INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

IJARIIT

ISSN: 2454-132X

Impact Factor: 6.078

(Volume 7, Issue 3 - V7I3-2101)

Available online at: https://www.ijariit.com

GEVAS: GPS based emergency vehicle detection system

Laxman Krishnan
laxmankrishnan99@gmail.com
Marian Engineering College, Kazhakuttom, Thiruvananthapuram

Pankaj Madhu
pankaj.madhu01@gmail.com
Marian Engineering College, Kazhakuttom, Thiruvananthapuram

C. Nirupama
nirpooja99@gmail.com
Marian Engineering College, Kazhakuttom, Thiruvananthapuram

Devi Saranya R. S.
devisaranyars@gmail.com
Marian Engineering College, Kazhakuttom, Thiruvananthapuram

Evankumar G.
evankumar.ec@marian.ac.in
Marian Engineering College, Kazhakuttom, Thiruvananthapuram

ABSTRACT

Emergency vehicles being stuck in traffic jams and various kinds of accidents caused due to speeding first response vehicles has become a very common scene in our country lately. The increased traffic congestion and underdeveloped roads along with the poor traffic regulation system has resulted in emergency vehicles like ambulance not reaching their required destination in time. So, the idea emphasizes mainly on enabling emergency vehicles to reach their required destination safe and smooth within least possible time on any road traffic conditions. This could be implemented by developing a GPS based device which notifies all other vehicles so that they can move out of the way in advance avoiding accidents and unnecessary traffic jams and clearing path for the emergency vehicle. The notification is either visual/sound based and the system is placed inside every vehicle so that the alert precedes the conventional siren alert system.

***Keywords**: Emergency Vehicle, GPS, NRF, Server, Notifier, Range, Location, Alert*

1. INTRODUCTION

Emergency vehicles in most cases help save lives but statistics show that every year 6 out of 10 accidents are caused due to speeding emergency vehicles. Increased number of vehicles on road is the biggest rising challenge faced by these emergency responders. In an overpopulated country like India where the roads are heavily packed with vehicles most of the time, the siren alert is usually rendered useless since the last-minute alert causes nothing but panic and severe traffic jams. An innovative improvement to the existing alert system has been long due and this system could be a viable improvement for the same.

The system mainly has two units: -

1. The Server System (Present in the emergency vehicle)
2. The Notifier System (Present in all other vehicles)

Wireless communication between the server and the notifier unit is accomplished by using an NRF24L01 module which serves to carry both GPS and alert data, among vehicles and from server to notifier respectively.

In the server unit, the location data of the emergency vehicle and the location data of all the other vehicles on the road within a set range around the emergency vehicle is obtained and the distance between these vehicles are found out by processing the data obtained from the respective GPS modules, this data is basically used for 2 purposes:

- To provide alert only to nearby vehicles.
- To find out whether the vehicle is in front of or behind the emergency vehicle.

The Notifier unit serves two important functions with all other vehicles:

- To transmit GPS data to the server unit.
- To provide alert within the vehicles.

The chart below shows the basic working plan and flow of communication between the units.

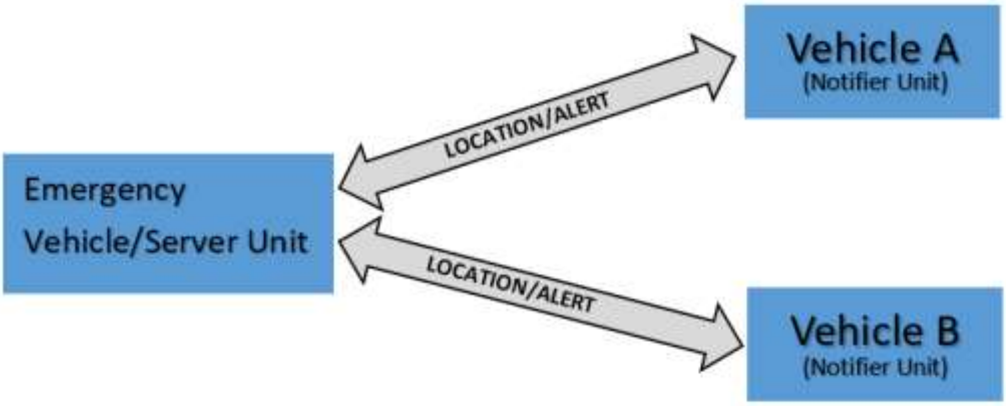

Chart-1: Basic Communication Flowchart

2. IMPLEMENTATION

2.1 Hardware

The components used in both the notifier and server unit are identical with the only difference being that there is an additional circuitry in the notifier unit which serves the purpose of providing the alert/notification.

