IECE Transactions on Intelligent Systematics
http://dx.doi.org/10.62762/TIS.2024.329126

RESEARCH ARTICLE

A Cyber-Physical System Based on On-Board Diagnosis (OBD-II) for Smart City

Syed Haider Ali[1], **Inam Ullah**[2,*], **Syed Ashraf Ali**[3], **M Ihtisham UL Haq**[4] **and Niamat Ullah**[5]

[1] Pakistan Council of Scientific & Industrial Research, Peshawar, Pakistan
[2] Department of Computer Engineering, Gachon University, Seongnam 13120, Republic of Korea
[3] Iqra National University Peshawar, Pakistan
[4] University of Agriculture Peshawar, Peshawar, Pakistan
[5] Department of Computer Science, University of Malakand, Pakistan

Abstract

This paper proposes designing and structuring a Cyber-Physical System (CPS) with a specific focus on vehicles equipped with on-board diagnosis (OBD-II). The purpose of the CPS is to collect and assess data pertaining to the vehicle's Electronic Control Unit (ECU), such as engine RPM, speed, and other relevant parameters. The OBD-II scanner utilizes the obtained data on mass airflow (MAF) and vehicle speed to compute CO_2 gas emissions and fuel consumption. The data is wirelessly communicated using a GSM module to a Semantic Web. The CPS also uses GPS tracking to ascertain the vehicle's whereabouts. A Semantic Web is utilized to construct a database management system that stores and manages sent data. A graphical user interface (GUI) is created to facilitate data analysis. It undergoes a sequence of qualification tests to verify the system's functionality. The results demonstrate that the system can accurately read parameters, process data, transfer information, and display readings.

Keywords: Automobiles, Intelligent Vehicle, Microcontroller, Cyber-Physical System, Embedded System.

Academic Editor:
Chunxi Yang

Submitted: 24 August 2024
Accepted: 13 September 2024
Published: 20 September 2024

Vol. 1, **No.** 2, 2024.
10.62762/TIS.2024.329126

***Corresponding author:**
✉ Inam Ullah
inam@gachon.ac.kr

Citation
Ali, S. H., Ullah, I., Ali, S. A., Haq, M. I. U, & Ullah, N. (2024). A Cyber-Physical System Based on On-Board Diagnosis (OBD-II) for Smart City. IECE Transactions on Intelligent Systematics, 1(2), 49–57.

1 Introduction

Cyber-physical systems (CPS) have recently become increasingly popular and have drawn the attention of researchers, engineers, and entrepreneurs. One particularly interesting use is vehicle monitoring. Research on real-time vehicle monitoring systems and driver behavior analysis has increased significantly in recent years [1].

The online monitoring systems provide access to various data, including diagnostic data, speed, engine RPM, position, and measurements from many engine sensors [2, 3]. A remote vehicle health monitoring system enables vehicle owners to protect themselves and their vehicles by providing access to historical and current diagnostic data. Automatic problem detection and reporting to automotive specialists is a key feature of this system [4]. The cloud-based architecture of this system facilitates the ability to access data remotely using cell phones or PCs. The engine and power transmission can be inspected during the vehicle's operation in a different location.

On-board diagnosis (OBD-II) devices [5] gather engine parameters and GPS data and communicate them to other microcontroller-based devices and a cloud server. A cloud server is a platform that facilitates online data transmission and processing, providing the visualization of information in graphical and data formats [6].

A microcomputer known as Raspberry Pi runs on the open-source Linux operating system [7]. Raspberry Pi 4B, introduced in 2019, features an integrated Wi-Fi module for wireless internet access. Several advantages of Raspberry Pi can be identified when comparing it with other microcontrollers like Arduino [8]. The suggested method recommends utilizing a Raspberry Pi for safety monitoring and vehicle location tracking [9]. The system includes the RFID, GSM, and GPS modules. However, the publication lacks implementation specifics and merely presents a system diagram. Furthermore, the system does not use OBD-II. The use of a Raspberry Pi with an OBD-II tool in a vehicle's diagnostic system is described in [10]. The system has a Bluetooth dongle for communicating with other Bluetooth devices, but it cannot upload data to a remote server for monitoring.

The authors describe a vehicle monitoring system that uses an OBD-II scan tool [11]. The SYM32 MCU controls the Wi-Fi module, takes in information from an OBD scan tool, and wirelessly sends it to other devices. This system presents and visualizes information on Bluetooth and Wi-Fi technology devices.

The work in [12] uses OBD-II and GPS data to predict driving behavior but does not specify the implementation of transmitting results online. There is a lack of Raspberry Pi and online data display specifics in the articles. There is a real need for comprehensive implementation instructions; knowledge of these specifics would assist working engineers and anyone interested in the subject. The key goals of our research are as follows:

1. A cost-effective CPS has been developed to monitor a vehicle's electronic control unit (ECU) data.
2. GPS coordinates complement the ECU data within the system to facilitate data analytics.
3. Creating an intelligent transportation systems-related Semantic Web platform involves integrating advanced technologies to facilitate smart mobility.
4. This platform aims to optimize transportation networks, enhance safety, and improve overall efficiency through data-driven insights and real-time monitoring.

