Prototyping of Smart Mine Detection Rover with Cloud Based Image Processing using MATLAB

Dr. Syeda Gauhar Fatima[1], Saquib Mohiuddin[2], Mohammed Samad[3], Khaja Masi Uddin[4]
[1]Professor, Department of Electronics and Communication Engineering, DCET
[2]Student, Department of Electronics and Communication Engineering, DCET
[3]Student, Department of Electronics and Communication Engineering, DCET
[4]Student, Department of Electronics and Communication Engineering, DCET

Abstract - *Robot application for human safety & security is an important topic in the research domain. From military to medical, many applications are being developed to ameliorate human life. Networked robotics suffer from inherent physical constraints such as, low speed on-board instruction execution, small size of memory, network latency, variable quality of service, downtime, and lack of intelligence.*

The limitations have motivated the researchers to think of new form of efficient robotic systems i.e., "Cloud Robotics". Cloud robotics may be described as a system that relies on the "Cloud Computing", Cloud computing and the IoT are two non-robotic enablers in creating distributed robotic systems. The Proposed System can also be referred as "Internet of Robotic Things".

***Key Words*: BigData, Raspberry Pi, Arduino, ESP8266, MATLAB, Image Processing, ZigBee, Python**

1.INTRODUCTION

Wilderness search and rescue entails performing a wide-range of work in complex environments and large regions. Given the concerns inherent in large regions due to limited rescue distribution, unmanned vehicle-based frameworks are a promising platform for providing imaging. In recent years, technological advances in areas such as micro-technology, sensors and navigation have influenced the various applications of UGVs. In this study, an all-in-one camera-based target detection and positioning system is developed and integrated into a fully autonomous UGV. The system presented in is capable of on-board, real-time target identification, post-target identification and location and image processing for further applications. This paper also contains Fuel Consumption Prediction using Machine Learning.

Application: Military, Search and Rescue

1.1 Existing System

- In this system, there is device to monitor the temperature and gas level.
- Alert is not. Only caregivers monitoring.

Drawbacks

- There is no wireless system for data transfer.
- There is no alert system.
- High cost.
- Monitoring not available

1.2 Proposed System

- The system is divided into 3 functional blocks:

  [1] Raspberry Pi Functional Block
  [2] Arduino Functional Block
  [3] Control Section

- While monitoring, if there is happening of any abnormal condition, the system will send the message to the appropriate persons.
- Camera is mounted on the Robo, it will capture the image of the intruder (if any) and allow the authenticate persons.
- Using this system, we can easily control Robo using Zigbee (wireless communication)
- Use of Machine Learning for Fuel Consumption Prediction under various environments such as Terrains.

Advantages

- Fast response.
- Auto alert system.

2. WORKING PRINCIPLE

An Ambitious Mine detecting robot along with Imaging and analyzing capabilities is a revolutionary military advancement and a lifesaving invention that can benefit

mankind. The robot is guided by a software that facilitates scanning the infected area, while a beat frequency oscillator is used as environment detection sensor placed on servo motor in front of the vehicle. This paper proposes a design of an environment detecting sensor and collision avoiding sensor and implementing it in a robotic prototype. The embedded proposed system is based on Raspberry Pi technology, and is interfaced with MATLAB Image

processing algorithms to facilitate recognition and authentication.

This system is controlled using Raspberry Pi, when the robot is connected to Raspberry Pi and interfaced with Zigbee module, to control, the Raspberry electronic Board makes the scanning operation functional, and capture the image of the person while the robot starts receiving navigation control signal over Zigbee. It is also equipped with several onboard sensors such as humidity, gas and temperature sensor's so we can monitor the exact status of gas, temperature and humidity levels. If any unauthorized person, wants to trespass the restricted area, then image will be captured through camera and will be processed in MATLAB using Image recognition algorithms for further identifications and facilitating decision making at control.