The list of components common both units are:

1. Arduino UNO
2. GPS Module (NEO-6M V2)
3. nRF24L01 Module

The components present only in the notifier unit are;

1. LED
2. 220 Ohm Resistor
3. 8 Ohm 5W Speaker
4. 12V 2A Adaptor

➢ Arduino Uno- is the microcontroller that is used to carry out all the calculation, analysis and alert mechanism initialization required to complete all the steps. It specifically carries out functions such as read GPS data, use the data to calculate distance between notifier unit and server units and thereby provide alert to the notifier unit.

Arduino UNO supports various communication protocols such as SPI, UART, I2C etc and this was a basic requirement during interfacing components such a NRF module and GPS Module.

➢ GPS Module (NEO-6M V2)- This module is used to obtain the latitude and longitude data of the vehicles. UART protocol is used to establish communication with Arduino microcontroller. It collects location data by tracking up to 22 satellites on 50 different channels. Baud rates ranging from 4800bps to 230400bps are used with a default of 9600. The module operates at a voltage range from 2.7 to 3.6V. It contains an EEPROM, which along with a rechargeable battery helps the Battery Backed RAM(BBR) thus retaining the latest position and module configuration.

➢ nRF24L01 Module – Wireless communication between the sever and notifier unit is established using radio frequency communication through nRF24L01 module. It communicates over a 4-pin Serial Peripheral Interface (SPI) using serial communication protocol. It has a data range of 10Mbps and operates at frequency 2.4Ghz ISM Band. The frequency and output parameters can be varied and configured using the SPI Interface. It operates at voltages ranging frin1.9V to 3.6V consuming up to 12mA during the time of transmission.

