TAMPERE UNIVERSITY
OF APPLIED SCIENCES

REAL TIME EMEBEDDED SYSTEM BASED INTELLIGENT LAND ROVER

Kushagar Tyagi

Bachelor's Thesis
April 2018
ICT Engineering

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
ICT Engineering
Exchange Student

KUSHAGAR TYAGI:
Real Time Embedded System Based Intelligent Land Rover

Bachelor's thesis 22 pages, appendices 8 pages
April 2018

This thesis has intended to provide enough information in order to understand land rovers, Internet of Things to be able to draw some positive conclusions.

Bots or droids are mostly used as substitute of human labors to perform tasks in remote areas which are difficult to access. Droids can be brought into use in various number of ways which can significantly reduce the manual work specifically in remote areas where human accessibility is a big pitfall. The main applications where the bots or droids have exhibited their excellence include surveillance, tracking targets for military purposes and also for disaster management endeavors like searching and rescuing victims. The bot that has been designed and presented in this paper can be used for defense, domestic as well as commercial purposes. This bot captures the live motion using the camera and also records inputs from various sensors embedded in it. It then sends the camera feed using Bluetooth HC-05 module and sensor inputs to the admin via email using wireless communication technique, on any device which has the privilege credentials to access the input, on the same network.

Keywords: Internet of things (IoT); sensors; raspberry pi; wireless communication; arduino; node-red.

INDEX

ABBREVIATIONS AND TERMS

IoT	Internet of Things is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data.
WSN	Wireless Sensor Networks refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location.
AUV	Autonomous Underwater Vehicle
ARM	Advanced RISC Machine is a family of reduced instruction set computing (RISC) architectures for computer processors, configured for various environ-ments.
AVR	AVR is a family of microcontrollers developed by Atmel
Raspberry Pi	The Raspberry Pi is a tiny and affordable computer that you can use to learn programming through fun, practical projects.
Arduino UNO	The UNO is the most used and documented board of the whole Arduino family, Arduino Uno is a microcontroller board based on the ATmega328P
GUI	Graphic User Interface
LEDs	Light-Emitting Diodes

1. INTRODUCTION

Robotics was grown leaps and bounds over the past century. With the development in technology, bots and their peripheral equipment's have become more sophisticated and compact. Due to their reliability, they have penetrated all walks of life like entertainment, military etc. They are being extensively used in environment monitoring and prediction industry. The main purpose of this industry is to measure several parameters like temperature, humidity, pressure etc. by using sensors. The advanced environment monitoring system uses wireless sensors and provides remote access to the measured data. The bot designed and presented in the paper incorporates an environment monitoring system[1]. This bot provides data from various sensors to update the status of various physical processes of that area which includes temperature, humidity, and smoke concentration. The main objective of the designed bot is to update the user about any real time change in the environment of a particular region, which can be mapped and averaged by controlling the bot remotely. This bot is very effective in hard terrains like dense forests, nuclear radiation affected areas etc. The bot can also find its scope in applications like terrain analysis, preengagement reconnaissance, finding whether the place is fit for human access, and in finding the status quo of specific area etc[2]. The proposed design uses the concept of internet of things (IoT). IoT mainly deals with machine to machine communication and person to computer communication. The key technologies used include WSN (wireless sensor networks), nanotechnology and miniaturization[3]. With the proposed design, human beings can very easily judge whether or not that area is fit for human access, can average out the conditions of that place for construction, establishment purposes and also for exploration purposes.

2. RELATED WORK

Various bots have been developed throughout the world, for serving highly specific and general surveillance, monitoring and mapping purposes. These include bots designed exclusively for underwater condition monitoring of parameters like water purity level, radio tagged fishes, to bots designed for monitoring pollution levels in specific areas of any major establishment. A few have been hereby mentioned along with references:-

2.1 STARTBUG:

This bot, as mentioned in the paper, is a hybrid AUV (Autonomous Underwater vehicle) concept developed by the CSIRO robotic reef monitoring team, with primary motivation to deploy a design keeping in mind the typical highly unstructured terrain of the Great Barrier Reef any AUV must navigate. Aim of its research was to develop an AUV for less than AUS$10,000 which requires less than one person/operator per AUV. Key performance specifications include a mass of 26 kg, length 1.2m (folding to 0.8m for transport), max. Forward thrust 20N, max speed 1.5 m/s, speed for max. Range 0.7m/s, battery capacity 6.4Ah (4X12V sealed lead acid batteries)[4].

