery.

For creating the mobile weather station on top of the platform a humidity and temperature (DHT 22) is being used. This particular sensor is more sensitive and accurate which made us choose the sensor over other available counterparts like DHT11 and DHT14.

A separate section is denoted to the sensor readings in the mobile application where the readings from sensor sent to the microcontroller are sent to the bluetooth module and then to the smartphone app interface. Hence, we have a two way communication and data is being sent both from the smartphone app to the microcontroller to actuate motors and from sensors to microcontroller to smartphone to monitor the weather conditions online. We have also included a ultrasonic sensor to calculate the distance to any obstacle in front of the robot. This distance also has a designated space in the smartphone app to be monitored. Moreover, we have implemented code in such a way that if the distance from the obstacle is too small the robot will automatically stop in order to avoid collision.

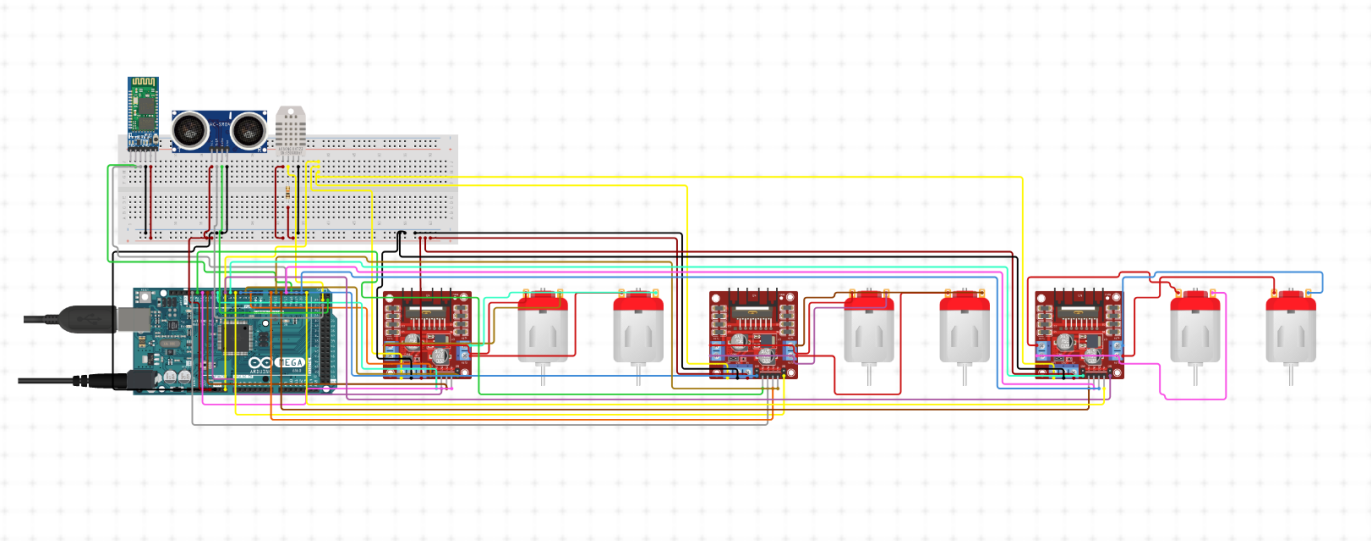


Fig. 5. Circuit Diagram

The circuit design for the robot is depicted in Figure 5 which shows six DC motors connected to three L298N motor drivers. One motor driver is being used to control two motors. We are driving the motors using a separate 12v Li-polymer battery. This is done to keep the microcontroller separate from the actuator circuit which is essential here as a safeguard measure.

