Mody University International Journal of Computing and Engineering Research
3 (2), 2019, 98-100
http://www.publishingindia.com/micer

A Study of Automated Vehicular Safety Systems Using OBD II and Raspberry Pi

Siddhanta Kumar Singh[1*] and Ajay Kumar Singh[2]

[1]Department of Computer Science & Engineering, SET, Mody University of Science and Technology, Lakshmangarh, Sikar, Rajasthan, India. Email: singhsiddhant@yahoo.com
[2]Department of Computer Science & Engineering, SET, Mody University of Science and Technology, Lakshmangarh, Sikar, Rajasthan, India.
Email: aksingh.cet@modyuniversity.ac.in
*Corresponding Author

Abstract: The road accidents are on the rise on Indian roads and it is a serious problem for the public and therefore it has become a burning issue in almost all the countries including India. There are many factors for road accidents such as vehicle's internal electro-mechanical system problem as well. To solve this problem, OBD II port is used to diagnose the vehicle's various health conditions which connects the CAN bus with an OBD II bridge construction and a smartphone using Bluetooth, WIFI or USB.

Keywords: CAN, DTC, ECU, ELM327, OBD II, pyOBD.

I. Introduction

According to Govt. of India, Ministry of Road Transport and Highways, the statistical figures for accidents on the roads, injuries and fatalities due to various road accidents is shown in the Fig. 1 [1], and this number of casualties continues to worsen year after year. There were 151417 death cases and 469418 cases of injuries reported out of total 469418 cases of road accidents that took place in 2018. A rise in 46% of the road accident was observed as compared to the last year and the fatality rate went up by 2.37% but the number of persons injured reduced by 0.33%.

	Total Number of Road Accidents (in numbers)	%change	Total Number of Persons Killed (in numbers)	%change	Total Number of Persons Injured (in numbers)	%change
2014	4,89,400		1,39,671		4,93,474	
2015	5,01,423	2.46	1,46,133	4.63	5,00,279	1.38
2016	4,80,652	-4.14	1,50,785	3.18	4,94,624	-1.13
2017	4,64,910	-3.28	1,47,913	-1.90	4,70,975	-4.78
2018	4,67,044	0.46	1,51,417	2.37	4,69,418	-0.33

Fig. 1: Road Accident Cases

Also the trends of road accidents, deaths, and injuries are shown in the following Fig. 2 [1].

These are the reports which are published officially by Govt. of India about road accidents, but there are huge number of accidents that also go unreported across the length and breadth of India.

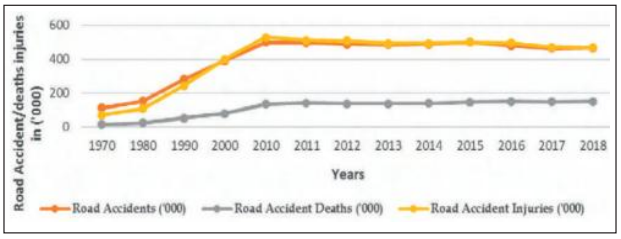

Fig. 2: Trends of Road Accidents

The Controller Area Network i.e. CAN [2] bus is used as an internal LAN's communications network in a vehicle. The outside system can not receive the relevant internal information of the engine from the internal LAN's network in vehicle. In this work, we use the OBD II port as shown in Fig. 3 [3] to construct an intercommunication network between vehicle's internal LAN's communication and external network. The CAN bus is the modern protocol which is added to the OBD II specification and it is made compulsory for all the vehicles manufactured after 2008 and newer models. The system can read various vehicular diagnostic data like Speed, MAF, RPM, Oxygen level, Water temperature, Coolant etc. in real time, which can be processed and monitored efficiently to decrease the risk of major road accident occurrences. Moreover, we can track immediate vehicular location along with date and time which can effectively help to find the position of lost vehicle. The OBD II scanner device is connected into the car's OBD II port which is usually called Data Link Connector (DLC) and it is found under dashboard in the steering wheel side. The

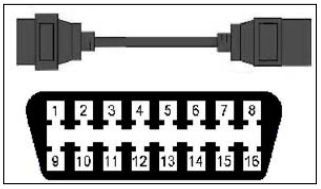

Fig. 3: OBD II Port

The OBD II reads the vehicular data from the ECU and sends the diagnostic data to Raspberry Pi computer. The system helps the automobile's driver to diagnose health of their own vehicles and if any faults are detected in the vehicle by the ECU [4]

unit, the system informs about the problem to the driver of the vehicle so that they can fix the problem for vehicle safety.