2.2 ROBOVOLC:

Robovolc project includes a simple mobile platform (wheel) based robot which was primarily developed to increase knowledge of volcano processes to enhance percentage of eruption forecasting and to reduce the risk of volcanologists operating in proximity of dangerous volcanic openings and potential regions which can be affected by volcanic eruptions . The robot is used for autonomous exploration of highly irregular and rough unstructured environments. It is designed, making it resistant to high temperatures and contaminated atmosphere. It comprises of sample collecting hardware for sampling of molten gas, molten lava or rocks after they deem fit (temperature wise) for the robotic sensors and hardware to carry out assessment. It also measures gas temperatures of the surroundings, along with measurement of physical parameters of volcanic jets, lava stream. To conclude, this robot has been highly successful in determining volcanic vents morphology and topography[5].

3. DESIGN CONCEPT

The bot was designed using microcontrollers and sensors. Node.js based Node-Red and some other software are used for configuration of the user interface, along with the custom user interface of the camera, which is already provided.

3.1 Hardware Specification and Design

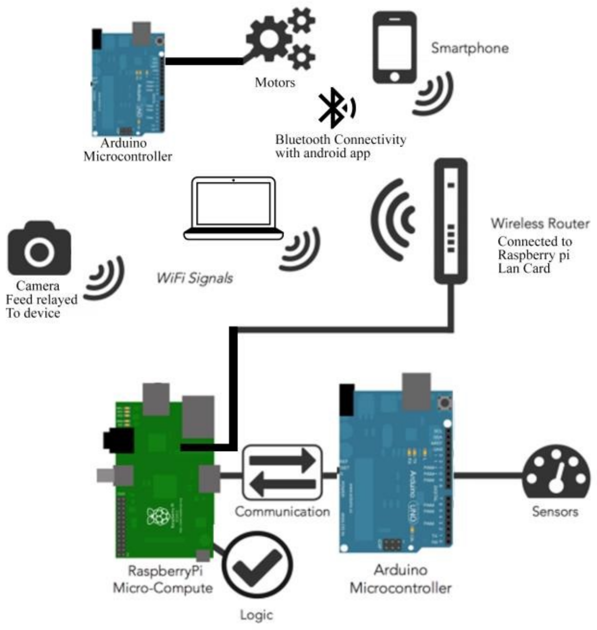

Motors
Smartphone
Arduino Microcontroller
Bluetooth Connectivity with android app
Wireless Router
Connected to Raspberry pi Lan Card
Camera Feed relayed To device
WiFi Signals
Communication
Sensors
RaspberryPi Micro-Compute
Arduino Microcontroller
Logic

Fig.1 Hardware design of the robot

In Fig. 1, an independent arduino board is used for making the “vehicle” element of this bot. Equipped with a HC-05 module and powered independently with a power supply of 5A-2.1A, it is controlled by an android application on a smartphone serving as a remote for the bot. All sensors are connected to another arduino board’s pins which is itself connected to the raspberry pi via USB, with the sensor inputs being relayed to raspberry pi. The raspberry pi (using its 802.11n wireless LAN card) and the host device(s) are connected to a network provided by the wireless router.

For this bot the DC motors used provided enough speed with all sensors and boards loaded on the chassis, along with their light weight of 54.4 grams. The IC used to control the motor movement from arduino was L293D, which is most suitable to drive small DC-Geared motors, which are required for the design of the robot[7]. For controlling the bot, Arduino Uno based on ATmega328 microcontroller has been preferred. The ATmega328 provides 6 analog inputs along with many more digital inputs and outputs, which is plenty of I/O for our sensor interfacing requirements[8]. For the design of robot, RaspberryPi has been used for its ease to integrate DHT11 temperature sensor in our node-red user interface and also to receive sensor inputs via packetized serial transfer approach and process them for proper output. Another benefit of using Raspberry pi is that it allows more options for communication than just packetized serial. The RaspberryPi 3 provides a 4× ARM Cortex-A53, 1.2GHz CPU, 10/100 Ethernet, 2.4GHz 802.11n wireless LAN[9], a Broadcom Video Core IV GPU[9]. All the processing part is done by raspberry pi and the data of the user interface is transferred via the 802.11n wireless LAN card of raspberry pi to the router and then to the device connected to the same network to which it is connected.

The cell phone camera used here is Intex Aqua Ace 2, running Android version 5.1, relaying feed independently to any device connected to it on the network. Power is supplied using an Intex IT-PB10K Polymer 10000mAh Power Bank, making both arduino and raspberry pi functional on the go. All the sensors except DHT11 temperature sensor are connected to the pins of the arduino connected to raspberry pi. The sensor used for temperature measurement is DHT11 basic temperature-humidity sensor as it is good for 0-50°C temperature readings ±2°C accuracy with 3 to 5V power and I/O[9]. We have used a GY-61 DXL335 3-Axis Accelerometer with extremely low noise and power consumption, suitable for the design of robot[11]. The photoresistor used has an operation range of environmental temperature 30-70 degree Celsius, fit for our usage[12].