The article's structure continues: Section 2 presents the related work, while Section 3 details the module methodology. Section 4 presents the results and analysis related to it. Finally, Section 5 concludes this work with future directions.

2 Related Work

Various OBD-II logging devices are available, some of which are included in Table 1. For our research, we needed a low-cost device that fulfilled all the requirements in the table. The most affordable devices were the Arduino platform and the Bluetooth scanner. Many new projects have emerged in OBD-II loggers in the last few years.

A low-cost fleet monitoring prototype model was developed [13]. The model involves attaching sensors and ECUs to each vehicle in the fleet to collect various parameters and location details. The communication between the vehicles' CAN-bus and the Raspberry PI 3 is achieved using PICAN2. The connection between PICAN2 and a vehicle's OBD-II is established through OBD-II cables. To obtain location details, the prototype utilizes the Adafruit Ultimate GPS module. Through a cellular network, the Hologram Nova makes communicating easier for the Raspberry PI and the server. The fleet management data is visualized through a web application. The Python programming language collects fleet data, and Raspbian OS is the platform. The database is MongoDB, and Node.js is utilized to operate the fleet management service.

The suggested solution is designed to facilitate predictive maintenance within the military environment by using the data obtained from onboard diagnostics. Using statistical techniques and diagnostic signals from the CAN-bus, it also integrates proactive maintenance [14]. The European OBD-II standard monitors many aspects of fuel and air intake. This includes metrics such as fuel injection pressure, intake air temperature, ignition advance, intake air quality, and exhaust settings for control, like lambda sensors. The analysis incorporates various standard operational data inputs, such as vehicle speed, engine RMP, oil condition, and coolant temperature. The system also monitors various aspects such as braking, safety, transmission, brake pads, brake fluid, spark plug conditions, active chassis, and motor oil quality. These parameters are key characteristics of the proposed system.

Several methods have been proposed to measure and minimize pollution, with the transportation sector accounting for 29% of GHG emissions. An approach that uses machine learning techniques and the OBD-II data from the vehicle to estimate pollution levels was proposed [15]. They observed that vehicle RPM and speed were positively correlated with CO_2 emissions. Their study concluded that the driving pattern impacts GHG emissions.

For fleet management and driver behavior analysis, Türk and Challenger [16] designed an Internet of Things-based solution. This system collects sensor data, including fuel consumption, speed, brake use, RPM, and steering angle, by connecting to the vehicle's onboard diagnostics interface. The data and the driver's details are uploaded to a server for future analysis. The system also includes analysis software to analyze driver behavior and vehicle status thoroughly. The authors [17] suggested a system for tracking and diagnosing automotive issues using an Arduino Uno and OBD-II. This system enables proactive vehicle maintenance to prevent accidents.

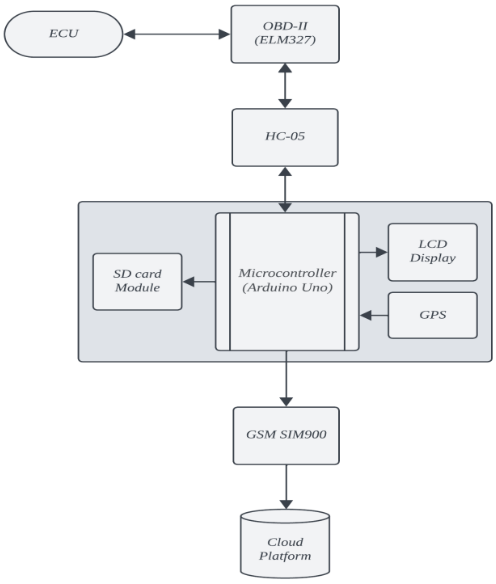


Figure 1. Block diagram of Cyber-Physical Systems.

