ILKOM Jurnal Ilmiah Vol. 18, No. 2, August 2026, pp. 320-332
Accredited 2nd by KEMDIKTISAINTEK No. 10/C/C3/DT.05.00/2025; E-ISSN 2548-7779 | P-ISSN 2087-1716

Optimization of Intelligent Traffic Control Based on iot and Reinforcement Learning for Congestion Reduction in Smart Cities

Tri Aristi Saputri [a,1,*]; Budi Sutomo [a,2]; Dimas Akbar Maulana [a,3]; Hendika Purnomo [a,4]
[a] *Universitas Dharma Wacana, Jl. Kenanga no. 3 Mulyojati 16c. West Metrot, Metro City, Lampung, Indonesia 34125*
[1] *aristy@dharmawacana.ac.id;* [2] *budi.atmel@gmail.com;* [3] *dimasmaulan76@gmail.com;* [4] *hendikapurnomo481@gmail.com*
* *Tri Aristi Saputri (aristy@dharmawacana.ac.id)*

Article history: Received December 12, 2025; Revised January 10, 2026; Accepted May 22, 2026; Available online August 10, 2026

Abstract

Traffic congestion has become a major challenge in Indonesian urban areas due to rapid vehicle growth and the limited adaptability of conventional traffic signal control systems. Most existing Deep Reinforcement Learning (DRL)-based traffic signal control studies adopt a free-phase selection approach, which assumes full agent freedom in determining signal phases — an assumption fundamentally incompatible with fixed phase-sequence regulations in Indonesian urban infrastructure — and rely on synthetic traffic data that fails to represent motorcycle-dominated traffic conditions. Furthermore, existing DQN-based approaches treat all traffic density conditions uniformly, without utilizing IoT-derived density categories for context-aware decision-making. To address these gaps, this study proposes a manual phase rotation mechanism with constrained actions (15, 30, and 60 seconds) compatible with existing fixed-phase infrastructure without hardware modifications, real-world IoT CCTV data from four intersections in Metro City processed using the YOLOv11 model to generate Low, Medium, and High traffic density categories as a representative training foundation for Indonesian urban traffic conditions, and a category-based action bias mechanism that adjusts DQN Q-value estimates according to IoT-derived traffic density, enabling context-aware signal duration selection. The DQN agent interacts with the SUMO simulation environment through the TraCI interface, receiving real-time traffic states comprising vehicle count, queue length, waiting time, average speed, density category, and delta queue, and selecting optimal green signal durations based on an epsilon-greedy exploration strategy and experience replay mechanism over 1,100 training episodes. Training yielded a 39.2% improvement in total reward and a 6.6% reduction in average waiting time. The best-performing model, obtained at episode 1050, achieved an 8.6% reduction in average waiting time and an 11.7% increase in traffic throughput compared to the fixed-time baseline. These results demonstrate that the proposed framework contributes three concrete advances for adaptive traffic signal control, a constrained-action DQN that is fully compatible with real-world fixed-phase infrastructure, a real-world IoT CCTV dataset as a representative data foundation for Indonesian traffic conditions, and a category-based bias mechanism for context-aware control — collectively offering a deployable, infrastructure-compatible, and replicable solution for traffic authorities and local governments advancing the smart city agenda in Indonesia.

Keywords: IoT; SUMO; Reinforcement Learning; DQN; Smart Transportation.

Introduction

Traffic congestion remains one of the major challenges in urban transportation management, particularly in Indonesian cities experiencing rapid population and vehicle growth [1]. Dense and dynamic traffic conditions reduce mobility efficiency, increase travel time, and contribute to greenhouse gas emissions and air pollution [2]. Conventional fixed-time traffic signal systems are considered inadequate for handling real-time traffic dynamics because they rely on static green signal durations [3], [4]. Within the smart city paradigm, the integration of Internet of Things (IoT) technology and artificial intelligence has emerged as a promising approach for developing adaptive and efficient transportation systems.[5]

Deep Reinforcement Learning (DRL), particularly the Deep Q-Network (DQN) algorithm, has been widely applied in Adaptive Traffic Signal Control (ATSC) research due to its capability to handle high-dimensional state spaces and discrete action spaces [3], [6]. However, several limitations in previous studies still hinder the practical deployment of RL-based traffic signal control systems, especially in Indonesian urban environments.

http://dx.doi.org/10.33096/ilkom.v18i2.3260.320-332

Most existing RL-based traffic signal control studies assume that the RL agent has full freedom to determine traffic signal phases at each simulation step, known as the free-phase selection approach [6], [7].This assumption is fundamentally incompatible with Indonesian urban intersections, where traffic signal systems operate under fixed phase-sequence regulations defined by traffic authorities, and hardware modifications require significant regulatory adjustments [8]. Furthermore, most previous studies rely primarily on synthetic or simulation-generated traffic data [9], [10], which fail to represent the heterogeneous and motorcycle-dominated traffic characteristics commonly found in Indonesian cities. Existing DQN-based approaches also treat all traffic density conditions uniformly, without utilizing IoT-derived traffic density categories for decision-making [3], [4]. These gaps collectively highlight the need for a constrained, infrastructure-compatible, and data-driven approach to adaptive traffic signal control in the Indonesian urban context.