3.2 Software Specification and Design

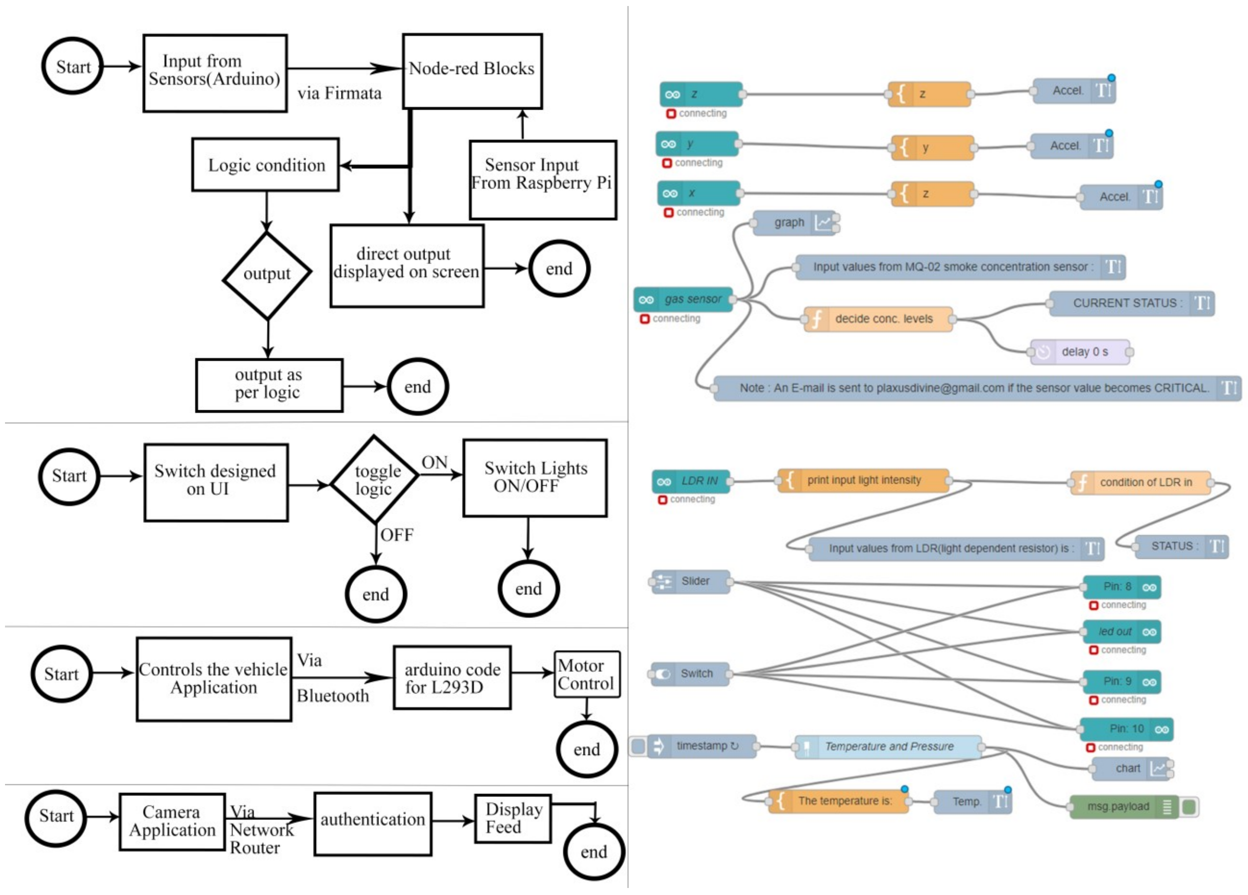


Fig.2 Software design of the robot (Left side- Different Levels, Right side- node-red flow layout)

In Fig. 2, for the design of the robot software development is done at various levels mainly for interfacing of arduino and raspberry pi for sensor data transfer, issuing commands from an Android device, sending camera input and GUI for user interaction. The method of data relaying between RaspberryPi and the Arduino for sensor data transfer is a library called Firmata. Firmata is an application that is installed on Arduino permitting great control of the board via serial communication. The code file in File->Examples->Firmata->StandardFirmata[13] is uploaded to arduino and thereby the sensor inputs become available to the GUI development framework – node-red[14]. We used Node.js based framework- node-red for developing the UI as it is based on Node.js and thus we can take maximum advantage of its event-driven model. We developed various flows (interconnected blocks) on node-red which included relaying input to test conditions and presenting the output. For getting camera feed, we used an android application which directly relayed data to any device connected to the network with a specified IP address, providing the feature of authentication, if specified, every time the user tried to access the IP address. An android application is developed with simple controls like forward, backward etc. which controlled the motors connected to arduino via serial communication, through Bluetooth.