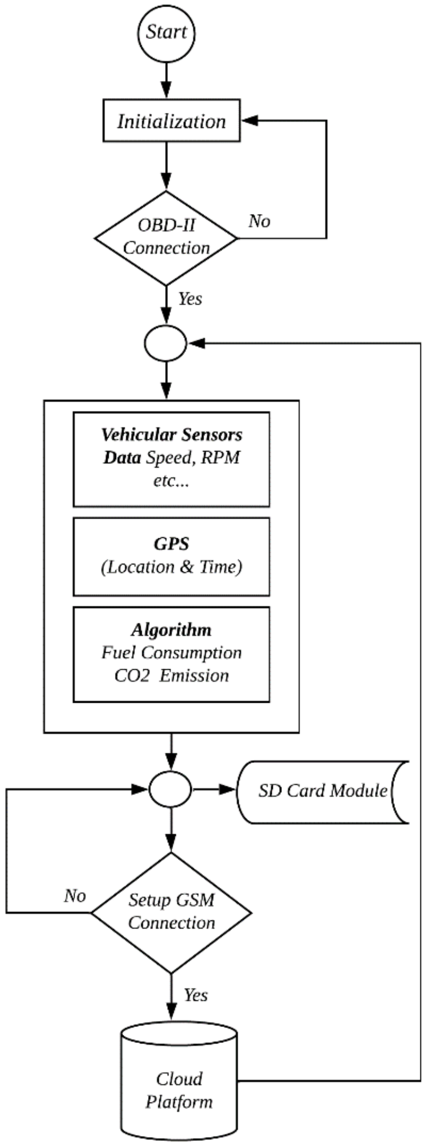


Figure 2. The Cyber-Physical System flowchart for vehicles.

An OBD-II-interfaced smartphone app was proposed [18] for monitoring vehicle energy consumption and engine performance. This solution's ability to remotely monitor vehicles and create energy-efficient control algorithms for route selection makes it very helpful for fleet management. The mobile app receives an accident alert with location information if the vibration sensor detects an impact above a predefined threshold. [19] developed an IoT-based Car Data Recorder to monitor vehicles and report accidents. They suggested the parts for the system as an Arduino Mega 2560, an HC-05 Bluetooth module, an accelerometer, an ELM327 OBD-II reader, and a GSM module SIM800L. An alert is transmitted via SMS whenever an impact with a magnitude above 4G is detected.

The primary difference between this paper's solution and those previously discussed is that our proposal utilizes a bottom-up strategy. Our research focuses on developing a universal and cost-effective platform to collect data from a vehicle's interior and exterior. Because of its open and scalable architecture, intelligent transport services can be implemented on this platform.

3 Methodology

The proposed system utilizes OBD-II technology and the Semantic Web to provide a low-cost, real-time

Table 1. Comparison of OBD Loggers.

Technology	Stand-Alone	GSM-Capable	Config-urable	GPS	Data Recorder	CAN
OBD-II Scanner (Bluetooth)	×	✓	✓	✓	✓	×
OBD-II Scanners	✓	×	×	×	×	✓
OBD Mini Logger [20]	✓	×	×	✓	✓	×
Car Chip Pro [21]	✓	×	×	✓	✓	×
CarTwin Data Logger [22]	×	✓	×	×	✓	✓
DashDyno SPD ProPack [23]	✓	×	×	×	✓	✓
IOSiX OBD-II Datalogger [24]	✓	×	×	×	✓	×
Proposed CPS	✓	✓	✓	✓	✓	✓

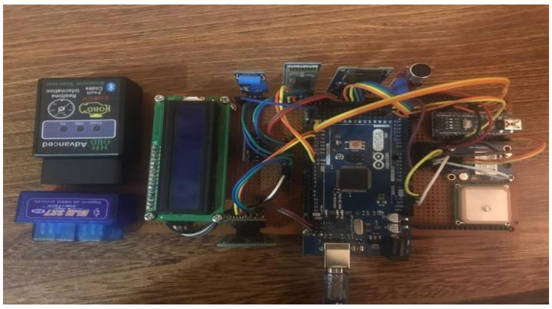
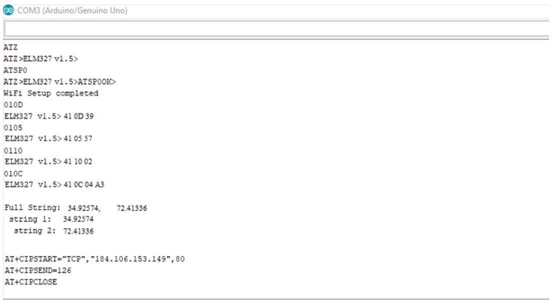


Figure 3. Overview of the proposed prototype.



```
ATZ
ATZ>ELM327 v1.5>
ATSP0
ATZ>ELM327 v1.5>ATSP0OK>
WiFi Setup completed
010D
ELM327 v1.5> 41 0D 39
0105
ELM327 v1.5> 41 05 57
0110
ELM327 v1.5> 41 10 02
010C
ELM327 v1.5> 41 0C 04 A3

Full String: 34.92574,    72.41336
 string 1:  34.92574
  string 2: 72.41336


AT+CIPSTART="TCP","184.106.153.149",80
AT+CIPSEND=126
AT+CIPCLOSE
```

Figure 5. A snapshot of the Test Program.