To address these limitations, this study proposes three main novelties compared to existing DQN-based traffic signal control research. First, a manual phase rotation mechanism based on constrained actions is implemented, where the phase sequence remains fixed and the DQN agent is restricted to adjusting green signal durations through three discrete actions: 15 s, 30 s, and 60 s — eliminating the need for hardware modifications. Second, real-world IoT CCTV traffic data collected from four intersections in Metro City, Lampung, is utilized and processed using the YOLOv11 model to generate low, medium, and high traffic density categories, replacing synthetic simulation data. Third, a category-based action bias mechanism is introduced to adjust DQN decision-making according to IoT-derived traffic density categories, enabling more context-aware signal control.

The main contributions of this study are threefold. First, a DQN-based adaptive traffic signal control framework is developed using a constrained-action approach that is fully compatible with fixed-phase real-world traffic infrastructure, without requiring hardware or regulatory changes. Second, a real-world IoT CCTV traffic dataset collected from four operational intersections in Metro City is utilized for model training and evaluation, providing a more representative data foundation for Indonesian urban traffic conditions. Third, a category-based action bias mechanism is implemented to improve DQN adaptability and decision-making performance under varying traffic density conditions.

Based on the identified research gaps and proposed novelties, the objectives of this study are as follows: To develop an adaptive traffic signal control system using IoT CCTV data, the YOLOv11 vehicle detection model, and the Deep Q-Network (DQN) algorithm within the SUMO simulation environment. To implement a manual phase rotation mechanism with constrained actions as an infrastructure-compatible traffic signal control approach for existing fixed-phase intersections. To evaluate the effect of category-based action bias on DQN decision-making performance under varying real-world traffic density conditions.

Method

A. Research Stages

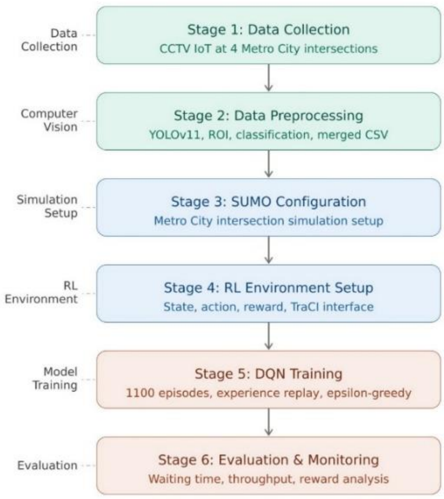

Figure 1. Research stages diagram

This research consists of six structured stages as illustrated in **Figure 1**. The overall methodology emphasizes a data-driven principle by utilizing real-world traffic data obtained from IoT CCTV devices at four strategic locations in Metro City [11]. In the first stage, traffic data are collected from four intersections in Metro City using IoT CCTV devices during both peak and off-peak periods to capture temporal variability in traffic conditions [12]. In the second stage, the collected video data are processed using the YOLOv11 model to detect and classify vehicles into three traffic density categories, namely Low, Medium, and High, with outputs stored in CSV format [13], [14]. In the third stage, the preprocessed dataset is used to configure the SUMO traffic simulation environment, where the CSV data serves as vehicle demand input to generate realistic traffic flow patterns [4], [15]. In the fourth stage, the reinforcement learning environment is defined by specifying the state, action, and reward components, integrated with SUMO through the TraCI interface to enable real-time interaction between the agent and the simulation [3], [10]. In the fifth stage, the DQN model is trained over 1,100 episodes using an epsilon-greedy strategy and experience replay mechanism to improve learning stability and policy performance [6], [7]. In the final stage, the trained model is evaluated using performance metrics including average waiting time, traffic throughput, and cumulative reward to assess the effectiveness of the proposed approach [16], [17].

