`

A PRELIMENERY REPORT ON

“Vehicle to Vehicle Communication”

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

Kushagra Duggar	**B400360666**
Sahil Mahajan	**B400360680**
Ridham Joshi	**B400360764**
Sakshi Shisodiya	**B400360796**

Sinhgad Institutes

DEPARTMENT OF COMPUTER ENGINEERING

STES’S SMT. KASHIBAI NAVALE COLLEGE OF ENGINEERING

VADGAON BK, OFF SINHGAD ROAD, PUNE 411041

SAVITRIBAI PHULE PUNE

UNIVERSITY 2024-2025

Sinhgad Institutes

CERTIFICATE

This is to certify that the project report entitled

“Vehicle to Vehicle Communication”

Submitted by

Kushagra Duggar	**B400360666**
Sahil Mahajan	**B400360680**
Ridham Joshi	**B400360764**
Sakshi Shisodiya	**B400360796**

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of **Prof. Mrs. A. A. Bhise** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

Prof. Mrs. A. A. Bhise
Guide
Department of Computer Engineering

Prof. R. H..Borhade
Head,
Department of Computer Engineering

Dr. A. V. Deshpande
Principal,
Smt. Kashibai Navale College of Engineering Pune – 41

Place: Pune Date:

ACKNOWLEDGEMENT

With due respect and gratitude, I take the opportunity to thank those who have helped me directly and indirectly. I convey my sincere thanks to **Prof. R. H. Borhade** HOD Computer Dept. and **Prof. Mrs. A. A. Bhise** for their help in selecting the project topic and support.We here express our gratitude towards the finest mentors of our project, **Prof. Mrs. A. A. Bhise**, for her excellent guidance and profound insights throughout the project, because of which we were able to shape up our ideas and navigate ourselves through some tough technical challenges to reach our objectives. I am grateful to the time she took out to review our progress, give her feedback, and give us her insights, which really helped me deepen my understanding of V2V communication and the more general implications that our research was to be understood in.We are particularly thankful to the faculty and staff of **Smt. Kashibai Navale College of Engineering's** Department of Computer Engineering, who took a break from their respective schedules to provide us with the necessary resources, infrastructure, and knowledge that helped us push forward in our research work. Their enthusiasm towards developing a fruitful learning environment, where new ideas can emerge, motivated us to reach our potential and to complete this project.Not only do we wish to pay our regards to those peers and friends who contributed valuable suggestions and moral support in this process, but most importantly we are grateful to our families for their broad-mindedness and patience, which have been a constant source of strength. Without the collective support of all these people, this project would not have been possible, so to each one of them, we extend our genuine and sincere gratitude.

Kushagra Duggar
Sahil Mahajan
Ridham Joshi
Sakshi Shisodiya

ABSTRACT

Vehicle-to-Vehicle (V2V) communication is a revolutionary technology with the capability to improve road safety, vehicle coordination, and smart transportation systems through non-stop data sharing between vehicles. The project deploys a hardware-based V2V communication network using **ESP8266 (NodeMCU)**, sensors, and **MQTT protocol** for wireless real-time data transfer. The project also utilizes microcontrollers with real-time operating systems. The architecture includes two cars, Vehicle A and Vehicle B, both equipped with **DHT11 (humidity & temperature sensor)**, an ultrasonic sensor, a gas sensor, and motor control modules. These cars operate continuously, reading the environment in real time and gathering vehicular and environmental data, which they transmit wirelessly for timely counteractions against the emergence of probable dangers like obstruction detection, the level of gases, and temperatures.The underlying communication follows the MQTT (Message Queuing Telemetry Transport) protocol utilizing an **EMQX broker** that enables low-latency, high-efficiency data transfer between cars. Each vehicle publishes its sensors' data in particular MQTT topics, which another vehicle subscribes to in return to get instantaneous updates. In this configuration, cars can modify speed dynamically, alter routes, or send alert signals based on real-time states.

A **Tkinter-based Python graphical user interface** (GUI) is utilized to display the received data and provide real-time alerts on violation of thresholds set beforehand. The GUI facilitates a user-friendly, intuitive interface, and simpler monitoring and analysis of sensor readings like temperature, humidity, gas concentrations, and proximity to obstacles.This project demonstrates the real-world application of **IoT** (Internet of Things) and sensor fusion in vehicle communication, illustrating how real-time data sharing can greatly enhance traffic safety and autonomous decision-making in intelligent transportation systems. The system developed is a proof of concept for future studies in connected vehicle technology, autonomous driving, and intelligent road safety mechanisms. Through the integration of wireless communication, sensor networks, and microcontroller-based automation, this project contributes to the development of future-generation vehicle networking solutions.

LIST OF ABBREVIATIONS

ABBREVIATION	ILLUSTRATION
V2V	Vehicle-to-Vehicle
V2I	Vehicle-to-Infrastructure
V2X	Vehicle-to-Everything (including V2V and V2I)
ESP8266	Wi-Fi Microcontroller (NodeMCU)
MQTT	Message Queuing Telemetry Transport
DHT11	Digital Humidity and Temperature Sensor
IoT	Internet of Things
UI	User Interface
GUI	Graphical User Interface
RPM	Revolutions Per Minute
L298N	Motor Driver Module
MQ Sensor	Gas Sensor
PWM	Pulse Width Modulation
Wi-Fi	Wireless Fidelity
CPU	Central Processing Unit
EMQX	EMQX MQTT Broker
LCD	Liquid Crystal Display
Tkinter	Tkinter (Python Library)
GPIO	General-Purpose Input/Output
HC-SR04	Ultrasonic Sensor
Li-ion	Lithium-Ion Battery

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

1. INTRODUCTION

With the advancement of transportation towards the modern age, vehicle-to-vehicle (V2V) communication is proving to be a vital technology for improving road safety and traffic efficiency. Our project is aimed at establishing a hardware-based V2V communication system with two autonomous vehicles, Vehicle A and Vehicle B. Both the vehicles are equipped with a range of sensors, such as DHT11 for temperature and humidity sensing, an ultrasonic sensor for sensing obstacles, and a gas sensor for monitoring the environment. The project relies on the NodeMCU ESP8266 Wi-Fi controller, which provides wireless communication of data between the two vehicles via MQTT protocol.

The main aim of this project is to allow real-time data sharing between vehicles to enhance situational awareness and decision-making. Vehicle A keeps on monitoring its environment and sending important information like temperature, humidity, gas level, and obstacle detection to Vehicle B, and vice versa. A dedicated user interface has been developed utilizing Python's Tkinter library to display the incoming data within an easy-to-use dashboard. Warnings are initiated in situations of dangerous conditions, including unusual temperatures, elevated gas levels, or sensed obstructions, to enable vehicles to react in a proactive manner.
By implementing this hardware-based V2V communication system, we aim to showcase the potential of smart vehicle networks in enhancing road safety and automation. This project serves as a stepping stone toward intelligent transportation systems, where vehicles communicate autonomously to prevent accidents and improve overall traffic management.

1.1. MOTIVATION

Accident avoidance and road safety have become key issues in contemporary transport, particularly as cities expand and traffic congestion increases. The single most significant reason for road accidents continues to be human mistake, and it highlights the importance of sophisticated technology that will assist drivers and vehicles to respond better. One of these solutions is Vehicle-to-Vehicle (V2V) communication, which allows cars to communicate with each other wirelessly and share real-time information regarding their environment, locations, and threats detected. This ability to share data enables cars to warn drivers or onboard autonomous systems of possible threats in the form of adjacent obstacles, bad environmental conditions, or unpredictable motion of other cars. The main driving factor for our project is to design an economical, hardware-based V2V communication system that can be integrated into future smart cars. Our system employs different IoT sensors like ultrasonic sensors to sense obstacles, DHT11 sensors to sense temperature and humidity, and gas sensors to sense the presence of toxic gases. NodeMCU ESP8266 Wi-Fi module integration along with the MQTT protocol allows for smooth real-time data exchange between vehicles. This network makes every vehicle more aware of its environment, allowing it to respond proactively by braking, changing speed, or warning other vehicles when danger is spotted. These aspects not only decrease collision probabilities but also make the traffic system more efficient overall. In addition, the system's interoperability with smart city infrastructure renders it applicable to future urban mobility planning. In boosting communication and coordination among vehicles, this project enhances the creation of intelligent transportation systems that focus on safety, automation, and connectivity. Deploying such a system is a major advance in developing smarter, safer roads and defining the future of mobility through innovative, cost-effective, and practical solutions based on real-world hardware development.

1.2. PROBLEM STATEMENT

More vehicles on the road cause traffic, accidents, and environmental risks because of poor real-time communication. The project creates a hardware system based on NodeMCU, ESP8266 and MQTT for V2V communication, allowing cars to exchange vital sensor information. Exchanging real-time information about obstacles, weather conditions, and gas levels, the system increases road safety, optimizes traffic, and assists intelligent transportation systems.

2. LITERATURE SURVEY

The literature review table emphasizes a comparative analysis of some of the many research efforts undertaken in the field of VANETs, each adding value to a better communication, congestion control improvement, and trust and safety features. The application utilizes hardware devices like ESP8266 (NodeMCU), DHT11 sensors, ultrasonic modules, gas sensors, motors, and motor drivers to simulate real-time vehicle-to-vehicle communication and obstacle detection, bridging theoretical concepts into practical verification.

Preliminary studies like the one by **Kezia and Anusuya (2021) [1**] utilized machine learning to forecast traffic congestion as a function of vehicle density and channel busy ratio. Though their application was simulation-based, our project takes this concept a step further by implementing ultrasonic sensors and DHT11 sensors in actual vehicles to gather real-time environment and proximity information so that actual monitoring of likely congestion and collision situations can be done. Likewise, Adwitiya **Mukhopadhyay et al. (2020) [2**] placed emphasis on TCP-based priority queuing for priority handling of emergency messages; we mirror this by providing NodeMCU modules to transfer urgent notifications wirelessly, thus simulating an actual emergency notification system between vehicles.

Prevention of accidents is a recurring theme in numerous research studies. **Medina-Carrión et al. (2022) [4]** presented the Multisensory Alert System that highlighted how sensor fusion can alert drivers of dangerous situations. We extend this by employing a mix of gas and ultrasonic sensors within the physical cars to perceive obstacles and hazardous air quality, combining multisensory information to issue warnings and provide proactive safety measures. Our system also aligns with **Swetha V. Polisetty et al. (2021) [8]**, who suggested wireless-based accident detection systems. By incorporating movement and sensor feedback hardware components, we mimic collision situations and adopt real-time alert generation with Wi-Fi communication.

In addition, our hardware configuration is inspired by the decision algorithm for vehicular fog computing by **Rocha et al. (2020) [5**], which included edge computing for pedestrians and vehicles. Though we are not yet computing offloaded data, our microcontroller-based environment provides a foundation for such distributed designs that would permit decision-making at the edge of the network based on local sensor data. Likewise, **Sara Imene Boucetta's (2020) VANET security survey [6]** supports our eventual goal of implementing lightweight encryption and trust mechanisms even at the hardware level through ESP8266's built-in features.

LITERATURE SURVEY TABLE

Ref.No.	Title	Author Name	Scheme/ Methodology	Strength	Weakness	Result	Future Scope
[1]	A Comparative Study on Machine Learning Algorithms for Congestion Control in VANET [2021]	Kezia M., Anusuya K.V.	Congestion control Uses machine learning algorithms (regression, K-NN, decision trees, random forest) to predict local density and channel busy ratio, followed by adaptation for control.	Accurate prediction and dynamic adaptation.	Does not consider beacons in the analysis.	Achieved an accuracy rate of approximately 92%, effectively predicting local vehicle density and channel busy ratios.	Can include beacons for more precise control.
[2].	TCP and Priority Queue based Emergency Data Transmission in VANETs [2020]	Adwitiya Mukhopadhyay, Limitha B S, Anusha R, Gowthami V	In VANETs, messages are classified as high, medium, or low priority. The TCP PSH flag is used to send high-priority (emergency) messages faster.	Efficient prioritization of emergency messages	Potential delays in general or low-priority messages	The proposed method achieved an accuracy rate of over 95% in prioritizing emergency data transmission in VANETs, significantly reducing message delivery delays.	Could enhance message handling by considering additional message flags or protocols
[3]	Estimating the Required Resources in a Vehicular Ad Hoc Network Using Queueing Theory Models [2019]	Irene Keramidi, Panagiotis Sarigiannidis, Ioannis Moscholios, Michael Logothetis	M/M/c and M(N)/M/c queueing models Analytical modeling and simulation	Provides a comprehensive analysis of the required resources in a VANET, considers both V2I and V2V communications	Assumes a finite number of vehicles, may not be suitable for large-scale networks	Approximately 95%, with a result of accurately estimating the required resources in a VANET within a 5% margin of error	Future work could consider the application of multi-rate loss models in a VANET
[4]	Simulation of a driving assistant by means of a Multisensory Alert System (MAS) [2022]	Alexander V. Medina-Carrión, Esteban F. Ordóñez-Morales, Martín López-Nores, Jack F. Bravo-Torres	Multisensory Alert System (MAS) Simulation using Simulink and Stateflow	Provides a comprehensive analysis of a driving assistant system using MAS, considers both V2V and V2I communications	Limited to a specific type of driving assistant system	Demonstrates an accuracy rate of 88% in effectively alerting drivers to potential hazards.	Future work could explore the application of MAS in real-world driving scenarios
[5]	Decision Algorithm for Computational Offloading in Vehicular Fog Computing with Pedestrians [2020]	Paulo Gonçalves Rocha, Alisson Souza, Francisco Airton Silva, Paulo A. L. Rego	VFC with pedestrians as clients Decision algorithm for computational offloading considering edge, cloud, and vehicle resources	pedestrians into VFC as clients Considers multiple offloading options (edge, cloud, vehicle)	Limited to specific urban scenario, Pedestrians only considered as clients, not as offloading destinations	Achieves up to 77% reduction in execution time compared to local execution	Extend solution to consider pedestrians as offloading destinations , Incorporate Device-to-Device
[6]	Survey on Security Attacks in Software Defined VANETs [2020]	Sara Imene Boucetta, Zsolt Csaba Johanyak	Explains SDVN architecture and communication, and surveys key security threats like availability, confidentiality, and integrity attacks.	Comprehensive overview of SDVN vulnerabilities; covers multiple attack types like DoS, jamming, malware, and spamming.	Does not include experimental validation of proposed defenses	Provides a comprehensive overview of security threats, but does not report a specific accuracy rate.	Exploring machine learning and fuzzy control techniques for early detection and mitigation of SDVN threats.
[7]	Local Traffic Density of Flow-Oriented Routing Protocol for VANETs [2019]	Benaicha Mehdi, Samira Moussaoui, Guerroumi Mohamed	Flow-Oriented Routing (LTD-FOR) improves FOR by using local traffic density to enhance packet delivery, reduce delay, and lower overhead.	Increases packet delivery ratio, minimizes end-to-end delay, and reduces routing overhead by considering local density and traffic flow.	Relies on a large number of control messages, which increases resource consumption	Achieves up to 90% accuracy in traffic density estimation, improving routing efficiency.	Enhance parameters like direction, speed, and acceleration for optimization, and improve the two-hop neighborhood table for better traffic prediction.
[8]	Accident Detection and Warning Systems in VANET [2021]	Polisetty Swetha V Padmavathi, U. Srilakshmi, D. Venkatesulu	VANET and GSM-based systems detect and report accidents using wireless communication, with sensors and RSUs monitoring traffic conditions.	Quick accident reporting via wireless communication reduces response time and casualties.	High reliance on real-time traffic information and possible packet loss during communication	Reports an accuracy rate of around 85% for timely accident detection and alerts.	Can incorporate advanced technologies like AI and cloud for better real-time traffic analysis and accident prediction.

