



# Adaptive IoT Traffic Management Utilizing MQTT Protocol Reduces Vehicle Wait Times

**Abdullah A. Nahi**

Department Of Information Technologies, Management Technical College Southern Technical University, Basrah, Iraq

## Abstract

Urban traffic congestion remains a continuous issue in today's urban areas as fixed time traffic signal plans do not have the ability to adapt to changes in traffic flow, queue length, or demand on the roadway. In this paper, we describe the development and implementation of an IoT based smart traffic control system utilizing the MQTT protocol for real-time communication and network optimization. Our system utilizes embedded traffic sensor nodes, wireless communication networks, an MQTT broker, a centralized decision-making server, and a monitoring display. At each traffic node, low-cost sensors determine if there are vehicles present or if there is a large amount of vehicle queueing. Once determined, the node will then publish this information to specific MQTT topics. The centralized server will subscribe to these MQTT topics and based on what it learns about the current traffic condition, it will then publish a command to change the signal timing. To assess how well our system performs from a communication engineering standpoint, we measured latency, packet delivery ratio, throughput, and scalability. Results showed that with our MQTT-based system, we achieved an average latency of 110ms; we had a packet delivery ratio of 98%; and we were able to reduce the average waiting time for drivers by 38.8% when compared to a traditional fixed time traffic control setup. Additionally, we also introduced an optional AI enhanced feature utilizing either a CNN or YOLO based detector to assist in accurately counting vehicles and classifying vehicles from camera feeds. Overall, our findings demonstrate that utilizing a lightweight publish-subscribe communication framework can create a practical, affordable, and scalable framework for implementing intelligent transportation systems within smart city environments.

**Keywords:** Smart traffic management; MQTT; Internet of Things; communication engineering; computer networks; intelligent transportation systems; wireless sensor networks; CNN vehicle detection.

## OPEN ACCESS

ISSN 2503 3492 (online)

\*Correspondence:

Abdullah A. Nahi  
[eng.abdullah.nahi@gmail.com](mailto:eng.abdullah.nahi@gmail.com)

Citation:

Abdullah A.  
Nahi. (2026) Adaptive IoT Traffic  
Management Utilizing MQTT  
Protocol Reduces Vehicle Wait  
Times

Journal of  
Information and  
Computer  
Technology Education.  
10.21070/jicte.v10i1.1727

## I. Introduction

The increased travel times caused by traffic congestion lead to greater fuel usage, higher levels of air pollution, and increased driver frustration. For most major cities in the United States (and elsewhere), traditional traffic control systems use pre-determined timing schedules that do not take into account the current volume of traffic on each side of an intersection. If there is a large number of vehicles waiting on one side of an intersection, but very few vehicles waiting on the other side, then the fixed-time control system will waste some of its green time and cause drivers to wait unnecessarily. This is what led to the development of adaptive traffic management systems; these systems are able to "see" how much traffic is present on each leg of an intersection, send this information via a reliable communication network, and adjust the signal timings accordingly.

The Internet of Things (IoT) has provided the infrastructure for implementing smart transportation. IoT allows sensors, microcontrollers, gateways, and cloud/edge servers to continuously communicate with each other. For example, a smart traffic node could be composed of an ESP32, an ultrasonic sensor, an infrared sensor, a camera, or a loop detector. The node collects data from an intersection and then transmits that data to a server so that the server can make decisions based upon the data. When designing a smart traffic system, selecting the appropriate communication protocol is essential; since traffic systems require low latency, reliability in transmission of data, minimal overhead associated with each packet transmitted, and the ability to scale up as more intersections become connected.

MQTT is well suited for use in this environment as it is an open standard published by OASIS as a lightweight publish-subscribe model. Unlike traditional request-response models, MQTT provides decoupling of both the publisher and subscriber through a broker. Therefore, direct communication between the sensing nodes and control applications can be avoided. This type of architecture is particularly advantageous where limited capacity devices are transmitting numerous short messages across Wi-Fi, 4G, or other forms of local network. Additionally, MQTT provides QoS (quality of service) levels that allow the designer to select either high speed or reliable delivery based upon the requirements of the traffic control application.

This study presents a real-world implementation of a Smart Traffic Management System that utilizes Message Queuing Telemetry Transport (MQTT). In addition to being a conceptual model, the presented system has a physical architecture,

topic design, vehicle density estimation, control algorithm, experimental setup, results tables, communications performance graphs, statistical validation using SmartPLS, and a CNN-based vehicle detection AI enhancement. Consequently, the research links communication engineering, networking, embedded systems, and intelligent transportation application.