B. System Architecture

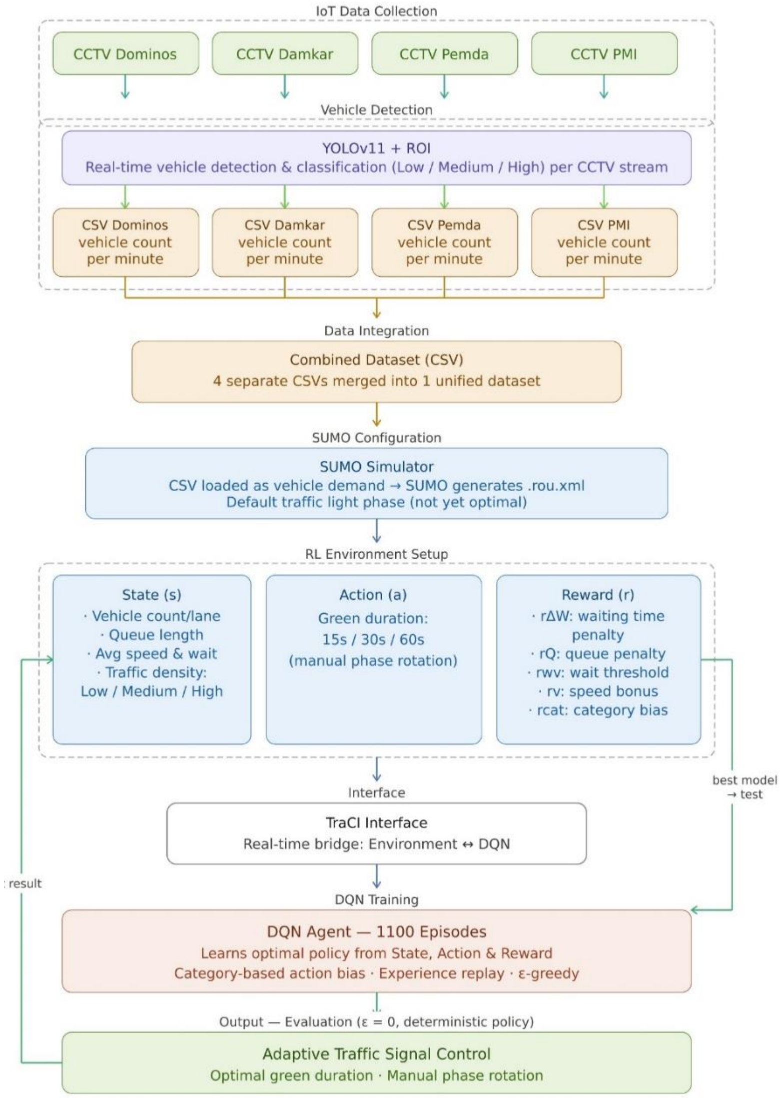

Figure 2. System Architecture

The proposed system architecture, illustrated in **Figure 2**, consists of five integrated stages. IoT CCTV data from four Metro City locations (Dominos, Damkar, Pemda, PMI) are processed using YOLOv11 with Region of Interest (ROI) filtering to classify traffic density into Low, Medium, and High categories, exported as CSV files [11],

[13].These files are integrated into a unified dataset used as vehicle demand input for the SUMO simulator, generating route configuration files (.rou.xml) [4], [15].

The RL environment defines state, action, and reward components, with the DQN agent interacting through the TraCI interface in real time. The agent is trained using ε-greedy exploration and category-based action bias, with the best model evaluated in deterministic mode (ε = 0) [3], [6], [7], [18].

C. Data

Primary data in this study were obtained from CCTV recordings integrated within an Internet of Things (IoT) network across several road segments in Metro City, namely Pemda, PMI, Damkar, and Domino's. These locations were selected based on their diverse traffic characteristics and varying levels of daily congestion, enabling them to represent the dynamics of urban traffic conditions [1], [11]. The selected intersections reflect different traffic patterns, including variations in vehicle density, flow distribution, and temporal congestion behavior, which are essential for developing a robust and generalizable traffic control model [12].

At each location, approximately 8–9 video recordings with a duration of 10 minutes each were collected during both peak and off-peak periods. This data collection strategy was designed to capture temporal variability in traffic conditions and ensure a balanced representation of both high-demand and low-demand scenarios [2]. In total, 32 to 36 IoT-based CCTV video samples were obtained, providing a sufficiently diverse dataset to support subsequent analysis and simulation processes [1], [14].