4. MECHANISM

The robot is controlled with an android application relaying its input over a Bluetooth module, controlling the rover's movement in linear directions. The rover navigated by being remotely (wirelessly) controlled over Bluetooth in forward, left and right direction in real time. All the sensors mounted on another Arduino board (which is connected to Pi) sent their readings continuously in raw form to raspberry pi over the firmata protocol, programs including firmata library is uploaded on Arduino UNO board for making microcontroller-software (here Node-Red) interaction possible. In this project, Standard Firmata sketch (Arduino program) is deployed over the UNO board, which is located in the Arduino IDE in File → Examples → Firmata[13].

The camera used for real time feed is of an android cell phone (which is connected to the same network). The video feed, upon web based authentication, is projected on the desktop or mobile device using various protocols. Raspberry pi deployed the sensor data on the specified blocks of the created GUI, ran on a port number, which can be showcased on any desktop and mobile device on that network.

Whenever the sensor input exceeded a specified threshold, a warning e-mail is issued to the authorities. If there existed a need to bring up the light around the robot for better camera feed, LEDs have been mounted on the bot for the same.

4.1Robot Navigation

Since navigation is a very crucial part of this rover, we have used 4 simple DC motors, which are controlled by an android application over a Bluetooth channel using HC-05 Bluetooth module. For proper control, a L293D motor driver IC is connected to the motors. The android application sends input commands over the

Bluetooth channel, which are received by the Bluetooth module and then directed to the Arduino board. It compares the commands with the code uploaded on it, then sends the corresponding output commands to the L293D motor driver, which makes the motor movement possible.

4.2 Controller unit

Controller unit is the main functioning unit of the robot. For this design we have used Arduino UNO to take in various sensor inputs namely – an accelerometer, smoke sensor and LDR along with a Raspberry Pi 3 to take in inputs from a DHT11 temperature sensor, also using same Raspberry Pi 3 to create an interactive GUI and display the input data on it. It is also used to deploy the GUI service on a port for remote access on a system (which may include a cell phone, laptop, desktop etc.) on that network.

4.3 Cell phone camera

A cell phone camera (here of an android cell phone) has been used for live video transmission over the internet. The video input has been sent using the transmission and security features of the TCP/IP protocol. This camera has been mounted on the robot and the camera demands proper user authentication before displaying the feed.

4.4Power supply

For the navigation purpose, the motors are supplied with 5V-2A DC power supply. To make Raspberry Pi 3, arduino and all the sensors to properly work, a constant supply of 5V-1A is given using a power bank. The rechargeable power bank has 2 output power slots: 5V-1A and 5V-2A.

4.5 Communication system

RaspberryPi and the cell phone camera both are connected to the same internet enabled network. The RaspberryPi transfers the sensor input from Arduino to an interactive GUI created on a node.js based platform (Node-Red), which runs on the static IP of RaspberryPi on port number 8080, all of which is easily accessible on devices in that network. The mentioned protocol is used for communicating with microcontrollers from software on a computer. The firmata library is brought into use for communicating with software on the host computer. This allows us to write custom firmware without having to create our own protocol and objects for the programming environment that we were using[11].

5. RESULTS

The surveillance robot is assembled successfully and has been successfully tested.