2.2 Circuit Diagram

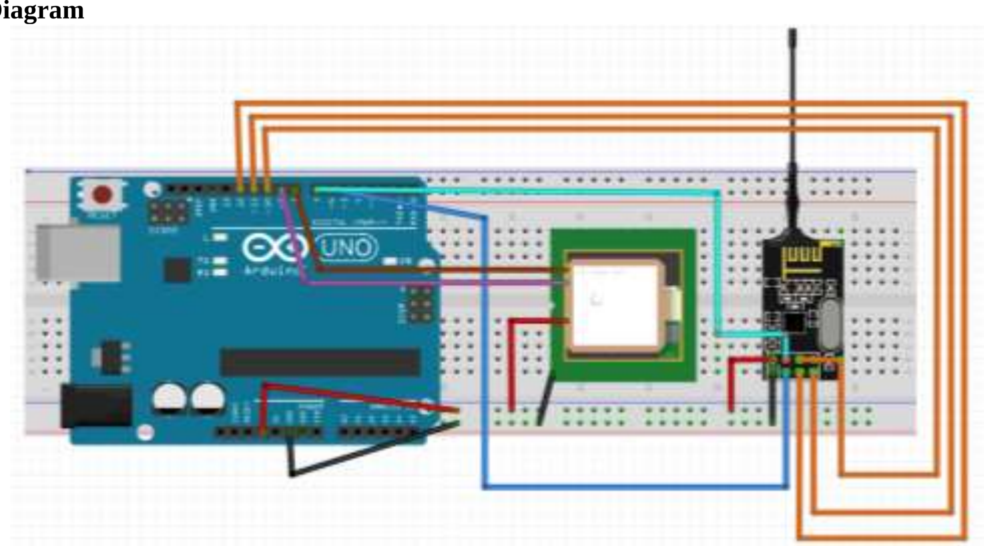

Fig. 1: Circuit of Server Unit

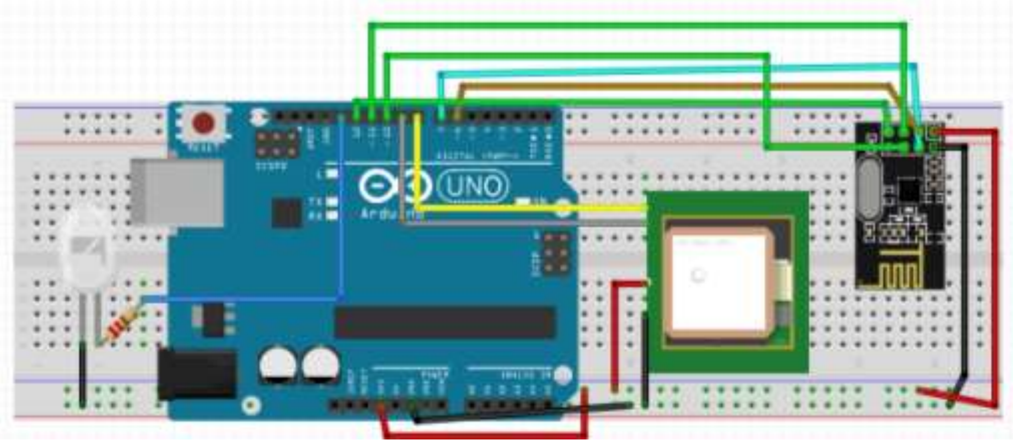

Fig. 2: Circuit of Notifier Unit

The circuit for both the server and the notifier are identical, having only one difference which is the presence of a notification unit in the notifier unit which is visual having an LED or sound based having a speaker.

2.3 Server Unit Setup

The server unit is present inside the emergency vehicle and it is connected to a PC/Laptop to observe the various analysis and calculations done by it. Data transfer takes place at the rate of 960Kbps and it is set to a baud rate of 9600. It sets the NRF module to start or stop listening depending upon on whether or not a radio is available. If radio is available, the receiver is to start listening so as to obtain location. After receiving, the server unit stops listening and begins to analyse the received data to obtain the distance.

2.3.1 Location Analysis: The GPS data is used to analyse the distance between vehicles and provide alert. The latitude and longitude data of other vehicles and the location data of the vehicle in which the server is present is stored in different variables and then used to calculate distance. This process is to be repeated every millisecond so as to provide continuous alert. The latitude and longitude data from these vehicles also helps to determine the vehicle which is in the front. This is important as this helps to eliminate the error factor that leads to unwanted panic if the vehicles behind the emergency vehicle are alerted. The speed of the emergency vehicle and the speed of other vehicles in the same direction and the opposite direction is also taken into consideration while setting the refresh rate. The delay is set to be as minimum as possible and higher transfer rates are used so that alerts are not missed due to high travelling speed of the emergency vehicle.

```
void calc_dist(float lat1, float lon1, float lat2,float lon2)
      {
      float distLat = abs(lat1 - lat2) * 111194.9;
      float distLong = 111194.9 * abs(lon1 - lon2) * cos(radians(lat1 + lat2) / 2);
      distance = sqrt(pow(distLat, 2) + pow(distLong, 2));
      }
```

Fig. 3: Distance Calculation Formula

This formula acts as the basis to demonstrate the working of our prototype. It is written as a user defined function which accepts the latitude and longitude values from the GPS module and uses it to obtain the distance between the units. This calculation is to be done simultaneously for two vehicles and embedding the formula as a user defined function helps in reducing the length and complexity of the code. This also helps us to expand and add more parameters so as to obtain an expanded demonstration or test scenarios. The data input into this formula is in the form of random numbers obtained from GPS module in both server and notifier units. This data is used to calculate the distance, which is the output of this user defined function and this output is presented in metres (unit of distance).

