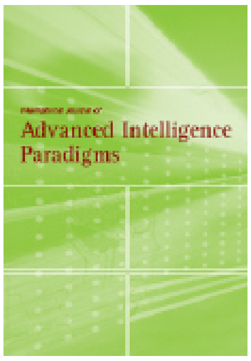


International Journal of Advanced Intelligence Paradigms

ISSN online: 1755-0394 - ISSN print: 1755-0386
https://www.inderscience.com/ijaip

An IoT-based accident severity detection for automobiles with alerting the appropriate location of the accident - an innovative attempt

J. Anudeep, G. Kowshik, G.I. Aswath, Shriram K. Vasudevan

DOI: 10.1504/IJAIP.2024.10062355

Article History:

Received:	25 May 2018
Last revised:	03 June 2018
Accepted:	03 June 2018
Published online:	22 February 2024

An IoT-based accident severity detection for automobiles with alerting the appropriate location of the accident – an innovative attempt

J. Anudeep

Department of Electronics and Communication Engineering,
Amrita School of Engineering,
Amrita Vishwa Vidyapeetham,
Coimbatore, India
Email: janudeep339@gmail.com

G. Kowshik

Department of Electronics and Instrumentation Engineering,
Amrita School of Engineering,
Amrita Vishwa Vidyapeetham,
Coimbatore, India
Email: kowshik700@gmail.com

G.I. Aswath

Department of Electrical and Electronics Engineering,
Amrita School of Engineering,
Amrita Vishwa Vidyapeetham,
Coimbatore, India
Email: giaswath@gmail.com

Shriram K. Vasudevan*

K. Ramakrishnan College of Technology,
Kariyamanikam Road, Samayapuram, Trichy – 621112, India
Email: shriramkv@gmail.com
*Corresponding author

Abstract: Even though occupant protection systems are aiding the present means of transportation, the deaths on highways are not decreasing due to lack of medical attention and delay in medical help. For an instance consider a place on a long highway where there exists only one hospital and let us suppose that four to five accidents happened near the hospital. If the ambulance is sent to the nearest place where the severity of the accident is very low, the person with a big hit will succumb to death fast. So, to overcome this problem we introduce a system to measure and intimate the severity of the accident, location, time of the accident. This system mainly strives in decreasing the delay in time of arrival of medical services to the place of the accident and also helps the hospitals in deciding the right place for the medical services to be sent.

Keywords: severity level; GPS data; force sensitive resistor; FSR; collision; webpage.

Reference to this paper should be made as follows: Anudeep, J., Kowshik, G., Aswath, G.I. and Vasudevan, S.K. (2024) 'An IoT-based accident severity detection for automobiles with alerting the appropriate location of the accident – an innovative attempt', *Int. J. Advanced Intelligence Paradigms*, Vol. 27, No. 1, pp.29–42.

Biographical notes: J. Anudeep is a third year Electronics and Communication Engineering student who has got abundant interest in the IoT. He is a student with lot of passion to work on embedded systems and products for healthcare. He has participated in many contests and works on interesting research statements.

G. Kowshik is a third year Electronics and Instrumentation engineering student who has got abundant interest in the IoT. He has participated in many contests and Hackathons and won prizes. He is interested in embedded systems, IoT and accident avoidance systems.

G.I. Aswath is a final year student from the Electrical and Electronics Engineering. He is very much interested in machine learning, deep learning and embedded systems applications. He has participated in many contests at national level. He also has got papers published in reputed journals.

Shriram K. Vasudevan is an Embedded System Engineer with 15 years of experience in IT and academics. He has authored 40 books for various reputed publishers across the globe. He has also written a lot of research papers. He has been awarded/recognised by ACM, Intel, IEI (India), Wipro, Infosys, ICTACT and VIT University, etc. for his technical contributions. He has received his Master's and Doctorate in Embedded Systems. He is currently associated as Principal with the K. Ramakrishnan College of Technology, Trichy. He was earlier associated with Wipro Technologies, Aricent Technologies, VIT University, and Amrita University.

1 Introduction

The challenge we are trying to solve with this system is to intimate the severity level of an accident to hospitals thereby priority in providing medical aids is given to the accident with highest severity level by using internet of things (IoT) and sensors. According to some of the surveys, it is found that the average response time for an ambulance to reach the site of an accident inside a district or town in India is approximately 18 to 19 minutes and with three ambulances it was brought down to 15 to 16 minutes (https://timesofindia.indiatimes.com/city/coimbatore/new-108-ambulances-to-bring-down-response-time/articleshow/60126988.cms). So from the above statistics, we can say that it takes approximately 20 minutes for the treatment to start which is a huge time for the patient. Now, if we extend the same problem for multiple accidents occurred at the same time in near premises from each other, it will become a knotty situation for the hospital authorities and creates confusion in sending the ambulance and due to lack of knowledge on the severity of the accident, wrong priority allocation takes place and medical aids may be sent to places with lesser severity. So, in order to overcome these types of problems we have come up with a solution which can predict the severity of the accident and conveys the location coordinates, Google maps link to the hospitals in order to reduce the chances of wrong priority allocation.