## SOFTWARE DOCUMENTATION

Before any description, it is necessary to mention the communications taking place in the project, Refer fig. 6.
Basically there are four main nodes in this communication network. We are giving commands to move robot from a mobile phone to the Firebase based realtime DB, then these commands are being fetched by raspberry Pi from the firebase webserver, then these commands are being forwarded to arduino using USB serial communication. Moreover, we wished to use same channel from sensor mounted on arduino, sent back to raspberry pi and then over internet sent to the mobile app, but we have currently opted for bluetooth communication from arduino to nearby mobile phone due to lack of resources. We have described below all different codes for each of these nodes of the project.
The programming effort in this project consisted of followin domains:

1. Firebase Database Setup
2. Arduino program
3. Raspberry Pi program
4. Android application program

**Firebase:** We have done setup for a realtime DB using google service called Google-Firebase (https://console.firebase.google.com).
Creating such a database on Firebase website (you will have to log in with your Google account) requires following steps:

1. Click on the *Get Started* button which will take you over to the firebase console.
2. Create a new project by clicking on the *Add Project* button, fill in the requirements (name, details, etc) and complete by clicking on the *Create Project* button.
3. Select *database* from the menu on the left-hand side.
4. Click on the *Create Database* button, select the *test mode* option.
5. Set the database to a *realtime database*. Select the *rules* tab and change them to *true*.
6. Finally click on the *data* tab and copy the database URL.

**Arduino:** Once the setup of database is done, through arduino program, we are primarily using serial communication to take inputs from the user

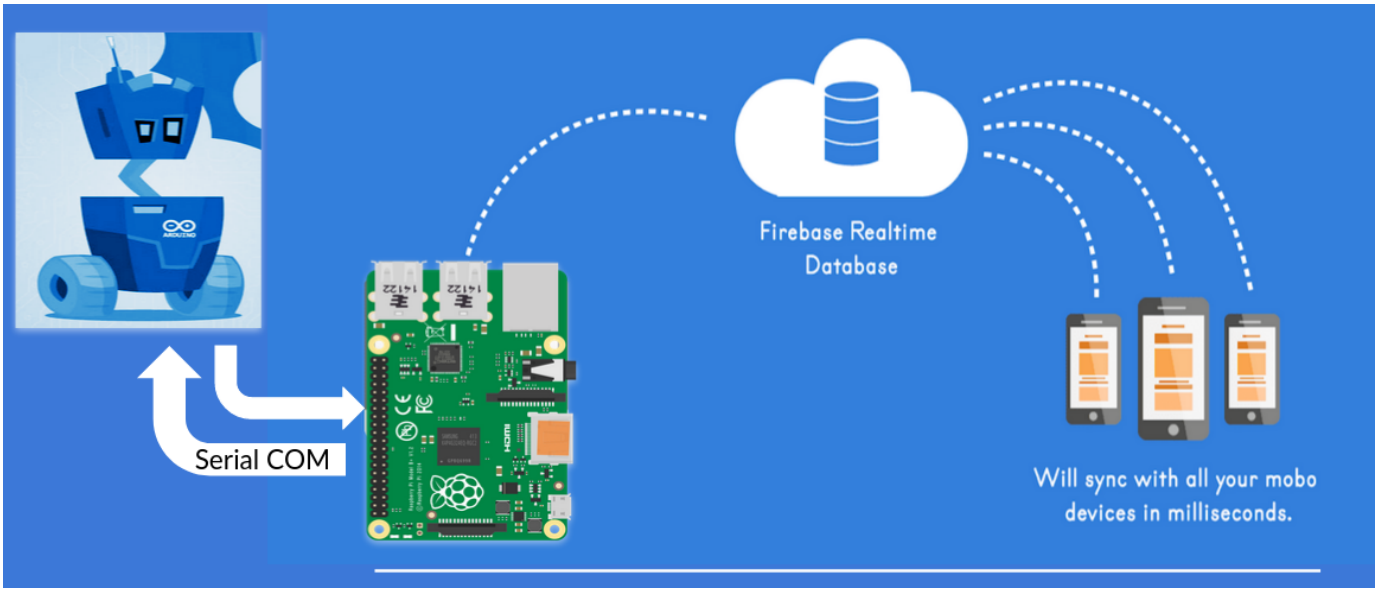



Fig. 6. Communication

of android app(coming from Raspberry Pi via firebase) and using those inputs to send commands to motors in order to drive motors. Along with this function we have also deployed in the code, a functionality to send processed sensor data to the android app so that the user is notified of the weather conditions surrounding robot using bluetooth. We have included following libraries for our code in form of header files:

1. SoftwareSerial
2. DHT_U
3. Servo

Here, SoftwareSerial library has built-in support from arduino hardware and is being used for serial communication. This uses universal asynchronous receiver-transmitter (UART) which allows the Atmega chip to receive serial communication even while working on other tasks, as long as there room in the 64 byte serial buffer. This library is being used here for USB serial communication between RPi and arduino Mega as well as for bluetooth communication between arduino and smartphone. Basically, we are using two serial communication channels, one to transfer data between arduino and RPi, and another for bluetooth.

Second library used here, i.e. DHT_U (Adafruit DHT Humidity & Temperature Sensor Library) is an Arduino library for the DHT series of low-cost temperature/humidity sensors. This library is being used to send data to the android app using serial communication.

We have used baud rate at 9600 for both Serial and Bluetooth, as used in most of the sensor communications. Some basic calculations are also done to get the distance in centimeters from the ultrasonic readings. Such calculations were not required for DHT sensor virtue of available library. a function for manipulator is also shown to represent manipulator to be mounted on the robot. The full program used is shown in Figure 7, 8, 9 and 10.

**Raspberry Pi:** We need some modeules in Pi before getting started. These include:
1. python-firebase module, and
$ sudo apt-get python-firefox)

2. RPi-Cam-Web-Interface module
($ git clone https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git)
$ cd RPi_Cam_Web_Interface
$ ./install.sh

```
#include <SoftwareSerial.h>
#include <DHT.h>
#include <DHT_U.h>

#define echopin  40 // echo pin
#define trigpin 44 // Trigger pin
#define DHTPIN 2     // Digital pin connected to the DHT sensor
#define DHTTYPE    DHT22     // DHT 22 (AM2302)
#define dht_pin 2 // Pin sensor is connected to

DHT_Unified dht(DHTPIN, DHTTYPE);
SoftwareSerial BT(22,24);


 int MotorAinput1 = 36;
 int MotorAinput2 = 34;
 int MotorBinput1 = 4;
 int MotorBinput2 = 5;
 int MotorCinput1 = 6;
 int MotorCinput2 = 7;
 int MotorDinput1 = 8;
 int MotorDinput2 = 9;
 int MotorEinput1 = 10;
 int MotorEinput2 = 11;
 int MotorFinput1 = 12;
 int MotorFinput2 = 14;

 int state;
 int Speed = 130;

 uint32_t delayMS; // humidity sensor.

 int temp;
 int hum;

 int timer = 0;

 int distanceFwd;
 long duration;

 int chk = 0;
 int set = 10;

// Functions
void backward(){
    digitalWrite(36,HIGH);digitalWrite(34,LOW);digitalWrite(4,LOW);digitalWrite(5,HIGH);digitalWrite(6,HIGH);digitalWrite(7,LOW);digitalWrite(8,LOW);digitalWrite(9,HIGH);digitalWrite(10,LOW);digitalWrite(11,HIGH);digitalWrite(12,LOW);digitalWrite(14,HIGH); }

void forward(){
```

