Proceedings of the 7th International Conference on Innovative Data Communication Technologies and Application (ICIDCA-2025)
IEEE Xplore Part Number: CFP25CR5-ART; ISBN: 979-8-3315-1414-3

Application of MQTT Protocol to Visualize the Behavior of Autonomous Vehicle in a Virtual Environment

Vaidehi M
Information Science and Engineering
Dayananada Sagar College of Engineering
Bengaluru,India
vaidehi-ise@dayanandasagar.edu

Vaibhav Magadum
Information Science and Engineering
Dayananada Sagar College of Engineering
Bengaluru,India
vaibhavmagadum859@gmail.com

Sharadhi Hegde
Information Science and Engineering
Dayananada Sagar College of Engineering
Bengaluru,India
sharadhih00@gmail.com

Shravya R
Information Science and Engineering
Dayananada Sagar College of Engineering
Bengaluru,India
shravyar024@gmail.com

Sahaj Gupta
Information Science and Engineering
Dayananada Sagar College of Engineering
Bengaluru,India
sahajguptarocks@gmail.com

2025 7th International Conference on Innovative Data Communication Technologies and Application (ICIDCA) | 979-8-3315-1414-3/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICIDCA66325.2025.11280377

***Abstract* — This study focuses on the design and implementation of an MQTT Protocol to Visualize the Behavior of Autonomous Vehicle in a Virtual Environment using the IoT technology. The primary motive of the study is to stimulate and monitor the behavior of autonomous vehicle in a virtual environment. A virtual environment having vehicle-to-environment interaction. The existing traditional vehicles will soon be replaced by autonomous vehicles due to advancements in the field of IoT in Transport. Hence, there should be a system which allows interaction between vehicles and environment in a connected ecosystem. Based on the changes in the environment the autonomous vehicles should be able to adapt changes. As the existing system completely depends on the Sensor data and Perception Systems but these are limited to a self-environment. The paper focus on building a network in which each vehicle take decision based on the behavior of the other vehicles and choose most efficient path based on the accidents and geofences. This study proposes a system which constantly monitor and sends data such as speed, battery life, GPS location, etc. to a centralized system using various sensors, low powered wireless communication protocol (MQTT) to collect, assess and alert using Internet of Things (IoT). The results demonstrate the feasibility and effectiveness of IoT technology for enhancing Autonomous Vehicle Network. The collected data can be used in implementation of Fleet Management System which help to visualize the performance of the Autonomous Vehicles in form of Graphs and Charts.**

Keywords—IoT, autonomous vehicle network, MQTT protocol, fleet management.

I. INTRODUCTION

A. Motivation

The autonomous vehicle technology is a booming industry which will soon be widely spread across the globe [4], [7]. To enhance the performance of autonomous vehicles, it is essential to enable both vehicle-to-environment (V2E) and vehicle-to-vehicle (V2V) interactions [5], [16]. Despite advances in autonomous vehicle technologies, there is still a significant lack of real-time communication between vehicles and central monitoring environments [3]. The paper emphasizes creating a virtual environment where most of the vehicles are autonomous, replacing traditional vehicles. A vehicle-to-vehicle network is created where each autonomous vehicle is connected to a central server, to which each vehicle shares its data like GPS location, speed of the vehicle, and battery life [8]. The centralized server acts as a hub for any specific company that uses the fleet management system, which works on the data collected from the autonomous vehicles [2], [11].

In this paper, we are using the Termux app to share data such as the user's GPS location, speed, and other details, which will be used to represent the vehicle's location. We are using mobile GPS to act as vehicle location [13] [14]. It's a simple setup: your phone becomes the vehicle, sending its location data to a central server. This server, built with FastAPI, receives the data, processes it, and then instantly updates a live map on your screen using WebSocket's [17]. All data is stored into an SQLite database for a detailed history [18]. This system lets us continuously monitor and display a vehicle's location in real-time without needing any actual hardware.