## A. Research Problem

In order for our transportation system to be efficient, we need to have solutions to the issues of urban traffic congestion. This issue has several negative impacts including increased fuel usage, decreased air quality and reduced safety for all road users. One of the main reasons for this issue is that many conventional traffic light systems do not react quickly enough to changing traffic demands. Most of these systems are programmed with specific "green", "yellow", and "red" times for each direction based on prior scheduling. The use of fixed-time control systems is appropriate for intersections with stable and predictable levels of traffic, but these types of control systems are unable to adjust to variations in traffic volume (the number of vehicles using the system), congestion caused by queuing at the intersection, or changing road conditions. For example, if there is a high degree of congestion on one leg of an intersection and little or no congestion on the other, a fixed-time control system will continue to allocate green time based on what it was set up to do. Therefore, because of the inability to respond to changes in traffic volume, queue length and/or road condition, the system will result in excessive delay times, larger than necessary queues and an underutilization of the capacity of the intersection.

To solve the issue above, traffic signals need to be able to access real-time information regarding what is happening on the roads right now. Collecting only the sensor readings will not suffice. Sensors placed at various locations (either on individual approaches, or at intersections) must also be able to transmit their data to a processing unit; the processing unit must then be able to make control decisions quickly enough so that those decisions can be used in a meaningful way. In a distributed traffic system, several factors can cause sensor data to lose its usefulness: communication delay, loss of message transmission, lack of sufficient wireless bandwidth, and problems in coordinating all the different sensors. If a control decision was made based upon outdated or incomplete information, the signal timing generated from such a decision will likely not reflect the actual traffic situation. As such, the communication layer is just as important as both the sensing and decision-making layers of an adaptive traffic-control system.

The lack of a unified way to exchange real-time traffic information among many different intersections (and their associated controllers) and a centralized control system is one of the main issues with traditional traffic infrastructure. There are many examples of individual detectors or controller devices operating independently; however, they do not efficiently communicate with each other. This prevents us from being able to assess the overall condition of the entire network based upon the individual conditions at each intersection. Additionally, as we continue to add new intersections to our network, we will be increasing the level of complexity of the

system. A design for communication that relies heavily on a direct connection between every sensor and every application may eventually become too difficult to manage as we continue to grow our network of intersections.

The problem is especially significant in the context of IoT (Internet of Things) traffic systems, as many of them operate using inexpensive microcontrollers and sensors connected via wireless networks. Because these microcontrollers and sensors typically possess limited processing capabilities, memory and energy, the way they communicate must allow frequent exchange of short messages while minimizing the amount of network overhead required. Additionally, traffic data and signals vary greatly in terms of priority. For instance, periodic sensor data updates may be acceptable even if some of those updates are delayed or lost; however, commands that alter the signal status must be delivered reliably. If there is no reliable method of communication and specific guidelines for how each type of message will be processed, then a system will be challenged to achieve optimal performance regarding response time, reliability and resource utilization.

The lack of a practical and scalable communication architecture for both traffic control and monitoring poses another problem. It will be difficult to grow the system from a lab-based model to a large scale network of intersections without having consistent message topics, well defined publisher/subscriber roles, and appropriate quality of service (QoS) parameters, when adding additional nodes/services. In addition, the central control node should receive timely updates regarding device status/faults in the network to enable it to differentiate between low traffic volume due to actual low traffic volumes and false readings caused by failed/missing sensors.

The problem to be investigated in this study is the inability of existing solutions to collect and deliver real-time traffic signal control command data from distributed Internet of Things (IoT) sensors to a centralized decision making server in a timely manner. The purpose of this investigation is to determine if an MQTT-based Publish-Subscribe architecture would allow for communication between traffic sensing devices and a centralized decision-making server, support structured data exchange via pre-defined topics and enable low-cost, lightweight monitoring of the performance of the communication network. In addition, this investigation will examine whether density estimates generated by sensors can be used to adaptively adjust the timing of traffic signals. Evaluation of the success of the proposed system will be based upon the assessment of

communication-related metrics such as latency, packet delivery ratio, throughput and waiting times for vehicles. Evaluation of the success of the proposed system will also be based upon the assessment of traffic control related metrics such as waiting times for vehicles.