3. RASPBERRYPI FUNCTIONAL BLOCK DIAGRAM

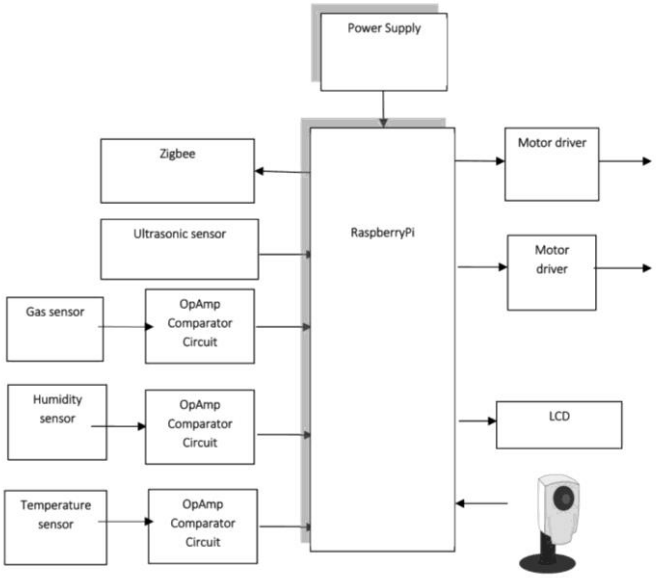


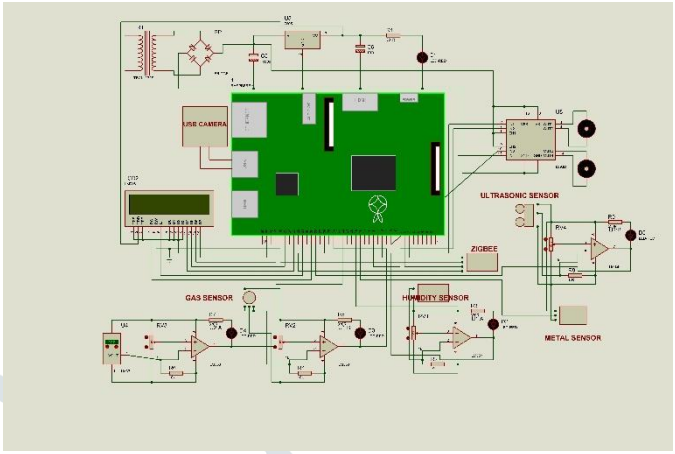
Fig: Block Diagram of Raspberry Pi Functional block

The robot is controlled via software at control room, while it surveys its designated area. It is powered using a battery and a power bank. Since the Motors require 12V, the voltage is increased using L293D paired with IN4007 Diode which is used to rectify voltage and keep the circuit stable. ZigBee is used for transmission of data.

The Raspberry Pi doesn't support analog input hence, we are using LM03501 DC Convertor paired with LM 358 Op-Amp as comparator that compares predefined threshold value with current value received from sensor and compares for any abnormalities such as Increase in heat signatures, or humidity or presence of gas.

Ultrasonic sensor is used to detect any intruder as well as avoid collision with any unwanted object

The Raspberry Pi is also paired with High-resolution 25MP camera which scans and capture images of intruders and transmit to control via cloud where images are processed at control section for further analysis.



The Components used are:

LM35 – Temperature, MQ2 – Gas Sensor, Ultrasonic sensor, L293D Motors, INV004 Diode, Humidity sensor, ZigBee Module, LM3501 DC Convertor