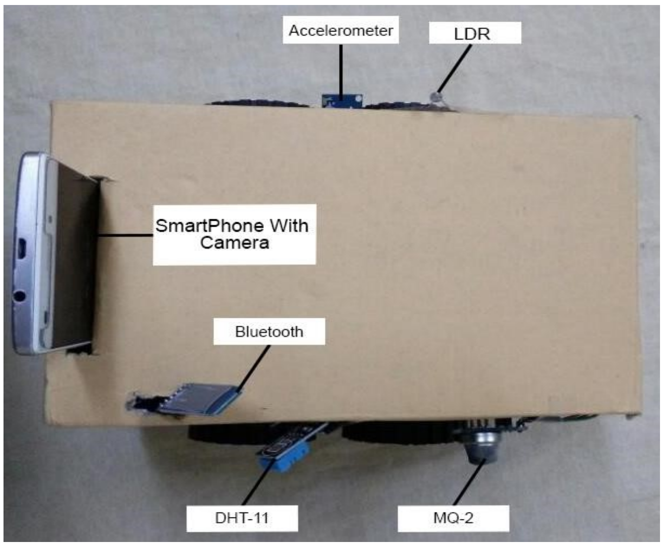
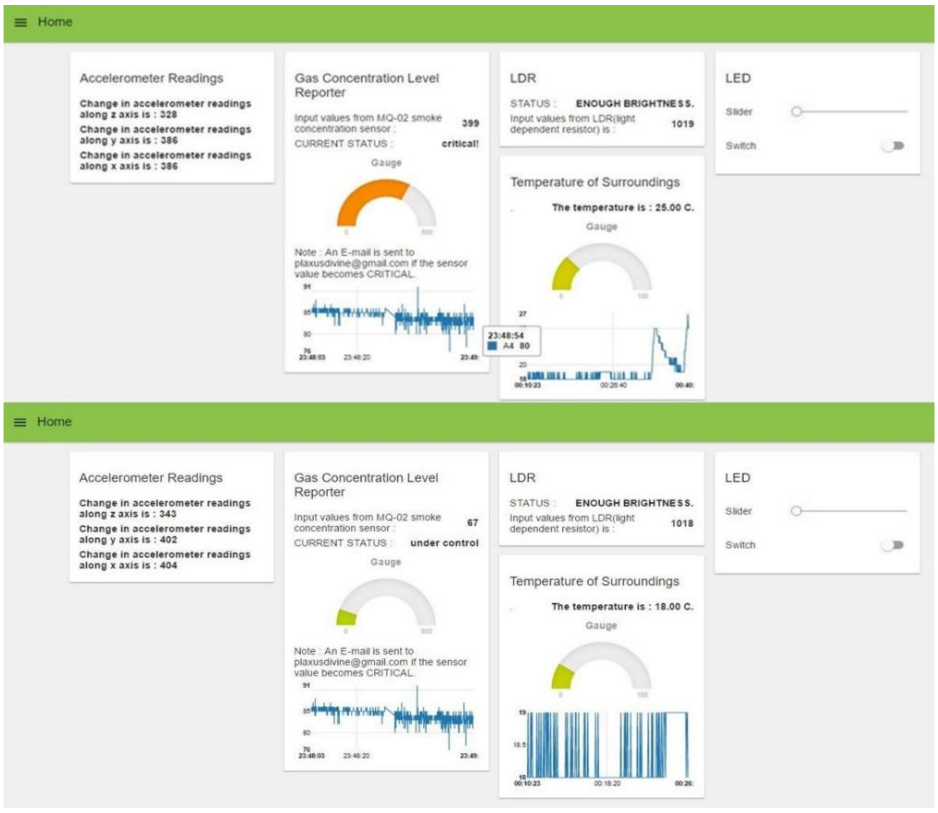


Fig.3 shows the hardware setup of the designed robot assembly

In Fig.3, it shows the hardware setup and how the sensors connected to the circuit, including LDR and GY-61 accelerometer. GY-61 is kept parallel to the ground surface so as to obtain correct readings. The other sensors include DHT11 temperature and humidity, MQ-02 smoke, HC-05 Bluetooth module and an android based smartphone. In this robot the Arduino UNO is used as the navigation controller and Raspberry Pi for GUI creation and network purposes.



Home

Accelerometer Readings

Change in accelerometer readings along z axis is : 328
Change in accelerometer readings along y axis is : 386
Change in accelerometer readings along x axis is : 386

Gas Concentration Level Reporter

Input values from MQ-02 smoke concentration sensor : 399
CURRENT STATUS : critical!

Note : An E-mail is sent to plaxusdivine@gmail.com if the sensor value becomes CRITICAL.

LDR

STATUS : ENOUGH BRIGHTNESS.
Input values from LDR(light dependent resistor) is : 1019

Temperature of Surroundings

The temperature is : 25.00 C.

LED

Slider

Switch

Home

Accelerometer Readings

Change in accelerometer readings along z axis is : 343
Change in accelerometer readings along y axis is : 402
Change in accelerometer readings along x axis is : 404

Gas Concentration Level Reporter

Input values from MQ-02 smoke concentration sensor : 67
CURRENT STATUS : under control

Note : An E-mail is sent to plaxusdivine@gmail.com if the sensor value becomes CRITICAL.

LDR

STATUS : ENOUGH BRIGHTNESS.
Input values from LDR(light dependent resistor) is : 1018

Temperature of Surroundings

The temperature is : 18.00 C.

LED

Slider

Switch

Fig.4 Sensor inputs on the output screen, inputs updated at a regular interval along with change in graphs

In Fig.4, it shows the inputs of accelerometer, MQ-02 smoke sensor, LDR, DHT11 Temperature sensor on the GUI created over Node-Red. A switch with a slider is provided to control the LEDs embedded in the robot. The graphs indicate change in sensor readings along x axis and time along y axis. The inputs from MQ-02 sensor are in form of voltage signals, unit being millivolts. Using the algorithm for value conversion described above, the input from sensor is converted to variable resistance, which can then be used to calculate concentration in ppm (parts per million).