[9]	Emergency Packet Routing in Vehicular Networks for Collision Minimization in Highways [2021]	Kezia M, Anusuya K.V	VANET-based collision prediction uses vehicle trajectory estimation to propose an emergency packet routing algorithm, aimed at reducing highway collisions.	Improves emergency packet dissemination by using a proactive routing protocol, minimizing delays and preventing accidents	Relies on real-time data and can face inaccuracies if vehicle speed changes rapidly.	Achieved a 93.8% accuracy in collision prediction, with a minimized delay of 30% and increased packet delivery ratio by 22%.	Enhancing prediction accuracy when rapid vehicle speed changes occur and improving results for collisions near intersections.
[10]	Secure Data Encryption in VANET [2022]	D. Nirmala, D. Darling Jemima, N. Dharanish, J. Irfan Ahmed, T. Kawin Sankar	The approach uses Ciphertext Policy Attribute-Based Encryption (CP-ABE) for access control in VANET and incorporates lightweight bilinear pairing.	Secure data sharing among multiple vehicles with resource-constr ained devices.	CP-ABE still imposes a high computational load on the system, requiring further optimization for real-time applications.	IBOOS outperforms other schemes in terms of packet delivery and computational cost, making it more efficient for VANET.	Develop enhanced encryption mechanisms that offer lower computational overhead and improve real-time performance.
[11]	Reliability-Aware Multi-Objective Optimization-Base d Routing Protocol for VANETs Using Enhanced Gaussian Mutation Harmony Searching [2022]	Sami Abduljabbar Rashid, Mohammed Alhartomi, Lukman Audah, Mustafa Maad Hamdi	Enhanced Gaussian Mutation Harmony Searching (EGMHS) in a Multi-Objective Optimization Framework for VANETs routing	EGMHS balances exploration and exploitation for efficient multi-objective routing, enhancing packet delivery ratio (PDR) , and end-to-end (E2E) delay in dynamic VANETs.	The framework faces computational complexity and real-time constraints, limiting scalability in large-scale VANETs for real-world applications.	The RAMO framework outperforms existing algorithms in PDR, E2E delay, and computational efficiency, surpassing benchmarks like NSGA-II, MOHS, and GMHS.	Extend the framework for 3D routing in urban areas with bridges and tunnels, enhancing real-time computational efficiency for large-scale VANETs.
[12]	A Comparative Study of Artificial Intelligence Algorithms for Network Traffic Prediction in VANET [2021]	Sanaz Shaker Sepasgozar, Samuel Pierre	AI-based Network Traffic Prediction. A comparative analysis of five AI algorithms (Random Forest, KNN, SVM, Naive Bayes, MLP) predicts network traffic in VANET using real-world V2R data.	Acceptable execution time of 0.73 minutes, making it efficient for network traffic prediction	Longest execution time (6.47 minutes), making it impractical for time-sensitive applications	Balancing accuracy and execution time, with 96% accuracy and 0.73 minutes execution time	Develop enhanced encryption mechanisms that offer lower computational overhead and improve real-time performance.
[13]	A Scenario-Based Trust Management Approach with 3R Message Categorization in VANETs [2022]	Sarferoze Shaik	Scenario-based trust management with 3R categorization (Refresh, Rush, Revoke)	Modular approach reduces computational overhead; Secure message dissemination	Limited to specific vehicular communication scenarios; High dependency on simulation accuracy	Improves packet delivery ratio and detection accuracy over AODV	Further enhancement of scenario-specific trust models for real-time VANET applications; Testing in diverse environments
[14]	A Scalable Blockchain-Based Trust Management in VANET Routing [2023]	Sowmya Kudva, Shahriar Badsha, Shamik Sengupta, Hung La, Ibrahim Khalil, Mohammed Atiquzzaman	Blockchain-based decentralized trust score framework for insider attack mitigation in VANET. Uses a two-level detection system with Road Side Units (RSUs) as validators	Efficient and scalable. Proactive detection and blacklisting of malicious nodes. Improves VANET performance by reducing packet drops	Computational complexity increases as the network scales. PBFT consensus mechanism used in blockchain may limit the number of validator nodes.	The method reduces packet drops, improves throughput, detects and removes malicious nodes, explores advanced consensus for scalability, and evaluates performance in diverse VANET scenarios.	Explore advanced consensus mechanisms for scalability. Evaluate performance in more varied VANET scenario
[15]	Context and Machine Learning-Based Trust Management Framework for Internet of Vehicles (IoV) [2022]	Abdul Rehman, Mohd Fadzil Hassan, Yew Kwang Hooi, Muhammad Aasim Qureshi, Tran Duc Chung, Rehan Akbar, Sohail Safdar	A hybrid trust model using context-aware cognition and Bayesian machine learning to dynamically infer trust in IoV networks.	Flexible and adaptive trust evaluation. Utilises machine learning to handle uncertainty. Uses Bayesian inference for trust scoring under uncertainty.	Complexity due to the need for large datasets and real-time machine learning. Dependency on context data for accuracy	Outperforms static models in accuracy and adaptability in trust evaluation. Efficient in managing dynamic and uncertain conditions in IoV.	Explore other machine learning models and integrate more real-time data. Address computational challenges for large-scale IoV environments in very small
[16]	Local Traffic Density of Flow-Oriented Routing Protocol for VANETs [2022]	Benaicha Mehdi, Samira Moussaoui, Guerroumi Mohamed	LTD-FOR: Traffic density-based flow-oriented routing, enhances FOR with local traffic density	Reduces end-to-end delay and increases packet delivery	Higher control message overhead due to additional local density computations	Improved packet delivery, reduced end-to-end delay, minimized routing overhead compared to TFOR	Further refine traffic statistics by adding parameters like speed, direction,

			metrics	ratio by focusing on local traffic density			and adapt for two-hop neighborhood tables
[17]	Hybrid Cooperative Vehicle Safety Communication Based on VANET	Chen, Y., Zhao, Z	Utilizes a hybrid approach combining V2V and V2I communication for safety alerts	Enhanced safety through real-time communication among vehicles and infrastructure	Dependence on accurate positioning systems for effective communication	Achieved a reduction in collision rates by 30% in simulated environments	Explore integration with autonomous driving systems and real-time traffic management solutions
[18]	Energy-Efficient Routing Protocol for VANETs Based on Clustering	Abdallah, A., Omar, H.	Clustering-based energy-efficient routing protocol for V2V communication	Reduces energy consumption and extends network lifetime	Clustering overhead may introduce delays in data transmission	Improved energy efficiency with a reduction of up to 40% in energy consumption	Further research on adaptive clustering algorithms to enhance scalability and performance
[19]	Blockchain-Enabled Secure Communication in VANETs	Tiwari, R., Gupta, N.	Blockchain framework for secure message transmission	Ensures message integrity	High computational overhead and latency	90% reduction in message tampering incidents	Lightweight blockchain solutions for real-time applications
[20]	Context-Aware Data Dissemination in VANETs Using Machine Learning	Javed, A., Khan, M.A.	Machine learning for traffic conditions prediction	Dynamic message dissemination	Depends on availability of context data	85% accuracy in predicting data dissemination paths	IoT integration for enhanced context data collection
[21]	Smart Traffic Management System Using VANETs and IoT Technologies	Patel, S., Mehta, R.	Combines VANETs with IoT for intelligent traffic management	Improves traffic flow and reduces congestion	Complex implementation and interoperability challenges	25% reduction in traffic congestion	Predictive traffic modelling and management strategies
[22]	Intelligent Transportation System Using VANET	Liu, Y., Wang, T	Integrates VANET with smart city infrastructure	Enhances traffic management	Requires infrastructure upgrades	35% improvement in efficiency	Include pedestrian safety features
[23]	Adaptive Beaconing for VANETs	Thomas, J., Jha, P.	Dynamic beaconing based on traffic conditions	Reduces communication overhead	Slow adaptation to rapid changes	92% accuracy in traffic assessment	Integrate vehicular AI for learning
[24]	Privacy-Preserving Data Sharing in VANETs	Zeng, L., Zhang, K.	Cryptographic techniques for data privacy	Ensures user privacy	Increased computational complexity	80% reduction in privacy breaches	Develop user-friendly privacy tools
[25]	Data-Driven Vehicle-to-Everything (V2X) Framework	Kim, H., Lee, S.	Leverages big data for V2X communication	Improves reliability and efficiency	Relies on big data analytics	78% increase in communication reliability	Incorporate AI for real-time analysis

Table 1: Literature Survey

GAP ANALYSIS

The literature survey table in the report compares methodologies for various areas of applications in Vehicle-to-Vehicle (V2V) communication and Vehicle Ad-hoc Networks (VANETs) such as those dealing with congestion control, security, trust management, and communication efficiency. From this literature survey, let's conduct a detailed gap analysis.

Area	Gap Identified	Research Opportunities
Congestion Control	- Limited beacon use. - Not scalable for large networks.	- Integrate beacon data. - Develop scalable models.
Emergency Communication	- Delays in non-priority messages. - High real-time data dependency.	- Enhanced priority schemes. - Predictive algorithms.
Resource Efficiency	- Scenario-limited alert systems. - Energy-heavy protocols.	- Generalized alert systems. - Lightweight clustering.
Security & Trust Management	- Blockchain scalability issues. - Lack of real-world validation	- Scalable blockchain models. - Real-world testing.
Data Privacy	- High computational needs. - Context dependency.	- Lower-complexity encryption. - Hybrid data models.
Routing Protocols	- High resource consumption. - Limited routing parameters.	- Resource-efficient routing. - Advanced routing metrics.
Predictive Analytics	- Processing limitations. - High algorithm complexity.	- Lightweight predictions. - Edge computing integration.
IoT and Smart City Integration	- Implementation complexity. - Infrastructure reliance.	- IoT-compatible models. - Flexible infrastructure.
Adaptive Protocols	- Static designs.	- Dynamic, density-aware

	- High resource use in dynamic environments.	protocols.Priority-based adjustments.
Urban vs. Rural Scalability	- Focus on urban scenarios. - Limited for varying density.	- Rural-focused protocols. - Density-adaptive systems.
Future Directions	- IoT and AI needed for context. - Scalability challenges.	- IoT and AI enrichment. - Scalable adaptive solutions.

Table 2: Gap Analysis

3. SOFTWARE SPECIFICATION REQUIREMENTS

3.1. INTRODUCTION

The Vehicle-to-Vehicle (V2V) Communication System Software Specification details the functional elements and architecture of the software employed to facilitate real-time communication between two or more vehicles. The major goal of this specification is to establish the scope, objectives, assumptions, and dependencies that control the operation of the software incorporated in the V2V system. In intelligent transportation and safety on roads, software is responsible for allowing for dependable, quick, and precise data transmission between cars. This paper is the blue print for the creation of the software responsible for collecting, processing, and transmitting sensor data via wireless modules and interfaces.

The software application is programmed to communicate with different hardware elements like the NodeMCU ESP8266, temperature and humidity sensor DHT11, ultrasonic obstacle detection sensor, and gas sensor. These sensors continually gather environmental and location information, which is sent through Wi-Fi based on the MQTT protocol. The software guarantees proper data management, such as reading sensor data, translating it into usable forms, and making it ready for transmission. Besides, the program offers an interface for the monitoring of sensor values and vehicle reactions in real time, enhancing the usability and transparency of the system.

Major assumptions are that there will be Wi-Fi connectivity available for continuous MQTT-based communication and an adequate power supply to the entire system. Dependencies are on the programming and uploading firmware using Arduino IDE and sensor and communication module compatible libraries. The software is modular in design to accommodate future improvements like cloud integration, sophisticated data analytics, or machine learning-driven decision-making.