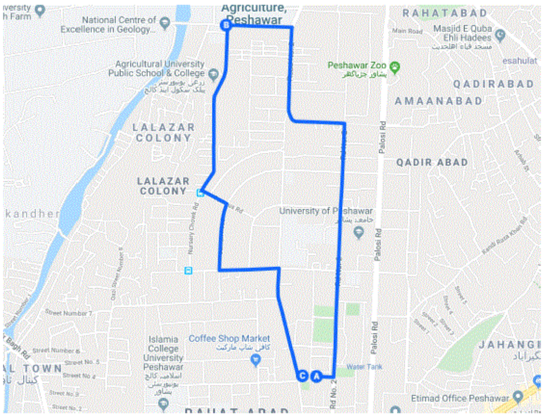


Figure 4. Route of Travel.

CPS. Vehicle owners can access historical and real-time vehicle data through the Semantic Web. The system allows users to view real-time charts displaying OBD parameter variations and provides live location tracking on a Google map. Furthermore, a Semantic Web maintains trip-related information, including fuel consumption, CO_2 emissions, and duration. The system can notify users of problems and maintenance needs via the cloud server.

3.1 OBD Scanner and On-Board Diagnostics

Since 1996, automobile manufacturers have initiated the integration of electronic sensors into automobiles, commonly referred to as the ECU. The primary objective was to monitor car sensors to increase fuel efficiency and decrease pollution. OBD was an additional advantage of ECU that enabled the diagnosis and logging of vehicle data for technical evaluation. OBD and ECU are connected via DLC, and PID codes capture data from vehicle sensors.Interoperability problems emerged in OBD's early days since no universal standard had been established. The OBD-II standard interface was suggested to solve this issue [25]. The industry has now adopted several protocols as standards, with the main difference being the placement of communication pins, such as ISO 9141-2, ISO 14230 KWP2000, ISO 15765 CAN, SAE J1850 PWM, and SAE J1850 VPW. The sensor data for this study is collected by connecting the sensor node to the test vehicle's ECU using the ELM327 OBD-II scanner. The sensor node and the ELM327 scanner are connected via the HC-05 Bluetooth module, as seen in Figure 1.

3.2 Sensor node

The Arduino Uno ATmega328P is the compute board for the sensor node shown in Figure 1. This microcontroller is based on the AVR architecture and operates at 16MHz. It has 32 KB of flash memory and offers 14 digital and six analog I/O pins for connecting sensors and peripherals [26]. The sensor node integrates a Bluetooth HC-05 module to facilitate connection with the ELM327 OBD-II scanner. Using

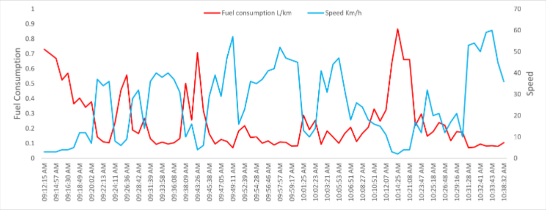


Figure 6. Fuel consumption and speed.

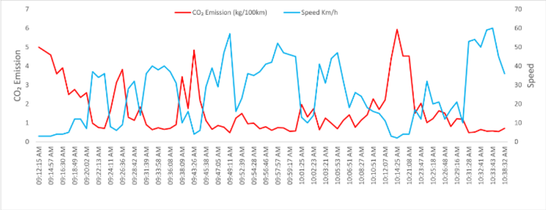


Figure 7. Carbon dioxide (CO_2) Emission and Speed.

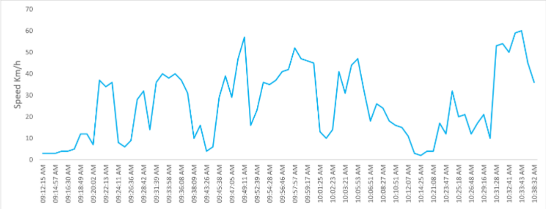


Figure 8. Speed of the Vehicle.

this module, a wireless connection can be established between the sensor node and the OBD-II scanner. The parameters of vehicle sensors are gathered by the OBD-II scanner ELM327 at 12-sec intervals. These parameters include coolant temperature, RPM, speed, and MAF (Mass Air Flow) of the vehicle. The data from these sensors is essential for determining the vehicle's fuel efficiency and CO_2 emissions, as explained in Section 3.3.

Integrating a GPS module (NEO-6M) enables the vehicle's sensor data to be geo-referenced to the road network, providing essential information such as GPS coordinates and timestamps. These GPS-based references and timestamps are critical in evaluating numerous trip variables such as fuel consumption, journey time, and CO_2 emissions with precise locations.

The sensor node incorporates a GSM module that transmits the measured parameters (RPM, speed, MAF, etc.) to ThingSpeak, a free and open-source Semantic Web. The values are transmitted to the Semantic Web and stored on an SD card module, as shown in Figure 1. In the case of a communication interruption and unavailability of Wi-Fi access, this local logging is executed to ensure data integrity.