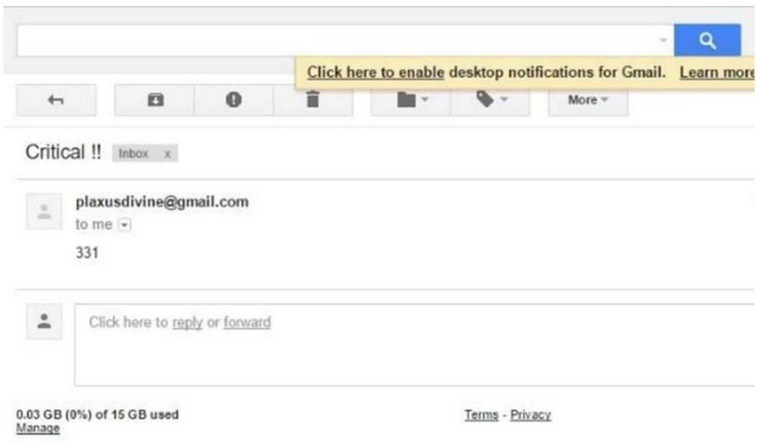

Click here to enable desktop notifications for Gmail. Learn more
More
Critical !!
Inbox x
plaxusdivine@gmail.com
to me
331
Click here to reply or forward
0.03 GB (0%) of 15 GB used
Manage
Terms - Privacy

Fig.5 E-mails sent with sensor values and input status (whenever sensor input crossed a certain threshold)

In Fig.5, it shows that an e-mail is sent with sensor values and input status. All the operating blocks can be configured in the GUI to send e-mails to any e-mail ID with pre-decided delay time. The e-mails contain direct sensor readings with a subject. These e-mails can then be used to take further action. The robot navigates freely in linear directions as per the user inputs and sends in all the inputs at a decided interval of time.

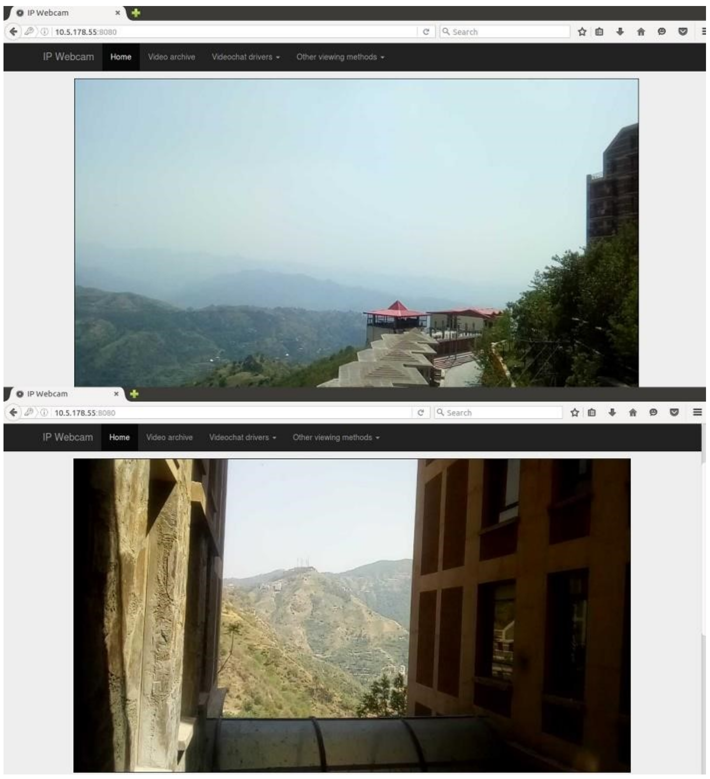


Fig.6 Camera feed images taken when tested

In Fig.6, images are shown when the robot was tested for proper functioning at 2 places providing different environment for the robot to operate. The first was on the hills where all the readings shown on the laptop screen were not close to the extremes. The second scenario was inside a small, closed room where wood was lit up. The robot showed heavy smoke concentration and high rise in temperature, with camera feed being heavily blurred. In both cases, the results obtained with the designed robot have given expected inputs with slight variations.

6. CONCLUSION

The surveillance robot has been designed with AVR and ARM microcontrollers using embedded platform. It monitored the location and other physical conditions of the environment of itself and updated the user in real time about all the minor changes in the condition, thus providing top quality surveillance, high precision and low machine errors. A cell phone camera has been used which monitored the place and sent the information to the user. The accelerometer has recorded the changes in acceleration at various points in the trail of the rover, thus measuring the level of craters and steep surfaces in its way. DHT11 has provided accurate reading of temperature and humidity around the rover. MQ-02 has provided smoke or a specific gas level concentration around the rover. The L293D motor driver used has provided the movement of robot with greater control compared to the conventional method. The future scope of the designed robot has many openings which include amendments in the communication techniques used, reliance on long range networks rather than short range networks, integration of camera into a singular UI with other inputs etc. These can be enhanced for various future applications in monitoring and controlling etc.