The data is transmitted using the MQTT protocol [10], [12], [15], which is well-suited for sending small, real-time data packets to show vehicle locations, their status, and any alerts. The frontend interface is built using React and Vite to display vehicle positions, status, and alerts, allowing users to monitor the network in real time. The system also supports fleet management [2], [11], where a central server collects and manages data from multiple autonomous vehicles. This means organizations can track their whole fleet, check each vehicle's performance, and use that information to make smart decisions about improving efficiency and safety [9].

As self-driving vehicles become more common, we need to test their behavior safely and affordably [1]. Real-world testing is too expensive and risky, and it's hard to get a full picture of what's happening [4], [7]. This paper solves that by creating a virtual system that's fast, scalable, and easy to use [6]. By simulating things like GPS tracking, accident alerts, and even the vehicle's dashboard, we're giving developers and fleet managers a safe and powerful way to study and improve how intelligent vehicles behave before they ever hit the road [8], [17]. With modern technologies like IoT [7], [16], lightweight protocols such as MQTT [10], [12], and real-time

frontend tools, it's now possible to create virtual systems that are fast, scalable, and easy to use. This allows us to monitor and test autonomous vehicle behavior [3]. By simulating GPS tracking, accident alerts, and dashboard displays, this paper gives developers, researchers, and fleet managers a safe and useful tool to study and improve how intelligent vehicles behave [2], [9].

B. Contributions

The novelty of this work lies in the design of a lightweight and cost-efficient virtual testbed that facilitates the evaluation of autonomous vehicle IoT systems without the need for specialized hardware [1], [3]. The key contributions of this paper are summarized as follows:

Lightweight MQTT-driven architecture: We propose an IoT framework built on MQTT and FastAPI that enables real-time communication among vehicles, fleet managers, and external entities in a fully virtualized environment [10], [12], [15].

Dual-dashboard framework: We introduce a two-tier dashboard system that provides distinct interfaces for vehicle users and fleet managers, supporting role-specific monitoring, visualization, and control capabilities [2], [11].

Virtual test environment for autonomous systems: We demonstrate how a standard mobile device can be transformed into a virtual autonomous vehicle, thereby lowering the cost and complexity associated with prototyping and experimentation [6], [8].

Scalable and extensible design: The proposed system is designed to seamlessly integrate with advanced functionalities such as geofencing, alert mechanisms, and real-time visualization, with potential extensions into AI-driven route optimization and autonomous control [5], [7], [9].

II. Related Work

A. Literature Survey

Over the last few years, IoT-based systems have demonstrated a remarkable ability to collect real-time data across several domains including autonomous vehicle and traffic, etc. The field has witnessed growing interest in the optimization of autonomous vehicular networks. The following works have significantly contributed to this field. [1] proposed a framework a map management framework that uses the Message Queuing Telemetry Transport (MQTT) protocol., enabling vehicles to broadcast and subscribe to traffic-related messages.

In [2] proposed a IoT driven system called the "Intelligent dispatching and health monitoring system" (IDHMS) to address various challenges in fleet management system. This system integrates both software and hardware providing the flexible network infrastructure, protocols and proper security measures.

In [3] work provides a proper summary of the past development made in the field of autonomous driving and intelligent vehicles.

In. [4] work focused on a problem faced by autonomous vehicular system as each vehicle requires cooperation of other vehicles in its surrounding. Also, the inefficiency of the sensors to share huge data, resulting in communication delay. Hence, they came with Realtime Collaborative Vehicular Communication based on Bird's-Eye-View (BEV) map which is 2D representation for accurate object detection and data sharing.

In [5] works proposed a smart traffic navigation system that mainly focuses on edge computing and fault tolerance. Also proposed a neuromorphic approach for more efficient navigation in Internet of Vehicle (IoV) application. Experimental results demonstrate that the proposed model improves communication distance, load balancing, and maximum throughput.