All the above given functionality is realized using python code as Raspberry Pi supports Python. The Code is as follows:

```
import lcd
import RPi.GPIO as GPIO
import sys
import time
import wifi
import datetime
import mcp3202
import webpage
import os
import camera
import ftp
import mail
import serial
import thread
#from flask import Flask
recpt = "naveen.ganeshwarapu@gmail.com"
ser = serial.Serial('/dev/ttyAMA0',9600)
ser.write("WELCOME\r\n")
lcd.lcd_init()
lcd.stringlcd(0x80,"WELCOME")
wifi.Connect_wifi()
m0 = 11
m1 = 13
m2 = 15
m3 = 19
led = 21
temp = 22
gas = 24
ultra = 26
hum = 29
metal = 18
buz = 16

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
```

```
GPIO.setup(m0,GPIO.OUT)
#GPIO.output(m0,True)
GPIO.setup(m1,GPIO.OUT)
#GPIO.output(m1,False)

GPIO.setup(m2,GPIO.OUT)
#GPIO.output(m2,True)
GPIO.setup(m3,GPIO.OUT)
GPIO.output(m3,False)

GPIO.setup(led,GPIO.OUT)
GPIO.output(led,False)
GPIO.setup(temp,GPIO.IN)

GPIO.setup(ultra,GPIO.IN)
#GPIO.output(r4,True)
GPIO.setup(gas,GPIO.IN)
#GPIO.output(g4,True)
GPIO.setup(hum,GPIO.IN)
#GPIO.output(r4,True)
#ser.write(msg+chr(26))
GPIO.setup(buz,GPIO.IN)
#GPIO.output(g4,True)

GPIO.setup(metal,GPIO.IN)
GPIO.output(m0,False)
GPIO.output(m1,False)
GPIO.output(m2,False)
GPIO.output(m3,False)
lcd.stringlcd(0x80,"WELCOME")
time.sleep(3)
#camera.capture("image.jpeg")
#ftp.sendftp("ULTRASONIC")
#time.sleep(10)
ser.timeout = 1
while(1):
	#lcd.stringlcd(0x80,"WAIT FOR CONTROL")
	try:
		rec=ser.read()
	except:
		rec = 0
		pass
	if(rec=='8'):
		lcd.stringlcd(0x80,"FRONT")
		ser.write("FRONT\r\n")
		GPIO.output(m0,True)
		GPIO.output(m1,False)
		GPIO.output(m2,True)
		GPIO.output(m3,False)
		time.sleep(3)
	if(rec=='2'):
		lcd.stringlcd(0x80,"BACK")
		ser.write("BACK\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,True)
		GPIO.output(m2,False)
		GPIO.output(m3,True)
		time.sleep(3)
	if(rec=='4'):
		lcd.stringlcd(0x80,"LEFT")
		ser.write("LEFT\r\n")
		GPIO.output(m0,True)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
		GPIO.output(m0,True)
		GPIO.output(m1,False)
		GPIO.output(m2,True)
		GPIO.output(m3,False)
		time.sleep(3)
	if(rec=='6'):
		lcd.stringlcd(0x80,"RIGHT")
		ser.write("RIGHT\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,True)
		GPIO.output(m3,False)
		time.sleep(3)
		GPIO.output(m0,True)
		GPIO.output(m1,False)
		GPIO.output(m2,True)
		GPIO.output(m3,False)
		time.sleep(3)
	if(rec=='5'):
		lcd.stringlcd(0x80,"STOP")
		ser.write("STOP\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
	if(GPIO.input(22) == False):
		lcd.stringlcd(0x80,"TEMP       IS
HIGH")
		ser.write("TEMPERATURE         IS
HIGH\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
	if(GPIO.input(24) == False):
		lcd.stringlcd(0x80,"GAS IS HIGH")
		ser.write("GAS IS HIGH\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
	if(GPIO.input(26) == True):
		lcd.stringlcd(0x80,"ULTRASONIC")
		ser.write("ULTRASONIC\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
		camera.capture("image.jpeg")
		ftp.sendftp("ULTRASONIC")
		time.sleep(10)
	if(GPIO.input(29) == False):
		lcd.stringlcd(0x80,"HUM IS HIGH")
		ser.write("HUMIDITY IS HIGH\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
	if(GPIO.input(16) == False):
		lcd.stringlcd(0x80,"METAL
DETECTED")
		ser.write("METAL DETECTED\r\n")
		GPIO.output(m0,False)
		GPIO.output(m1,False)
		GPIO.output(m2,False)
		GPIO.output(m3,False)
		time.sleep(3)
	if(GPIO.input(16) == True):
		time.sleep(3)
	print 'Loop'
```

4. ARDUINO FUNCTIONAL BLOCK

The Raspberry Pi doesn't support analog sensors, hence to aid analysis at command and control, we obtain values from sensor in real time via Arduino paired with ESP8266 Wi-Fi Module which streams data in real time and can be used to perform analysis.