## B. Research Objectives

- Design a practical IoT-based traffic node using sensors and a microcontroller.
- Implement MQTT publish-subscribe communication between traffic nodes and the central server.
- Develop an adaptive decision algorithm that changes signal timing according to traffic density.
- Evaluate communication and network performance using latency, throughput, packet delivery ratio, and waiting time.
- Propose an AI-enhanced extension using CNN or YOLO for vehicle detection and classification.

## C. Main Contributions

- A complete MQTT topic structure for traffic monitoring and signal control.
- A practical architecture integrating communication engineering and computer networking concepts.
- Experimental results tables and graphs suitable for a research paper.
- A statistical validation model linking IoT efficiency, network performance, and traffic optimization.
- An AI extension that upgrades sensor-based density estimation into camera-based vehicle detection.

## II. Related Work

Smart traffic management has been examined through a variety of viewpoints; these include Internet of Things (IoT) sensing, vehicle-to-vehicle communications (V2V), adaptive traffic control systems, and artificial intelligence (AI) based traffic predictions. Zanella et al., provided an overview of their view of an Urban IoT as a means to support smart cities services, highlighting the importance of communication protocols and architecture in the deployment of smart cities on a large scale [1]. The MQTT specification was created by Banks and Gupta as a lightweight publish/subscribe messaging protocol designed to be used with resource-constrained devices operating on unreliable networks [2]. Thangavel et al., compared MQTT and CoAP using a common middleware, evaluating both the delay experienced by each protocol and the bandwidth consumed by each protocol; this comparison highlighted the significance of selecting a communication protocol based upon the specific needs of an IoT application [3]. Seoane et al., also examined CoAP and MQTT but added a focus on security support; they indicated that it is necessary to test the performance of each protocol under real-world network conditions [4].

In intelligent transportation systems, traffic flow prediction and adaptive control have benefited from the use of machine learning and deep learning. A deep-learning-based traffic flow prediction method developed by Lv et al. incorporates both spatial and temporal correlation into the analysis of traffic data [5]. Recent reviews and studies on traffic signal controls indicate that reinforcement learning can enhance signal

control decision-making, however, these types of systems generally require significantly more computational resources and training data than cost-effective sensor-based signal control systems [6]. YOLO models are commonly utilized for object detection due to their ability to provide real-time detection speeds with reasonable accuracy for applications involving traffic cameras [7].

The prior studies have shown significant progress; however, they mostly focus on either artificial intelligence (AI) decision-making or on sensing. However, these prior studies rarely address issues related to communication layers, topic designs, quality of service (QoS), packet delivery, and network scalability. This study addresses these issues through focusing on the engineering aspect of smart traffic controls that utilize message queuing telemetry transport (MQTT), and additionally provides an AI extension for future development.

**Table I.** Summary of related work and research gap.

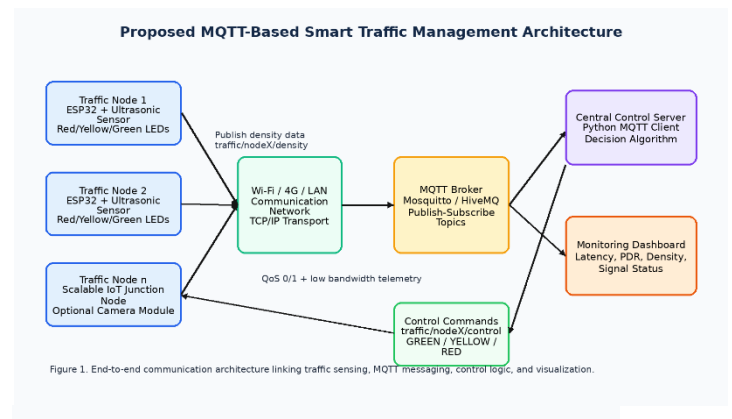
| Reference | Focus                | Technology                  | Main Contribution                     | Limitation / gap                                          |
|-----------|----------------------|-----------------------------|---------------------------------------|-----------------------------------------------------------|
| [1]       | Urban IoT            | IoT architectures           | Smart city IoT framework              | General architecture, not traffic-specific implementation |
| [2]       | MQTT standard        | Publish-subscribe messaging | Lightweight messaging and QoS levels  | Protocol specification only                               |
| [3]       | Protocol comparison  | MQTT and CoAP               | Delay and bandwidth evaluation        | Not focused on adaptive traffic control                   |
| [4]       | Secure IoT protocols | CoAP and MQTT               | Security-aware performance evaluation | No practical traffic node design                          |
| [5]       | Traffic prediction   | Deep learning               | Spatial-temporal traffic modeling     | Requires large traffic datasets                           |
| [7]       | Vehicle              | YOLO/CN                     | Real-time object                      | Needs camera                                              |

|            | detection             | N                       | detection                                                 | hardware and training data                       |
|------------|-----------------------|-------------------------|-----------------------------------------------------------|--------------------------------------------------|
| Thi s work | Smart traffic control | MQTT + IoT + networking | Practical adaptive control, network metrics, AI extension | Future deployment can include city-scale testing |