In [6] work proposed a layered approach for sharing the emergency message in Internet of Vehicle (IoV) using the MQTT protocol which enhanced the delivery rates and reduced the latency hence increasing road safety and efficient traffic management. In [7] works was systematic review on the existing technologies for the fleet management and charge scheduling. The review provides a foundation for building a efficient fleet management system.

In [8] work was deploying a autonomous surface vehicle for water quality measurements in a lake using MQTT. The usage of MQTT protocols provides reliable and efficient transmission of data over remote servers, enabling real time data processing and analysis. V. Rahatal [9] study concludes that the proposed IoT-based communication system effectively enhances the operational efficiency and safety of autonomous electric vehicles.

In [10] study addresses the critical need for ultra-reliable and low-latency communication (URLLC) in Connected Autonomous Vehicles (CAVs). The authors propose a novel Multipath TCP (MPTCP) framework that leverages multiple wireless communication paths, such as high-speed Wi-Fi and 5G networks, to enhance data transmission reliability and reduce latency.

B. Research Gap

From the reviewed literature, it is evident that IoT-enabled vehicular systems have advanced in areas such as trajectory planning, cooperative perception, edge computing, fleet management, and emergency communication [1] – [3]. However, several limitations remain: Most works emphasize domain-specific or isolated scenarios (e.g., single-vehicle monitoring, emergency alerts) without addressing holistic fleet-level monitoring. Many solutions are hardware-intensive or theoretical, limiting scalability, cost-effectiveness, and adaptability for real-world prototyping. Visualization of vehicle behavior through interactive, real-time dashboards is largely missing. Existing research often treats reliability, low-latency, and fault tolerance in isolation, rather than as an integrated requirement [2].

Hence, there is a clear need for a unified IoT-based vehicular communication and visualization framework that leverages lightweight protocols like MQTT to support scalable, low-latency, and fault-tolerant monitoring of autonomous vehicles in a virtual environment [1], [3]. Our work addresses this gap by implementing a simulation platform that integrates real-time telemetry, multi-vehicle dashboards, and flexible fleet management features "Tab 1".

Table 1: Research Gap Analysis of Existing Works

Reference (Year)	Existing Work / Focus	Gap & Our Contribution
[1] Mustafa et al. (2025)	Edge intelligence for IoT devices enabling AI-driven decision-making at the edge.	Focused on device-level intelligence, not fleet-level IoT communication. Our work combines lightweight MQTT + dashboards for scalable AV simulation.
[2] Farahpoor et al. (2024)	IoT-driven fleet management system (IDHMS) for industrial vehicles.	Hardware-intensive and costly. Our work provides a virtual, low-cost fleet simulation using smartphones and dashboards.
[3] Yan (2024)	Integration of edge computing in autonomous vehicles for real-time decision-making.	Focuses on efficiency but lacks visualization/testbed. Our work adds dashboards and virtualized simulation.
[4] Chen et al. (2023)	Survey of milestones in autonomous driving and intelligent vehicles.	Purely theoretical, no prototype. Our work builds a working IoT-based testbed for AV simulation.
[5] Ngo et al. (2023)	Cooperative perception using V2V Bird's-Eye-View mapping.	Addresses perception but not IoT telemetry or visualization. Our work applies MQTT dashboards for real-time fleet monitoring.
[6] Yang et al. (2023)	Smart traffic navigation with fault-tolerant edge computing.	Focus on communication optimization, not simulation/monitoring. Our work adds visualization and extensible dashboards.
[7] Biswas & Wang (2023)	IoT + Edge + 5G + Blockchain integration for AVs.	High-level conceptual integration, lacks practical lightweight simulation. Our work offers a deployable MQTT-driven testbed.
[8] Szántó et al. (2023)	Trajectory planning of automated vehicles with real-time map updates via MQTT.	Path optimization only, no visualization. Our work uses MQTT telemetry integrated with dashboards.
[9] Limmer et al. (2023)	Evolutionary approach for scheduling shared EV fleets.	Optimization-centric, not IoT communication or monitoring. Our work provides real-time dashboards for vehicle and fleet behavior.
[10] Pragathi et al. (2022)	Emergency message dissemination using MQTT in IoV.	Limited to emergency alerts. Our work integrates full telemetry (GPS, speed, battery, accidents, geofencing).