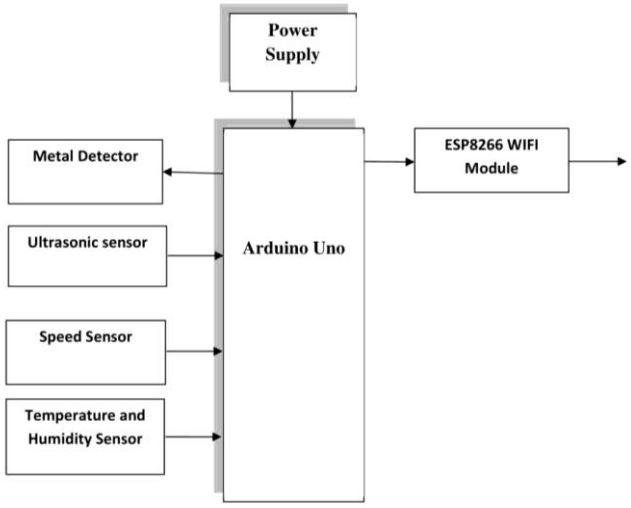


Fig: Arduino Functional block

The above requirements are realized by using following code in Arduino IDE:

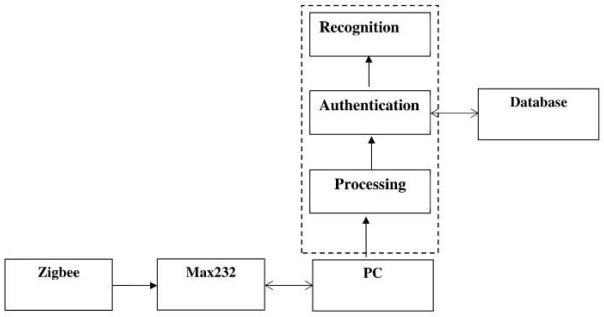
```
#include<SoftwareSerial.h>
SoftwareSerial client(2,3); //RX, TX

String webpage="";
int i=0,k=0;
String readString;
int x=0;

boolean No_IP=false;
String IP="";
char temp1='0';

String name="<p>Circuit Digest</p>";    //22
String          dat="<p>Data          Received
Successfully.....</p>";     //21

void check4IP(int t1)
{
  int t2=millis();
  while(t2+t1>millis())
  {
    while(client.available()>0)
    {
      if(client.find("WIFI GOT IP"))
      {
        No_IP=true;
      }}}}
void get_ip()
{
  IP="";
  char ch=0;
  while(1)
  {
    client.println("AT+CIFSR");
    while(client.available()>0)
    {
      if(client.find("STAIP,"))
      {
        delay(1000);
        Serial.print("IP Address:");
        while(client.available()>0)
        {
          ch=client.read();
          if(ch=='+')
          break;
          IP+=ch;
        }}
            if(ch=='+')
      break;
    }
    if(ch=='+')
    break;
    delay(1000);
  }
  Serial.print(IP);
  Serial.print("Port:");
  Serial.println(80);
}
void connect_wifi(String cmd, int t)
{
  int temp=0,i=0;
  while(1)
  {
    Serial.println(cmd);
    client.println(cmd);
    while(client.available())
    {
      if(client.find("OK"))
      i=8;
    }
    delay(t);
    if(i>5)
    break;
    i++;
  }
  if(i==8)
  Serial.println("OK");
  else
  Serial.println("Error");
}
void wifi_init()
{
   connect_wifi("AT",100);
      connect_wifi("AT+CWMODE=3",100);
      connect_wifi("AT+CWQAP",100);
      connect_wifi("AT+RST",5000);
      check4IP(5000);
      if(!No_IP)
      {
        Serial.println("Connecting Wifi....");
        connect_wifi("AT+CWJAP=\"1st
floor\",\"muda1884\"",7000);          //provide
your WiFi username and password here
     //           connect_wifi("AT+CWJAP=\"vpn
address\",\"wireless network\"",7000);
      }
      else
        {
        }
      Serial.println("Wifi Connected");
      get_ip();
      connect_wifi("AT+CIPMUX=1",100);
      connect_wifi("AT+CIPSERVER=1,80",100);
}
void sendwebdata(String webPage)
{
```

```
    int ii=0;
     while(1)
     {
      unsigned int l=webPage.length();
      Serial.print("AT+CIPSEND=0,");
      client.print("AT+CIPSEND=0,");
      Serial.println(l+2);
      client.println(l+2);
      delay(100);
      Serial.println(webPage);
      client.println(webPage);
      while(client.available())
      {
        //Serial.print(Serial.read());
        if(client.find("OK"))
        {
          ii=11;
          break;
       } }
            if(ii==11)
      break;
      delay(100);
     }}
void setup()
{
   Serial.begin(9600);
   client.begin(9600);
   wifi_init();
   Serial.println("System Ready..");
}
void loop()
{
  k=0;
  Serial.println("Please Refresh your Page");
  while(k<1000)
  {
    k++;
   while(client.available())
   {
    if(client.find("0,CONNECT"))
    {
      Serial.println("Start Printing");
      Send();
      Serial.println("Done Printing");
      delay(1000);
    }}
  delay(1);
 }}
void Send()
{
      webpage = "<h1>Welcome to Circuit
Digest</h1><body bgcolor=f0f0f0>";
      sendwebdata(webpage);
      webpage=name;
      webpage+=dat;
      sendwebdata(webpage);
      delay(1000);
     webpage = "<a
href="http://circuitdigest.com/";
     webpage+="\">Click Here for More
projects</a>";
     sendwebdata(webpage);
     client.println("AT+CIPCLOSE=0");
}
```