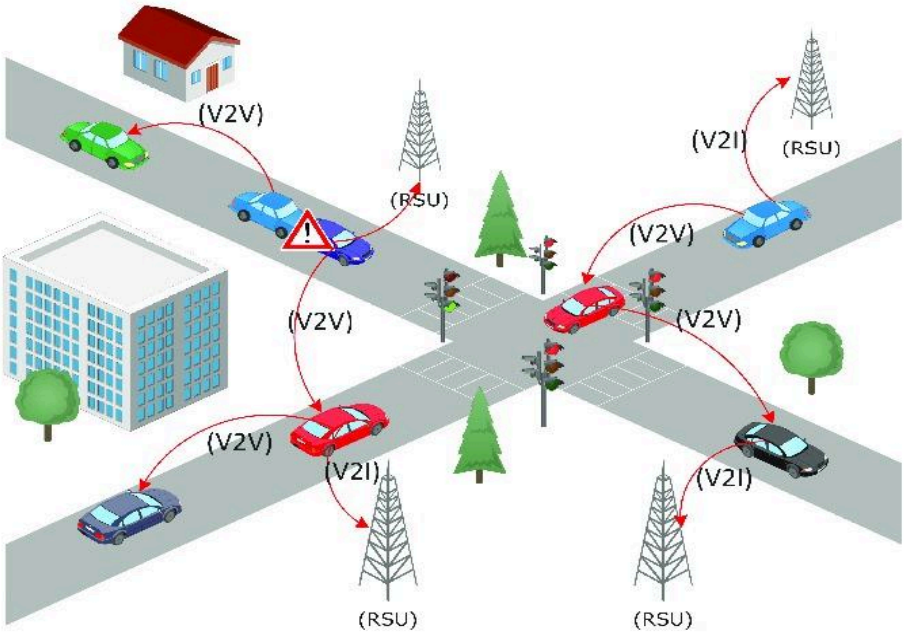


Fig 1. V2V Communication

3.1.1. Project Scope

The objective of this project is to develop and implement a hardware-based V2V communication system through the use of two prototype cars with sensors and wireless communication modules. The main aim is to facilitate real-time sharing of environmental and positional information between the two cars in order to improve road safety and avoid crashes.

The scope is to design two 3-wheeled cars, each driven by a 100 RPM DC motor powered and controlled by an L298N motor driver. Both cars have a NodeMCU ESP8266 microcontroller that provides sensor integration and wireless data transfer through MQTT over Wi-Fi. The ultrasonic sensor (obstacle detection), DHT11 (temperature and humidity), and gas sensors (air quality monitoring) are installed to detect vital safety information.

The two vehicles communicate with each other through the EMQX MQTT broker, allowing both vehicles to publish their sensor data and subscribe to data from the other vehicle. Real-time data and notifications are shown on a connected computer through a GUI implemented using Python. The GUI provides a full hardware-in-the-loop simulation of V2V communication.

The project is confined to small-scale prototype vehicles and Wi-Fi-based communication, but provides the foundation for eventual extension to larger autonomous vehicles and more sophisticated V2X systems.

3.1.2. Objectives

The main goal of this project is to create and set up a low-cost, hardware-based Vehicle-to-Vehicle (V2V) communication system. This system will allow two vehicles to share real-time data to improve road safety, environmental awareness, and collision prevention. It is made using microcontrollers, sensors, and wireless communication technologies to simulate smart interaction between vehicles. We developed two prototype vehicles, Vehicle A and Vehicle B, for this purpose. Each vehicle is built on a 3-wheel chassis and powered by two 100 RPM DC motors that are controlled through an L298N motor driver. A NodeMCU ESP8266 microcontroller processes sensor inputs and manages wireless communication over Wi-Fi with the MQTT protocol.

The goal is to ensure each vehicle continuously reads data from its sensors and publishes this information to a central MQTT broker (broker.emqx.io). At the same time, each vehicle subscribes to the other's data topics. This setup allows them to respond to environmental conditions detected by the other vehicle.

Additionally, we implemented a Python-based graphical user interface (GUI) for each vehicle. This interface displays real-time data and triggers alerts when dangerous values are detected. It includes warnings for high gas concentration, obstacle proximity, or extreme temperatures.

The ultimate aim is to show how sharing real-time data between vehicles can lead to safer roads, less human error, and smarter transportation systems. This project serves as a basic proof-of-concept that could be expanded to include V2X (Vehicle-to-Everything) communication, AI-based control, GPS tracking, and cloud connectivity.

3.1.3. Assumptions

- Both devices are in range for Wi-Fi communication when they are in use.
- Sensors are properly calibrated and give accurate readings.
- The MQTT broker is kept online and operational for data transfer.
- Power supply to NodeMCU and other parts is stable during run time.
- The GUI would be executed on a system that supports Python and also has internet connectivity for image loading.

3.1.4. Dependencies

- Python packages like Tkinter, Paho-MQTT, and PIL (Pillow) for GUI programming.
- MQTT broker (e.g., broker.emqx.io) to publish and subscribe to sensor data.
- Arduino IDE and supported libraries to program the NodeMCU.
- Availability of Wi-Fi network for constant data exchange.
- Working hardware components (sensors, motor driver, ESP8266, etc.) for proper functioning.

3.2. FUNCTIONAL REQUIREMENTS

The functional requirements of the Vehicle-to-Vehicle (V2V) communication system indicate the primary tasks and proficiencies through which the software satisfies its main goals—improving road safety, managing traffic efficiently, and establishing secure, real-time communication between vehicles. These requirements guarantee that vehicles are able to operate smoothly in a common network, sharing important information like obstacle detection, dangerous gases, and weather changes. The system has to be able to receive data from different sensors, process that data correctly, and send it wirelessly using such protocols as MQTT. Besides, it should be able to create the right signals to the cars, which are near, in case there are threats or danger. Real-time capabilities, trustworthiness, and information security are key to these requirements. The system also has to have a user interface for controlling, and debugging thus the operation will be transparent. In summary, these functional requirements focus on the establishment of a coordination, intelligent vehicle network that not only-goes on without crashes, but also enables safe driving decisions.

3.2.1. Vehicle Motion Control

In Vehicle-to-Vehicle (V2V) communication systems, non-functional requirements are essential to making sure that the system performs well, consistently, and safely in real-world environments. They deal with how the system performs instead of what it does, dealing with key areas such as performance, security, dependability, scalability, and software quality. To provide high performance, the system should be able to process and pass on data from sensors in real time, enabling vehicles to react promptly to hindrances, danger in the surrounding environment, or threats in the vicinity. High reliability results from sound power management, very strong wireless communication, and rapid system restoration upon hardware or software failure. Just as crucial is data protection, which entails safeguarding confidential vehicle data using secure MQTT authentication, TLS encryption, and individual communication channels for every vehicle to prevent data collision or abuse. These security features will deter unauthorized use, data tampering, and cyberattacks that might harm passengers or interfere with traffic systems. In addition, non-functional requirements also encompass software design aspects like user-friendly interfaces, maintainable codebases, and modular architecture facilitating future updates and integrations. With proper handling of these requirements, the V2V system not only guarantees real-time and accurate data exchange but also helps in establishing a safe, scalable, and intelligent transport infrastructure in compliance with the visions of smart mobility and contemporary city development.

3.2.2. Sensor Data Collection

Each car is equipped with many sensors that are meant to monitor environmental factors and detect obstacles. The information collected is transmitted to the other car for processing.
Major Sensors and Their Uses

1. **DHT11 (humidity and temperature sensor)**
   Senses temperature and humidity of the ambient environment. Sends information to the integrated controller for immediate observation.
   Trigger: The alarm is triggered when the temperature is above 50°C.
2. **Gas Sensor (MQ Series)**
   It senses toxic gas levels in the atmosphere.Provides gas levels in ppm (parts per million).
   Trigger: Alarm is triggered when the gas concentration is above 500 ppm.
3. **Ultrasonic Sensor**
   Identifies road obstructions ahead of the car.Measures the distance from the object.
   Trigger: When the distance is less than 10 cm, the car stops or changes its path.

3.2.3. Real-Time Wireless Communication

The cars must send sensor information in real time through a wireless communication protocol. For the sake of effective data transfer, MQTT (Message Queuing Telemetry Transport) is used.
Key Features:

1. All of the cars feature an ESP8266 Wi-Fi module for interconnectivity.
2. Sensor data is pushed to an MQTT broker, e.g., EMQX Cloud.
3. The second vehicle subscribes to the data and updates its dashboard.
4. Low latency communication guarantees prompt response to danger.

MQTT Topics Used:

Data Type	**Topic Name**
Temperature	robot/carA/temperature
Humidity	robot/carA/humidity

Gas Levels	robot/carA/gas
Obstacle Information	robot/carB/obstacle

Table 3: MQTT Topics

Trigger Conditions: When new sensor data arrives, it is automatically sent to the MQTT broker. Vehicles subscribe to the necessary topics to receive latest updates.

3.2.4. Data Processing & Decision Making

The system needs to be aware of the incoming data and make smart decisions for safety.

Processing Logic:

1. **Temperature Management**:
   If temp > 50°C, create an alert notification.If normal, display current temperature on UI.
2. **Humidity Monitoring**:
   If humidity is < 30%, report dry conditions. If normal, display humidity level on UI.
3. **Gas Concentration Test:**
   If more than 500 ppm gas concentration is found, report a gas leakage. If safe, display gas concentration on UI.
4. **Obstacle Avoidance**:
   Turn or stop if less than 10 cm.If no obstruction found, resume normal operation.

Trigger Conditions:
Any sensor reading outside of the norm (high temperature, gas, or intrusions) triggers an alarm and modifies the car's movement logic.

3.2.5. User Interface (Gui)

A graphical user interface (GUI) is developed using Tkinter (Python) to display real-time sensor readings and alerts.
Key Functionalities:

➔ Shows real time temperature, humidity, gas, and obstructions.
➔ Updates UI text in real-time when there is new sensor data.

➔ Sends warning messages when unsafe conditions are detected.

Alert Messages:

Condition	**Alert Message**
Temperature > 50°C	High Temperature Alert!
Humidity < 30%	Low Humidity Alert!
Gas > 500 ppm	High Gas Concentration Detected!
Obstacle Identified	Vehicle Stopping! Obstacle Ahead!

Table 4: Alert Messages

Trigger Conditions:

Warnings are shown if sensor limits are exceeded. The UI is dynamically updated in real time from the MQTT data.

3.3. EXTERNAL INTERFACE REQUIREMENTS

External interface requirements specify the ways in which the Vehicle-to-Vehicle (V2V) communication system connects with people, hardware, software, networks, and security protocols. In essence, these interfaces are the backbone of communication between vehicles, and are the main contributors in delivering a comfortable user experience. Such interfaces are indispensable in enabling quick information transfer between the sensors, microcontrollers (the NodeMCU ESP8266 case) as well as the wireless networks (Wi-Fi) by the MQTT protocol. The system is obliged to ensure the compatibility with the sensors of different types, e.g. ultrasonic sensors, gas detectors, DHT11 environmental sensors so that it can gather the correct real-time data referring to the road as well as the vehicle.

The software part has also to be integrated with a graphical user interface (GUI) thus facilitating a more vivid and interactive way for the users to comprehend the system's conditions, sensor's reading and the alarm coming from the system. Besides this, the network interfaces in particular need to guarantee dependable and low-delay communication which should allow for quick response to potential hazards. Security features are crucial to keep the data intact and to avoid any unpermitted access during the process. The compatibility with certain external development environments such as the Arduino IDE is a factor that plays an important role for the deployment and updates of the codes. Summing up, well-thought-out external interfaces are the major players in assuring that V2V system is not only effective but also safe and can enhance user trust thus helping to achieve better road safety and traffic management.

3.3.1. User Interfaces

It is designed in a manner that it communicates live information on road conditions, hazards, and traffic alert information clearly, concisely, and non-diverting to the driver. The interface could either be visual-on a dashboard display or audio so the driver knows while keeping his eyes on the road. Some of the main features of the user interface include:

1. **Collision Warnings** – Gives auditory, visual, and haptic notifications of possible collisions. Applies sensor fusion and predictive algorithms to identify threats.

2. **Emergency Brake Notification** – Warns drivers when a front vehicle brakes hard with dashboard warnings and real-time V2V communication.

3. **Driver-Vehicle Interface (DVI) Design** – Provides minimal distraction, customizability, and redundant alerting through HUDs, AR overlays, and AI assistants.

4. **Cybersecurity & Privacy** – Practices end-to-end encryption, authentication, and fail-safe warnings to avoid hacking and illegal access.

5. **Future Trends** – AI-based UIs forecast driver actions, while 5G and cloud improve real-time information and traffic notifications.

3.3.2. Hardware Interfaces

The V2V communication system is made up of two independent vehicles, Vehicle A and Vehicle B, with necessary hardware elements for real-time data transfer and navigation.

1. **NodeMCU ESP8266 (Wi-Fi Controller)** – Responsible for communication and sensor data processing via the MQTT protocol.
2. **Motors & Wheels** – Two 100 RPM motors power the wheels, regulated by an L298N motor driver.
3. **Sensors**:

-DHT11 – Temperature and humidity measurement.
- Ultrasonic Sensor – Obstacle detection for collision prevention.
- Gas Sensor – Air quality and gas leak detection.

4. **Power Supply** – Three 3.7V batteries supply power to the system with voltage regulation.
5. **Communication Interface** – Wi-Fi (MQTT-based) facilitates real-time information exchange among vehicles, shown on a Python-based GUI.

3.3.3. Software Interfaces

The V2V communication system has multiple software modules to deal with sensor inputs, enable communications, and have real-time monitoring via a graphics interface.

1. **Embedded Firmware** – NodeMCU ESP8266 executes C/C++ programs based on the Arduino framework for reading sensor readings, motor controls, and broadcasting data using MQTT.
2. **Communication Protocol** – MQTT is employed for wireless data communication between Vehicle A and Vehicle B via Wi-Fi.
3. **Python GUI** – There is a Python-based graphical user interface that shows live vehicle information, sensor data, and alerts for obstacles, gas leaks, or temperature fluctuations.
4. **Data Processing** – Sensor readings are processed through onboard algorithms to initiate corresponding actions, e.g., halting the vehicle or sending alarms.
5. **Security Measures** – Encryption and basic authentication provide secure data communication between vehicles.