3.3 Implementation and Algorithm

The proposed method is implemented in C++ within the Arduino IDE and executed on an ATmega328P-based Arduino Uno. Figure 2 shows the program flow of the software for the CPS implemented in the vehicle.

The DLC connector allows the sensor node to interact with the ELM327 OBD-II scanner while driving. This connection is established using Bluetooth. After establishing a connection, the sensor node initiates regular data retrieval from a range of vehicle sensors, such as RPM, speed, and MAF, at intervals of 12 sec. These sensor readings are then used to calculate the fuel consumption and CO_2 emissions for the entire trip. In the next stage, the integrated sensors detect and gather data from the sensed parameters, as shown in Figure 2. This data includes GPS coordinates and time, which are utilized to determine the geographical location, speed, fuel consumption, and CO_2 emissions for various segments of the road network. Further details regarding this process are listed in sections 3.4 and 3.5.

Sections 3.4 and 3.5 mention that the third step involves executing algorithms to estimate fuel usage and CO_2 emissions. The processed data is transmitted to the cloud platform (Thingspeak) using a GSM module. However, as shown in Figure 2, the data is also stored locally on an SD card module to limit the possibility of GSM connectivity interruptions. The duration of programmed execution is 15 sec. This results in four data sets uploaded to the cloud service (ThingSpeak) per minute.

3.4 Consumption of fuel instantly

Fuel consumption can be expressed as miles per gallon (MPG) or liters per km (l/km), the fuel used per travel unit. Various factors, including traffic conditions, vehicle type, driver conduct, and time of day, might affect fuel use. Instantaneous fuel consumption can be calculated using Eq.(1) [27, 28].However, it is important to note that this calculation relies on the availability of the vehicle's Fuel Flow parameter (PID 015E).

The instantaneous fuel consumption C (in $\frac{l}{km}$) can be calculated as the ratio of the fuel flow F (in $\frac{l}{h}$) to the vehicle speed V (in $\frac{km}{h}$). Thus, the fuel consumption is equal to the ratio of the fuel flow to the vehicle speed, as shown in the following formula:

$$C = \frac{F}{V} \tag{1}$$

Figure 9. Speed during Test.

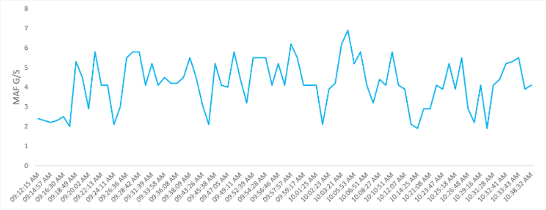


Figure 10. MAF of Vehicle.

However, the Fuel Flow parameter is not available in most vehicles. This could be due to a fuel flow sensor not being there or the manufacturer choosing not to include one between the fuel tank and the engine carburetor. Furthermore, the fuel flow parameter is unavailable on the test vehicle used to validate this research. In this case, Eq. (2) can be used to figure out the Fuel Flow. Fuel Flow can be calculated using the Air-to-Fuel ratio (AFR) (0144) and Mass Air Flow (MAF) (0110), as shown in Eq. (2).

$$\text{Fuel Flow}\left(\frac{l}{h}\right) = \frac{\text{MAF}}{\text{AFR} \times \text{FD}} \times 3600 \tag{2}$$

FD stands for fuel density, equivalent to 820 grams of petrol per cubic decimeter.

3.5 Emissions of Carbon Dioxide

The carbon and oxygen atomic masses can be utilized to calculate the amount of CO_2 created during the combustion of fuel containing carbon. The atomic masses of oxygen and carbon are 16 and 12, respectively. Therefore, the formula for CO_2 may be determined by adding the atomic masses of carbon and two oxygen atoms, resulting in a total of 44. Using this equation, we can calculate the estimated amount of CO_2 created after burning one kilogram of carbon content.

$$\frac{44}{12} \approx 3.67\,\text{kg of}\, CO_2 \tag{3}$$

The system uses the mass air flow sensor's values and the vehicle's speed data to determine the amount of CO_2 emissions. The OBD-II interface is used to read this data. The mass air flow sensor monitors how much air enters the engine's intake.The amount of petrol utilized in gallons per hour (GPH) can be calculated using the value acquired from the MAF sensor. The fuel consumption can subsequently be converted to liters per hour (LPH). Finally, the fuel efficiency in terms of km per liter(KPL) can be determined.

Figure 11. MAF during Test.

$$\text{GPH} = \text{Mass Air Flow} \times 0.0805\left(\frac{g}{h}\right) \tag{4}$$

$$\text{LPH} = \text{GPH} \times 3.75\left(\frac{l}{h}\right) \tag{5}$$

$$\text{KPH} = \frac{\text{Speed}}{\text{LPH}}\left(\frac{km}{l}\right) \tag{6}$$

Depending on the fuel type utilized, different fuels have different amounts of CO_2 emissions per liter. A diesel engine's CO_2 emissions per liter are 2.6 kilograms, compared to 2.3 kilograms in a gasoline engine.