Figure 3. CCTV in front of Damkar Metro City

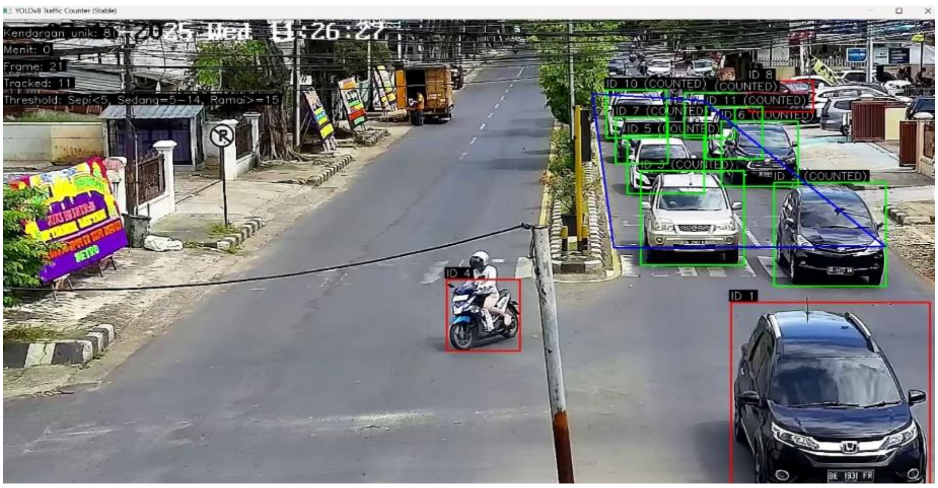

Figure 4. CCTV at Dominos traffic light

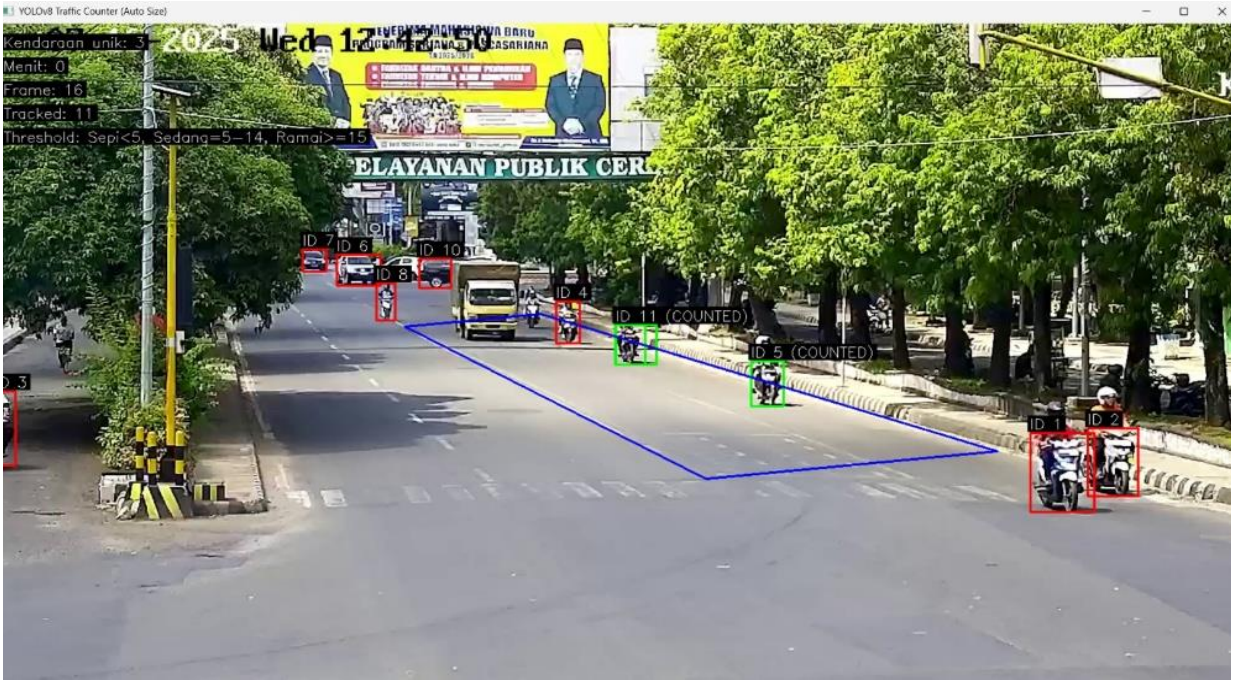

Figure 5. CCTV at Pemda traffic

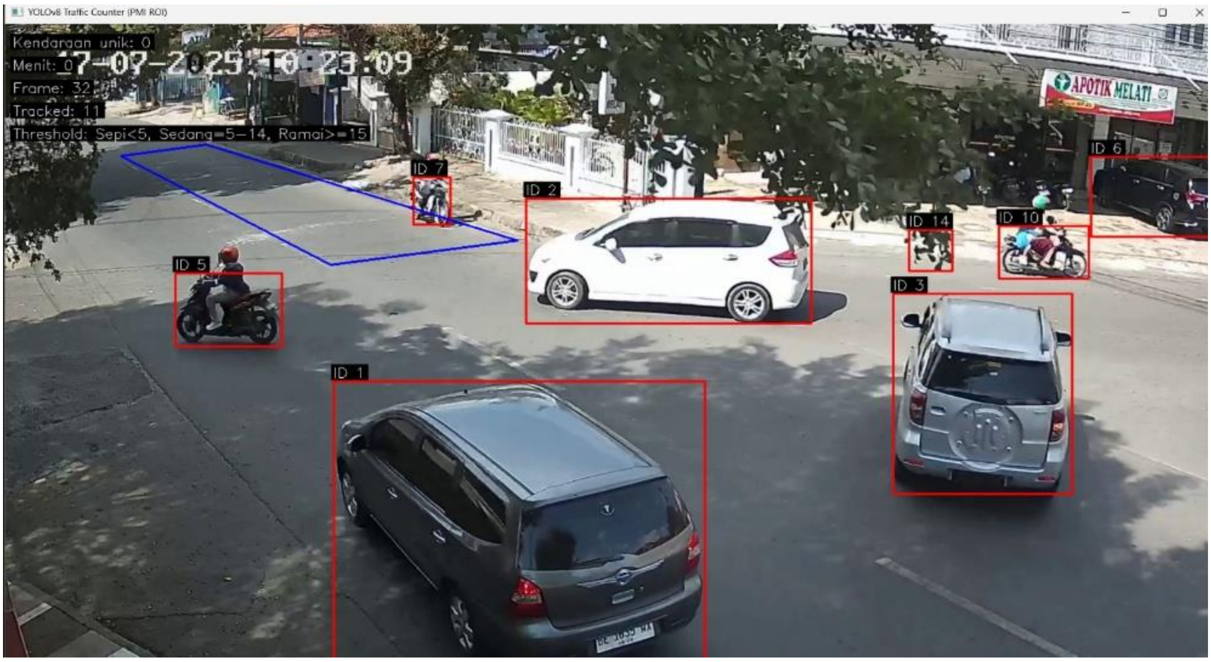

Figure 6. CCTV in front of PMI Metro City

D. Data Preprocessing

Data preprocessing ensures consistency and quality prior to model training. Frames are extracted at one frame per second using OpenCV and processed through YOLOv11 for vehicle detection with ROI filtering to reduce noise [5], [11]. Detection results are aggregated per minute and classified into three density categories: Low (< 5 vehicles/min), Medium (5–14 vehicles/min), and High (> 15 vehicles/min) [7], [19]. Missing values are handled via linear interpolation, duplicates removed, and temporal inconsistencies corrected before all location data are integrated into a unified CSV dataset [13], [18]. A sample of the final preprocessed output is presented in **Table 1** [3], [16].

Table 1. Traffic data per minute

Minute	Total	Category	Category_Num
0	19	Busy	2
1	8	Moderate	1
2	20	Busy	2
3	6	Moderate	1
4	3	Quiet	0
5	16	Busy	2
6	9	Moderate	1
7	0	Quiet	0
8	3	Quiet	0
9	16	Busy	2
10	9	Moderate	1
11	3	Quiet	0
12	0	Quiet	0
13	9	Moderate	1
14	0	Quiet	0

E. SUMO Configuration

Traffic simulation is conducted using SUMO version 1.18.0, with the road network configuration adapted to represent a real-world single intersection in Metro City [12], [15], [20]. The network is modeled in the *test_patched.net.xml* file and classified as an urban arterial road in accordance with SUMO standards [2], [16], [21], [22].