2.3.2 Alert Range

In our case, we have decided to set the range to provide alert as 100m. It means, only the vehicles which are within 100 metres range of the emergency vehicles receive the alert.

```
if(distance_car1<100)
{
 Serial.println("!!!!! Car 1 warning !!!!!");
 Serial.println("!!!!! RED LIGHT ON !!!!!");
 radio.openWritingPipe(car1);
 radio.stopListening();
 radio.write(alert, sizeof(alert) );
}
if(distance_car2<100)
{
 Serial.println("!!!!! Car 2 warning !!!!!");
 radio.openWritingPipe(car2);
 radio.stopListening();
 radio.write(alert, sizeof(alert) );

}
```

Fig. 4: Comparison of distance

In the server unit the minimum distance is to provide alert set and then implemented by using comparators. The distance is saved into two integer variables and the value is compared against the set range. If the distance value is less than the set range, then the notifier unit is made to provide alert. This process can be monitored on the serial monitor in the server unit vehicle. Details such as which vehicle is within the range and whether an alert has already been sent can all be seen on the serial monitor. The locations data and the detailed log of the server unit can be recorded or stored in an SD card and this data could be used later to plot the path of the emergency vehicle by plotting points on maps.

2.4 Notifier Unit Setup

The notifier unit is setup within every other vehicle and its purpose is to update the location of the vehicle to the server and also receive and display alerts. Similar to the server, implementation of both these steps is decided by assigning either stop listening or start listening to the wireless NRF module. This unit is also setup at a baud rate of 9600. We have provided both visual and sound alert with the help of an LED and a speaker. This unit is not connected to a PC/laptop and thus is aimed to be powered through the vehicle's power source (battery).

2.4.1 Transmit GPS Data: The GPS data decoded by Arduino is sent to the server unit when the wireless communication module is in stop listening mode. This data is saved onto variables and later used for calculation as seen before.

```
void loop()

 {
   while(gpsSerial.available())
   {

    if(gps.encode(gpsSerial.read()))
    {
      gps.f_get_position(&lat2,&lon2);
 data[0]=lat2;
 data[1]=lon2;
 data[2]=2;// car/vehicle 2
 // TXXXXXXXXXXXXXXXXX //
 radio.openWritingPipe(car1);
 radio.stopListening();
 radio.write( data, sizeof(data) );
```

Fig. 5: Transmission of GPS data

The notifier keeps on transmitting GPS data in a loop, thereby helping to increase the accuracy of the alert. This means that the location of the vehicle and the distance between it and the emergency vehicle is updated every few seconds. This reduces the chance of missing an alert and increases the time left for drivers to react.

2.4.2 Receive Alert from Server

The notifier keeps on transmitting till radio is available. If radio is available, the NRF module is switched to start listening mode so that it can listen to the alert request send by the server and thus provide and alert.

```
radio.openReadingPipe(1,car2);
radio.startListening();
if ( radio.available() )
{
Serial.println("!!!!! Car 2 warning !!!!!");
Serial.println("!!!!! RED LIGHT ON !!!!!");
digitalWrite(ledPin,HIGH);
delay(500);
digitalWrite(ledPin,LOW);
delay(500);
digitalWrite(ledPin,HIGH);
delay(200);
digitalWrite(ledPin,LOW);
delay(200);
digitalWrite(ledPin,HIGH);
delay(200);
digitalWrite(ledPin,LOW);
}
```

Fig. 6: Reception of Alert from server

The figure shows a sample mechanism in which an LED is made to blink if the vehicle is within range. During alert, for reference, the serial monitor is made to show details such as whether the LED is blinking and a warning message. This alert is provided till either the emergency vehicles passes the other vehicle or till the vehicle moves away from the emergency vehicle above the set range.