3.6 Semantic Web

ThingSpeak is a free and open-source Semantic Web for processing, storing, and visualizing data. It provides data visualization capabilities without programming and includes integrated MATLAB analytics functionality [29]. Built-in APIs in ThingSpeak allow data to be received and stored from HTTP-based sensor nodes. Moreover, it is capable of using the logged data for data analytics.

4 Results and Analysis

Tests were conducted using a CPS, as shown in Figure 3. The proposed approach was evaluated and tested in a 2013 Toyota passo with an automated gearbox and a 1000CC engine. The driving route through urban areas 5 km is shown in Figure 4. The software on the laptop manages communication with the OBD-II Link device. It was created using Arduino IDE C++, especially

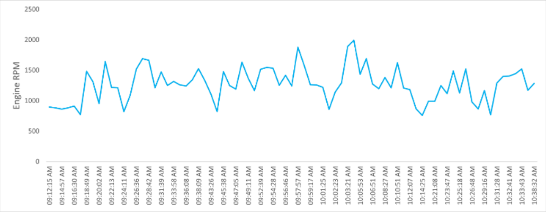


Figure 12. Engine RPM of Vehicle.

Figure 13. Engine RPM during the Test.

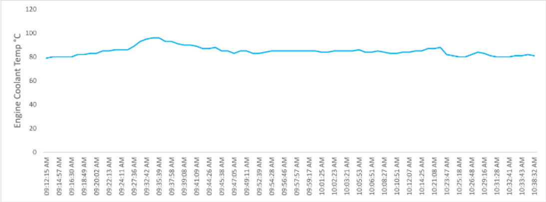


Figure 14. Coolant Temperature of Vehicle.

Figure 15. Collant temperature During Test.

for this research study. Multiple parameters, RPM, vehicle speed, and other essential parameters, were monitored and recorded throughout the designated route. Figure 5 shows the test program used to collect these results.

The capability and performance of the CPS to measure various vehicle parameters, such as RPM, speed, and others, are depicted in Figure 6 through Figure 15. The figure analysis shows that the maximum speed that could be obtained was approximately 60 km per hour. The maximum fuel consumption that could be seen was 0.866 liters per KM, as shown in Figure 6. It is important to note that fuel consumption increased at lower speeds. Furthermore, Figure 7 shows that the maximum CO_2 emission per 100 km was 4.99 kilograms, and the maximum observed speed was 60 km/h. Interestingly, lower speeds were also linked to increased CO_2 emissions.

Furthermore, the graphical representation of all supported OBD-II parameters is presented with the parameters above, as illustrated in Figs. 6-12. At the time of data acquisition, the engine rpm is 1027, as shown in Figure 13, and the speed is 15 km/h, as shown in Figure 9. A temperature of 95 degrees Celsius is also documented for the engine coolant.

The MAF sensor measures the mass flow rate of air entering the engine. This measurement is essential for determining the ideal spark timing and how much fuel to supply. The MAF value in the OBD-II system is given in g/sec and varies depending on the engine model and capacity. Figure 10 shows the MAF values and time stored in a database.

RPM is a measurement of how many times the crankshaft of an engine spins in one minute. This parameter is essential for drivers because it helps them figure out when to transfer gears for maximum fuel efficiency, especially when operating a car with a manual transmission. If the RPM exceeds the vehicle's speed, the engine is idling, which wastes fuel. In vehicles with manual gearboxes, fuel consumption might vary based on driver behavior. The RPM values recorded during the test drive in the Semantic Web are shown in Figure 12.

A vehicle's coolant temperature is a crucial sensor that shows the engine's temperature. Monitoring it is critical since a greater coolant temperature usually signals a higher engine temperature. The CPS periodically uses the PID 05 to retrieve the coolant temperature. The figure below illustrates the continuous monitoring of the values retrieved by the CPS.

5 Conclusion

Testing the OBD-II reader's communication with the Semantic Web confirmed that the system could retrieve sensor readings from an OBD-II compliant vehicle. Tests on a real vehicle showed that the system could measure RPM, speed, coolant temperature, etc. The approach minimizes costs by integrating technologies by eliminating the need to buy many devices with distinct functions. The design also incorporates the capability to measure metrics like fuel consumption, CO_2 emission, and GPS tracking that are not typically available with standard OBD-II interfaces. The system power cut off when the car's engines were turned off, which disabled wireless connectivity and GPS location tracking.