When there is an accident in a deserted place or long highways or in deep forest roads, there will be a lot of delay in noticing the accident which may cost the life of a person. In order to avoid it, our system immediately intimates the hospitals the place of the accident by which they can provide the medical aids immediately.

2 Existing solutions

The following citations are some of the related work done in implementing the above-mentioned necessity.

The system proposed by Hannan et al. (2008) works by detecting the frontal crash using pneumatic devices which works by principles of gases or air. It makes use of an algorithm for making precise collisions. This is useful only in order to detect the accident but not to measure how severe the accident is.

One more system which is designed by Le et al. (2009) makes use of pressure sensors that are mounted on the doors of cars and also uses an accelerometer sensor in order to detect the force on the car. When both the signals reach the threshold value the alert is generated but this is system is not designed to estimate the severity percentage and given an alert.

There is one more attempt in the direction of solving this challenge by Thompson et al. (2010) who has designed a system which detects whether crash has happened or not and intimates the data like position of the vehicle, location of the accident, etc., by using the onboard components inside a mobile phone. This is an efficient method in detecting and calculating the severity of the accident but at every instant of time, data may not be retrieved accurately from the mobile phones and when the accident happens one cannot rely on or wait for the mobile alert to be sent. There are many more disadvantages in it like the phone should be charged enough, the phone should be carried always whenever they are travelling there may be cases when we travel without a phone.

Table 1 Table of contrast between proposed and existing solutions

	Uses sensors	*Uses mobile*	*Provides location*	*Web application*	*Alerts*
Hannan et al. (2008)	✓	×	×	×	✓
Le et al. (2009)	✓	×	×	×	✓
Thompson et al. (2010)	×	✓	✓	×	✓
Proposed system	✓	×	✓	✓	✓

But even after the implementation of many methodologies, one cannot arrive at a point where the parameters to detect the severity are not confined. The contrast between the above systems and the proposed system are tabulated in Table 1.

3 Problem statement

To device an equipment to detect the severity of the impact during automobile collision while also recording the location of the accident for further action.

4 Architecture

Our proposed system uses a collision sensor which is practically used in cars for detecting whether the collision has happened or not, force sensitive resistor (FSR) which measures the impact experienced by the car, GPS module for getting the exact location of the car. If a collision has occurred then the FSR data, GPS data is sent to the server like Raspberry pi for data analytics to classify the severity level which immediately sends the timestamp, pin code, latitude, longitude, severity level and Google maps link data of the collision to a website for the reference of hospitals. This website can be used by the hospitals to know the information about nearby accidents and their severity level to provide emergency services accordingly. For the prototype purpose, we constructed a localhost database and HTML page for storing and displaying collision data using Xampp and phpmyadmin.

From Figures 1, 2, we can infer the steps through which the impact severity level is measured and intimated to the authorities. Firstly, data from the collision sensor, FSRs are collected and measured against the threshold. If the alert is needed to be triggered, the GPS sensor data is taken and severity levels are processed through a python script which is uploaded to the MySQL database which is again uploaded on a webpage using a PHP code.

Figure 1 Schematic of the proposed system (see online version for colours)

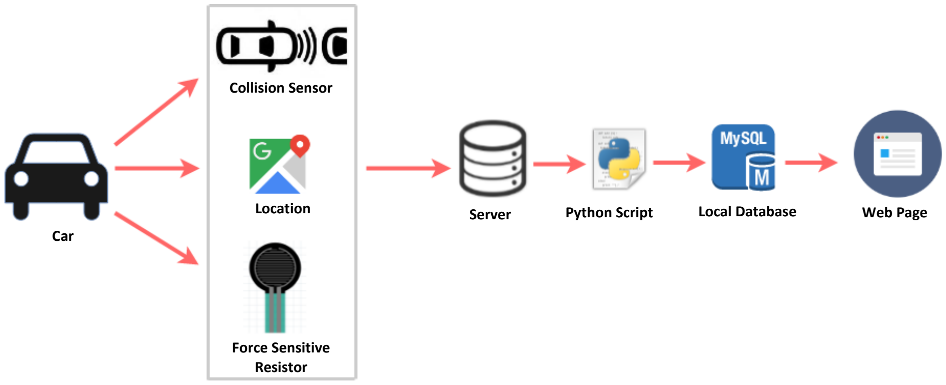
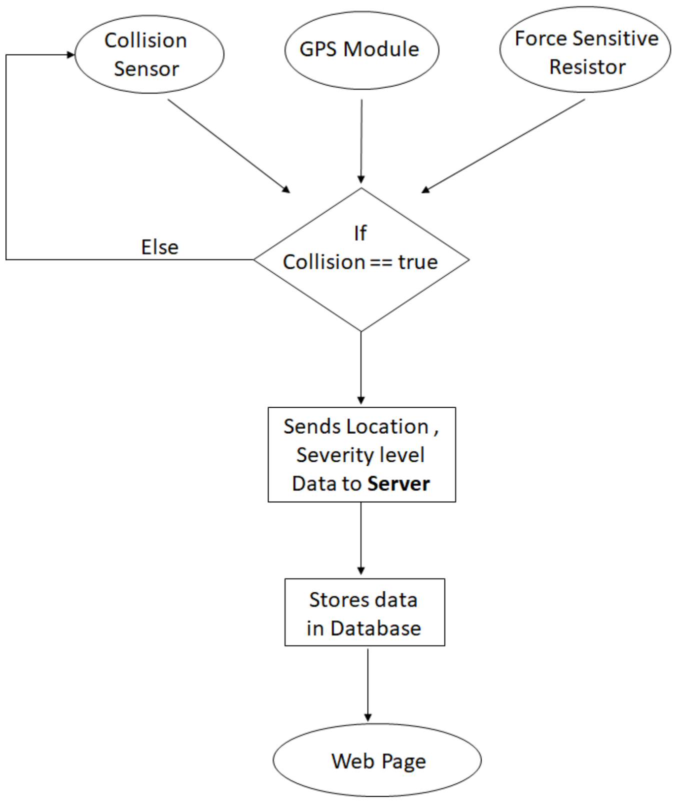