III. METHODOLOGY

A. System Architecture

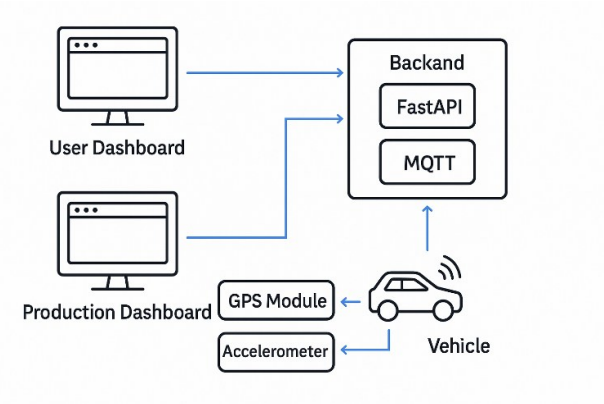

Fig. 1. System Architecture

The system follows a client-server architecture as shown in Fig. 1, where the vehicles act as clients and there is a centralized server built on FastAPI backend. A React-based frontend consisting of two parts: User Dashboard and Production Dashboard. User dashboard mainly focuses on the representation of the vehicles on the map [4][8] along with other options such as raise a SOS alert, Distance table, Geofences, weather, etc. Whereas the production mainly focuses on the analysis and visualization of the data collected from the vehicles using various charts and graphs [2]. The mobile of the user is assumed as a vehicle and used to transmit the data like GPS location, speed, etc. The overall paper is set up in order to test an autonomous vehicle network in a controlled environment without the requirement of an actual vehicle [1]. The data transmission within the system takes place using MQTT protocol i.e. from the Vehicle node to Server [10], whereas WebSockets are used for data transmission from backend to frontend [5].

B. Component Modules

a) Vehicle Module:

The vehicle module consists essentially of hardware components such as GPS module and accelerometer. The GPS module tracks the vehicle's location and speed, while the accelerometer detects sudden movements or impacts that may indicate accidents [6]. This sensor data plays a crucial role in enabling smart communication between vehicles and the system [7]. All collected information is transmitted to the backend server using the lightweight MQTT protocol [10],

ensuring efficient and timely updates for processing and monitoring.

b) Backend Module

The backend module serves as a centralized server. It is constructed utilizing MQTT for lightweight data transmission from the vehicle unit and FastAPI for managing API requests and WebSocket-based real-time interactions [2]. The backend receives incoming sensor data from the vehicles, processes it, and sends it to the respective dashboards.

c) User Dashboard:

The User Dashboard is designed to provide individual vehicle owners or users with a clear and interactive interface to monitor their vehicle in real time [4]. Users can examine important data on this dashboard, including speed, location, geofence status, and accident notifications. The dashboard fetches data from the backend and updates dynamically using WebSocket communication [5], ensuring that the user gets immediate insights into their vehicle's condition and movements.

d) Production Dashboard:

The Production Dashboard is intended for fleet managers or administrators who oversee multiple vehicles simultaneously. Its centralized view of all connected vehicles allows them to keep an eye on real-time data for the whole fleet, including position, speed, and emergency alerts [2][9]. This dashboard aids in operational decision-making, alerts management, and overall fleet efficiency.

e) Frontend Module:

The Frontend Module is the user interface layer of the system, developed to facilitate real-time interaction and visualization of vehicle data [8]. It includes both the User Dashboard and the Production Dashboard, each tailored for different stakeholders. Using modern web technologies (such as React or Next.js), the frontend communicates with the backend WebSocket [5] to show the most recent data, including alerts, vehicle location, speed, and other details. It provides a responsive and easy-to-use experience, allowing users to monitor and interact with the system efficiently from any device with internet access.