Future work could focus on improving battery backup systems that keep essential functions running even when the vehicle's motors are turned off. This would enable GPS location tracking and continuous wireless connection. To increase overall efficiency, it's also necessary to decrease the delay brought on by system initializations. Due to the current delay in preserving crucial initial parameter measurements, the automobile cannot be driven until all initializations are finished. To reduce this delay, new approaches must be investigated and developed.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Fridman, L., Brown, D. E., Glazer, M., Angell, W., Dodd, S., Jenik, B., ... & Reimer, B. (2019). MIT advanced vehicle technology study: Large-scale naturalistic driving study of driver behavior and interaction with automation. *IEEE Access*, 7, 102021-102038. [CrossRef]

[2] Naik, P., Kumbi, A., Telkar, N., Kotin, K., & Katti, K. C. (2017, December). An automotive diagnostics, fuel efficiency and emission monitoring system using CAN. In *2017 International Conference on Big Data, IoT and Data Science (BID)* (pp. 14-17). IEEE. [CrossRef]

[3] Santa, J., Sanchez-Iborra, R., Rodriguez-Rey, P., Bernal-Escobedo, L., & Skarmeta, A. F. (2019). LPWAN-based vehicular monitoring platform with a generic IP network interface. *Sensors*, 19(2), 264. [CrossRef]

[4] Shafi, U., Safi, A., Shahid, A. R., Ziauddin, S., & Saleem, M. Q. (2018). Vehicle remote health monitoring and prognostic maintenance system. *Journal of advanced transportation*, 2018(1), 8061514. [CrossRef]

[5] BinMasoud, A., & Cheng, Q. (2019, November). Design of an iot-based vehicle state monitoring system using raspberry pi. In *2019 International Conference on Electrical Engineering Research & Practice (ICEERP)* (pp. 1-6). IEEE. [CrossRef]

[6] Silva, M., Vieira, E., Signoretti, G., Silva, I., Silva, D., & Ferrari, P. (2018). A customer feedback platform for vehicle manufacturing compliant with industry 4.0 vision. *Sensors*, 18(10), 3298. [CrossRef]

[7] Mathe, S. E., Pamarthy, A. C., Kondaveeti, H. K., & Vappangi, S. (2022, February). A review on raspberry pi and its robotic applications. In *2022 2nd International Conference on Artificial Intelligence and Signal Processing (AISP)* (pp. 1-6). IEEE. [CrossRef]

[8] Karthikeyan, S., Raj, R. A., Cruz, M. V., Chen, L., Vishal, J. A., & Rohith, V. S. (2023). A systematic analysis on raspberry pi prototyping: Uses, challenges, benefits, and drawbacks. *IEEE Internet of Things Journal*, 10(16), 14397-14417. [CrossRef]

[9] Kumari, M., Kumar, A., & Khan, A. (2020, February). IoT based intelligent real-time system for bus tracking and monitoring. In *2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC)* (pp. 226-230). IEEE. [CrossRef]

[10] Manalu, S. R., Moniaga, J., Hadipurnawan, D. A., & Sahidi, F. (2017). OBD-II and raspberry Pi technology to diagnose car's machine current condition: study literature. *Library Hi Tech News*, 34(10), 15-21. [CrossRef]

[11] Pan, Y. J., Yu, T. C., & Cheng, R. S. (2017, May). Using OBD-II data to explore driving behavior model. In *2017 international conference on applied system innovation (ICASI)* (pp. 1816-1818). IEEE. [CrossRef]

[12] Baek, S. H., & Jang, J. W. (2015). Implementation of integrated OBD-II connector with external network. *Information Systems*, 50, 69-75. [CrossRef]

[13] Ben Othmane, L., Alvarez, V., Berner, K., Fuhrmann, M., Fuhrmann, W., Guss, A., & Hartsock, T. (2018, September). A low-cost fleet monitoring system. In *2018 IEEE International Smart Cities Conference (ISC2)* (pp. 1-2). IEEE. [CrossRef]

[14] Hassan, M. A., Javed, R., Granelli, F., Gen, X., Rizwan, M., Ali, S. H., ... & Ullah, S. (2023, March). Intelligent transportation systems in smart city: a systematic survey. In *2023 International Conference on Robotics and Automation in Industry (ICRAI)* (pp. 1-9). IEEE. [CrossRef]

[15] Dabarera, W. N. S., Jayatilake, N. T., Jayathissa, R. H. N. S., & Weerawardane, T. L. (2022). Towards an IoT based Vehicle Management System for Vehicle Tracking & Vehicle Diagnostics with OBD2 telem. [CrossRef]

[16] Türk, E., & Challenger, M. (2018, May). An android-based IoT system for vehicle monitoring and diagnostic. In *2018 26th Signal Processing and Communications Applications Conference (SIU)* (pp. 1-4). IEEE. [CrossRef]

[17] Wahl, H., Naz, E., Kaufmann, C., & Mense, A. (2016). Simplifying the complexity for vehicle health management system. In *2016 7th International Multi-Conference on Complexity, Informatics and Cybernetics, IMCIC 2016* (pp. 2-6). [CrossRef]

[18] Baghli, L., Benmansour, K., & Djemai, M. (2014, November). Development of a data acquisition and tracking system for vehicles. In *3rd International Symposium on Environmental Friendly Energies and Applications (EFEA)* (pp. 1-6). IEEE. [CrossRef]

[19] Nugroho, S. A., Ariyanto, E., & Rakhmatsyah, A. (2018, May). Utilization of Onboard Diagnostic II (OBD-II) on four wheel vehicles for car data recorder prototype.