On Board Diagnostic (OBD II port) system is fitted in most cars and light trucks today. The vehicle companies used electronics systems to diagnose vehicular problems and control functions of engines in seventies and early eighties. This was primarily used to comply with EPA emission norms. But OBD II standard is getting more and more sophisticated through the years and it provides control on engines and monitors various parts of the vehicle including the network. Now all the cars which are manufactured after 1996 in the US, early 2000 in Europe and 2013 in India must have OBD II port fitted in it. So the car manufacturers have begun incorporating OBD II port under dashboard in various models. Many of the cars which were manufactured earlier are not 100% compliant to OBD II port. The ECU stores the Diagnostic Trouble Codes (DTC) in the memory of vehicle. The ECU generates a string of length five known as trouble codes and store it in its memory whenever it encounters any problem in the vehicle. The trouble code would help one to identify the type of problem and fix the issue within the car. The DTC may be manufacturer dependent or it may be generic as well. The DTC codes generated and stored are unique to the type of problem (e.g. Air to Fuel ratio is imbalanced). By reading these DTC codes, one can pinpoint a particular problem and provide the user with an information to guide as where the fault might have occurred in the vehicle like Body, Powertrain, Chassis or Network. The DTC codes are to be used along with the vehicle's service manual to find which part of the system, subsystem, circuits or components should be tested to fully diagnose the problem with a car code reader or OBD II reader. Each trouble code consists of a string of five characters, out of which the first one is a letter and the rest are four digits such as P1212.

There are four categories of OBD II diagnostic trouble codes [5] which are shown in Fig. 4.

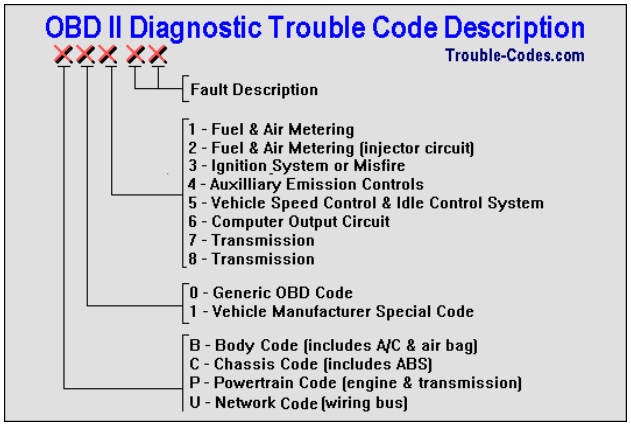

Fig. 4: DTC Codes

Body (B): It provides information about interior of the vehicle. The Body category provides information about the driver assistance, comfort, convenience and safety of the vehicle.

Chassis (C): It covers all the relevant functions which are related to exterior of the passenger compartment of the vehicle. The chassis includes all the mechanical subsystems external to the car like brakes, clutch, steering, suspension etc.

Powertrain (P): It provides information from power generating units and its related components.

Network and Vehicle Integration (U): It provides information about network connectivity and ECU.

To understand the DTC code, we read the leftmost character of the DTC code which marks the system related to the trouble codes. The OBD II scanner connects to the OBD II port and reads the diagnostic trouble codes from the cars computer system. The OBD II scan tool has the standard pinout system.

Pin 1: It is used by manufacturer

Pin 2: It is used by SAE J1850 PWM and VPW

Pin 3: It is used by Manufacturer

Pin 4: GND

Pin 5: GND

Pin 6: It is used by ISO 15765-4 CAN

Pin 7: K-Line of ISO 9141-2 and ISO 14230-4

Pin 10: It is used only by SAE J1850 PWM

Pin 14: It is used by ISO 15765-4 CAN

Pin 15: K-Line of ISO 9141-2 and ISO 14230-4

Pin 16: Power

We use Bluetooth Dongle to connect to Raspberry Pi which connects to the OBD II port using bluetooth. First of all, we install the Raspbian Operating Systems in the Raspberry Pi. After that we use the pyOBD [6] which is available for free of cost and it is an OBD II compliant scanning tool that is written entirely in Python. pyOBD is specially designed to interface with OBD II scanner. It has many API's that allow the programmer to interact with their vehicle's ECU and read and display the various diagnostic trouble codes and process it according to the fault codes.