B. Data Flow

The three components of the system are connected in a linear manner where the Vehicle node is connected to FastAPI backend using MQTT. The backend is connected to the front end using WebSocket.

Fig. 2 illustrates the flow of data in the system. The vehicle node is built on Termux app where a python script is used to fetch the GPS location of the mobile which is considered as a vehicle for this system. The Termux application eliminates the requirements of the Hardware components such as GPS, accelerometer etc. The fetched data received from this is considered the sensor data which is sent to the Backend for processing and storage using a light-weight protocol called MQTT. The stored data is sent to the frontend for user and production dashboard for analysis and visualization of real-time data.

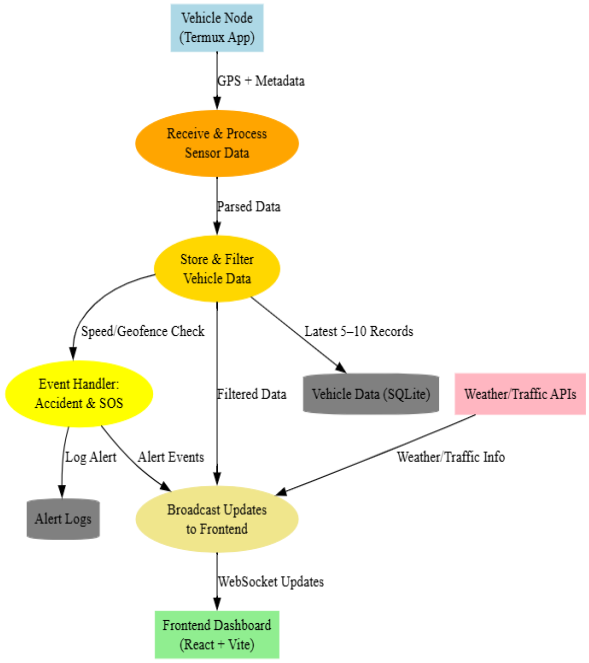

Fig. 2. Dataflow diagram

C. Tools and Technologies

- Backend**:** FastAPI, SQLite, WebSocket
- Frontend**:** React, Leaflet, Vite
- Data Transmission**:** MQTT via Termux (mobile GPS)

IV. Implementation

A. Overview

The Smart Vehicle System is designed to enable autonomous V2V communications for the safety, efficiency, and real-time exchange of information among the vehicles. The system comes with two dashboards, one for the user and the other for the production side: these dashboards provide the main features for the end-user (driver) and fleet operators. The communication among the vehicles is done through GPS modules and accelerometer, while the MQTT protocol is used for data transmission to the backend.

The communication flow of system is depicted in Fig. 3. The Vehicle node sends the data from GPS and Accelerometer to Backend of the system. After parsing and validating, backend queries SQLite Database for events like speeding, SOS triggers, or geofence crossing. For providing additional context, the Backend interacts with external APIs like Traffic or Weather to consider environmental factor before triggering alerts. The alert status is stored in the database and is sent to Vehicle node if necessary. Simultaneously this alert status is pushed to Frontend dashboard, where operators can monitor their vehicle in real-time and take appropriate actions.

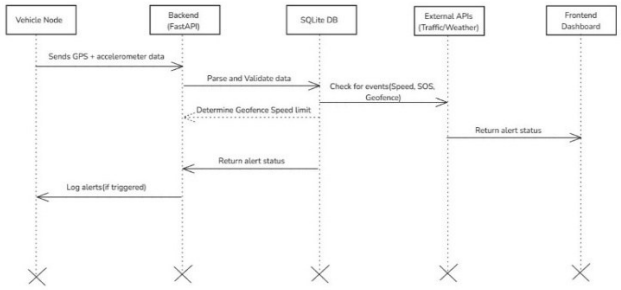

Fig. 3. Sequence diagram

B. Technologies Used

The backend server was hosted locally using Uvicorn, the ASGI server for FastAPI. Mobile devices ran Termux (an Android terminal emulator) and executed a Python script to send real-time GPS data via MQTT. The MQTT broker (Mosquitto) was configured to run on the same server that hosted the backend. The frontend was served on localhost during development and connected to the backend WebSocket. The SQLite database was divided into two table one named **vehicle_data** with columns id, vehicle_id, latitude, longitude, speed, and timestamp. The other table was **alerts** columns vehicle_id, alert_type, latitude and longitude.