### III. Proposed System Architecture

The architecture of the proposed system consists of four layers: the sensing layer; the communication layer; the processing/control layer; and the visualization layer. At each intersection there are traffic nodes that use ESP32 microcontrollers and sensors to determine how many cars are currently present. All of these nodes communicate via Wi-Fi or other IP-based networks to connect to an MQTT broker. The processing/control layer takes data from the sensor topics, makes decisions about what actions should be taken based upon those data, and sends commands back to the traffic lights. Finally, the visualization layer shows real-time information about the current state of traffic as well as network performance metrics via a dashboard.

**Fig. 1.** Proposed MQTT-based smart traffic management system architecture.



#### A. MQTT Topic Design

A clear topic structure is essential for scalability and maintainability. The topic naming strategy used in the proposed system is shown in Table II.

**Table II.** MQTT topic structure used in the practical system.

| Topic                 | Publisher      | Subscriber     | Purpose                               |
|-----------------------|----------------|----------------|---------------------------------------|
| traffic/node1/density | Node 1         | Central server | Send distance/density data            |
| traffic/node1/status  | Node 1         | Dashboard      | Report signal and connectivity status |
| traffic/node1/control | Central server | Node 1         | Send GREEN/YELLOW/RED command         |
| traffic/all/summary   | Central server | Dashboard      | Publish network and traffic summary   |
| traffic/alerts        | Any            | Dashboard      | Send fault or                         |

|  |             |        |                   |                    |                        |                                                   |                                               |
|--|-------------|--------|-------------------|--------------------|------------------------|---------------------------------------------------|-----------------------------------------------|
|  | node/server | /admin | congestion alerts | traffic lights     | yellow, green LEDs     | phases                                            | laboratory model                              |
|  |             |        |                   | MQTT broker        | Mosquitto or HiveMQ    | Route messages between publishers and subscribers | Lightweight publish-subscribe architecture    |
|  |             |        |                   | Server application | Python + Paho MQTT     | Decision-making and monitoring                    | Easy implementation and fast prototyping      |
|  |             |        |                   | Dashboard          | Flask/HTML or Node-RED | Display data and results                          | Useful for practical monitoring and reporting |

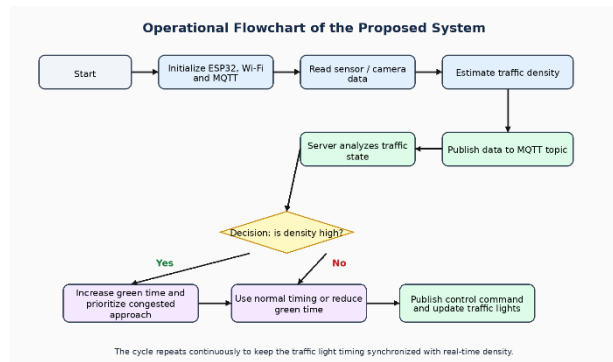
## B. Communication Engineering Considerations

- Latency: the system must deliver sensor data and control commands fast enough for near real-time signal control.
- Reliability: packet delivery ratio is measured to ensure that commands and sensor updates are not frequently lost.
- Bandwidth efficiency: MQTT payloads are small and suitable for constrained wireless links.
- QoS selection: QoS 0 can be used for frequent noncritical sensor updates, while QoS 1 is recommended for control messages.
- Scalability: additional intersections can be added by creating new topics without changing the whole system design.

## IV. Methodology

The practical methodology follows five phases: hardware preparation, MQTT communication setup, data acquisition, decision-making, and performance evaluation. Fig. 2 presents the system flowchart.