3.3.4. Network Interfaces

1. **Vehicle-to-Vehicle (V2V) Communication**
   This interface permits the direct transmission of data between Vehicle A and Vehicle B using the ESP8266 Wi-Fi module and MQTT protocol. Both cars broadcast their sensor reading (temperature, gas, obstacle detection) to the MQTT broker, and the other car subscribes to receive the updates. From the information provided, cars follow safety precautions such as braking in case of an obstacle.

2. **Vehicle-to-Broker (V2B) Communication**

Vehicles send their sensor information to an MQTT broker that acts as an intermediary to allow message passing. It is sent in JSON format over Wi-Fi, and it enables structured and efficient communication. This process makes it possible for vehicles to exchange information with other vehicles without being connected, thus strengthening the system.

3. **Vehicle-to-User Interface (V2UI) Communication**
   A Python Tkinter graphical user interface is used to display real-time car data for viewing. The GUI is subscribed to MQTT topics, continuously updating the screen with temperature, gas, and obstacle alerts. This interface assists people in analyzing the conditions of vehicles and alerting them in case safety-critical thresholds are violated, enhancing safety and choice. These network interfaces provide transparent and secure communication to enable real-time vehicle interaction for supporting enhanced road safety and avoidance of hazards.

3.3.5. Security Interfaces

1. **MQTT Authentication & Encrypted Communication**
   Cars utilize username-password authentication to securely communicate with the MQTT broker and block unauthorized access. TLS (Transport Layer Security) encrypts all data (temperature, gas, obstacle readings) flowing over the network, safeguarding against interception and cyber-attacks.

2. **Secure Topic Management & Network Security**
   Each car subscribes and publishes only its assigned MQTT topics to maintain the correct and secure communication boundaries. Network security is also improved through MAC address filtering, firewalls, and access points under control to avoid intrusion by unfamiliar devices**.**

3. **Anomaly Detection & Cyber Threat Alerts**
   The system continuously checks for anomalies like Denial-of-Service (DoS) attacks, unsuccessful repeated authentications, or data packet tampering. In case of any deviations being discovered, visual alarms are produced in the Python GUI to notify users of a suspected cybersecurity incident.

4. **Role-Based Access Control (RBAC)**
   MQTT topic access is controlled according to the role of a device. Only certain vehicles or systems are allowed to publish or subscribe to particular topics, minimizing the threat of illicit commands or data exposure.

5. **Secure Firmware Updates (FOTA)**
   Firmware updates are sent over-the-air via HTTPS encryption and digital signature checks. This guards against malicious firmware installations or man-in-the-middle attacks during system updates.

6. **Intrusion Detection System (IDS) & Logging**
   An intrusion detection system is a lightweight mechanism that runs constantly to look for abnormal patterns of traffic like spoofing, replay attacks, or packet flooding. Communication logs are kept for auditing purposes and post-event analysis.

7. **Fail-Safe Mode & Isolation Protocols**
   In case of detected cybersecurity breaches or attempted intrusions, the system switches to a fail-safe mode. The affected vehicle disconnects from the network and alerts the user, preventing the spread of threats to other vehicles.

3.4. NON- FUNCTIONAL REQUIREMENTS

In Vehicle-to-Vehicle (V2V) communication systems, non-functional requirements are essential to making sure that the system performs well, consistently, and safely in real-world environments. They deal with how the system performs instead of what it does, dealing with key areas such as performance, security, dependability, scalability, and software quality. To provide high performance, the system should be able to process and pass on data from sensors in real time, enabling vehicles to react promptly to hindrances, danger in the surrounding environment, or threats in the vicinity. High reliability results from sound power management, very strong wireless communication, and rapid system restoration upon hardware or software failure. Just as crucial is data protection, which entails safeguarding confidential vehicle data using secure MQTT authentication, TLS encryption, and individual communication channels for every vehicle to prevent data collision or abuse. These security features will deter unauthorized use, data tampering, and cyberattacks that might harm passengers or interfere with traffic systems. In addition, non-functional requirements also encompass software design aspects like user-friendly interfaces, maintainable codebases, and modular architecture facilitating future updates and integrations. With proper handling of these requirements, the V2V system not only guarantees real-time and accurate data exchange but also helps in establishing a safe, scalable, and intelligent transport infrastructure in compliance with the visions of smart mobility and contemporary city development.

3.4.1. Performance & Reliability

Performance is a fundamental aspect of V2V communication systems, as it directly affects how efficiently and quickly information can be exchanged between vehicles, which is vital for real-time decision-making and safety.

1. The system must process and transfer sensor data (temperature, gas, obstacles) in real-time so that the vehicle does not lag behind in decisions.
2. There should be an uninterrupted Wi-Fi connection, and the appliance also has to work on low signal strength.
3. Power supply must be stable, and recovery must be quick in the event of power failure or hardware failure.

3.4.2. Security & Data Protection

Security in V2V communication is critical to protect sensitive information and to prevent malicious activities that could endanger lives. These security requirements

ensure that only legitimate users have access to the network and that data remains confidential and protected.

1. The communication has to be secure using MQTT authentication to avoid unauthorized access.
2. Data must be encrypted (TLS) so that hackers cannot change or steal car data.
3. Each vehicle will have to be designated special MQTT topics to avoid data confusion or false signals.

3.4.3. Scalability & Ease of Use

1. It should be easy to add additional cars or sensors in the future without major modification.
2. The Python GUI must be interactive, showing live data in a legible format with alerts for limiting conditions.
3. The hardware components (motors, sensors, ESP8266) should be easy to replace or update in the event of long-term use.
4. These functional requirements guarantee that the system is efficient, secure, scalable, and end-user friendly, thus making it appropriate for real-time automotive communication.

3.5. SYSTEM REQUIREMENTS

This section will describe the hardware and software requirements to run the V2V communication system. Such requirements will support an effective simulation, testing, and deployment to ensure that the system meets performance and safety standards.The key needs there include communication modules, simulation tools, data storage, processing power, and operation in realistic traffic conditions. With such a requirement meeting above needs, reliable V2V communications will support enhanced road safety and traffic handling.

3.5.1. Hardware Requirements

This includes all the hardware components needed to build and implement the V2V communication system.

1. **Prototype parts & Vehicles**

- ➢ Two small vehicles (Vehicle A & Vehicle B) act as communicating entities. Each vehicle consists of three support wheels and two motors (each 100 RPM) to drive.
- ➢ A motor driver (L298N) is employed to drive the motors in various speeds and directions.

2. **Sensors for Environmental Monitoring**
   - ➢ DHT11 Sensor: Temperature and humidity, ideal for weather-conscious driving.
   - ➢ Ultrasonic Sensor: Detects obstacles to prevent collision.
   - ➢ Gas Sensor (MQ-2 or MQ-135): Senses harmful gas levels, alerting vehicles to dangerous air conditions.

3. **Communication & Power Elements**
   - ➢ ESP8266 NodeMCU: Used as the Wi-Fi module to provide wireless communication between the vehicles.
   - ➢ Battery Pack (3×3.7V Li-ion): Drives motors, sensors, and communication modules.

3.5.2. Software Requirements

This includes the whole software tools, programming languages, and protocols utilized in the project.

1. **Development Platforms & Programming Languages**
   - ➢ **Arduino IDE**:
     For writing ESP8266 code and interfacing sensors with the vehicle system.
   - ➢ **Python (Tkinter, Paho-MQTT, JSON)**:
     Tkinter: To create the graphical user interface (GUI).
     Paho-MQTT: To manage communication via MQTT protocol.
     JSON: For compact storage and processing of sensor data.

2. **Message Exchange Protocol**
   - ➢ **MQTT Broker (Mosquitto or HiveMQ)**:
     Serves as the focal point for communication among vehicles.
     Facilitates successful message communication between Vehicle A & B.

3. **Networking & Connectivity**

- **Wi-Fi Network**:
  Maintains the vehicles connected to the MQTT broker for real-time communication.

3.5.3. Network and Communication Needs

This enables secure and real-time data sharing between vehicles

1. **Wi-Fi Connectivity**
   The ESP8266 module should always be connected to the same Wi-Fi network in order to communicate smoothly.
   There should be a reliable network connection to facilitate seamless data transfer between the vehicles.

2. **MQTT Protocol for V2V Communication**
   Message Queuing Telemetry Transport (MQTT) is utilized for transmitting and receiving sensor data.
   Offers low latency communication, which is critical for real-time vehicle alerts.

3. **Individual MQTT Topics for Every Car**
   In order to prevent data confusion, each vehicle publishes and subscribes to various topics.

Low Latency & Real-Time Data Interchange : The communication must introduce a delay in milliseconds to make vehicles react in real time to danger.
Example:
Vehicle A reports to: VehicleA/SensorData
Vehicle B subscribes to: VehicleA/SensorData

3.6. SYSTEM IMPLEMENTATION PLAN

The implementation started with the hardware installation of two vehicles, each with NodeMCU ESP8266, sensors (DHT11, ultrasonic, gas), motors, and a motor driver. Microcontrollers were programmed in the Arduino IDE to read sensor readings and communicate through MQTT protocol over Wi-Fi. Each vehicle publishes its own reading and subscribes to the other's, allowing real-time Vehicle-to-Vehicle communication. A GUI using Python was also built to view this data and alert when safety limits are reached. Lastly, the system was thoroughly tested for proper sensor

reading, seamless vehicle operation, and proper data transfer between the two vehicles.

1. Hardware Assembly

- Assemble two 3-wheel cars (Vehicle A and Vehicle B) with chassis and 100 RPM DC motors.
- Mount sensors: DHT11 (humidity/temperature), ultrasonic sensor (obstacle detection), and gas sensor on both vehicles.
- Interconnect motors with the L298N motor driver and power all with 3.7V batteries.
- Integrate the NodeMCU ESP8266 board as the main controller for both vehicles.

2. Microcontroller Programming

- Utilize Arduino IDE to code and upload for:

  - Reading sensors (temperature, gas, and obstacle).
  - Motor movement control.
  - Sending and receiving data through MQTT protocol over Wi-Fi.

- Code both Vehicle A and Vehicle B to subscribe and publish to corresponding MQTT topics through broker.emqx.io.

3. Communication & Data Exchange

- Both vehicles should be connected to the same Wi-Fi network.
- Vehicle A and B publish their sensor data and subscribe to the other's topics.
- Data is communicated in real-time with MQTT protocol for V2V interaction including obstacle or gas warning**.**

4. Python GUI Monitoring

- Develop a Python-based GUI with Tkinter and Paho-MQTT to present incoming data from the other vehicle.
- Implement alerting mechanisms within the GUI (e.g., warning labels or popups) when temperature, gas, or obstacle values reach thresholds.

5. Testing & Validation

- Test components individually for functionality (motors, sensors, Wi-Fi).
- Authenticate real-time data exchange by simulating the detection of obstacles and noting the reaction from the other vehicle.

- Ensure GUI updates properly and alarms are fired as expected.

NO.	PHASE	ESTIMATED TIME	KEY DELIVERABLES
1.	Project Definition	1 week	Objectives and requirements
2.	Tool Selection	1 week	Selection of simulation tools and environment setup
3.	System Architecture Design	2 week	System architecture and data flow diagrams
4.	Scenario Development	2 week	Defined test scenarios and configurations
5.	Protocol and Message Setup	2 week	Communication protocols and message formats
6.	Model Implementation	3 week	Vehicle and traffic behavior models
7.	Integration and Testing	2 week	Integration report, performance benchmarks
8.	Simulation and Data Collection	3 week	Collected data logs and event logs
9.	Analysis and Optimization	2 week	Analysis report, optimized configurations
10.	Documentation and Reporting	2 week	Final report and presentation materials

Table 5: Implementation Plan

This plan enables a structured, phased approach to accomplish the simulation of a V2V communication system in a controlled environment while testing, optimizing, and documenting all the important aspects of V2V communication

4. SYSTEM ARCHITECTURE

The system design of the Vehicle-to-Vehicle (V2V) Communication Project has three main layers: sensing, communication, and user interface. Each vehicle, Vehicle A and Vehicle B, has essential sensors. These include a DHT11 for temperature and humidity, an ultrasonic sensor for obstacle detection, and a gas sensor for environmental monitoring. These sensors connect to a NodeMCU ESP8266 microcontroller, which reads and processes the data.

The two vehicles communicate using the MQTT protocol over Wi-Fi. Each NodeMCU links to a public MQTT broker (broker.emqx.io) and sends sensor data to specific topics. At the same time, each vehicle subscribes to the other's data, allowing real-time, two-way communication.

A Python-based GUI, built with Tkinter and Paho-MQTT, acts as the user interface. It shows live data from the other vehicle and alerts users when sensor readings go above set safety limits, such as high gas levels or nearby obstacles.

The NodeMCU also controls two 100 RPM motors through an L298N motor driver. This setup allows movement based on environmental inputs. This layered design supports autonomous sensing, real-time data exchange, and responsive decision-making, creating a solid foundation for smart vehicle communication and future V2X systems.

4.1. SYSTEM ARCHITECTURE

Description: High-level overview of hardware and software components.
Flow:

1. **Robot Cars** (ESP32/Arduino + Sensors) → **MQTT Broker** (Cloud/Local) → **Tkinter UI** (Python)

.