The frontend dashboard consisted of a full-screen map view with real-time vehicle markers. The Leaflet map displayed icons representing each vehicle, where user-entered vehicle ID was highlighted with a red color other in blue color. Only vehicles within a 2 km radius of the reference vehicle were shown. Additional buttons were added to trigger SOS and accident alerts, which displayed corresponding messages on the map. Also, a table containing vehicle-specific data like speed, distance, battery life, etc. Fleet management plays a pivotal role in ensuring coordinated, efficient, and safe operation of multiple vehicles in dynamic environments. The data from these vehicles are collected and visulaised using various charts and graphs.

C. Result

The results demonstrate that the proposed system successfully provides two operational modes with distinct functionalities: User Mode enabling real-time vehicle tracking and accident reporting, and Production Mode offering fleet-level performance monitoring and analytics.

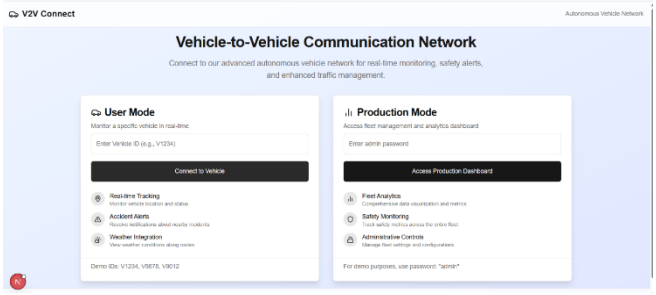

Fig. 4. Landing page

Fig. 4 shows the landing page of the system, with options for User Mode and Production Mode. In User Mode, the user is prompted to enter a vehicle ID, which then takes them to the user dashboard. In Production Mode, a password is required to access the production dashboard.

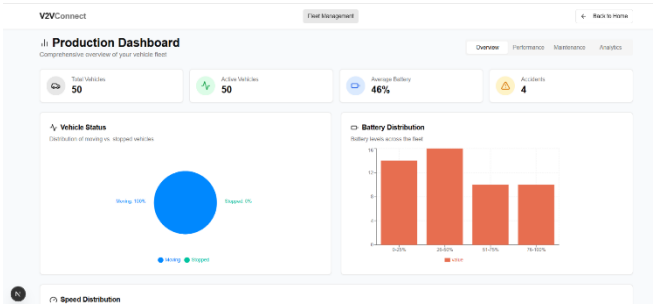

Fig. 5. Production Mode Dashboard

Fig. 5 shows the Production Mode dashboard, which displays various graphs related to the vehicle fleet's performance. These graphs include data on vehicle performance, maintenance, and analytics. For example, performance graphs might show the speed and battery levels of vehicles, maintenance graphs could track when vehicles need servicing, and analytics graphs provide insights into overall fleet efficiency.