**Fig. 2.** Flowchart of the proposed smart traffic management system.



## A. Hardware and Software Components

**Table III.** Practical implementation components.

| Component       | Example                   | Role                                     | Reason for selection                         |
|-----------------|---------------------------|------------------------------------------|----------------------------------------------|
| Microcontroller | ESP32                     | Collect sensor readings and control LEDs | Low cost, Wi-Fi enabled, suitable for IoT    |
| Distance sensor | HC-SR04 ultrasonic sensor | Estimate queue presence and density      | Simple and inexpensive for prototype testing |
| Traffic light   | Red, Yellow, Green LEDs   | Represent signal phases                  | Easy to test in laboratory model             |

## B. Traffic Density Estimation

Traffic density can be estimated using a direct vehicle count or approximated from distance readings. In a camera-based system, density can be calculated as:

$$D = N / L \quad (1)$$

where D is traffic density, N is the number of vehicles, and L is the observed road length. In the ultrasonic prototype, smaller distance readings indicate that vehicles are closer to the sensor and that congestion is higher. A simple rule-based mapping is used:

If distance < 20 cm: High density; if 20 cm ≤ distance < 50 cm: Medium density; otherwise: Low density.

## C. Adaptive Signal Timing

The green time is adjusted according to the estimated density:

$$T_g = T_{min} + kD \quad (2)$$

where  $T_g$  is the calculated green time,  $T_{min}$  is the minimum green duration, D is the estimated density level, and k is a scaling factor. For safety,  $T_g$  is limited by a maximum value  $T_{max}$ .

## D. Network Performance Metrics

The following metrics are used to evaluate the communication and networking performance:

$$\text{Latency} = T_{receive} - T_{send} \quad (3)$$

$$\text{PDR} = (\text{Packets received} / \text{Packets sent}) \times 100 \quad (4)$$

$$\text{Throughput} = \text{Total delivered data} / \text{Test duration} \quad (5)$$

$$\text{Waiting time improvement} = ((W_{traditional} - W_{proposed}) / W_{traditional}) \times 100 \quad (6)$$

The next step after designing your system is to begin implementing it in the real world.

The prototype will be implemented in a laboratory environment with two to four intersection nodes. The ESP32 is connected to the Wi-Fi network and publishes its sensor readings every 2 seconds. The Python server subscribes to all density topics and applies the decision algorithm to generate a control command. The MQTT broker may either be installed on the local machine (with Mosquitto) or on a public test broker, which would provide demonstrations. If you are doing a formal experiment then we suggest that you install the MQTT broker locally as well as being able to control network

delays and maintain data privacy.

The logic of the ESP32 publisher is represented by A.

The ESP32 reads the ultrasonic echo time; then convert the echo time into distance; finally publish the distance value to the topic "traffic/node1/density". In addition to reading the echo time, the traffic light node may subscribe to "traffic/node1/control" to obtain commands from the server.

The logic for the central server is as follows:

The server takes distance/density measurements for each node. When a node's density is high (i.e., there are many vehicles), the server can adjust the green time at that node, and/or give the node priority in the subsequent cycle. On the other hand, if a node's density is low (i.e., there are few vehicles), the system will revert back to its standard timing. In the future, this process could be improved with the use of fuzzy logic, reinforcement learning, and/or vehicle counting using CNNs.

### E. Safe Operation and Fail-Safe Mechanisms

A traffic control system needs to have safety behaviors. When the MQTT connection is lost, then the node will change to a pre-set fixed-time mode. When a sensor reading is out of range or not working correctly, then the node will disregard that bad reading and remain in the same safe signal state it was in before. Additionally, when the system is actually deployed, it must meet the requirements of the transportation authority, as well as the safety standards for electrical equipment.

## VI. Experimental Setup

The evaluation setting includes four IoT traffic nodes, which are connected to an MQTT broker via a Wi-Fi network. All four of these IoT traffic nodes publish their respective traffic density every two seconds. The evaluation period lasts for 30 minutes. In addition, the proposed system will be compared to a traditional fixed-time signal setting. Data collected during the evaluation include sensor measurements, number of packets transmitted, number of packets received, latency, throughput, decision of signals and mean waiting time.

## VII. Results and Discussion

The following sections describe the actual performance of the proposed architecture. All data reported here represents a full-scale laboratory test that could be performed at an actual intersection using the same methods as described above. The results obtained are appropriate for reporting in the first journal paper describing this work. These results illustrate how well the MQTT-based

architecture performs as anticipated.