Fig. 7. Arduino Code 1

```
// Functions
void backward(){
    digitalWrite(36,HIGH);digitalWrite(34,LOW);digitalWrite(4,LOW);digitalWrite(5,HIGH);digitalWrite(6,HIGH);digitalWrite(7,LOW);digitalWrite(8,LOW);digitalWrite(9,HIGH);digitalWrite(10,LOW);digitalWrite(11,HIGH);digitalWrite(12,LOW);digitalWrite(14,HIGH); }

void forward(){
    digitalWrite(36,LOW);digitalWrite(34,HIGH); digitalWrite(4,HIGH);digitalWrite(5,LOW); digitalWrite(6,LOW);digitalWrite(7,HIGH);digitalWrite(8,HIGH);digitalWrite(9,LOW);digitalWrite(10,HIGH);digitalWrite(11,LOW);digitalWrite(12,HIGH);digitalWrite(14,LOW);}

void turnRight(){
    digitalWrite(36,LOW);digitalWrite(34,HIGH);digitalWrite(6,LOW);digitalWrite(7,HIGH);digitalWrite(10,HIGH); digitalWrite(11,LOW);digitalWrite(4,LOW);digitalWrite(5,HIGH);digitalWrite(8,LOW);digitalWrite(9,HIGH);digitalWrite(12,LOW);digitalWrite(14,HIGH);}

void turnLeft(){
    digitalWrite(36,HIGH);digitalWrite(34,LOW); digitalWrite(6,HIGH);digitalWrite(7,LOW); digitalWrite(10,LOW);digitalWrite(11,HIGH); digitalWrite(4,HIGH);digitalWrite(5,LOW);digitalWrite(8,HIGH);digitalWrite(9,LOW);digitalWrite(12,HIGH);digitalWrite(14,LOW);}

void Stop(){
    digitalWrite(36,LOW); digitalWrite(34,LOW); digitalWrite(4,LOW); digitalWrite(5,LOW); digitalWrite(6,LOW); digitalWrite(7,LOW);digitalWrite(8,LOW);digitalWrite(9,LOW);digitalWrite(10,LOW);digitalWrite(11,LOW); digitalWrite(12,LOW); digitalWrite(14,LOW);}

long data()
{  digitalWrite(trigpin,LOW); delayMicroseconds(2);  digitalWrite(trigpin,HIGH);  delayMicroseconds(10);  duration=pulseIn (echopin,HIGH);
  return duration / 29 / 2; }

void setup() {
// ultrasonic

    pinMode (trigpin, OUTPUT);
    pinMode (echopin, INPUT );

// Temp and humidity

dht.begin();
  // Print temperature sensor details.
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);

  // Print humidity sensor details.
  dht.humidity().getSensor(&sensor);

  // Set delay between sensor readings based on sensor details.
  //delayMS = sensor.min_delay / 1000;


// Motor
  pinMode(MotorAinput1, OUTPUT);
  pinMode(MotorAinput2, OUTPUT);
  pinMode(MotorBinput1, OUTPUT);
  pinMode(MotorBinput2, OUTPUT);
  pinMode(MotorCinput1, OUTPUT);
  pinMode(MotorCinput2, OUTPUT);
```

Fig. 8. Arduino Code 2

The code in Pi is written in Python and it basically imports serial library to communicate with arduino. It also imports a module called firebase and creates an object from class firebase using function FirebaseApplication. then in continously checks for incoming data from firebase and saves it as variale direction, later sends it to arduino using function ser.write()
Full code is showin in figure 11.

Code documentation for Android app is described in next section.