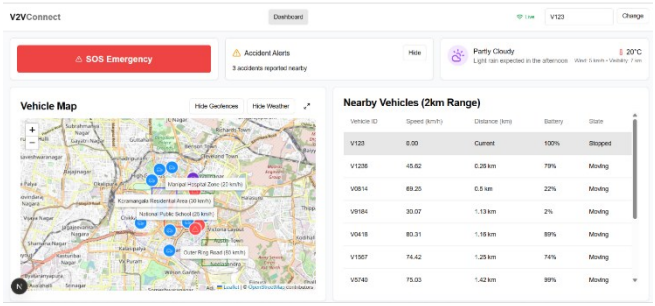

Fig. 6. User Mode Dashboard

Fig. 6 shows the User Mode dashboard, which displays a map with different vehicles, and the vehicle ID entered by the user is highlighted. The map also shows accident locations and geofences. Users can view weather conditions and have the option to report an accident, indicating the intensity of the incident. There is also an option to mark the accident on the map. Additionally, a table shows data for vehicles within a 2 km range, including their vehicle IDs, speed, distance from the user's vehicle, battery status, and whether they are moving or not.

V. Conclusion

This paper explains how real-time GPS tracking, accident monitoring, and fleet management can be applied to autonomous and semi-autonomous vehicles in a virtual environment. By making use of technologies such as MQTT, Termux, FastAPI, and WebSocket, the system makes communication and data sharing between vehicles and a central server simple and reliable. The platform comes with two main modes: User Mode for everyday users and Production Mode for fleet managers, each designed to give helpful insights into vehicle behavior and performance. By recreating vehicle interactions inside a controlled environment, this paper offers a cost-friendly and scalable method for testing autonomous systems before they are

deployed on real roads. With live data visualization and a dashboard that supports fleet operations, the system aims to improve both safety and efficiency. This paper builds an important step toward reliable autonomous driving, and with stronger technology and development, it could be expanded into real-world use, supporting wider adoption. During development, several challenges were encountered. First, fluctuations in GPS readings from mobile devices caused false vehicle alerts. To mitigate this, the Termux accuracy parameter was employed to filter out low-accuracy readings, ensuring that only reliable GPS coordinates were processed. Second, multiple alert icons often appeared simultaneously on the map, leading to clutter and confusion. This was addressed by implementing an alert queue with timestamp-based ordering, which regulates the sequence and timing of alert display. Third, the frontend WebSocket connections experienced timeouts or disconnections, particularly when idle or when multiple browser tabs were active. To resolve this, a periodic ping mechanism was added on the client side, complemented by server-side heartbeat checks, ensuring stable and persistent real-time connections.

REFERENCES

[1] I. Mustafa, I. Umer and M. J. Arshad, "Edge Intelligence in IoT: Enabling Smarter, Faster Autonomous Devices Through Artificial Intelligence and Edge Computing," *Spectrum of Engineering Sciences*, vol. 3, no. 5, pp. 241-248, 2025.

[2] M. Farahpoor, O. Esparza and M. C. Soriano, "Comprehensive IoT-Driven Fleet Management System for Industrial Vehicles," *IEEE Access*, vol. 12, pp. 193429-193444, 2024, doi: 10.1109/ACCESS.2023.3343920.

[3] X. Yan, "Integration of Edge Computing in Autonomous Vehicles for System Efficiency, Real-Time Data Processing, and Decision-Making for Advanced Transportation," *Applied Research in Artificial Intelligence and Cloud Computing*, vol. 7, no. 6, pp. 25-45, 2024.

[4] L. Chen et al., "Milestones in Autonomous Driving and Intelligent Vehicles: Survey of Surveys," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1046-1056, Feb. 2023, doi: 10.1109/TIV.2022.3223131.

[5] H. Ngo, H. Fang and H. Wang, "Cooperative Perception With V2V Communication for Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 11122-11131, Sept. 2023, doi: 10.1109/TVT.2023.3264020.

[6] S. Yang, J. Tan, T. Lei and B. Linares-Barranco, "Smart Traffic Navigation System for Fault Tolerant Edge Computing of Internet of Vehicle in Intelligent Transportation Gateway," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 13011-13022, Nov. 2023, doi: 10.1109/TITS.2022.3232231.

[7] A. Biswas and H.-C. Wang, "Autonomous Vehicles Enabled by the Integration of IoT, Edge Intelligence, 5G, and Blockchain," *Sensors*, vol. 23, no. 4, Article 1963, 2023, doi: 10.3390/s23041963.