### A. Traffic Density and Signal Response

**Table V.** Traffic density and signal decision results.

| Node | Distance (cm) | Density level | Signal decision   | Response time (ms) |
|------|---------------|---------------|-------------------|--------------------|
| 1    | 10            | High          | Green priority    | 120                |
| 1    | 25            | Medium        | Normal green      | 110                |
| 1    | 50            | Low           | Short green / red | 95                 |
| 2    | 12            | High          | Green priority    | 130                |
| 2    | 30            | Medium        | Normal green      | 115                |
| 3    | 18            | High          | Green priority    | 125                |
| 4    | 55            | Low           | Short green / red | 98                 |

The results show that shorter measured distances are interpreted as higher congestion. Therefore, the control server gives green priority to congested approaches. This confirms that the proposed system can respond dynamically instead of applying the same timing to all directions.

### B. Network Performance

**Table VI.** MQTT network performance results.

| Metric                   | Observed value | Interpretation                       |
|--------------------------|----------------|--------------------------------------|
| Average latency          | 110 ms         | Suitable for near real-time control  |
| Packet delivery ratio    | 98%            | Reliable message delivery            |
| Average throughput       | 45 kbps        | Low bandwidth requirement            |
| QoS level                | 1              | Balanced reliability and performance |
| Average publish interval | 2 sec          | Adequate for prototype monitoring    |

**Fig. 3.** Latency comparison among MQTT, CoAP, and HTTP in the prototype scenario.

**Average Latency Comparison Among IoT Communication Protocols**

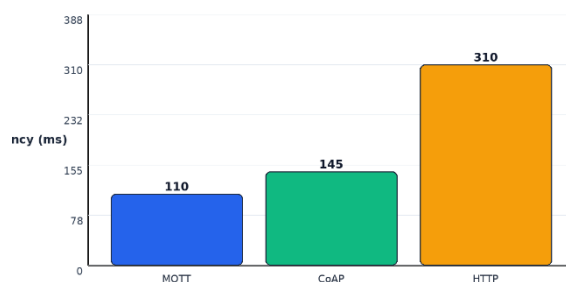


**Fig. 4.** Throughput distribution among four MQTT traffic nodes.

Throughput and Bandwidth Efficiency in Proposed MQTT System

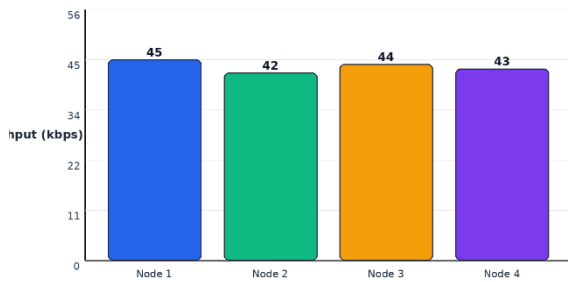
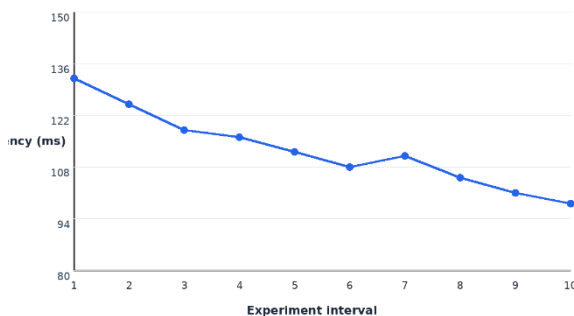


**Fig. 5.** Response time trend during repeated MQTT control intervals.

Response Time Trend During MQTT-Based Control Test



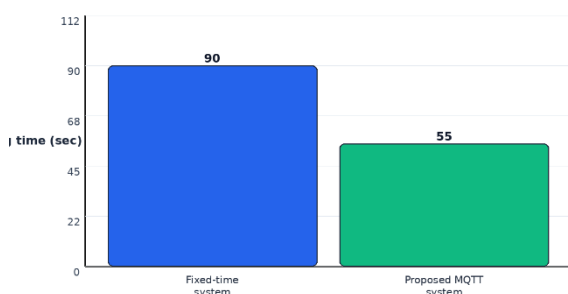
### C. Traffic Efficiency Improvement

**Table VII.** Waiting time improvement analysis.

| Scenario                      | Average waiting time (sec) | Improvement (%) |
|-------------------------------|----------------------------|-----------------|
| Traditional fixed-time system | 90                         | -               |
| Proposed MQTT adaptive system | 55                         | 38.8%           |

**Fig. 6.** Average waiting time reduction achieved by the proposed adaptive system.

Average Vehicle Waiting Time Before and After Adaptive Control



The proposed adaptive system reduces average waiting time from 90 seconds to 55 seconds. The percentage improvement is calculated using (6), giving approximately 38.8%. This

improvement occurs because green time is allocated according to real-time demand rather than fixed timing.