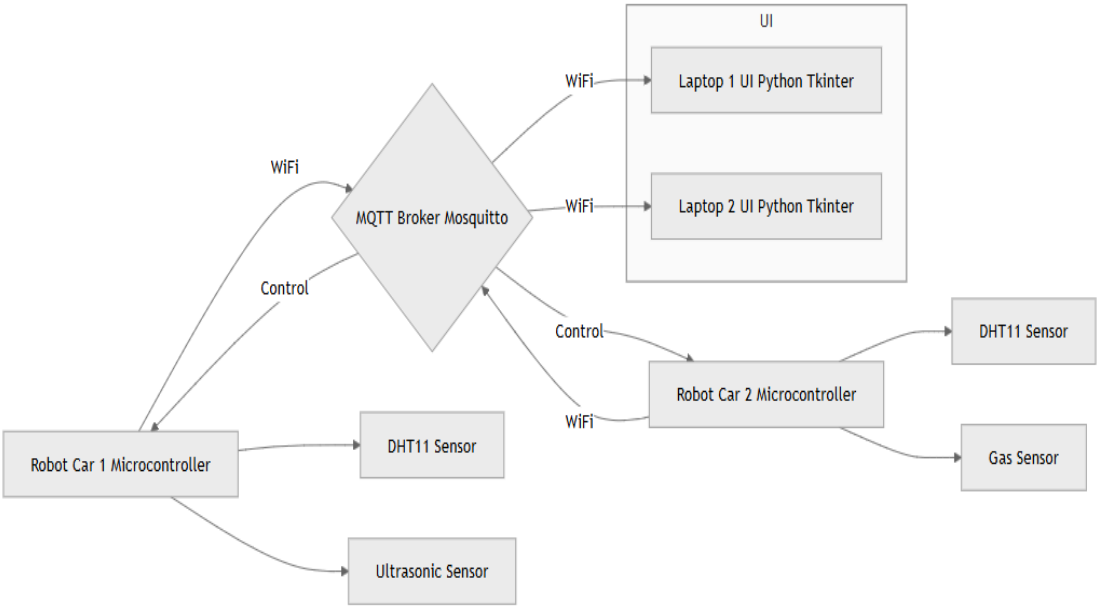


Fig 2. System Architecture

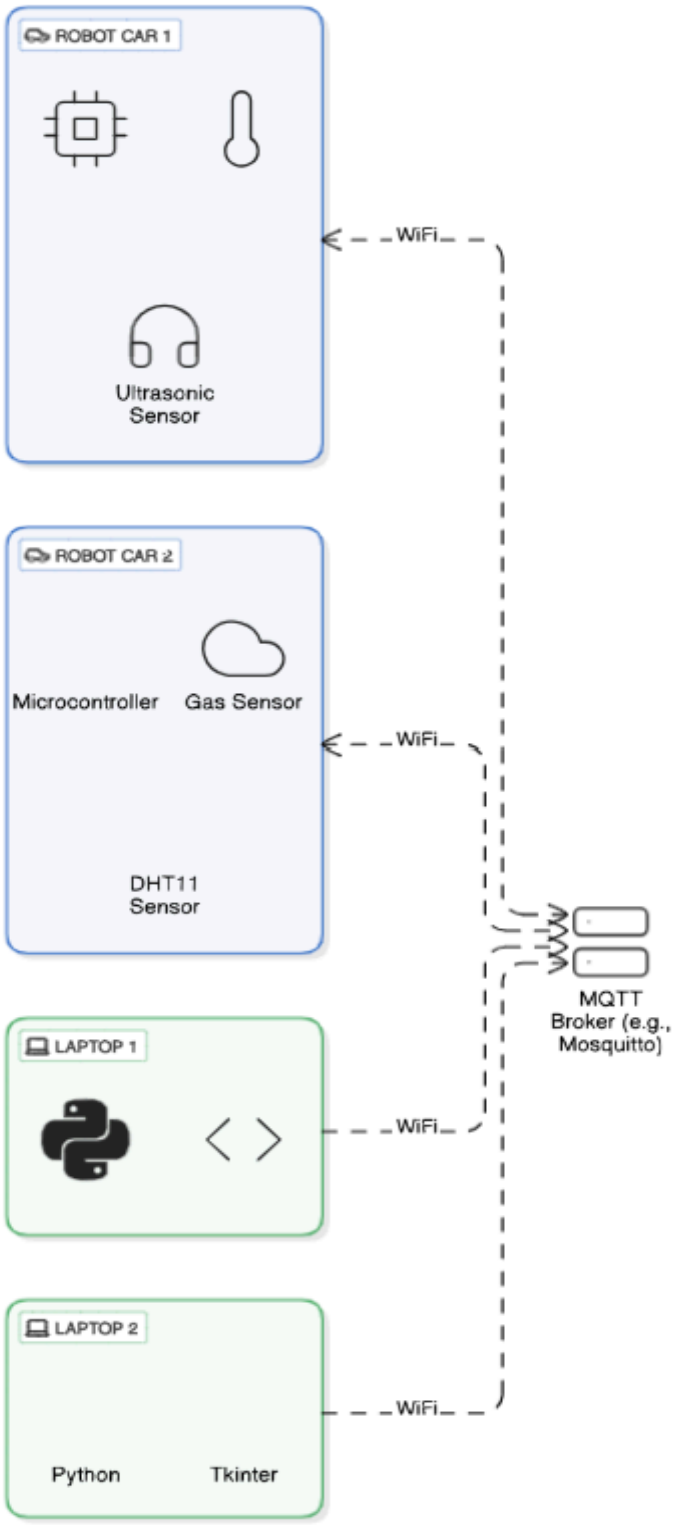


Fig 2.1. System Architecture 2

4.2. DATA FLOW DIAGRAM

Level 0 (Context Diagram):

- **External Entities**: Robot Cars, Laptops
- **Process**: MQTT Broker
- **Data Flows**: Sensor Data → Broker → UI

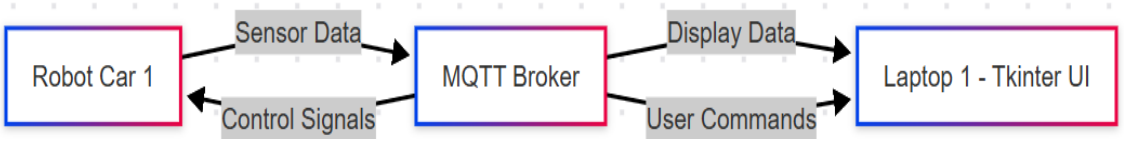


Fig 3. Data Flow Diagram

4.3. ENTITY RELATION DIAGRAM

Description: Only relevant if you're storing data in a database (e.g., SQLite).
Entities:

- **Robot** (robot_id, type)
- **SensorData** (timestamp, robot_id, temperature, humidity, distance, gas_level)

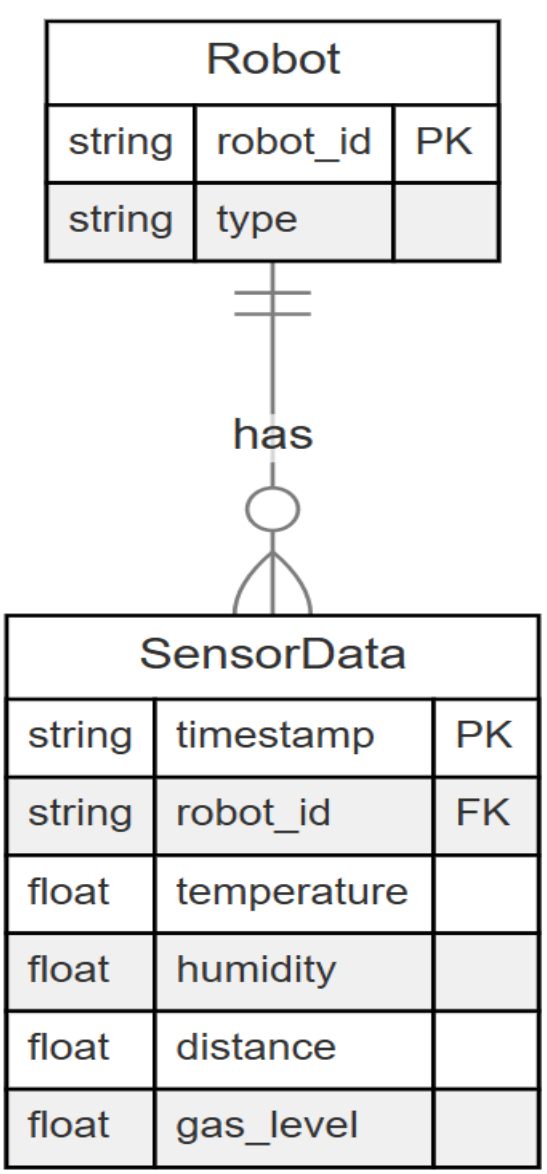


Fig 4. ER Diagram

4.4. UML DIAGRAMS

Unified Modeling Language (UML) diagrams for the V2V communication system depicts the structural and behavioral nature of this system. UML diagrams are essentially a blueprint about how all the components of the system interplay, how data flows between them, and how they react to various events. Such diagrams play a very crucial role in defining architecture when developing the system and ensure clear communication of design principles. It gives a pictorial form of interaction.

The most prominent UML diagrams for this V2V communication project are as mentioned below:

- **Use Case Diagram** : It depicts the overall actors-vehicle nodes, data processing units-and their interaction in the system and encapsulates the most important functionalities, for example, data exchange, alert generation, and vehicle status updates.

- **Class Diagram**: This diagram represents the fundamental classes of the V2V communication system along with their attributes and methods, their relationships. This diagram depicts a static structure placing emphasis on entities like Vehicle, Network Interface, and Traffic Controller.

- **Sequence Diagram**: This diagram elucidates interactions between objects during activities such as data transmission. It describes the sequence of messages between vehicles and infrastructures in terms of alerts or emergency notifications, thus offering real-time communication flow.

4.4.1. Use Case Diagram

Description: Shows interactions between users (actors) and the system components.
Components:

- **Actors**: Robot Car 1, Robot Car 2, User (via Laptop UI)
- **Use Cases**:
    - Publish sensor data (DHT11, Ultrasonic, Gas)
    - Subscribe to MQTT topics
    - Display data on Tkinter UI
    - Control robot movement (if applicable)

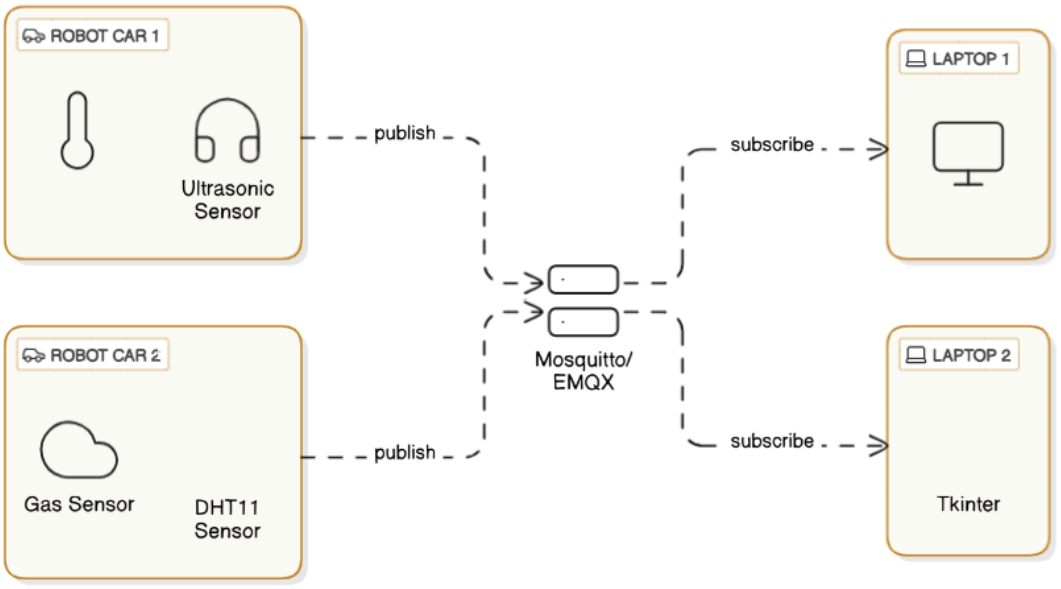


Fig 5. Use Case Diagram

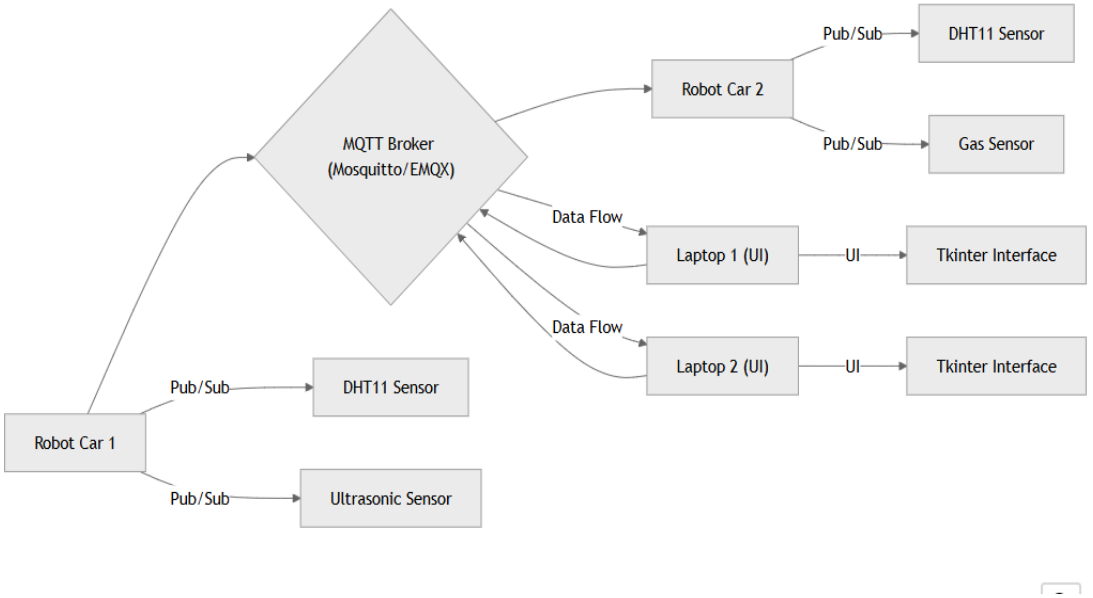


Fig 5.1. Use Case Diagram 2

4.4.2. Class Diagram

Description: Represents the structure of classes in the system.
Classes:

1. **RobotCar**
    - Attributes: robot_id, sensors
    - Methods: publish_data(), subscribe_data(), move()
2. **Sensor** (Parent)
    - Child Classes: DHT11, UltrasonicSensor, GasSensor
    - Methods: read_data()
3. **MQTTClient**
    - Attributes: broker_ip, port
    - Methods: connect(), publish(), subscribe()
4. **TkinterUI**
    - Attributes: window, data_labels
    - Methods: update_display(), handle_controls()