## APP DEVELOPMENT

We have used the services of MIT App Inventor to develop our Mobile application to control the bot remotely. We had used the app for phase-1 of the project but there are many changes to the previous version of the app. The primary role of this app is to be the bridge between the bot and human controller. The app has to have dual-connection back-end interface where it connects to the bot and retrieves the sensor data via Bluetooth module and simultaneously sends data to the bot via Firebase-Raspberry pi-Arduino communication.
The first step in the app building was to make

```
// Motor
  pinMode(MotorAinput1, OUTPUT);
  pinMode(MotorAinput2, OUTPUT);
  pinMode(MotorBinput1, OUTPUT);
  pinMode(MotorBinput2, OUTPUT);
  pinMode(MotorCinput1, OUTPUT);
  pinMode(MotorCinput2, OUTPUT);
  pinMode(MotorDinput1, OUTPUT);
  pinMode(MotorDinput2, OUTPUT);
  pinMode(MotorEinput1, OUTPUT);
  pinMode(MotorEinput2, OUTPUT);
  pinMode(MotorFinput1, OUTPUT);
  pinMode(MotorFinput2, OUTPUT);

  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  BT.begin(9600); // Setting the baud rate of Software Serial Library
}

char cmmd;

void loop() {
    //if some date is sent, reads it and saves in state
    if (Serial.available()) {
    cmmd = Serial.read();
    //Serial.print("You sent Arduino: ");
    Serial.println(cmmd);


    distanceFwd = data();
    Serial.println(distanceFwd);

  if((distanceFwd<set) && (chk==1)){chk = 2; Stop();}
  if(distanceFwd>set){chk = 0;}

  // if the state is '1' the DC motor will go forward initial
  if ((cmmd == '1') && (chk==0)){chk = 1; forward();Serial.println("Go Forward!");}

  // if the state is '2' the motor will Reverse initial
  else if (cmmd == '4'){backward();Serial.println("Reverse!");}

  // if the state is '3' the motor will turn left initial
  else if (cmmd == '2'){turnLeft();Serial.println("Turn LEFT");}

  // if the state is '4' the motor will turn right initial
  else if (cmmd == '3'){turnRight();Serial.println("Turn RIGHT");}
```

Fig. 9. Arduino Code 3

```
  // if the state is '4' the motor will turn right initial
  else if (cmmd == '3'){turnRight();Serial.println("Turn RIGHT");}

  // if the state is '5' the motor will Stop initial
  else if (cmmd == '0') {Stop();Serial.println("STOP!");}
}


if(BT.available() >= 0){
      //Serial.println("Connected");
      state = BT.read();
      //Serial.println(state);
      if(state > 10) { Speed = state;}

       timer = timer+1;
//BT.print("timer: ");
//BT.println(timer);

if(timer==200){
if(distanceFwd>200){distanceFwd=200;}
 BT.print("A");
 BT.print(":");
 BT.print(distanceFwd); //send distance to MIT App
 BT.println(";");

}
  //delay(5);
if(timer>300){
 sensors_event_t event;

 dht.temperature().getEvent(&event);
 temp = event.temperature;

 dht.humidity().getEvent(&event);
 hum = event.relative_humidity;
 BT.print("B");
 BT.print(":");
 BT.print(temp); //send distance to MIT App
 BT.print(";");
 BT.print(hum); //send distance to MIT App
 BT.println(";");
 timer = 0;
}
//delay(1);
}
}
```

Fig. 10. Arduino Code 4

the app connect the Bluetooth module (HC-05), so that sensor data from the DHT-22 temperature sensor and HC-SR04 Ultrasonic sensor is received. A button for the same has been provided

Block-1( refer fig: 13 )represents that app switches its Bluetooth connection on-and lets the user connect to HC-5. If the Bluetooth of device is not enabled, it displays an error message.

Block-2 ( refer fig:14 ) represents that a clause has been added where, if the Bluetooth isn't connected, the text is label-1 which is a welcome message is changed to "Disconnected" and the color is changed to blue. A text-dictation patch has been added to send a dictation notification.

A disconnect button has also been added so that the user can disconnect Bluetooth module after the task is finished.

Block-3 (refer fig: 15) represents the navigation block setup to send the user-input feed to Firebase-dataset in order to navigate the app. The navigation buttons forward-backward-left-right are assigned an integer respectively to send to the firebase. For example, if the user wants to navigate in forward direction, he/she presses the forward button which triggers the block and sends '1' to firebase which

```python
#!/usr/bin/env python3
import serial
from firebase import firebase
import time

if __name__ == '__main__':
    ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
    ser.flush()
    firebase = firebase.FirebaseApplication('https://wildfire-robot.firebaseio.com/', None)

    while True:
       # ser.write(b"Hello from Raspberry Pi!\n")
        line = ser.readline().decode('utf-8').rstrip()
        direction = firebase.get('/wildfire-robot', 'data')
        print(line)
        print("\n")
        ser.write(str(direction))
```