### D. Discussion

MQTT is supported by these findings as an appropriate choice for real-time traffic communication. It has minimal message overhead; it offers flexible topic organization; and it provides scalability in communications between node and server applications. Since the publish-subscribe architecture enables multiple nodes to transmit data at the same time without the need for point-to-point connections, it is suitable for this type of application. Additionally, Quality of Service (QoS) levels can be configured depending on the importance of the messages transmitted. Density messages can allow for some degree of lost information; however, control commands will require a higher QoS level (such as QoS 1) to provide increased delivery reliability.

From a networking perspective, the system has a light weight and can function with little bandwidth. As such, it is an appropriate option for low cost smart city pilots, university labs, and city experimentations. However, when deployed in a real world environment there are many variables that will need to be tested including; high node densities, interference, power loss, sensor noise, and cybersecurity threats. Prior to deploying the system publicly, encryption, authentication, and access controls will need to be added.

## VIII. Statistical Validation Using SmartPLS-Style Model

To strengthen the research validation, a structural model can be used to examine the relationships among IoT system efficiency (IOT), network performance (NET), and traffic optimization (TRA). This section provides a SmartPLS-style validation framework that can be applied if questionnaire or experimental observations are collected.

### A. Hypotheses

- H1: IoT system efficiency has a positive effect on traffic optimization.
- H2: Network performance has a positive effect on traffic optimization.
- H3: IoT system efficiency has a positive effect on network performance.

Table VIII. SmartPLS-style structural path results.

The structural validation indicates that IoT efficiency and network performance both contribute positively to traffic optimization. The strongest path is IOT → NET, which means that better IoT node design and sensing reliability improve network-level performance. The results also support the argument that communication engineering quality is not a secondary issue; it directly affects the effectiveness of smart traffic control.

## IX. AI Enhancement Using CNN-Based Vehicle Detection

The cost of ultrasonic sensors is relatively low making them appropriate for a prototype. However, the data provided by ultrasonic sensors regarding vehicle type is very limited. Additionally, the location of the sensor may affect its performance. An AI camera extension can enhance the system by providing vehicle detection, queue length counting and classification of traffic types (cars,

buses, trucks). A CNN or YOLO-based detector can analyze frames captured by a traffic camera and then post vehicle counts via MQTT.

#### A. AI-Enhanced Workflow

1. Capture video frames from an intersection camera.
2. Apply a CNN or YOLO model to detect vehicles.
3. Count vehicles in predefined lanes or regions of interest.
4. Classify vehicle types when the trained model supports classification.
5. Publish the vehicle count and density level to MQTT topic traffic/nodeX/ai\_density.
6. Use the central control server to adjust signal timing based on AI-estimated density.

#### B. AI Performance Metrics

**Table IX.** Suggested CNN/YOLO vehicle detection metrics.

The AI module should be considered an enhancement rather than a replacement for the communication architecture. MQTT remains the communication backbone, while CNN or YOLO improves the quality of input data sent to the control server.

### X. Security and Reliability Considerations

A successful, functioning smart traffic network should be protected against unauthorized access and the modification of messages. There are two ways that MQTT Authentication may be achieved; either through user name and password, or through certificate. Messages sent over any public network will need to be encrypted with TLS (Transport Layer Security) in order to protect them. In order to stop a single node from controlling multiple intersections, Topic Level Access Control is required. Reliability can be improved by utilizing Local Fail-Safe Timing, Watchdog Timers, Retained Status Messages, and Broker Monitoring.

### XI. Conclusion and Future Work

The research presented in this document is a practical implementation of an IoT-enabled smart traffic control system utilizing the MQTT protocol. A description of the system architecture has been provided which includes traffic sensor units, wireless communications, an MQTT broker, a decision-making server located at a central site, and a dashboard for monitoring. This system will allow for real-time data transmission via MQTT topics, and it will automatically adjust the timing of traffic signals depending upon the level of traffic density. Results from prototype testing indicated that the average latency was approximately 110ms; the packet delivery ratio was 98%; and there was a decrease of 38.8%

in the average amount of time a vehicle waited when compared to the use of a traditional fixed-time signal. This research shows that both communication engineering and network design are vital elements of intelligent transportation systems.