References

[1] Satvir Singh and Dr. Rajeshwar Singh, *Design of Environment Monitoring and Control System*, International Journal of Engineering And Computer Science ISSN: 2319-7242 Volume 4 Issue 5 May 2015, Page No. 1198011984.

[2] V.Divya, S.Dharanya, S. Shaheen and A.Umamakeswari, *Amphebious Survelliance Robot with Smart Sensor Nodes*, Indian Journal of Science and Technology.

[3] Sean Dieter Tebje Kelly, Nagender Kumar Suryadevara, and Subhas Chandra Mukhopadhyay, Fellow, IEEE, *Towards the Implementation of IoT for Environmental Condition Monitoring in Homes*, IEEE SENSORS JOURNAL, VOL. 13, NO. 10, OCTOBER 2013

[4] Dunbabin, M.; Roberts, J.; Usher, K.; Corke, P. A new robot for environmental monitoring on the Great Barrier Reef. In: Barnes, N.; Austin, D. eds, editor/s. Australasian Conference on Robotics and Automation (ACRA 2004); 2004; Australian National University. 2004.

[5] Hazelwood, Z. C., & Sbenaty, S. M. (2014, June), *Designing, Building, and Testing an Autonomous Search and Rescue Robot — An Undergraduate Applied Research Experience* Paper presented at 2014 ASEE Annual Conference & Exposition, Indianapolis, Indiana. https://peer.asee.org/20271

[6] “*Robovolc Leaflet.*”Robovolc, "A robot for Volcano Exploration".Web http://www.robovolc.dees.u nict.it/download/files/robovolc_leaflet.pdf

[7] “L293D Motor Driver IC - Nex Robotics”.Nex Robotics. Web <.http://www.nex-robotics.com/products/motordrivers/l293d-motor-driver-ic.html>

[8] “Arduino Uno”. Arduino. Web. http://arduino.cc/en/Main/ArduinoBoardUno

[9] “DHT11 basic temperature-humidity sensor”.Adafruit. Web. <https://www.adafruit.com/product/386>

[10] “Raspberry Pi 3 is out now! Specs, benchmarks &amp; more - The MagPi MagazineThe MagPi Magazine”.Raspberry Pi. Web. <https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/>

[11] “GY-61 DXL335 3-Axis Accelerometer Module”. Robotpark. Web. http://www.robotpark.com/GY-61DXL335-3-Axis-Accelerometer-Module

[12] “GL55 Series Photoresistor”.CdS Photoresistor Manual. Akizukidenshi. Web. <https://www.akizukidenshi.com/download/ds/senba/GL55%20Series%20Photoresistor.pdf>

[13] ”*Arduino-Firmata*”.Arduino.Web. <https://www.arduino.cc/en/reference/firmata>

[14] ”Node-RED : Interacting with Arduino”. Node-RED. Web. <https://nodered.org/docs/hardware/arduino>

APPENDIX

Requirements:

- dht11 Temprature and Humidity Sensor
- Bluetooth
- MQ2 Gas sensor
- Accelerometer
- Smart Phone with camera
- Raspberry Pi
- Arduino Uno

Usage

- Do the connections as shown in the figures.
- SSH into your Client to access Raspberry Pi GUI.
- Burn all your codes into Raspbeery Pi and Arduino
- Start Node-Red and do the following connections on it.
- Get the GUI using Node-Red