Figure 2 Workflow of the proposed system



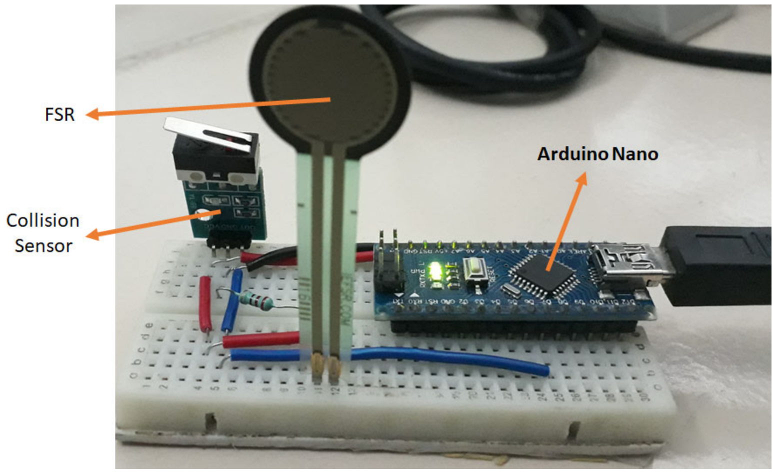
4.1 Data from car

Data is collected from FSR, a collision sensor, GPS module placed inside the car. The working and usage of these sensors are explained below:

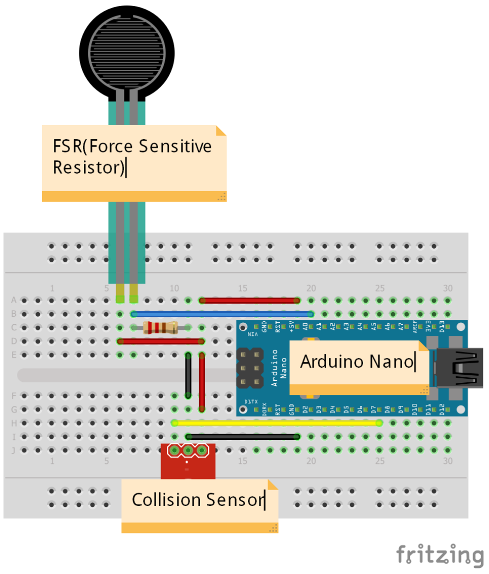
4.1.1 Collision sensor

it is a microcontroller interfaceable sensor which is used for collision detection in most of the cars. This sensor sends a signal immediately when a crash has occurred which in real-time is used for triggering airbags, locking the doors, etc. But in our proposed system we used it for sending FSR, GPS data to the webpage when a crash has been detected. The interfacing diagrams are presented as Figure 3.

Figure 3 Experimental setup along with the labelled components (see online version for colours)



(a)



(b)

4.1.2 *Force sensitive resistors*

FSR works on the principle of variable resistance with the variation of force applied to it. When no force is applied on the FSR then its resistance is very much high, as the applied force increases FSR resistance decreases by which we can determine the force experienced by the FSR. By using this principle, the severity level of the collision can be easily found and sent to the webpage for emergency aids. By training the proposed system we can determine the exact position of FSR to be placed inside the car by which correct level of impact experienced can be known. By placing many FSR sensors at different places inside the car can improve the accuracy in detection of impact experienced by the car.