Fig. 11. Python Code

is relayed to arduino which advises the motors and motion is achieved, all in real-time. '0' is set as a base value when nothing is pressed and the bot stops when it receives zero.
Block-4 (refer fig:16) represents the Manipulator actuation via firebase link. The second block in the image represents the connection to retrieve the visual feed. The user enters the I.P address of his/her network and as soon as connect button is pressed, the app via local network connects to the rpi-web-interface and displays the visual feed on the app which helps the user to navigate through.

The process begins with users clicking the 'Connect Bluetooth' button on the screen and entering IP address of raspberry pi in the textbox provided to get started. This automatically connect phone to RPi over internet and gives them options to connect with the nearby active Bluetooth clients nearby, HC-05 module in our case.

After successfully connecting with the module and in turn with Mega microcontroller, application has buttons enabled to move/steer the robot in four directions .i.e. forward, reverse, twist right, twist left. There is also provision for users to keep checking the humidity level and temperature around the robot base at any instant of time, as it is always visible on the application in the bottom region. Moreover, there is also data available for how far the nearest obstacle is, in front of the robot. App UI is shown in figure 12

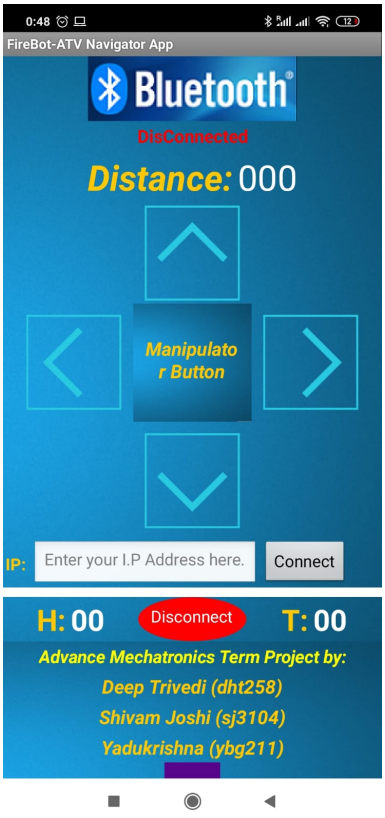



Fig. 12. User Interface of the Mobile Application

## Challenges and Results

Major challenges we faced during this project were majorly in communication setup from firebase to raspberry Pi. Using firebase with Pi has a learing