CODES

1. **Arduino Code**

```
const int motorA1  = 5;

const int motorA2  = 6;

const int motorB1  = 10;

const int motorB2  = 9;



const int bluein = 2;

int perVolt;

float voltage = 0.0;

int level;

int i=0;

int j=0;
```

```
 int state;

 int vSpeed=200;


void setup() {

   pinMode(motorA1, OUTPUT);

   pinMode(motorA2, OUTPUT);

   pinMode(motorB1, OUTPUT);

   pinMode(motorB2, OUTPUT);


   pinMode(bluein, INPUT);

   Serial.begin(9600);

}


void loop() {

    if(digitalRead(bluein)==LOW) { state='S';

   }


   if(Serial.available() > 0){

     state = Serial.read();

   }

   if (state == '0'){

     vSpeed=0;}

   else if (state == '1'){

     vSpeed=100;}

   else if (state == '2'){
```

```
  vSpeed=180;}
else if (state == '3'){
  vSpeed=200;}
else if (state == '4'){
  vSpeed=255;}
if (state == 'F') {
      analogWrite(motorA1, vSpeed);
      analogWrite(motorA2, 0);
    analogWrite(motorB1, 0);
      analogWrite(motorB2, 0);
}
else if (state == 'G') {
      analogWrite(motorA1, vSpeed);
      analogWrite(motorA2, 0);
    analogWrite(motorB1, 200);
    analogWrite(motorB2, 0);
}
else if (state == 'I') {
      analogWrite(motorA1, vSpeed);
      analogWrite(motorA2, 0);
    analogWrite(motorB1, 0);
      analogWrite(motorB2, 200);
}
else if (state == 'H') {
      analogWrite(motorA1, 0);
```

```
          analogWrite(motorA2, vSpeed);
     analogWrite(motorB1, 200);
       analogWrite(motorB2, 0);
  }

  else if (state == 'J') {
       analogWrite(motorA1, 0);
 analogWrite(motorA2, vSpeed);
     analogWrite(motorB1, 0);
analogWrite(motorB2, 200);
  }
  else if (state == 'L') {
       analogWrite(motorA1, 0);
       analogWrite(motorA2, 0);
     analogWrite(motorB1, 200);
       analogWrite(motorB2, 0);
  }

  else if (state == 'R') {
       analogWrite(motorA1, 0);
       analogWrite(motorA2, 0);
     analogWrite(motorB1, 0);
       analogWrite(motorB2, 200);
  }
    state='n';
  }
  else if (state == 'S'){
      analogWrite(motorA1, 0);
        analogWrite(motorA2, 0);
      analogWrite(motorB1, 0);
        analogWrite(motorB2, 0);
  }



}
```

2. Firmata Library

The Firmata library implements the Firmata protocol for communicating with software on the host computer. This allows you to write custom firmware without having to create your own protocol and objects for the programming environment that you are using.

2.1 Methods

```
begin();
//start the library
begin(long); //start the library and override the default baud rate
begin(Stream &s);
//start the library using a [Stream](http://www.arduino.cc/en/Reference/Stream)
other than Serial (eg Serial1 or EthernetClient)
printVersion();
```

//send the protocol version to the host computer

blinkVersion():

//blink the protocol version on the build in LED (typically pin 13)

printFirmwareVersion();

//send the firmware name and version to the host computer

setFirmwareVersion(byte major, byte minor);

//set the firmware name and version, using the sketch's filename, minus the '.ino'

setFirmwareNameAndVersion(const char *name, byte major, byte minor);

//set both the name and version of the firmware

2.2 Sending Messages

sendAnalog(byte pin, int value); //send an analog message

sendDigitalPort(byte portNumber, int portData); //send an 8-bit port in a single digital message

sendString(const char* string); //send a string to the host computer

sendString(byte command, byte bytec, byte *bytev); //send a string to the host computer using a custom command type

sendSysex(byte command, byte bytec, byte* bytev); //send a command with an arbitrary array of bytes

write(byte c); //write a byte to the Stream

2.3 Receiving Messages

available(); //check to see if there are any incoming messages in the buffer

processInput(); //process incoming messages from the buffer, sending the data to any registered callback functions

attach(byte command, callbackFunction myFunction); //attach a function to an

incoming message type
detach(byte command); //detach a function from an incoming message type

2.4 Utility methods

sendValueAsTwo7bitBytes(int value); //writes value as 2 bytes
startSysex(void); //starts a sysex message
endSysex(void); //ends a sysex message

2.5 Callback Functions

In order to attach your function to a message type, your function must match the standard callback function. There are currently three types of callback functions in Firmata: generic, string, and sysex.

GENERIC

void callbackFunction(byte pin, int value);

SYSTEM_RESET

void systemResetCallbackFunction(void);

STRINGS

void stringCallbackFunction(char *myString);

SYSEX

void sysexCallbackFunction(byte command, byte byteCount, byte *arrayPointer);

2.6 Message Types

There are various message types that you can attach callback functions to.

ANALOG_MESSAGE //the analog value for a single pin

DIGITAL_MESSAGE //8-bits of digital pin data (one port)

REPORT_ANALOG //enable/disable the reporting of an analog pin

REPORT_DIGITAL //enable/disable the reporting of a digital port

SET_PIN_MODE //change the pin mode between INPUT/OUTPUT/PWM/etc.

STRING_DATA //C-style strings, uses stringCallbackFunction for the function type

SYSEX_START //generic, arbitrary length messages (via MIDI SysEx protocol), uses sysexCallbackFunction for the function type

SYSTEM_RESET //message to reset firmware to its default state, uses systemResetCallbackFunction for the function type