The below code is used in getting the collision sensor, FSR data from the Arduino Nano and sending it to the Python code:

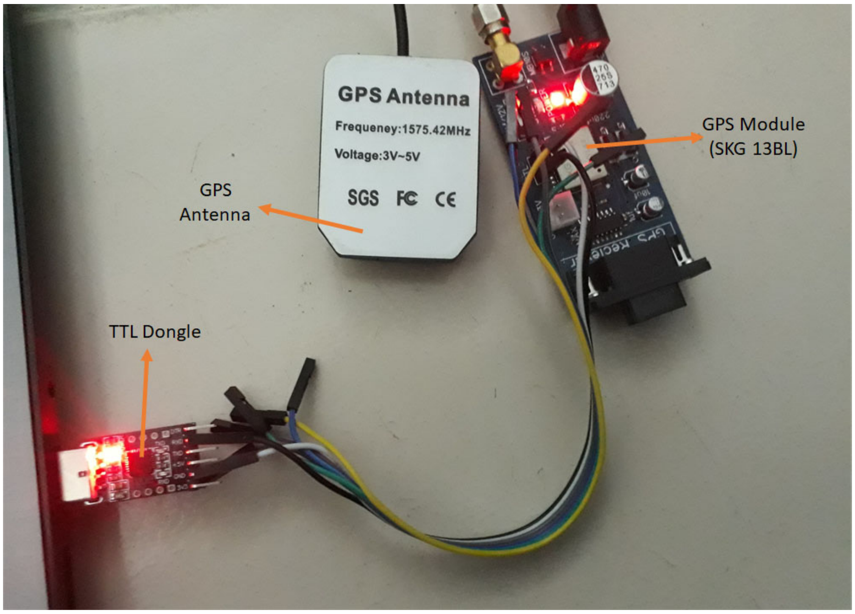
```
int fsr = A0;
int col = 7;
void setup() {
  Serial.begin(9600);
  pinMode(fsr, INPUT);
  pinMode(col, INPUT);
}
void loop() {
  if (digitalRead(col) == 0){
    Serial.println(analogRead(fsr));
    delay(1000);}
}
```

4.1.3 GPS module

In the proposed system, GPS module is used for getting the location data of the car at the time of collision which is very much required for the emergency services to reach the exact accident location. We have used an SKG13BL Skylab GPS module and a TTL dongle to transmit the GPS data directly to the server.

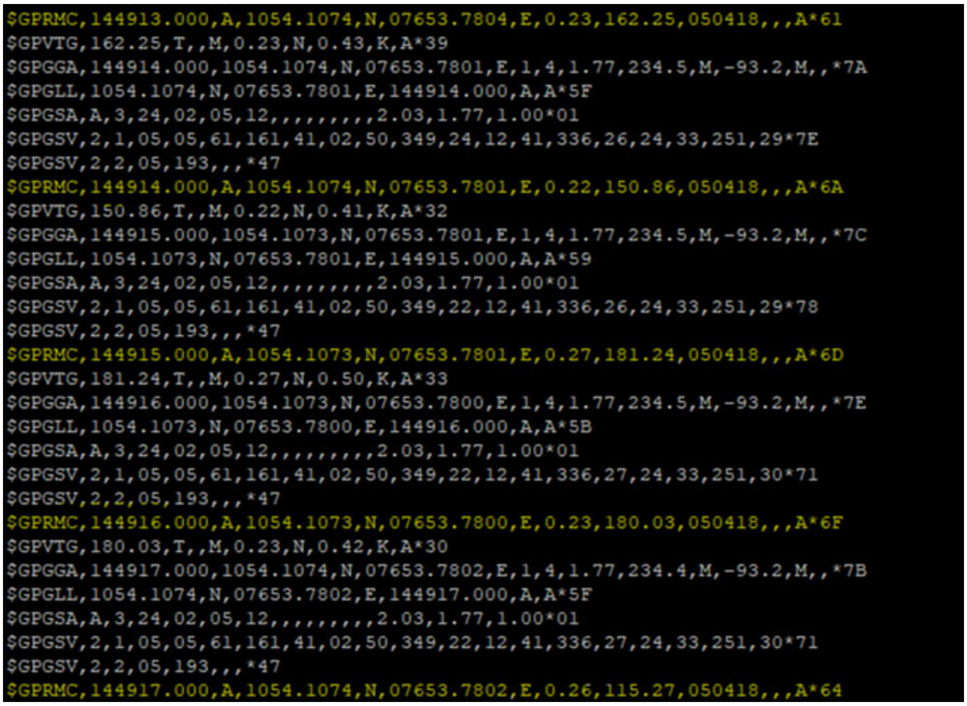
The experimental setup is given in Figure 4.

Figure 4 GPS module used for locating the accident (see online version for colours)



The above-mentioned GPS antenna connects to a maximum of nine satellites and works without errors only when it connects with at least three satellites, as the connection with the number of satellites increases the data accuracy increases. It fetches data from which we can get the velocity of the GPS module (if it is moving), timestamp, latitude, longitude, altitude, number of satellites connected, etc., which is very useful in locating the accident. This data is generally called as NMEA data and by parsing it we are able to collect the latitudes and longitude details.