5. CONTROL SECTION

The Control sections is the command and control center of the rover, its where the rover is controlled, Image Processing is done, analysis is performed on Data extracted from Images and Values Obtained from the Arduino Functional Block using Deep Learning and Big Data.

6. FUNCTIONAL BLOCK DIAGRAM OF CONTROL SECTION

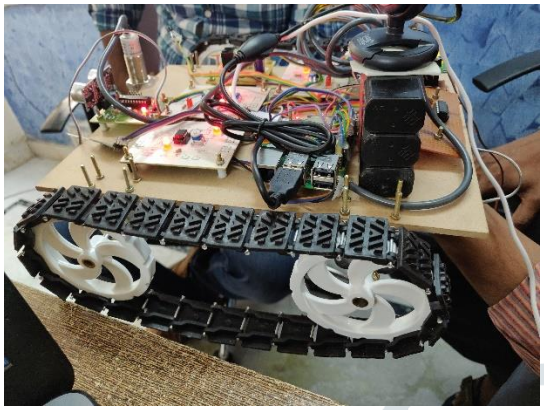


The Image downloaded from cloud to PC is processed in MATLAB using following code:

```
clc
close all;
clear all;
warning ('off');
a=[];
ccf=[];
dirr = cd;
vid = webcam;
snapback=snapshot(vid);
figure,imshow(snapback);
imwrite(snapback,'Background.jpg');
pause(2);
%  [file,  path]  =  uigetfile('*.jpg','Select
File');
% snapback = imread([path file]);
figure,imshow(snapback);
I = snapback;
FDetect = vision.CascadeObjectDetector;
BB = step(FDetect,I);
Img = imcrop(I,BB);
Img = imresize(Img,[256 256]);
GRAY = rgb2gray(Img);
dir1 = [dirr '\Database'];
cd(dir1);
AllFiles = dir('*.jpg');
for i = 1:length(AllFiles)
    MM=AllFiles(i).name;
    IN = imread(MM);
    BB1 = step(FDetect,IN);
    Img1 = imcrop(IN,BB1(end,:));
    Img1 = imresize(Img1,[256 256]);
    GRAY1 = rgb2gray(Img1);
    [m, n] = size(GRAY1);
    c=(GRAY-uint8(GRAY1));
    c1=graythresh(c);
    c2=im2bw(c,c1);
    BWdfill = imfill(c2, 'holes');
    d=strel('diamond',1);
    d1=imopen(BWdfill,d);
    bord = imclearborder(d1, 4);
    bord=double(bord);
    gg=regionprops(bord,'all');
    if size(gg,1)==0 || (gg.FilledArea>200 &&
gg.FilledArea<(m*n/10))
        disp('AUTHORISED PERSON');
        ccf=1;
        break;
    else
        if i==length(AllFiles)
```

```
            disp('UNAUTHORISED PERSON');
            ccf=0;
        end
    end
end
cd(dirr);
```