In *2018 6th International Conference on Information and Communication Technology (ICoICT)* (pp. 7-11). IEEE. [CrossRef]

[20] OBD Mini Logger [Online]. Available: https://hemdata.com/products/dawn/obd-mini-logger/

[21] CarChip - Model Fleet Pro - Data Logger [Online]. Available: https://www.environmental-expert.com/products/carchip-model-fleet-pro-data-logger-311723

[22] CarTwin Data Logger [Online]. Available: https://cartwin.ai/data-logger/

[23] DashDyno SPD ProPack [Online]. Available: http://www.auterraweb.com/dashdynopropack.html

[24] IOSiX OBD-II Datalogger [Online]. Available: http://caflor.net/datalogger.html

[25] De Rango, F., Tropea, M., Serianni, A., & Cordeschi, N. (2022). Fuzzy inference system design for promoting an eco-friendly driving style in IoV domain. *Vehicular Communications*, 34, 100415. [CrossRef]

[26] Signoretti, G., Silva, M., Dias, A., Silva, I., Silva, D., & Ferrari, P. (2019, June). Performance evaluation of an edge obd-ii device for industry 4.0. In *2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4. 0&IoT)* (pp. 432-437). IEEE. [CrossRef]

[27] Kondaveeti, H. K., Kumaravelu, N. K., Vanambathina, S. D., Mathe, S. E., & Vappangi, S. (2021). A systematic literature review on prototyping with Arduino: Applications, challenges, advantages, and limitations. *Computer Science Review*, 40, 100364. [CrossRef]

[28] Meseguer, J. E., Toh, C. K., Calafate, C. T., Cano, J. C., & Manzoni, P. (2017). Drivingstyles: A mobile platform for driving styles and fuel consumption characterization. *Journal of Communications and networks*, 19(2), 162-168. [CrossRef]

[29] Balakrishna, S., & Thirumaran, M. (2020). Semantic interoperability in IoT and big data for health care: a collaborative approach. In *Handbook of data science approaches for biomedical engineering* (pp. 185-220). Academic Press. [CrossRef]

Syed Haider Ali earned his B.Sc. and M.Sc. in Electrical Engineering from UST Bannu and UET Peshawar, Pakistan, in 2016 and 2019, respectively. He lectured at Swat Polytechnic Institute from 2016 to 2019 and later worked at the National Center for Big Data and Cloud Computing and Real Engineering Services. Currently, he is engaged in additive manufacturing and reverse engineering at PCSIR. His research focuses on embedded systems, cyber-physical systems, UAVs, IoT, wireless communications, and energy harvesting. (engrsyedhaiderali@yahoo.com)

Inam Ullah received a B.Sc. degree in Electrical Engineering from the Department of Electrical Engineering, University of Science and Technology Bannu, Pakistan, in 2016 and a Master's and Ph.D. degree in Information and Communication Engineering from the College of Internet of Things Engineering, Hohai University, China, in 2018 and 2022, respectively. He completed his postdoc with BK21, Chungbuk National University, S Korea, in 2023. He is currently an Assistant Professor at the Department of Computer Engineering, Gachon University, S Korea. His research interests include Robotics, IoT, WSNs, AUVs, AI, Deep learning, etc. He has authored more than 100 articles and five books as an editor.(inam@gachon.ac.kr)

Syed Ashraf Ali earned his BS in Electronic Engineering from BUITEMS, Pakistan, in 2012 and his MS in Electrical Engineering from Iqra National University, Pakistan, in 2017. He is currently pursuing a PhD in Electrical Engineering at Iqra National University. He worked as a Lab Engineer at the same university from 2014 to 2018 and now serves as a Lecturer. His research interests include electronics, wireless communication, renewable energy, power systems, and sustainable energy. (Ashrafali@inu.edu.pk)

M Ihtisham UL Haq received a BS degree in information technology from the University of Agriculture Peshawar, Peshawar, Pakistan, in 2018 and 2022. From 2022 to 2023 Network Assistant at the University of Peshawar, Peshawar, Pakistan. Currently, He is pursuing a Master's in Software Engineering from Northeastern University, Shenyang, China. (ihtishamkhn004@gmail.com)

Niamat Ullah completed a BS in Computer Science from the University of Malakand, Pakistan. His research interests include computer vision, machine learning, and deep learning. (niamatabbas4@gmail.com)