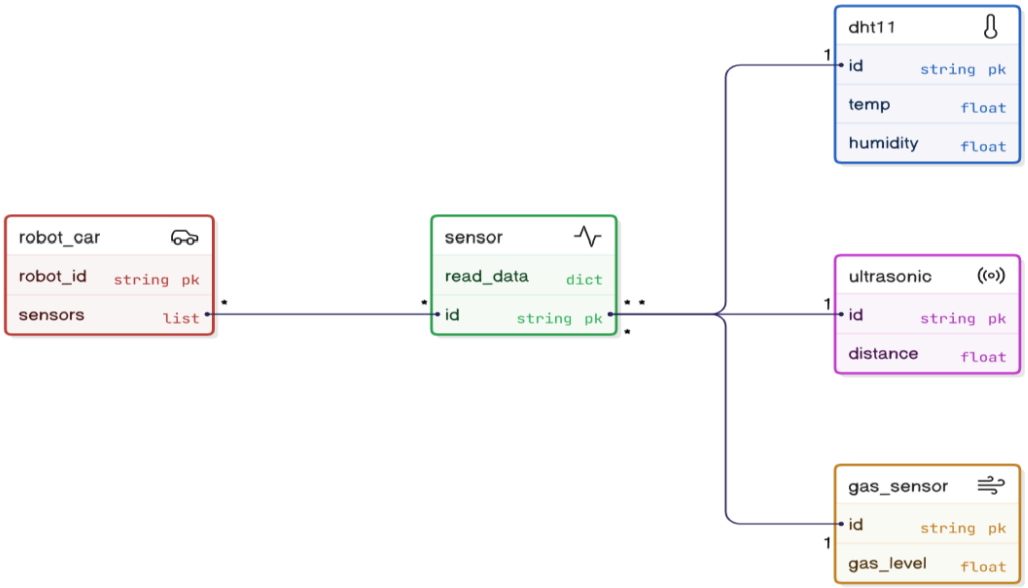


Fig 6. Class Diagram

4.4.3. Sequence Diagram

Description: Shows the step-by-step interaction between components.

Steps:

1. Robot 1 reads DHT11/Ultrasonic data.
2. Publishes to MQTT topic robot1/sensors.
3. Broker forwards to subscribed UIs and Robot 2.
4. Robot 2 reads Gas Sensor data and publishes to robot2/sensors.

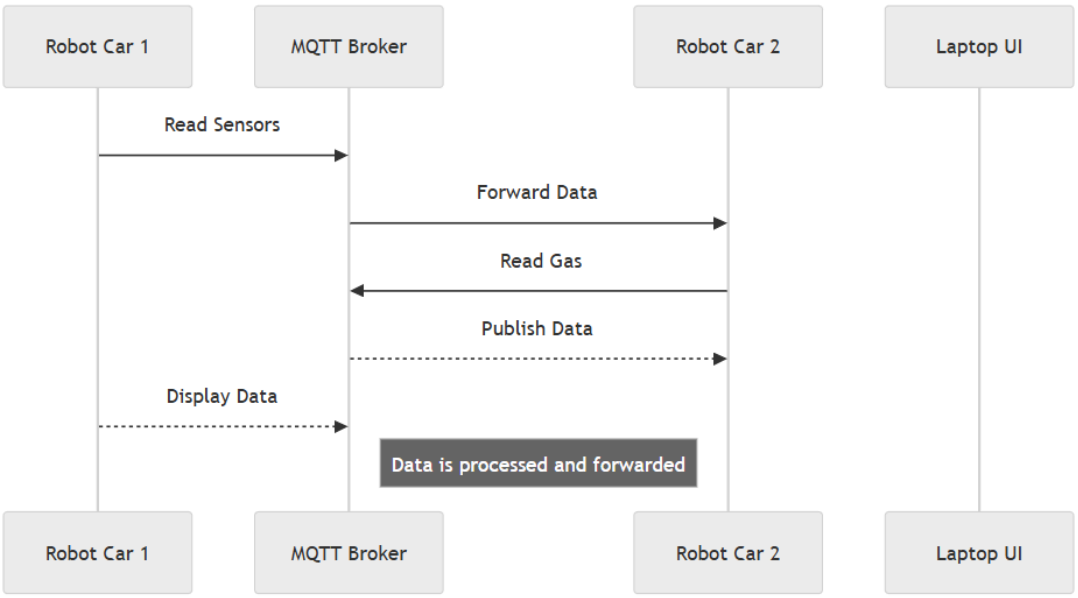


Fig 7. Sequence Diagram

5. OTHER SPECIFICATION

5.1. ADVANTAGES

1. **Enhanced Road Safety**
   The system also picks up on obstructions, leaks, and temperature variations in real-time.
   Alerts help prevent collisions, hazardous gas exposure, and extreme weather-related accidents.

2. **Real-Time Data Exchange**
   The ESP8266 and MQTT protocol provide the ability to share sensor readings in real time between cars. Low-latency communication will enable real-time decision-making with minimal reaction time in critical situations.

3. **Cost-Effective and Scalable**
   It uses low-cost hardware like ESP8266, DHT11, ultrasonic, and gas sensors, so it is low-cost. The system can be extended with additional vehicles and sensors without needing to make extreme changes.

4. **Less Traffic Congestion**
   Vehicles may exchange traffic conditions (e.g., road construction, accidents, or traffic jams) and suggest alternative routes. Facilitates efficient traffic control through elimination of excess congestion.

5. **Minimal Power Consumption**
   Employ low-power sensors and ESP8266 that can be powered by batteries. The system is meant for extended use without regular replacement of batteries.

5.2. LIMITATIONS

1. **Dependency on Network Connectivity**
   Wi-Fi or Internet connectivity is required for MQTT communication. Cars will be unable to trade sensor information sufficiently if the network is unstable or interrupted.

2. **Limited Range of Communication**
   The ESP8266 Wi-Fi module's operating range is shorter (typically 30-50 meters of free space). Cars need to be within the range of Wi-Fi for ongoing data transmission.

3. **Hardware Constraints**
   Low accuracy sensors such as DHT11 and ultrasonic may fail in bad weather conditions. The battery-powered system is prone to constant recharging, which affects long-term utilization.

4. **Security Risks**
   If there isn't proper authentication and encryption in place, then the system will be vulnerable to hacking.

5. **Unauthorized access**
   Manipulation of data can lead to spurious alarms and improper vehicle selection.

6. **Environmental Interference**
   Rain, fog, or electromagnetic interference can affect sensor readings, reducing system accuracy. Gas sensors may give false results due to external influences like humidity or temperature variation.

7. **Conclusion**
   The V2V system significantly improves road safety, sharing of real-time data, and cost reduction but is subject to some limitations like network, hardware dependency, and security vulnerabilities. Overcoming these limitations makes the system more efficient and feasible to implement under actual conditions.

6. CONCLUSION AND FUTURE WORK

6.1. CONCLUSION

The Vehicle-to-Vehicle (V2V) Communication System designed in this project effectively proves the feasibility of improving road safety, environmental consciousness, and autonomous decision-making through real-time data sharing among vehicles. The integration of temperature, humidity, gas, and obstacle detectors with the NodeMCU ESP8266 and the MQTT communication protocol in the project allows wireless communication between two vehicles to take place smoothly.

Employing a Python GUI supports the system even further with real-time visualization of data and prompt alerts, ensuring the system is user-friendly and informative. Hardware-based development guarantees practical utility and bridges the gap between theory and practice concerning V2V models and implementations.

This project sets the stage for subsequent improvements in intelligent transport systems, such as GPS incorporation, AI-driven decision making, and intelligent infrastructure interconnectivity. In general, it offers a proof of concept on how to construct smarter, safer, and more networked vehicular networks.

6.2. FUTURE WORK

The existing V2V communication system effectively proves real-time data sharing between two vehicles through sensors and Wi-Fi-based communication. Nevertheless, there are a few areas where the project can be enhanced and expanded in the future:

1. Integration with Cloud Services: The system can be integrated with cloud platforms such as AWS, Azure, or Google Cloud to store past sensor data, conduct advanced analytics, and facilitate remote monitoring.
2. Mobile Application Development: A mobile application can be created to receive real-time updates, alarms, and notifications from the vehicles, increasing the accessibility and convenience of the system.
3. GPS-Based Location Tracking: Installing GPS modules in both vehicles will enable location-based alarms and enhance the precision of collision avoidance and route planning.
4. AI-Based Decision Making: Machine learning models can be trained to forecast driver behavior or unsafe conditions and execute automated safety measures.
5. Battery Optimization: Incorporating low-power modes and intelligent energy management can increase battery life, particularly for extended-range applications.
6. Multi-Vehicle Communication:Expand the system to accommodate inter-vehicle communication, emulating an intelligent traffic system and increasing scalability.
7. Advanced Security Features: Integrating sophisticated encryption algorithms, secure authentication, and intrusion prevention systems will ensure resistance to cyber threats and unauthorized use.
8. Utilization of 5G or LTE-V2X: Substituting conventional Wi-Fi with 5G or LTE-V2X can enhance speed, range, and reliability of vehicle communication in real-world scenarios.

7. APPENDIX A

Project Title: Vehicle-to-Vehicle (V2V) Communication
Team Members:
Kushagra Duggar
Sahil Mahajan
Ridham Joshi
Sakshi Shisodiya

Institution: Stes's Smt. Kashibai Navale College Of Engineering
Vadgaon Bk, Off Sinhgad Road, Pune 411041

Department: Computer Engineering

Hardware Components Used:

1. NodeMCU ESP8266 Wi-Fi module
2. DHT11 temperature and humidity sensor
3. Ultrasonic distance sensor (HC-SR04)
4. MQ gas sensor
5. L298N motor driver
6. 100 RPM DC motors (2 per vehicle)
7. 3.7V Li-ion batteries (3 per vehicle)
8. 3-wheel chassis (per vehicle)
9. Jumper wires, resistors, and breadboard

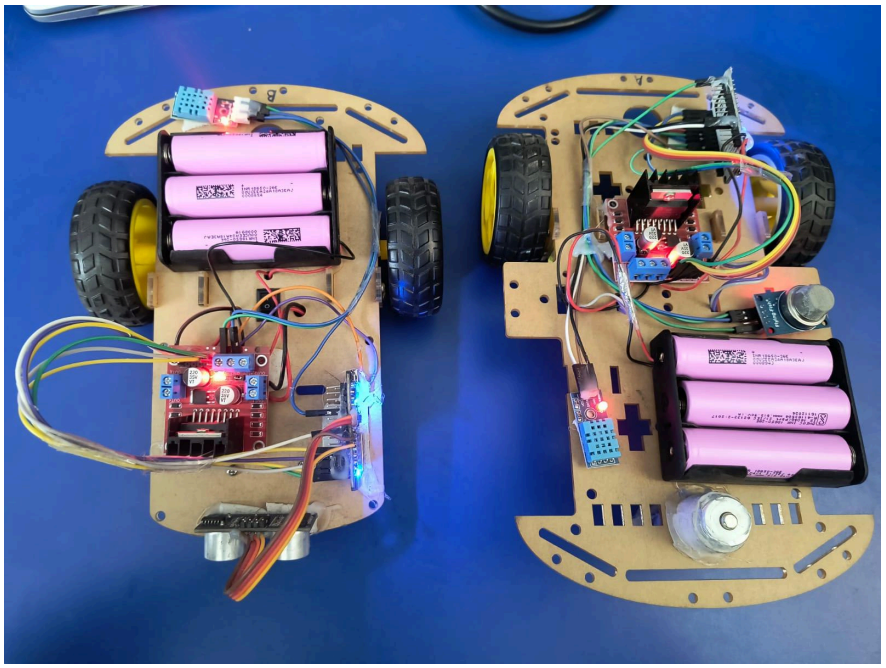
Software Used:

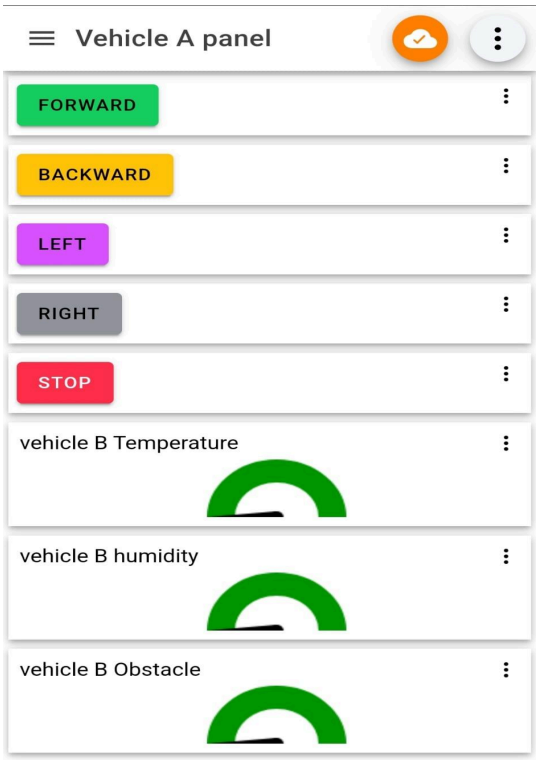
1. Arduino IDE (for embedded programming)
2. Python 3.x (for GUI dashboard)
3. Tkinter (GUI library)
4. Paho-MQTT (MQTT client for Python)
5. EMQX MQTT Broker (broker.emqx.io)
6. PIL (Python Imaging Library)

Communication Protocol:

1. MQTT (Message Queuing Telemetry Transport) over Wi-Fi
2. Tools Used:
3. Laptop/PC for programming and testing
4. USB cables for NodeMCU upload
5. Internet connection for MQTT and GUI operation.

Screenshots & Result :





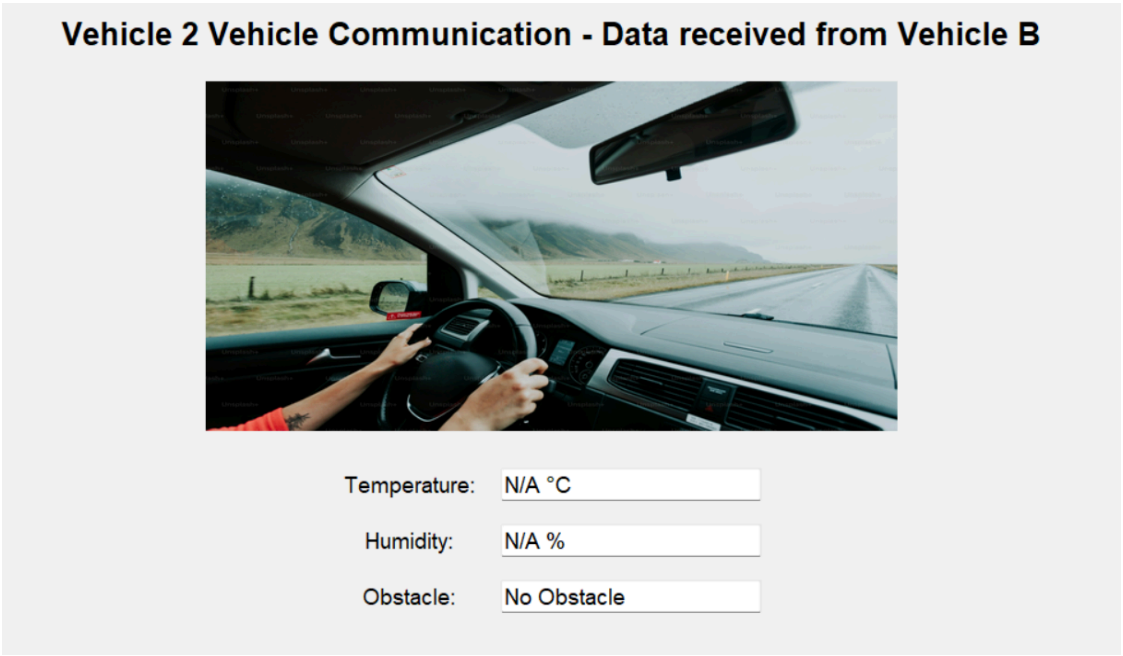


Fig 8. Output photos

8. APPENDIX B

Name of the journal:
International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025
https://www.irjmets.com/

Certificates:

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500216994

DOI: https://www.doi.org/10.56726/IRJMETS77322

Date: 27/05/2025

Certificate of Publication

This is to certify that author "**Prof. Arti Bhise** " with paper ID "**IRJMETS70600062572**" has published a paper entitled "VEHICLE TO VEHICLE COMMUNICATION" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025**

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500216994

DOI: https://www.doi.org/10.56726/IRJMETS77322

Date: 27/05/2025

Certificate of Publication

This is to certify that author "**Sakshi Shisodiya** " with paper ID "**IRJMETS70600062572**" has published a paper entitled "VEHICLE TO VEHICLE COMMUNICATION" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025**

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500216994

DOI: https://www.doi.org/10.56726/IRJMETS77322

Date: 27/05/2025

Certificate of Publication

*This is to certify that author "**Ridham Joshi** " with paper ID "**IRJMETS70600062572**" has published a paper entitled "VEHICLE TO VEHICLE COMMUNICATION" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025***

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500216994

DOI: https://www.doi.org/10.56726/IRJMETS77322

Date: 27/05/2025

Certificate of Publication

This is to certify that author "**Kushagra Duggar** " with paper ID "**IRJMETS70600062572**" has published a paper entitled "VEHICLE TO VEHICLE COMMUNICATION" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025**

Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

International Research Journal Of Modernization in Engineering Technology and Science

(Peer-Reviewed, Open Access, Fully Refereed International Journal)

e-ISSN: 2582-5208

Ref: IRJMETS/Certificate/Volume 07/Issue 05/70500216994

DOI: https://www.doi.org/10.56726/IRJMETS77322

Date: 27/05/2025

Certificate of Publication

This is to certify that author "**Sahil Mahajan** " with paper ID "**IRJMETS70600062572**" has published a paper entitled "VEHICLE TO VEHICLE COMMUNICATION" **in International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 07, Issue 05, May 2025**

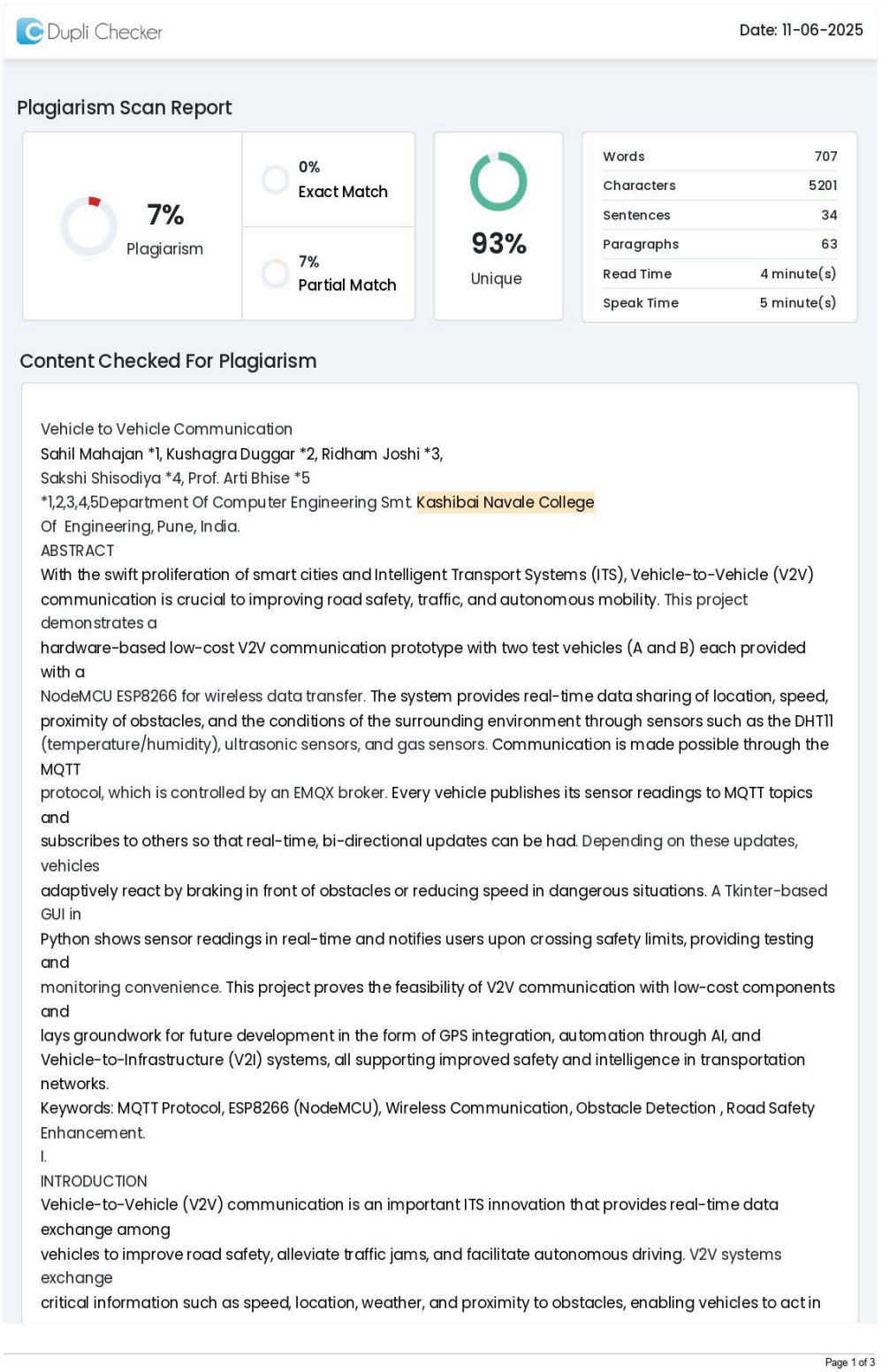
Editor in Chief

IRJMETS Impact Factor 8.187

We Wish For Your Better Future

www.irjmets.com

9. APPENDIX C



Dupli Checker

Date: 11-06-2025

Plagiarism Scan Report

7% Plagiarism

0% Exact Match

7% Partial Match

93% Unique

Words	707
Characters	5201
Sentences	34
Paragraphs	63
Read Time	4 minute(s)
Speak Time	5 minute(s)

Content Checked For Plagiarism

Vehicle to Vehicle Communication
Sahil Mahajan *1, Kushagra Duggar *2, Ridham Joshi *3,
Sakshi Shisodiya *4, Prof. Arti Bhise *5
*1,2,3,4,5Department Of Computer Engineering Smt. Kashibai Navale College
Of Engineering, Pune, India.
ABSTRACT
With the swift proliferation of smart cities and Intelligent Transport Systems (ITS), Vehicle-to-Vehicle (V2V) communication is crucial to improving road safety, traffic, and autonomous mobility. This project demonstrates a
hardware-based low-cost V2V communication prototype with two test vehicles (A and B) each provided with a
NodeMCU ESP8266 for wireless data transfer. The system provides real-time data sharing of location, speed, proximity of obstacles, and the conditions of the surrounding environment through sensors such as the DHT11 (temperature/humidity), ultrasonic sensors, and gas sensors. Communication is made possible through the MQTT
protocol, which is controlled by an EMQX broker. Every vehicle publishes its sensor readings to MQTT topics and
subscribes to others so that real-time, bi-directional updates can be had. Depending on these updates, vehicles
adaptively react by braking in front of obstacles or reducing speed in dangerous situations. A Tkinter-based GUI in
Python shows sensor readings in real-time and notifies users upon crossing safety limits, providing testing and
monitoring convenience. This project proves the feasibility of V2V communication with low-cost components and
lays groundwork for future development in the form of GPS integration, automation through AI, and Vehicle-to-Infrastructure (V2I) systems, all supporting improved safety and intelligence in transportation networks.
Keywords: MQTT Protocol, ESP8266 (NodeMCU), Wireless Communication, Obstacle Detection , Road Safety Enhancement.
I.
INTRODUCTION
Vehicle-to-Vehicle (V2V) communication is an important ITS innovation that provides real-time data exchange among
vehicles to improve road safety, alleviate traffic jams, and facilitate autonomous driving. V2V systems exchange
critical information such as speed, location, weather, and proximity to obstacles, enabling vehicles to act in

Page 1 of 3

concert
and react dynamically. This project offers a hardware implementation of V2V communication using low-cost and
low-power devices. Two prototype cars, Vehicle A and Vehicle B, are built with NodeMCU ESP8266
microcontrollers, each having a DHT11 temperature and humidity sensor, an ultrasonic sensor to detect obstacles, and
an environmental gas sensor. Motor drivers facilitate motion, and rechargeable batteries drive the cars independently.
Communication among the vehicles is done through the MQTT (Message Queuing Telemetry Transport) protocol due
to its lightness and adaptability to low-latency IoT networks. Topic-based messaging is handled by an EMQX broker
wherein each vehicle publishes its sensor readings and subscribes to updates from the other. It provides constant
bi-directional communication and real-time awareness of environmental and vehicular conditions. To depict and track
this communication, a GUI driven by Python with Tkinter visualizes real-time sensor readings and issues warnings
when thresholds are hit. This improves system usability and yields instantaneous feedback during run-time. Overall,
the project presents a practical demonstration of IoT, embedded systems, and wireless communication applied to
vehicle networking. It is a proof of concept for future evolution of connected vehicle technology, and it provides a
scalable platform for more sophisticated features such as smart routing and real-time hazard avoidance in smart city
environments.

(A) LITERATURE REVIEW:–

The literature review table emphasizes a comparative analysis of some of the many research efforts undertaken in the
field of VANETs, each adding value to a better communication, congestion control improvement, and trust and safety
features. The application utilizes hardware devices like ESP8266 (NodeMCU), DHT11 sensors, ultrasonic modules,
gas sensors, motors, and motor drivers to simulate real-time vehicle-to-vehicle communication and obstacle
detection, bridging theoretical concepts into practical verification.

e-ISSN: 2582-5208
International Research Journal of
Technology and Science
(Peer-Reviewed, Open Access,
Volume:07/Issue:20/May-2025
Modernization in Engineering
Fully Refereed International
Journal)
Impact Factor- 8.187
www.irjmets.com

Preliminary studies like the one by Kezia and Anusuya (2021) [1] utilized machine learning to forecast traffic
congestion as a function of vehicle density and channel busy ratio. Though their application was simulation-based, our
project takes this concept a step further by implementing ultrasonic sensors and DHT11 sensors in actual vehicles to
gather real-time environment and proximity information so that actual monitoring of likely congestion and collision

situations can be done. Likewise, Adwitiya Mukhopadhyay et al. (2020) [2] placed emphasis on TCP-based priority
queuing for priority handling of emergency messages; we mirror this by providing NodeMCU modules to transfer
urgent notifications wirelessly, thus simulating an actual emergency notification system between vehicles.

Matched Source

Similarity 2%

Title:ESP8266 Two Wheel Robot (NodeMCU and Stepper Motor)

Mar 23, 2022 · ESP8266 Two Wheel Robot (NodeMCU and Stepper Motor). Generally, robot cars are built on a chassis with 2 DC motor wheels and a bovine wheel.Missing: Vehicle

https://www.pcbway.com/project/shareproject/Two_Wheel_Robot_ESP8266_WiFi_d878b6ab.html

Similarity 2%

Title:Smt Kashibai Navale College of Engineering Vadgaon: Fees, Admission ...