7. PCITORIAL REPRESENTATION



8. FUEL CONSUMPTION PREDICTION USING MACHINE LEARNING

Machine Learning is used to predict consumption of Fuel in Real Life Rovers, this can be done by examining data from several different sources describing road, vehicle, driver and weather characteristics & a regression to a fuel consumption is measured in liters per distance. The thesis is done for Scania and uses data sources available to Scania. We evaluate which machine learning methods are most successful, how data collection frequency affects the prediction and which features are most influential for fuel consumption. We find that a lower collection frequency of 10 minutes is preferable to a higher collection frequency of 1 minute. We also find that the evaluated models are comparable in their performance and that the most important features for fuel consumption are related to the road slope, vehicle speed and vehicle weight.

This study evaluates methods of machine learning (ML) and statistical analysis for predicting fuel consumption in heavy vehicles. The idea is to use historical data describing driving situations to predict a fuel consumption in liters per distance.

PREDICTOR 1	PREDICTOR 2	PREDICTOR 3	PREDICTOR 4
SPEEDOMETER	DRIVER BEH. (SENSOR)	WEIGHT IN CAR	SUDDEN BRAKE SYESTE VS SPEED
60	0001	400	
50	0100	500	
40	0011	550	
30	0101	600	
20	0010	650	

9. DATA COLLECTION & PROCESSING

The data that will be used for training comes from four different sources. The first and arguably the most important data source is the fm data that is collected by Scania and stored in their own database system. To complement the fm data there is also map data and information about road characteristics which comes from Scania's parent company Volkswagen, as well as weather data which is accessed through smhi's web-based interface and vehicle configuration data from an internal Scania system.

Road data

The road data comes from the system Digital Reality 3.0 which is provided by Scania's parent company Volkswagen. The Digital Reality system is developed by Volkswagen as part of the companion project. It is a GPS routing system and the desktop installation consists of a frontend GUI, or workbench, which allows high level interactions with the underlying map data. Using the workbench one can visualize map data and routes and have access to methods for routing between two or more waypoints. The system also includes a low-level Java api which provides methods for querying the map database using the Java programming language. The database itself is in the Navigation Data Standard (nds) format and has been purchased from TomTom which is a navigation and mapping company. All roads in the database are broken down into links. A link, in the terms of the GPS routing software, is the longest possible piece of road on which a vehicle can travel without any updated navigation instructions. For example, if a road has an intersection or a roundabout the link will be broken and new links will be added. The intersection is itself not a link, but rather a link-connector element. In terms of the systems data model a new link will be created whenever a value of the fixed attribute set changes along a road, e.g. whether or not the road is a bridge or a tunnel or if it passes through an urban area. Each such link has a number of intrinsic properties (for instance the members of the fixed attribute set) as well as a number of computed properties such as the average slope and the average speed.

10. WEATHER DATA

Table: Variables of interest in the driver behavior data.

Variable name	Description
Distance with trailer	The total distance driven with a trailer attached during the period
Time over speeding	Time spent over 80 km/h during the period
Time overrevving	Time spent in high revolutions during the period
Harsh brake applications	Number of harsh brakes during the period

Brake applications	Number of brakes during the period
Harsh accelerations	Number of harsh accelerations during the period
Time out of green band driving	Time spent in environmentally optimal revolution during period
Distance with vehicle warnings	Distance driven with vehicle warnings during the period
Distance with CC active	Distance driven with cruise control during the period
Distance moving while out of gear	Distance travelled in neutral gear during the period
Calculated vehicle weight	An estimate of the vehicle weight, measured by the suspension

11. DATA CONSOLIDATION

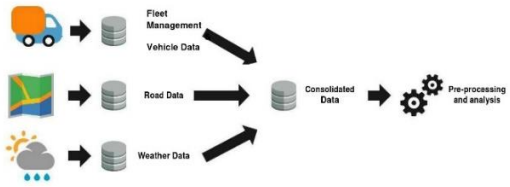


Fig: Schematic view of the data consolidation process.