3. RESULTS

3.1 Unit Wiring and Setup

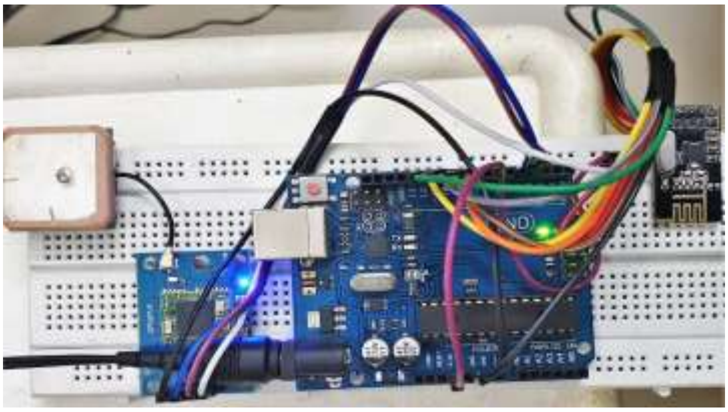

Fig. 7: Server Unit

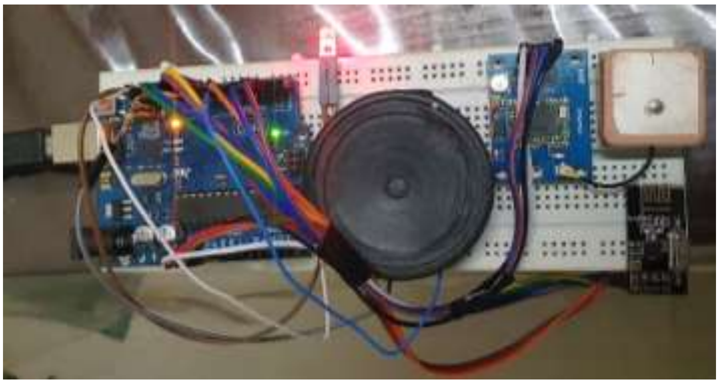

Fig. 8: Notifier Unit

The server and the notifier unit are both setup based on the circuit diagram and then tested. The aim of these tests were to find out various parameters such as real-life range of NRF24L01 module, accuracy of GPS module, finetune the system and finally find out the ideal range to provide the alert.

3.2 Serial Monitor Log

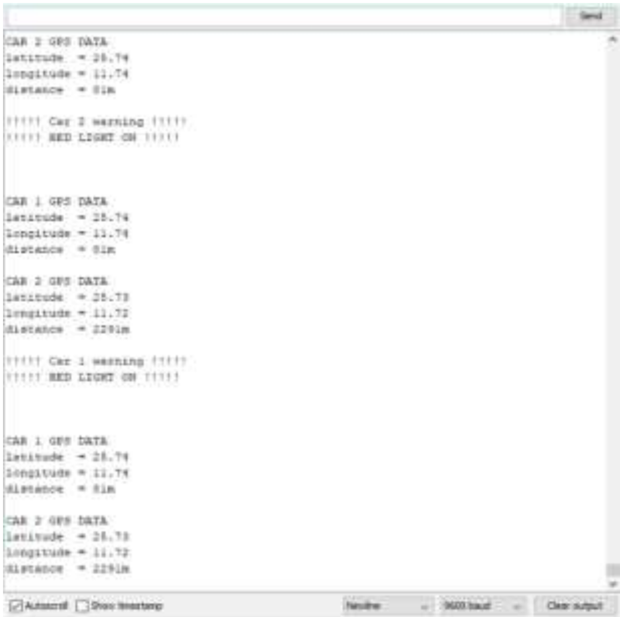

Fig. 9: Monitor Log

In the serial monitor, we can see the collection of GPS data from both car 1 and car 2. When the distance of car (1 or 2) is less than 100m, the respective car is given alert by the server. The alert in the notifier unit is shown in Figure-8 by using an LED.

4. CONCLUSION

The prototype on nearing completion has made us understand the various improvements that could be made to this project such as using a different wireless communication module etc.

The sole purpose of this project was to provide a universal solution to the challenges faced by emergency vehicles all over the world. This solution could ultimately overcome the disadvantages of current alert systems but still has rooms for improvements. Our final aim is to upgrade this solution using Google Maps API so that this system could be integrated along with the modern car infotainment systems which detect and pin point the location of an emergency vehicle behind us by providing an alert which has the capability to suppress music currently playing on the system. Such a device, in the real world would provide the best solution to this problem.

5. REFERENCES

[1] https://www.instructables.com/Smart-Buoy-GPS-Radio-NRF24-and-a-SD-Card-Module-/
[2] https://create.arduino.cc/projecthub/Rad_Silviu/speaker-with-arduino-be454c
[3] https://www.electronicwings.com/arduino/nrf24l01-interfacing-with-arduino-uno
[4] https://cloud.google.com/blog/products/maps-platform/how-calculate-distances-map-maps-javascript-api
[5] https://lastminuteengineers.com/neo6m-gps-arduino-tutorial/
[6] https://www.arduino.cc/reference/en/libraries/neogps/
[7] https://www.arduino.cc/reference/en/libraries/rf24/