Figure 5 NMEA data fed to python script (see online version for colours)



```
$GPRMC,144913.000,A,1054.1074,N,07653.7804,E,0.23,162.25,050418,,,A*61
$GPVTG,162.25,T,,M,0.23,N,0.43,K,A*39
$GPGGA,144914.000,1054.1074,N,07653.7801,E,1,4,1.77,234.5,M,-93.2,M,,*7A
$GPGLL,1054.1074,N,07653.7801,E,144914.000,A,A*5F
$GPGSA,A,3,24,02,05,12,,,,,,,,,2.03,1.77,1.00*01
$GPGSV,2,1,05,05,61,161,41,02,50,349,24,12,41,336,26,24,33,251,29*7E
$GPGSV,2,2,05,193,,,*47
$GPRMC,144914.000,A,1054.1074,N,07653.7801,E,0.22,150.86,050418,,,A*6A
$GPVTG,150.86,T,,M,0.22,N,0.41,K,A*32
$GPGGA,144915.000,1054.1073,N,07653.7801,E,1,4,1.77,234.5,M,-93.2,M,,*7C
$GPGLL,1054.1073,N,07653.7801,E,144915.000,A,A*59
$GPGSA,A,3,24,02,05,12,,,,,,,,,2.03,1.77,1.00*01
$GPGSV,2,1,05,05,61,161,41,02,50,349,22,12,41,336,26,24,33,251,29*78
$GPGSV,2,2,05,193,,,*47
$GPRMC,144915.000,A,1054.1073,N,07653.7801,E,0.27,181.24,050418,,,A*6D
$GPVTG,181.24,T,,M,0.27,N,0.50,K,A*33
$GPGGA,144916.000,1054.1073,N,07653.7800,E,1,4,1.77,234.5,M,-93.2,M,,*7E
$GPGLL,1054.1073,N,07653.7800,E,144916.000,A,A*5B
$GPGSA,A,3,24,02,05,12,,,,,,,,,2.03,1.77,1.00*01
$GPGSV,2,1,05,05,61,161,41,02,50,349,22,12,41,336,27,24,33,251,30*71
$GPGSV,2,2,05,193,,,*47
$GPRMC,144916.000,A,1054.1073,N,07653.7800,E,0.23,180.03,050418,,,A*6F
$GPVTG,180.03,T,,M,0.23,N,0.42,K,A*30
$GPGGA,144917.000,1054.1074,N,07653.7802,E,1,4,1.77,234.4,M,-93.2,M,,*7B
$GPGLL,1054.1074,N,07653.7802,E,144917.000,A,A*5F
$GPGSA,A,3,24,02,05,12,,,,,,,,,2.03,1.77,1.00*01
$GPGSV,2,1,05,05,61,161,41,02,50,349,22,12,41,336,27,24,33,251,30*71
$GPGSV,2,2,05,193,,,*47
$GPRMC,144917.000,A,1054.1074,N,07653.7802,E,0.26,115.27,050418,,,A*64
```

As in our prototype, we need only the latitude, longitude for getting the location of the car remaining data can be neglected. In order to get latitude, longitude data it is sufficient to parse the `$GPRMC` data which is the highlighted text in Figure 5.

```
$GPRMC, 144,916.000, A, 1,054.1073, N, 07653.7800, E, 0.23,
180.03, 050418, A * 6F
```

We can get the latitude and longitude information from the above line by the following logic:

1 From the above line, highlighted text is taken separately,

   `1054.1073,N,07653.7800,E`

2 Let the number be of the format *xxxyy.zzzz* then exact latitude or longitude number is obtained by the formula: `xxx + yy.zzzz / 60`.

3 Now we will obtain exact latitude and longitude by using the formula in step 2

- The number corresponding to N or S is taken as latitude: `1,054.1073`.
- *Latitude:* `10 + 54.73 / 60 = 10.91216`.
- The number corresponding to E or W is taken as longitude: `07653.7800`.
- *Longitude:* `076 + 53.7800/60 = 76.8963`

4 The location is 10.91216, 76.8963 these coordinates are enough in detecting the location of the vehicle. By this logic from NMEA data, location coordinates can be derived.

For performing the above logic with ease, we used the following code.

```
if ‘GPRMC’ in data_gps:
    data_gps = data_gps.strip(‘\n’)
    data_gps = data_gps.strip(‘\r’)
    parsed.append(data_gps)
for x in range(0,len(parsed)):
    msg = pynmea2.parse(parsed[x])
    latit = msg.latitude
    longi = msg.longitude
```

After getting the latitude, longitude points we can directly get the address corresponding to the coordinate points in the form of an array from the Google maps by using the following commands:

```
try:
    poa = str(geolocator.reverse(loc))
    poa = poa.split(‘,’)
    for i in range(len(poa)):
        try:
            poa[i] = int(poa[i])
        except (ValueError or TypeError):
            continue
except GeocoderTimedOut as e:
    print(“Error: geocode failed on input with message”)
```

The Google map link can be obtained by following procedure.

The main template of Google maps is same for any coordinate but the x, y mentioned below should be changed,

<<`https://www.google.co.in/maps/@xxxxxxxx,yyyyyyyy,17.21z`>>

where

x – represents the latitude point

y – represents the longitude point.

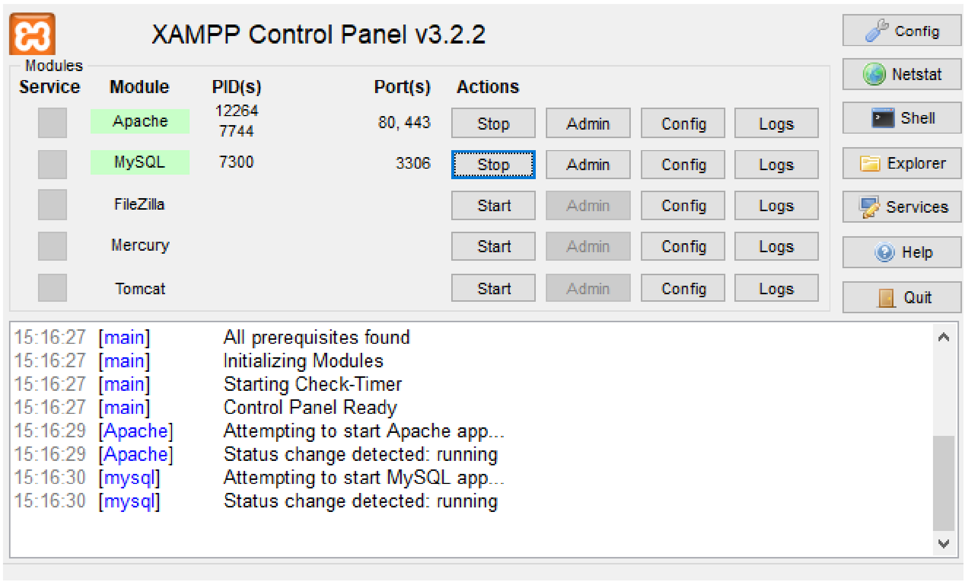
4.2 Uploading to the website

Our proposed system main aim is to convey the collision information to the hospitals. In case of hospitals conveying the information is more convenient in the form of a web page rather than in the form of mobile application. Now, we are provided with the parameters like pin code, latitude, longitude, timestamp, severity level and Google maps link. But, conveying this information to the hospitals is the crucial part. For the prototype purpose, we upload the data from the python script to a local database which will be hosted on a webpage using a PHP code.

4.2.1 Starting the server

In this prototype, we used Xampp for making a local server and database. From the Figure 6, we can see that Xampp Control Panel is used in starting Apache Server and MySQL database.

Figure 6 Xampp control panel for starting apache, MySQL modules (see online version for colours)



Modules

Service	Module	PID(s)	Port(s)	Actions			
	Apache	12264 7744	80, 443	Stop	Admin	Config	Logs
	MySQL	7300	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
15:16:27  [main]      All prerequisites found
15:16:27  [main]      Initializing Modules
15:16:27  [main]      Starting Check-Timer
15:16:27  [main]      Control Panel Ready
15:16:29  [Apache]    Attempting to start Apache app...
15:16:29  [Apache]    Status change detected: running
15:16:30  [mysql]     Attempting to start MySQL app...
15:16:30  [mysql]     Status change detected: running
```

4.2.2 Uploading to the database

For the connecting python script to the database named `accident_inf` and erase the previous data the below code is used:

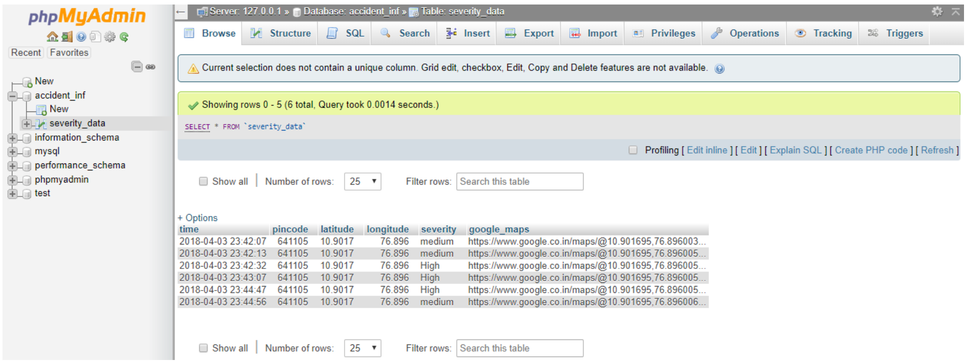
```
connection = pms.connect("localhost", "root", "",
"accident_inf")

curs = connection.cursor()

curs.execute("TRUNCATE TABLE severity_data")
```

Here the database name is `accident_inf`, the Table name is `severity_data` which can be seen in Figure 7. The below code is used in sending the data of the collision from the python script to the database and auto refreshing the database for uploading it to the webpage:

```
if (msg == 1 and sever != ‘no impact’):
    curs.execute(“INSERT INTO severity_data (time, pincode,
    latitude, longitude, severity, google_maps)
    VALUES(‘“+str(time_now)+”‘, ”+str(poa[val])+”, ”
    +str(latit)+”, ”+str(longi)+”, ‘”+str(sever)+”‘,
    ‘”+str(“https://www.google.co.in/maps/@”+str(latit)+”,
    ”+str(longi)+”, 17.21z”)+”‘);”)
    connection.autocommit(True)
```

Figure 7 shows the format in which the data from the python script is stored in the database.

Figure 7 Data obtained from python stored in the local database (see online version for colours)



Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 5 (6 total, Query took 0.0014 seconds.)

```
SELECT * FROM `severity_data`
```

time	pincode	latitude	longitude	severity	google_maps
2018-04-03 23:42:07	641105	10.9017	76.896	medium	https://www.google.co.in/maps/@10.901695,76.896003...
2018-04-03 23:42:13	641105	10.9017	76.896	medium	https://www.google.co.in/maps/@10.901695,76.896005...
2018-04-03 23:42:32	641105	10.9017	76.896	High	https://www.google.co.in/maps/@10.901695,76.896005...
2018-04-03 23:43:07	641105	10.9017	76.896	High	https://www.google.co.in/maps/@10.901695,76.896005...
2018-04-03 23:44:47	641105	10.9017	76.896	High	https://www.google.co.in/maps/@10.901695,76.896005...
2018-04-03 23:44:56	641105	10.9017	76.896	medium	https://www.google.co.in/maps/@10.901695,76.896006...

4.2.3 *Web page*

For designing the web page HTML, CSS code is used and for uploading the data from the database to the web page PHP code is used.

For designing the columns, colours of the webpage following HTML, CSS code is used:

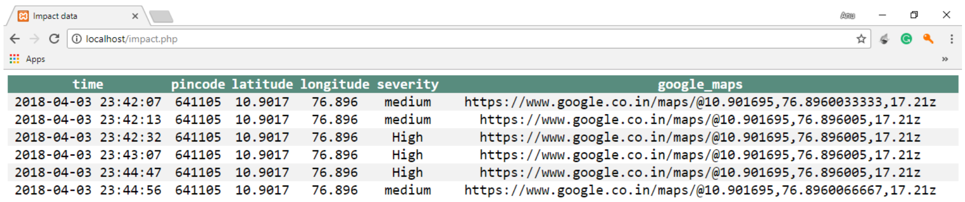
```
<html>
<head>
   <title>Impact data</title>
<style>
        table {
          border-collapse: collapse;
          width: 100%;
          colour: #000000;
          font-family: monospace;
          font-size: 20px;
          text-align: centre;
          }
      th {
        background-colour: #588c7e;
        colour: white;
          }
      tr:nth-child(even) {background-colour: #f2f2f2}
    </style>
</head>
<body>
    <table>
    <tr>
      <th>time</th>
      <th>pincode</th>
      <th>latitude</th>
      <th>longitude</th>
      <th>severity</th>
      <th>google_maps</th>
    </tr>
</table>
</body>
</html>
```

For uploading the data from the database to the webpage following PHP code is used:

```
<?php
$conn = mysqli_connect("localhost", "root", "", "accident_inf");
  if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
  }
  $sql = "SELECT
  time,pincode,latitude,longitude,severity,google_maps FROM
  severity_data";
  $result = $conn->query($sql);
  if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
    echo "<tr> <td>" . $row["time"] . "</td> <td>" .
    $row["pincode"] . "</td> <td>" . $row["latitude"] . "</td>
    <td>" . $row["longitude"] . "</td> <td>" . $row["severity"]
    . "</td> <td>". $row["google_maps"] . "</td></tr>";
}
echo "</table>";
} else { echo "0 results"; }
$conn->close();
?>
```

Figure 8 shows the webpage in which collision data is displayed which get uploaded data from the database using the above two set of PHP, HTML codes.

Figure 8 Collision data uploaded from database to the webpage (see online version for colours)



time	pincode	latitude	longitude	severity	google_maps
2018-04-03 23:42:07	641105	10.9017	76.896	medium	https://www.google.co.in/maps/@10.901695,76.8960033333,17.21z
2018-04-03 23:42:13	641105	10.9017	76.896	medium	https://www.google.co.in/maps/@10.901695,76.896005,17.21z
2018-04-03 23:42:32	641105	10.9017	76.896	High	https://www.google.co.in/maps/@10.901695,76.896005,17.21z
2018-04-03 23:43:07	641105	10.9017	76.896	High	https://www.google.co.in/maps/@10.901695,76.896005,17.21z
2018-04-03 23:44:47	641105	10.9017	76.896	High	https://www.google.co.in/maps/@10.901695,76.896005,17.21z
2018-04-03 23:44:56	641105	10.9017	76.896	medium	https://www.google.co.in/maps/@10.901695,76.8960066667,17.21z

5 Conclusions and future enhancements

The way IoT has emerged in the recent past in all sectors is amazing and in this paper we exploited the power of IoT and sensors (Shah et al., Forthcoming; Anudeep et al., Forthcoming; Giridhararajan et al., Forthcoming). Also, it is observed that the domains that IoT can be used is very vast and applications are developed in every possible domain with IoT (Datta et al., 2017; Velusamy et al., 2013). Our system is developed in order to detect the crashes and certainly to measure the percentage of the severity so that hospital authorities get alerted to the severity level and the needy is rescued in time. The system is developed with considering Indian and similar conditions. One would also appreciate that, the system is built with the costing 41 USD (Arduino Uno – 5 USD, collision sensor – 5 USD, FSR – 11 USD, and GPS sensor – 20 USD).

The system needs enhancements to be done in aspects like the accuracy of the sensors used and predominantly a safety system must be integrated so that victim can resist the injury before the medical aids reach him. This system is accurate in notifying the data like location and severity of the accident and tested for the foolproof. The system needs to be calibrated with various crash scenarios in order to increase the precision by adding more FSRs and thresholding the data.

References

Anudeep, J., Kowshik, G., Krishna, P.V., Shah, I. and Vasudevan, S.K. (Forthcoming) ‘An inventive and innovative system to detect fall of old aged persons – a novel attempt with IoT, sensors and data analytics to prevent the post-fall effects’, *International Journal of Medical Engineering and Informatics*, Inderscience Publishers, Scopus, Gale indexed, ISSN online: 1755-0661 ISSN print: 1755-0653 (listed in forthcoming articles of IJMEI).

Datta, B.S., Ganapathy, R., Vasudevan, S.K. and Abhishek, S.N. (2017) ‘An inventive and innovative alternate for legacy chain pulling system through internet of things’, *Indonesian Journal of Electrical Engineering and Computer Science*, Vol. 6, No. 3, pp.688–694.

Giridhararajan, R., Vasudevan, S.K., Shah, I., Karthikeyan and Abhishek, S.N. (Forthcoming) ‘A system to prevent toiletry (lavatory) based diseases as Norovirus, Staphylococcus, Escherichia and Streptococcus through IoT and embedded systems’, *International Journal of Advanced Intelligence Paradigms*, Inderscience Publishers (listed in forthcoming articles of IJAIP).

Hannan, M.A., Hussain, A., Mohamed, A. and Samad, S.A. (2008) 'Development of an embedded vehicle safety system for frontal crash detection', *International Journal of Crashworthiness*, Vol. 13, No. 5, pp.579–587 [online] https://www.tandfonline.com/doi/abs/10.1080/13588260802316714 (accessed 15 January 2018).

How Many Car Accidents Are There in the USA Per Day? Posted in Accident Statistics, *Car Accidents on September 18*, *2017* [online] http://branlawfirm.com/many-car-accidents-usa-per-day/ (accessed 18 March 2018).

Le, J., Chou, C.C., Clark, T.N., Tustanowski, R.N. and Barbat, S.D. (2009) *Vehicle side impact Crash Detection for Deployment of Curtain and Side Airbags*, Ford Global Technologies LLC, U.S. Patent 7,484,756.

Ministry of Transport (2017) *Speed*, New Zealand Government [online] http://www.transport.govt.nz/assets/Uploads/Research/Documents/Speed-2017.pdf/ (accessed 25 January 2018).

New '108' Ambulances to bring down Response Time, TNN, 19 August 2017, 13:23 IST [online] https://timesofindia.indiatimes.com/city/coimbatore/new-108-ambulances-to-bring-down-response-time/articleshow/60126988.cms (accessed 15 March 2018).

Shah, I., Vasudevan, S.K. and Sriharsha, P. (Forthcoming) 'Smart and efficient IoT based quality tracking system for perishables – pertaining to Indian conditions', *International Journal of Advanced Intelligence Paradigms*, Inderscience Publishers (listed in forthcoming articles of IJAIP).

Sivakumar, B. (2015) *27% of Deaths in India for Want of Medical Attention*, TNN, updated: 21 October, 15:33 IST [online] https://timesofindia.indiatimes.com/india/27-of-deaths-in-India-for-want-of-medical-attention/articleshow/49474537.cms (accessed 20 January 2018).

Thompson, C., White, J., Dougherty, B., Albright, A. and Schmidt, D.C. (2010) 'Using smartphones to detect car accidents and provide situational awareness to emergency responders', in *International Conference on Mobile Wireless Middleware*, *Operating Systems*, *and Applications*, Springer, Berlin, Heidelberg, June, pp.29–42 [online] https://dl.acm.org/citation.cfm?id=2991440 (accessed 1 February 2018).

Velusamy, K., Venkitaramanan, D., Vasudevan, S.K., Periasamy, P. and Arumugam, B. (2013) 'Internet of things in cloud', *Journal of Engineering and Applied Sciences*, Vol. 8, No. 9, pp.304–313.