We will be focusing on training the model to some distance implementing different parameters, from that training data various observations will be deducted and based on that, we will select a machine learning model based on which gives the best results

Performance metrics for regression methods:

To evaluate how well an ml method for regression describes the underlying relationship several metrics may be applied to the trained model.

12. LINEAR REGRESSION:

Linear regression is the fitting of a linear function of one or more inputs to an output. In the univariate case it is the fitting of a straight line with input x and output y on the form y = w1x + w0, where w0 and w1 are real-valued coefficients to be learned

Linear regression is the fitting of a linear function of one or more inputs to an output. In the univariate case it is the fitting of a straight line with input x and output y on the form y = w1x + w0, where w0 and w1 are real-valued coefficients to be learned [17]. When finding the weights in a linear regression problem it is most common to minimize the squared loss function. The problem is then to find the weight vector w∗ according to (3.3). Choosing the weights in this way guarantees that we'll find a unique global minimum

$$w* = \text{argmin}\ w\ \text{Loss}(hw)$$

where, hw(x) = w1x + w0

The multivariate case of linear regression is not much more complex than the univariate case. In multivariate linear regression each example xj is an n-element vector and the object are to find the hyperplane which best fits the outputs y according to some loss function, most commonly the squared loss function. The hypothesis space is given by the set of functions of the form defined by. The vector of weights that minimizes the loss function is given by

hsw(xj) = w|xj = X i wixj,i.

w∗ = argmin w X j Loss(yj , hsw)

13. CONCLUSION

This paper presents an approach which could be used for developing a Smart Mine Detection Rover with Image Processing and Deep Learning Capabilities which could aid in carrying out Search & Rescue Missions, Analysis of Location. It could also be used as a surveillance system to increase the security strength especially in the area where human interference is strictly prohibited and detect for any Intrusion.

14. REFERENCES

[1] Machine learning and statistical analysis in fuel consumption prediction for heavy vehicles
[2] HENRIK ALMÉR
[3] [1] R. Arvidson, et al., "NASA Mars Exploration Program Mars 2007 Smart Lander Mission Science Definition Report," 11 October 2001.
[4] [2] S. Krasner, "A Reusable State-Based Guidance, Control and Navigation Architecture for Planetary Missions," Proceedings of 2000 IEEE Aerospace Conference, March 2000.
[5] [3] http://research.hq.nasa.gov/
[6] [4] http://www.hq.nasa.gov/office/codeq/trl/
[7] [5] R. Volpe, I.A.D. Nesnas, T. Estlin, D. Mutz, R. Petras, H. Das, "The CLARAty Architecture for Robotic Autonomy," proceedings of the 2001 IEEE Aerospace Conference, Big Sky Montana, March 10-17 2001.
[8] [6] J. Yen and A. Jain, "ROAMS: Rover Analysis Modeling and Simulation Software," in International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Noordwijk, Netherlands, June '99.
[9] [7] L. Matthies, ``Stereo Vision for Planetary Rovers: Stochastic Modeling to Near Real-time Implementation'', International Journal of Computer Vision, 8(1), July 1992.
[10] [8] Y. Xiong and L. Matthies, "Error Analysis of a Real-Time Stereo System", IEEE Conference on Computer Vision and Patter Recognition (CVPR), Puerto Rico, June 17-19, 1997.
[11] Symposium on Artificial Intelligence, Robotics and Automation in Space, June, 1999. [11] P. Backes, K. Tso, J. Norris, G. Tharp J.
[12] https://circuitdigest.com/microcontroller-projects/sending-arduino-data-to-webpage