[8] M. Szántó, C. Hidalgo, L. González, J. P. Rastelli, E. Asua and L. Vajta, "Trajectory Planning of Automated Vehicles Using Real-Time Map Updates," *IEEE Access*, vol. 11, pp. 67468-67481, 2023, doi: 10.1109/ACCESS.2023.3291350.

[9] S. Limmer, J. Varga and G. R. Raidl, "An Evolutionary Approach for Scheduling a Fleet of Shared Electric Vehicles," in *Applications of Evolutionary Computation, EvoApplications 2023, Lecture Notes in Computer Science*, vol. 13989, Springer, Cham, 2023, pp. 3-18, doi: 10.1007/978-3-031-30229-9_1.

[10] N. K. B, P. V. M, S. L. Pragathi B, M. Arunachalam and C. Chin, "Emergency Message Dissemination through MQTT over the Internet of Vehicles - A Layered Approach," *2022 IEEE 19th India Council International Conference (INDICON)*, Kochi, India, 2022, pp. 1-6, doi: 10.1109/INDICON56171.2022.10039736.

[11] W. Tan, Y. Sun, Z. Ding and W.-J. Lee, "Fleet Management and Charging Scheduling for Shared Mobility-on-Demand System: A Systematic Review," *IEEE Open Access Journal of Power and Energy*, vol. 9, pp. 425-436, 2022, doi: 10.1109/OAJPE.2022.3215865.

[12] P. S. Yadav, "Enhancing Real-Time Data Communication and Security in Connected Vehicles Using MQTT Protocol," *Journal of Artificial Intelligence & Cloud Computing*, 2022.

[13] L. Reidher, B. Lampe, T. Woopen, R. van Kempen, T. Beemelmanns and L. Eckstein, "Enabling Connectivity for Automated Mobility: A Novel MQTT-Based Interface Evaluated in a 5G Case Study on Edge-Cloud Lidar Object Detection," *arXiv preprint arXiv:2209.03630*, Sept. 2022.

[14] M. H. Muneer and V. A. Haseena, "An ML-Based Fleet Management System for Electric Vehicles," *SAE Technical Paper 2022-28-0300*, 2022, doi: 10.4271/2022-28-0300.

[15] A. Sakuraba, Y. Shibata and M. Ohara, "An Implementation of V2R Data Delivery Method Based on MQTT for Road Safety Application," in *Advanced Information Networking and Applications (AINA 2022), Lecture Notes in Networks and Systems*, Springer, Cham, 2022, pp. 399-410, doi: 10.1007/978-3-030-99619-2_38.

[16] V. Rahatal, P. More, M. Salunke, S. Makeshwar and R. D. Joshi, "IoT Based Communication System for Autonomous Electric Vehicles," *2021 7th International Conference on Signal Processing and Communication (ICSC)*, Noida, India, 2021, pp. 66-72, doi: 10.1109/ICSC53193.2021.9673164.

[17] S. R. Pokhrel, N. Kumar and A. Walid, "Towards Ultra Reliable Low Latency Multipath TCP for Connected Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 8175-8185, Aug. 2021, doi: 10.1109/TVT.2021.3093632.

[18] I. Idris, "Real-Time Vehicle Monitoring and Positioning Using MQTT for Reliable Wireless Connectivity," Master's Thesis, Queensland University of Technology, 2017.

[19] A. Nigam, V. M., K. Agrawal, and A. Shree, "Application of IoT and Machine Learning Techniques for Smart Waste Management System," in *Futuristic Trends in IoT*, vol. 3, IIP Series, 2024, pp. 60–70.

[20] B. Agarwal, L. Ravichandra, M. Kumar, P. D. Thakkar and Vaidehi. M, "Design and Implementation of Virtual Laboratory using Unity with a Web Interface," *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*, Manama, Bahrain, 2024, pp. 728–732, doi: 10.1109/ICETSIS61505.2024.10459676.