II. Design Process

The system uses the iSaddle OBD II bluetooth scanner tool for reading the ECU data which is shown in the Fig. 5.

Fig. 5: iSaddle OBD II Scanner

The iSaddle scanner collects the diagnostic data from the memory of ECU and then it sends to the Raspberry Pi via Nano Bluetooth Dongle using Bluetooth communication. We use python language for processing the diagnostic data. For this, we use the open source pyOBD tool which is an written entirely in Python. It is used to interact with iSaddle/ELM 327 OBD II diagnostic device. It basically allows us to interface with car's ECU unit which is used to read and display the fault codes and also reads various real time parameters. The pyOBD displays data based on the OBD II DTC codes after the diagnostic data is sent to the Raspberry Pi as shown in Fig. 6 [7].

Fig. 6: Raspberry Pi

Node.JS [8] is used as the server side language to process the collected diagnostic data from the ECU memory. The Node. JS is a server side programming language which outperforms the classical PHP language in terms concurrency condition. This data is then encapsulated to MongoDB [9]. The MongoDB provides auto-sharding and rich queries. It provides replication and high availability. The users' mobile phone can use the JSON objects stored in MongoDB. The Node.JS then processes the JSON objects into ELM-USB command list. This lists are sent to iSaddle OBD II device using Bluetooth as shown in Fig. 7.

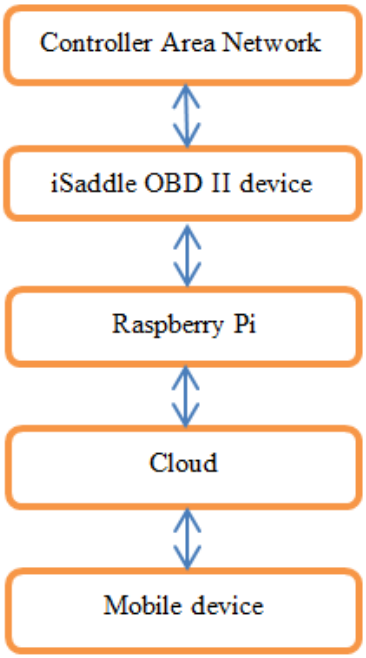

Fig. 7: OBD II Design Process

III. Conclusion

This intelligent vehicular safety system for driving safety supporting system has been proposed here. This system can achieve driving safety for reducing major road accidents occurrences. This system proposes a technique to study and analyze the diagnostics from vehicle which is connected to OBD II port and process the diagnostic data with Raspberry Pi. In the future research, we can extend this model to monitor driving habits to resolve any accidental issues.

References

[1] Govt. of India, Delhi, India. *Road accidents in India-2018*. Accessed: September 2019. [Online]. Available: https://morth.nic.in/sites/default/files/Road_Accidednt.pdf

[2] G. Signoretti, M. Silva, A. Dias, I. Silva, D. Silva, and P. Ferrari, "Performance evaluation of an edge OBD II device for industry 4.0," *II Workshop on Metrology for Industry 4.0 and IoT*, June 2019.

[3] P. R. Sawant, and Y. B. Mane, "Design and development of on-board diagnostic (OBD) device for cars," *4th International Conference on Computing Communication Control and Automation*, August 2018.

[4] Md. S. Ul Alam, "Securing vehicle electronic control unit (ECU) communications and stored data," *Thesis*, September 2018.

[5] J. V. Miniaga, S. R. Manalu, D. A. Hadipurnawan, and F. Sahidi, "Diagnostics vehicle's condition using OBD II and raspberry Pi technology: Study literature," *Journal of Physics Conference Series*, vol. 978, no. 1, March 2017.

[6] E. Husini, "Driving and fuel consumption monitoring with internet of things," *International Journal of Interactive Mobile Technologies*, vol. 11, no. 3, April 2017.

[7] A. Umera, and B. Bahu, "IOT based theft detection using raspberry Pi," *International Journal of Advance Research, Ideas and Innovations in Technology*, vol. 3, no. 6, pp. 131-134, 2017.

[8] H. Shah, and T. R. Soomro, "Node.js challenges in implementation," *Global Journal of Computer Science and Technology: E Network Web & Security*, vol. 17, no. 2, pp. 72-83, June 2017.

[9] R. Kaur, and J. K. Sahiwal, "A review of comparison between NoSQL databases: MongoDB and CouchDB," *International Journal of Recent Technology and Engineering*, vol. 7, no. 6S, pp. 892-898, March 2019.