The traffic signal control is defined using four primary phases, consisting of green and yellow phases for the north–south and east–west directions, with base durations of 30 s for green, 3 s for yellow, and an inter-phase red time of 2 s [3], [7], [23]. The system operates under a fixed-phase rotation scheme, where the phase sequence remains unchanged [3], [18], [19]. The Reinforcement Learning agent is limited to adjusting the duration of the active green phase through the TraCI interface [9], [24], [25].

Preprocessed data stored in CSV format are utilized as input within the SUMO configuration to generate traffic demand in the form of *.rou.xml* files [26], [27], [28]. This approach ensures that vehicle movement patterns in the simulation are derived from real-world traffic observations [29].

Through TraCI, various real-time simulation parameters, including vehicle count, queue length, average speed, and waiting time, can be accessed [4], [9], [30]. This configuration enables a realistic and controlled simulation environment while supporting the integration of Reinforcement Learning for adaptive traffic signal optimization [17], [24].

F. Reinforcement Learning Environment

The Reinforcement Learning (RL) environment is designed as an interface between the SUMO simulation and the Deep Q-Network (DQN) algorithm by adopting the standard OpenAI Gym architecture [4], [7]. The environment manages the interaction process between the SUMO simulator and the TraCI interface during the learning process [12], [15], [30].

The state space is represented as a multidimensional vector describing current traffic conditions, including the average number of vehicles per lane, queue length, cumulative waiting time, average vehicle speed, traffic density category derived from IoT CCTV classification, and changes in queue length from the previous step (delta queue) [19], [23], [31]. All states are updated every 10 s of simulation time and normalized before being provided to the DQN agent [18], [22], [32].