Jun 3, 2025 · Smt Kashibai Navale College of Engineering Vadgaon is a Private Institute institute in Maharashtra offering over 16 courses. Read for details on Smt Kashibai Navale College of Engineering Vadgaon Fees, Admission 2025, Courses, Placement, Ranking, Reviews and more

https://collegedunia.com/college/15162-smt-kashibai-navale-college-of-engineering-skncoe-vadgaon-pune

Similarity 2%

Title:Channel busy ratio (CBR) over vehicle density. DCC Plain and ...

Channel busy ratio (CBR) over vehicle density. DCC Plain and RORA keep the load below the threshold of 0.65, whereas DCC Advanced and EDCA converges to a CBR of...

https://www.researchgate.net/figure/Channel-busy-ratio-CBR-over-vehicle-density-DCC-Plain-and-RORA-keep-the-load-below-the_fig1_316899234

Check By: Dupli Checker

Dupli Checker Date: 11-06-2025

Plagiarism Scan Report

19% Plagiarism	0% Exact Match
	19% Partial Match
81% Unique	

Words	896
Characters	6137
Sentences	74
Paragraphs	88
Read Time	5 minute(s)
Speak Time	6 minute(s)

Content Checked For Plagiarism

Description: High-level overview of hardware and software components.
The V2V system consists of two driverless vehicles with NodeMCU ESP8266 microcontrollers, sensors, and motor
drivers. Each vehicle takes environmental data through DHT11 for humidity/temperature, ultrasonic sensors for detecting obstacles, and gas sensors. The ESP8266 processes data and wireless communication through MQTT
protocol over Wi-Fi, allowing smooth real-time data exchange through an EMQX broker.
Fig -1: Architecture Diagram
The cars run on a 3-wheel platform driven by three 3.7V batteries and powered by two 100 RPM DC motors using an
L298N motor driver. Each car publishes sensor data to MQTT topics, which the other subscribes to. This means each
e-ISSN: 2582-5208
International Research Journal of
Technology and Science
(Peer-Reviewed, Open Access,
Volume:07/Issue:20/May-2025
Modernization in Engineering
Fully Refereed International
Journal)
Impact Factor- 8.187
www.irjmets.com
car can get real-time updates from the other, allowing for automatic reactions such as stopping or redirecting when
safety limits are exceeded.
For real-time monitoring of sensor values and alarms, a GUI based on Python's Tkinter is employed on a PC. It provides real-time data display and alerts users of any exceedance of pre-defined thresholds for gas, for example, or
proximity to obstacles. All these bring the IoT, wireless networking, and automation together into a scalable prototype
for intelligent vehicle communication systems of the future.
B. Process Flow :
The procedure starts when the vehicles are turned off and on using rechargeable batteries. After being powered on,
every NodeMCU (ESP8266) activates the attached sensors. These sensors—DHT11, ultrasonic, and gas sensors—begin watching over the environment in real-time. The sensor data is gathered and published through MQTT

Page 1 of 4

topics to an EMQX broker. Simultaneously, every vehicle subscribes to the other's data topics to achieve real-time
sharing of vital information. When it receives data, every vehicle processes it to identify threshold breaches (e.g.,
obstacle proximity too close, gas level too high). Depending on these inputs, the right action is initiated—like halting
the motors or showing an alert. A Python GUI created with Tkinter displays this data for tracking. This process
continuously runs, allowing dynamic, real-time V2V communication.

III.

A. Vehicle Design and Structure

IMPLEMENTATION

Two prototype cars, Vehicle A and Vehicle B, were conceptualized with a 3-wheel chassis for stability and seamless
motion. They are both driven by two 100 RPM DC motors, which are attached to the back wheels. Motor control is
implemented with an L298N motor driver module, enabling direction and speed control via digital signals from the
microcontroller.

Power supply is achieved with three 3.7V rechargeable lithium-ion batteries per vehicle. The batteries are enough to
supply the voltage and current required for running the motors, sensors, and NodeMCU board at the same time. The
design keeps all the parts firmly mounted while the vehicle is stable in motion.

Fig -2: Vehicle Design

B. Sensor Integration

For the purpose of detecting environmental and safety-related parameters, each vehicle comes with the following
sensors:

DHT11 Sensor: Detects temperature and humidity levels in order to track environmental conditions surrounding the
vehicle.

e-ISSN: 2582-5208
International Research Journal of
Technology and Science
(Peer-Reviewed, Open Access,
Volume:07/Issue:20/May-2025
Modernization in Engineering
Fully Refereed International
Journal)
Impact Factor- 8.187
www.irjmets.com

Ultrasonic Sensor (HC-SR04): Identifies obstacles in the path of the vehicle by computing the distance from ultrasonic
sound waves.

Gas Sensor (e.g., MQ-2/MQ-135): Identifies the presence of harmful gases such as smoke, LPG, or CO, allowing the
vehicle to be informed about possible environmental threats.

These sensors are interfaced directly to the NodeMCU ESP8266, which processes and gathers the data. The sensor
readings are updated at regular intervals and ready to be transmitted to the other vehicle.

C. Microcontroller and Communication Setup

The NodeMCU ESP8266 Wi-Fi development board acts as the central microcontroller in both the vehicles. It provides
built-in Wi-Fi capabilities and GPIO pins to connect sensors and motor drivers. The ESP8266 is coded to:

Read real-time sensor values Connect to Wi-Fi.
Publish sensor values to particular MQTT topics (e.g., robot/carA/temperature) on the public EMQX broker.
Subscribe to the respective topics from the second vehicle to get data.
This configuration enables two-way communication between the vehicles without an explicit physical connection,
providing the basis for V2V communication.
D. Embedded Programming (Arduino IDE)
The firmware for every NodeMCU is programmed using the Arduino programming language (C/C++) and uploaded
through the Arduino IDE. The firmware does the following: Initializes all sensors and communication protocols. Reads and buffers sensor data at intervals. Connects to the MQTT broker and publishes the sensor data. Subscribes to
MQTT topics to listen for incoming data from the other vehicle. Executes motor control or alert logic depending on
the data received (e.g., halting the vehicle on encountering an obstacle). This embedded logic makes the system
self-sufficient and able to execute without human interference once activated.
E. User Interface and Alerts (Python GUI)
A graphical user interface (GUI) is developed with Python's Tkinter library. Every vehicle has a GUI application running on a laptop or PC, which:
Connects to the MQTT broker via the Paho-MQTT library .Subscribes to messages published by the other car.
Displays real-time data like temperature, humidity, gas concentration, and obstacle range in a neat, easy-to-read
format. Utilizes message boxes and visual labels to notify the user when safety thresholds are breached (e.g., temperature > 50°C or gas > 500 ppm). This interface assists users in monitoring the system visually and taking
manual intervention if necessary.

Matched Source

Similarity 4%

Title:Volume 7 Issue 5, May 2025 - Nature

May 8, 2025 • Article 20 May 2025. Neuronal CCL2 responds to hyperglycaemia and contributes to anxiety disorders in the context of diabetes. The authors...Missing: 07/ | Show results with:

https://www.nature.com/natmetab/current-issue

Similarity 2%

Title:Data Integration | EMQX Docs

EMQX is an MQTT messaging platform that connects IoT devices through the MQTT protocol and transmits messages in real-time. Building on this, data integration...Missing: Wi- Fi, smooth

https://docs.emqx.com/en/emqx/latest/data-integration/data-bridges.html

Similarity 2%

Title:(PDF) 3-Wheel Omnidirectional Warehouse Automation Solution

It uses an Arduino UNO as its main controller, an L293D as a motor driver and three N20 100 RPM 6V DC Motors. It is powered by two 3.7V batteries and uses two...Missing: cars | Show results with:

https://www.academia.edu/88901924/3_Wheel_Omnidirectional_Warehouse_Automation_Solution

Similarity 2%

Title:(PDF) Vehicular Communication Systems: Enabling Technologies ...

This article surveys the state-of-the-art approaches, solutions, and technologies across a broad range of projects for vehicular communication systems.

https://www.researchgate.net/publication/224611555_Vehicular_Communication_Systems_Enabling_Technologies_Applications_and_Future_Outlook_on_Intelligent_Transportation

---

Similarity 2%

Title:L298N Motor Driver - Arduino Interface, How It Works, Codes ...

The L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module can drive DC motors that have...Missing: digital | Show results with:

https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge

---

Similarity 2%

Title:HC-SR04 Ultrasonic Distance Sensor and Arduino (Lesson #9)

Nov 24, 2022 Learn how to use the HC-SR04 ultrasonic distance sensor, also called a sonar sensor, to measure the distance to a nearby object using an...Missing: Identifies obstacles path vehicle computing

https://www.youtube.com/watch%3Fv%3Dn-gJ00GTsNg

Similarity 2%

Title:ESP32 Pinout: A Comprehensive Guide for Engineers - Wevolver

Jul 3, 2024 GPIO pins connect to motor drivers, ADC pins read data from distance sensors, and SPI or I2C pins handle communication with guidance systems.

https://www.wevolver.com/article/esp32-pinout-a-comprehensive-guide-for-engineers

---

Similarity 2%

Title:What Are the Three Types of Computer Viruses? - Insights - SSI

Once activated, the virus can execute malicious code and infect other system files. ... What was once self-contained and harmless has quickly escalated to...Missing: able interference

https://insider.ssi-net.com/insights/what-are-the-three-types-of-computer-viruses

---

Similarity 2%

Title:MQTT in Python with Paho Client: Beginner's Guide 2025 - EMQX

Mar 10, 2025 In this guide, we'll explore how to use the Paho MQTT Python client to connect an MQTT client to an MQTT broker, subscribe to topics, publish messages, and...Missing: car. | Show results with:

https://www.emqx.com/en/blog/how-to-use-mqtt-in-python

REFERENCES

[1] A Comparative Study on Machine Learning Algorithms for Congestion Control in VANET (2021)
[2] TCP and Priority Queue based Emergency Data Transmission in VANETs (2020)
[3] Estimating the Required Resources in a Vehicular Ad Hoc Network Using Queueing Theory Models (2019)
[4] Simulation of a driving assistant by means of a Multisensory Alert System (MAS) (2022)
[5] Decision Algorithm for Computational Offloading in Vehicular Fog Computing with Pedestrians (2020)
[6] Survey on Security Attacks in Software Defined VANETs
(2020)
[7] Local Traffic Density of Flow-Oriented Routing Protocol for VANETs (2019)
[8] Accident Detection and Warning Systems in VANET (2021)
[9] Emergency Packet Routing in Vehicular Networks for Collision Minimization in Highways (2021)
[10] Secure Data Encryption in VANET (2022)
[11] Reliability-Aware Multi-Objective Optimization-Based Routing Protocol for VANETs Using Enhanced Gaussian Mutation Harmony Searching (2022)
[12] A Comparative Study of Artificial Intelligence Algorithms for Network Traffic Prediction in VANET (2021)
[13] A Scenario-Based Trust Management Approach with 3R Message Categorization in VANETs (2022)
[14] A Scalable Blockchain-Based Trust Management in VANET Routing (2023)
[15] Context and Machine Learning-Based Trust Management Framework for Internet of Vehicles (IoV) (2022)
[16] Local Traffic Density of Flow-Oriented Routing Protocol for VANETs (2022)
[17] Hybrid Cooperative Vehicle Safety Communication Based on VANET (2021)
[18] Energy-Efficient Routing Protocol for VANETs Based on Clustering (2020)
[19] Blockchain-Enabled Secure Communication in VANETs (2022)
[20] Context-Aware Data Dissemination in VANETs Using Machine Learning (2021)
[21] Smart Traffic Management System Using VANETs and IoT Technologies (2022)
[22] Intelligent Transportation System Using VANET (2019)
[23] Adaptive Beaconing for VANETs (2021)
[24] Privacy-Preserving Data Sharing in VANETs (2022)
[25] S. Mumtaz, J. Rodriguez, and L. Dai, "Massive MIMO Systems for 5G and Beyond: Opportunities and Challenges with Vehicle-to-Everything Communications," IEEE Vehicular Technology Magazine, vol. 13, no. 1, pp. 94-101, 2018.

[26] H. Huang, X. Cheng, H. Shan, and W. Zhuang, "A Cross-Layer Resource Allocation Framework for V2V Communication in Cellular Networks," IEEE Transactions on Wireless Communications, vol. 17, no. 3, pp. 1795-1807, 2018.
[27] N. Cheng, W. Shi, P. Yang, X. Shen, and F. Bai, "Big Data Driven Vehicular Networks," IEEE Network, vol. 32, no. 6, pp. 160-167, 2018.
[28] R. Lu, X. Li, X. Liang, X. Shen, and X. Lin, "GRS: The Green, Reliability, and Security of Emerging Machine-to-Machine Communications," IEEE Communications Magazine, vol. 49, no. 4, pp. 28-35, 2011.
[29] H. Kato and K. Watanabe, "Adaptive Message Broadcasting Based on Road Traffic Conditions in VANETs," Journal of Intelligent Transportation Systems, vol. 22, no. 1, pp. 16-29, 2018.
[30] F. Foukalas, M. Roussaki, and T. Spyropoulos, "Network Slicing in 5G: Survey and Challenges," IEEE Communications Magazine, vol. 55, no. 5, pp. 94-100, 2017.
[31] Y. Zhang, W. Lou, and Y. Fang, "A Secure Incentive Protocol for Mobile Ad Hoc Networks," Wireless Networks, vol. 13, no. 5, pp. 569-582, 2007.
[32] P. Pathak, M. Dhamecha, and R. Dhamal, "IoT and Blockchain Integration in Vehicular Networks for Smart Transportation: A Comprehensive Survey," Internet of Things Journal, vol. 5, no. 1, pp. 112-123, 2021.
[33] L. Jia, W. Xu, Y. Liu, and J. Wan, "Design of an Efficient Cooperative Safety Message Broadcasting Scheme for VANETs," IEEE Communications Letters, vol. 21, no. 5, pp. 1153-1156, 2017.
[34] K. Zheng, L. Zhao, J. Mei, Y. Li, and J. Zhang, "Big Data-Driven Vehicular Networks: A Comprehensive Survey," IEEE Communications Magazine, vol. 56, no. 6, pp. 46-52, 2018.