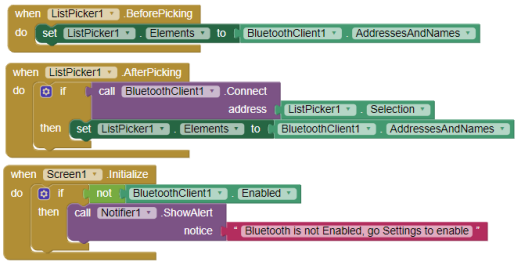



Fig. 13. Code Block1: MIT App Inventor

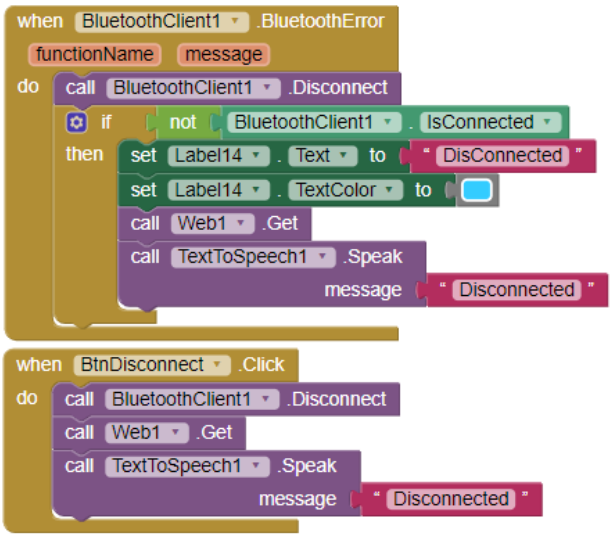



Fig. 14. Code Block2: MIT App Inventor

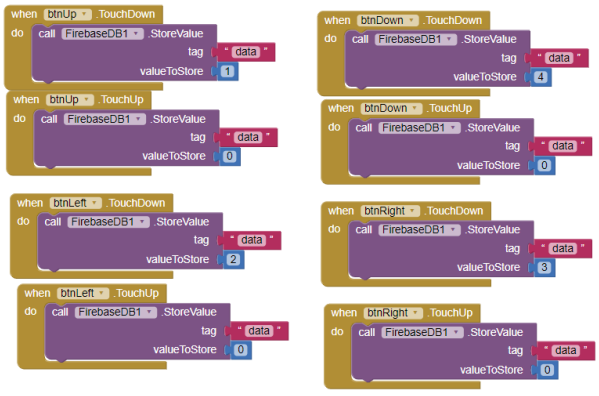
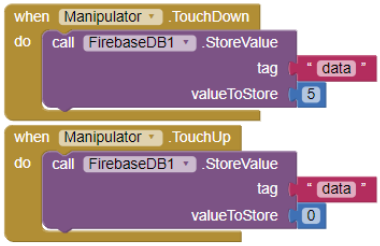



Fig. 15. Code Block3: MIT App Inventor



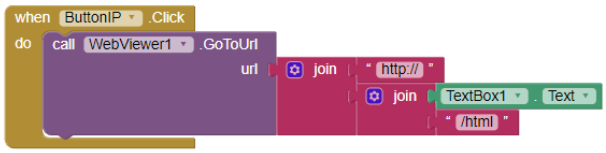




Fig. 16. Code Block4: MIT App Inventor

curve. Moreover, there we challenges related to Rpi web interface as well because it initiates a process in OS which does not let any other process thread like from $raspistill, or from openCV to run.
Another area which was somewhat challenging was the process of syncing the code for different sensors, drivers and android application together. The project served to be a good learning ground for enhancing our skills and knowledge.

## Conclusion and Future Scope

Future plans for the project follows usage of Computer vision methods to survey the live feed from the robot. We tried using tensorflow framework for the same and wish to build usable object detection application out of it.

## Link to the Video

Video Link : https://drive.google.com/drive/u/1/folders/1cd94DI9ZDl0HqJ6cy7ts

## Bill of Material

Following is the tabular representation of the components we have used in our project.

| Component | Rate | Quantity | Cost (in Dollars) |
|---|---|---|---|
| Arduino Mega | 35.00 | 1 | 35.00 |
| Raspberri Pi | 40.00 | 1 | 40.00 |
| RPi Camera | 25.00 | 1 | 25.00 |
| HC-05 | 6.99 | 1 | 6.99 |
| DHT22 | 8.11 | 1 | 8.11 |
| L298N | 2.91 | 3 | 8.73 |
| Li-ion Battery | 11.99 | 1 | 11.99 |
| Alkaline Battery | 1.99 | 1 | 1.99 |
| HC-SR04 | 1.99 | 1 | 1.99 |
| DC Motors | 14.00 | 6 | 84.00 |
| Thumper Wheels | 7.50 | 6 | 45.00 |
| U-PVC pipes | 15.00 | - | 15.00 |
| Others | 20.00 | - | 20.00 |
| Total | | | 303.00 |

## REFERENCES

[1] Robin R. Murphy, Rachel Brown, Reginald Grant and Clint T. Arnett. *Preliminary Domain Theory for Robot-Assisted Wildland Firefighting*. IEEE Denver, Colorado, USA, 2010 https://ieeexplore.ieee.org/document/5424143