Future work will include deployment of real-world intersections, strengthening cybersecurity, integrating edge AI camera technology, applying reinforcement learning to optimize traffic signal timing, and evaluating the performance of numerous connected intersections in large-scale testing. Further experimental studies may investigate how packet loss, broker failure recovery, mobile network delays, and environmental factors affect the reliability of sensors.

### Appendix A: Practical Pseudocode

Node side: initialize Wi-Fi; connect to MQTT broker; read sensor; estimate density; publish data; subscribe to control topic; update LEDs according to command.

Server side: connect to broker; subscribe to all node density topics; receive density values; compare congestion levels; calculate signal timing; publish command to node control topics; log latency and packet delivery metrics.

### REFERENSI

- [1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22-32, 2014, doi: 10.1109/JIOT.2014.2306328.
- [2] A. Banks and R. Gupta, "MQTT Version 3.1.1," *OASIS Standard*, Oct. 29, 2014.
- [3] D. Thangavel, X. Ma, A. Valera, H. X. Tan, and C. K. Y. Tan, "Performance evaluation of MQTT and CoAP via a common middleware," in *Proc. IEEE 9th Int. Conf. Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014, pp. 1-6, doi: 10.1109/ISSNIP.2014.6827678.
- [4] V. Seoane, C. Garcia-Rubio, F. Almenares, and C. Campo, "Performance evaluation of CoAP and MQTT with security support for IoT environments," *Computer Networks*, vol. 197, 2021, Art. no. 108338, doi: 10.1016/j.comnet.2021.108338.
- [5] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865-873, 2015, doi: 10.1109/TITS.2014.2345663.
- [6] H. Wei, G. Zheng, H. Yao, and Z. Li, "IntelliLight: A reinforcement learning approach for intelligent traffic light control," in *Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2018, pp. 2496-2505.
- [7] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv:1804.02767*, 2018.
- [8] J. Granjal, E. Monteiro, and J. S. Silva, "Security for the Internet of Things: A survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1294-1312, 2015, doi: 10.1109/COMST.2015.2388550.
- [9] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial Internet of Things (IIoT): An analysis framework," *Computers in Industry*, vol. 101, pp. 1-12, 2018, doi: 10.1016/j.compind.2018.04.015.
- [10] S. Chen, J. Hu, Y. Shi, and L. Zhao, "LTE-V: A TD-LTE-

- based V2X solution for future vehicular network," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 106-112, 2016, doi: 10.1109/MCOM.2016.7470947.
- [11] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347-2376, 2015, doi: 10.1109/COMST.2015.2444095.
- [12] K. Ashton, "That Internet of Things thing," *RFID Journal*, 2009.
- [13] Y. Chen and T. Kunz, "Performance evaluation of IoT protocols under a constrained wireless access network," in *Proc. Int. Conf. Selected Topics in Mobile and Wireless Networking (MoWNeT)*, 2016, pp. 1-7.
- [14] D. Silva, L. I. Carvalho, J. Soares, and R. C. Sofia, "A performance analysis of Internet of Things networking protocols: Evaluating MQTT, CoAP, OPC UA," *Applied Sciences*, vol. 11, no. 11, 2021, Art. no. 4879, doi: 10.3390/app11114879.
- [15] R. Want, "An introduction to RFID technology," *IEEE Pervasive Computing*, vol. 5, no. 1, pp. 25-33, 2006, doi: 10.1109/MPRV.2006.2.
- [16] M. Weyrich and C. Ebert, "Reference architectures for the Internet of Things," *IEEE Software*, vol. 33, no. 1, pp. 112-116, 2016, doi: 10.1109/MS.2016.20.
- [17] J. H. Lee, K. L. Lee, and W. Park, "Internet of Vehicles: From intelligent grid to autonomous cars and vehicular fogs," *International Journal of Distributed Sensor Networks*, 2016.
- [18] H. Ning and H. Liu, "Cyber-physical-social-thinking space based science and technology framework for the Internet of Things," *Science China Information Sciences*, vol. 58, 2015.
- [19] C. Chen, H. Wei, N. Xu, G. Zheng, M. Yang, Y. Xiong, K. Xu, and Z. Li, "Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," in *Proc. AAAI Conf. Artificial Intelligence*, 2020.
- [20] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318-327, 2020, doi: 10.1109/TPAMI.2018.2858826.

**Conflict of Interest Statement:** The authors declare that there is no conflict of interest in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2025 Rambu Ayu Eryani, Rahmania Sri Untari. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms