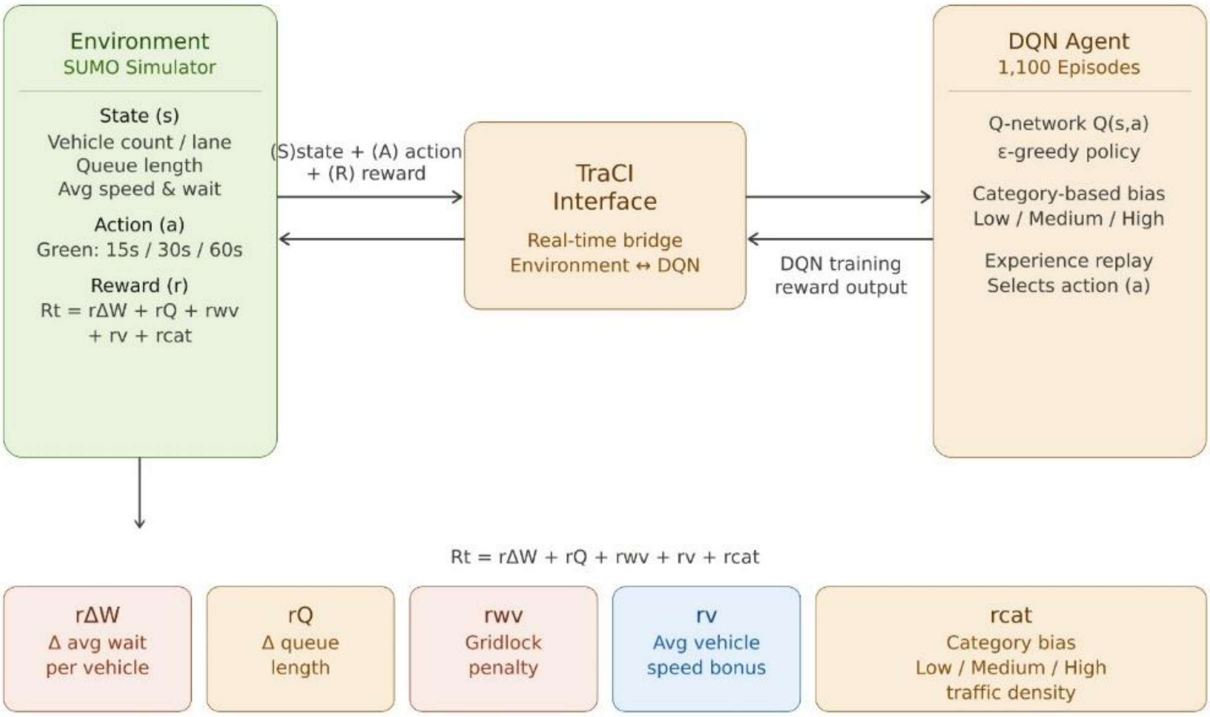

Figure 7. Environment and RL agent interaction diagram

Figure 7 illustrates the interaction mechanism between the SUMO environment and the DQN agent through the TraCI interface in real time [24], [30]. The environment generates traffic states that are transmitted to the DQN agent as learning inputs [9], [27]. The agent subsequently determines actions in the form of green phase durations of 15 s, 30 s, or 60 s using the ε-greedy policy and experience replay mechanism [3], [26], [29].

This study adopts a constrained action approach based on manual phase rotation, where the agent is only allowed to adjust the green signal duration without modifying the predefined traffic phase sequence. After the selected action is executed, the environment calculates the reward based on traffic condition changes and returns it to the agent as learning feedback [16], [20].

The reward function is designed to guide the agent in minimizing congestion and vehicle waiting time. The reward is calculated using the following formula:

$$Rt = r\Delta W + rQ + rwv + rv + rcat \quad (1)$$

where:

- R_t is the total reward value at time step t, clipped within the range $[-20, +20]$
- $r_{\Delta W}$ is the waiting time penalty with coefficient 0.005, calculated as $-0.005 \times \max(W_t - W_{t-1}, 0)$, where W_t is the total waiting time across all lanes at time step t
- r_Q is the normalized queue length penalty with coefficient 0.8, calculated as $-0.8 \times \min(\bar{Q}_t, 1.5)$, where $\bar{Q}_t$ is the average number of queued vehicles per lane
- r_{wv} is the waiting time threshold component with values ranging from -2.0 to $+1.5$, determined by the average waiting time per vehicle
- r_v is the speed bonus component with values ranging from -1.0 to $+1.5$, determined by the average vehicle speed $\bar{v}_t$ across all lanes
- r_{cat} is the category-based bias component with values ranging from -2.0 to $+2.0$, determined by the IoT CCTV traffic density classification (Low, Medium, or High) and the green signal duration d_t selected by the agent

The coefficients used in the reward function were determined through a series of preliminary experiments conducted prior to the main training phase. The coefficient for the waiting time penalty (rΔW) was set to 0.005 to scale down the raw waiting time difference, which can reach values in the hundreds of seconds, into a contribution range compatible with the other reward components [7], [19]. A larger coefficient was found to cause the agent to over-prioritize marginal waiting time reductions, leading to unstable Q-value estimates. The coefficient for the normalized queue penalty (rQ) was set to 0.8 after testing values in the range of 0.5–1.2; lower values underweighted queue buildup, while higher values caused the agent to aggressively clear queues at the expense of overall throughput [7], [19], [23]. The remaining components — rwv, rv, and rcat — are designed as threshold- and category-based terms whose magnitudes are bounded by fixed ranges (−2.0 to +1.5, −1.0 to +1.5, and −2.0 to +2.0, respectively), making explicit weighting coefficients unnecessary, as their contribution scales directly with observed traffic conditions [4], [10], [33].

Table 2. State, action, and reward components of the proposed DQN agent

Component	Description
State (s)	Vehicle count per lane, queue length, vehicle waiting time, average vehicle speed, traffic density category, and delta queue
Action (a)	Green signal duration adjustment of 15 s, 30 s, and 60 s
Reward (r)	Combination of five components: waiting time penalty (rΔW), queue penalty (rQ), waiting time threshold (rwv), speed bonus (rv), and category bias (rcat)

Table 2 summarizes the three main components of the reinforcement learning environment defined in this study [16], [20]. The state space captures real-time traffic conditions obtained from the SUMO simulation through the TraCI interface [3], [11], [12], [33], the action space is constrained to three discrete green signal duration options based on a manual phase rotation mechanism, and the reward function is designed to minimize congestion and vehicle waiting time.

The reward component described **in Table 2** is further elaborated through the reward function formulation presented in Equation (1), where each component is designed with empirically determined coefficients and mechanisms tailored to the characteristics of each traffic indicator.

G. Training of the Deep Q-Network (DQN) Model

Unlike supervised learning approaches, this study does not employ a conventional train/validation/test data split. The DQN agent learns the control policy directly through interaction with the SUMO simulation environment rather than from labelled datasets [9], [11]. The preprocessed IoT CCTV data is used as traffic demand input to generate vehicular routes within the SUMO simulation [11]. Model evaluation is performed based on cumulative rewards over 1,100 training episodes, while final testing is conducted using the best-performing model checkpoint in deterministic mode (ε=0) [18], [27].

The DQN model is implemented using a feedforward neural network with three hidden layers configured according to the dimensionality of the state space. The input layer receives the normalized traffic state vector, while each hidden layer employs the ReLU activation function. The output layer produces three Q-values corresponding to the green signal duration actions of 15 s, 30 s, and 60 s [21], [27].

To improve learning performance, this study incorporates experience replay and target network mechanisms. A replay buffer with a capacity of 300,000 transitions is utilized to reduce temporal correlations between training samples [24], [30]. A separate target network is periodically updated using a soft update mechanism to obtain more stable Q-value estimations [26], [29]. Network parameters are optimized using the Adam optimizer with a learning rate of 3×10^{-4} and a mean squared error (MSE) loss function. Exploration is handled using an ε-greedy strategy with a linearly decaying exploration rate during training. In addition, a category-based action bias mechanism is applied to adjust Q-values according to IoT-derived traffic density categories, encouraging the agent to select longer green durations under high-density traffic conditions [1], [12].

Training is conducted over 1,100 episodes, where each episode represents a complete traffic simulation period [16]. The training pipeline supports automatic checkpoint recovery, per-episode model saving, and best-model selection based on the average reward obtained over the previous ten episodes [17]. **Table 3** presents the hyperparameter configuration used in the DQN training process. All hyperparameter values were determined through preliminary experiments to balance convergence speed and learning stability [20].

Table 3. DQN training hyperparameters

Parameter	Value
Learning Rate	3×10^{-4}
Discount Factor (γ)	0.99
Replay Buffer Capacity	300,000
Batch Size	128
Training Episodes	1,100
Optimizer	Adam
Activation Function	ReLU
Loss Function	Mean Squared Error (MSE)
Exploration Strategy	ε-greedy
Action Space	15 s, 30 s, 60 s

Results and Discussion

A. Results of Data Collection and Processing

Data collection from four strategic locations in Metro City produced 32 to 36 CCTV videos with a total recording duration sufficient for analysis. Detection results using YOLOv11 demonstrated good capability in identifying vehicles with consistent bounding box accuracy. The generated data were successfully classified into three traffic categories using threshold values that align with the characteristics of urban traffic conditions in Indonesia. An integrated dataset was successfully created in CSV format compatible with SUMO simulation, with a data structure that enables comprehensive spatio-temporal pattern analysis [15], [27].

Table 4. Result of data intergration

Location	Average Vehicles/Minute	Dominant Category	Detection Accuracy
Dominos	28	High	85%
Pemda	22	Medium	87%
PMI	19	Medium	86%
Damkar	18	Medium	88%

Analysis of the integrated dataset shows that Dominos has the highest vehicle volume, with a total of 12,760 vehicles. The total dataset consists of 65,183 rows of frame-level detection data with 24 attribute columns. Traffic density classification was performed at the per-minute aggregation level, yielding a total of 2,090 classified minutes: 830 minutes (39.7%) categorized as Medium, 690 minutes (33.0%) as High, and 570 minutes (27.3%) as Low. The remaining frame-level rows represent raw detection results prior to minute-level aggregation and are therefore not subject to per-minute classification. These findings are consistent with studies integrating IoT data for urban traffic monitoring within the smart city context [1], [11].

B. Results of DQN Training Progress

Training over 1,100 episodes demonstrated consistent progress in improving RL agent performance. Monitoring through the Telegram Bot system provided real-time insight into learning progression. In the first episode, the total reward reached −5417.93 with an average waiting time of 1442.45 s, and 100% of actions were exploratory (RANDOM). As training progressed, significant improvement was observed, and by episode 1100, the total reward reached −3288.58 with an average waiting time of 1347.40 s, with 95.3% of actions being learned (LEARNED) [18], [28].

Table 5. Summary of DQN training progress

Episode	Total Reward	Average Waiting Time (s)	Learned Actions (%)
1	−5417.93	1442.45	0%
50	−5234.38	1483.95	13.8%
100	−5225.70	1525.04	25.4%
200	−4440.44	1439.76	44.4%
300	−4328.45	1478.64	58.9%
400	−3863.57	1399.47	70.5%
500	−3677.63	1413.93	77.7%
600	−3453.51	1335.27	83.9%
700	−3578.92	1382.04	87.9%
800	−3404.99	1375.56	90.3%
900	−3405.85	1372.77	92.8%
1000	−3258.88	1358.88	94.3%
1100	−3288.58	1347.40	95.3%

The total reward increased significantly by 39.2%, from −5417.93 in the first episode to −3288.58 in the final episode. The average waiting time decreased from 1442.45 s to 1347.40 s, representing an efficiency improvement of 6.6%. The transition from exploration to exploitation occurred gradually and in a controlled manner, with the proportion of learned actions increasing consistently from 0% to 95.3% by episode 1100.

C. Model Stability Analysis

To assess learning stability, the final training phase spanning episodes 900 to 1100 was examined. The total reward ranged from −3405.85 to −3258.88 (SD = 77.71 points), and the average waiting time stabilized within 1347.40–1372.77 s (SD = 12.70 s), confirming that the agent had reached a consistent control policy [7]. A temporary performance degradation was observed at episode 700 (reward: −3578.92), attributable to the high exploration rate of approximately 39.5% at that stage, which caused the agent to occasionally select suboptimal actions. Recovery at episode 800 (reward: −3404.99) confirmed that this degradation was transient and did not affect the overall convergence trajectory [18].

D. Model Evaluation Results

The best-performing model was obtained at episode 1050, where no further improvement in model performance was observed in subsequent episodes, indicating that the DQN agent had reached convergence. Unlike supervised learning, Reinforcement Learning operates as a single continuous training process in which the agent learns through trial-and-error interaction with the environment; therefore, the best model was selected based on the highest average reward over the preceding ten episodes rather than the final episode.

The best-performing model demonstrated significant performance with a total reward of −3045.01 and an average waiting time of 1318.91 s during the evaluation episode. These results indicate a 43.8% improvement in total reward compared to the initial training episode and an 8.6% reduction in average waiting time. The evaluation simulation ran for 1,970 steps with a throughput of 425 vehicles per hour, representing an 11.7% increase compared to the fixed-time baseline system [16]. Table 6 summarizes the performance metrics obtained from the evaluation of the best-performing model checkpoint in deterministic mode ($\varepsilon = 0$), obtained at episode 1050.

Table 6. Evaluation results

Metric	Value	Description
Total Reward	−3045.01	43.8% improvement from evaluation
Average Waiting Time	1318.91 s	8.6% reduction from evaluation
Number of Steps	1970	Optimal simulation duration
Throughput	425 vehicles/hour	11.7% increase from baseline

E. Performance Analysis and Comparison with Free-Phase RL

The evaluation results confirm that the proposed constrained-action DQN framework achieved measurable improvements over the fixed-time baseline. However, when compared to free-phase RL approaches reported in previous studies, which typically achieve waiting time reductions of 15–30% [7], the proposed method achieved an 8.6% reduction. This performance gap is primarily attributable to the fixed phase sequence constraint, which prevents the agent from dynamically prioritizing high-demand directions under asymmetric traffic conditions. Under such conditions, the system continues to follow the predefined phase rotation even when significant queue length imbalances exist across directions, limiting the agent's ability to allocate green time optimally.

F. Deployability vs. Optimality Trade-off

This performance trade-off is intentional and reflects a deliberate design choice. The constrained-action approach preserves full compatibility with existing fixed-phase traffic signal infrastructure, eliminating the need for hardware modifications or regulatory adjustments — a critical consideration for practical deployment in Indonesian urban environments [8], [28]. While free-phase RL methods offer higher theoretical optimality, their implementation requires significant changes to physical signal controllers and traffic authority protocols, making them less feasible for near-term deployment. The proposed method therefore prioritizes deployability over maximum optimality, representing a pragmatic middle ground between static fixed-time control and fully autonomous RL-based systems.

G. Scalability Implications

The current framework is designed for single-intersection control. Extending this approach to multi-intersection networks would require additional coordination mechanisms, such as multi-agent reinforcement learning (MARL) or centralized control architectures, to prevent locally optimal decisions from degrading network-wide traffic performance. Nevertheless, the modular design of the TraCI-based simulation environment and the category-based action bias mechanism provide a reusable foundation that can be adapted for larger-scale deployments. Future work incorporating vehicle-to-infrastructure (V2I) communication and shared state representations across intersections could further enhance the scalability of the proposed framework within the smart city context in Indonesia [20], [24].

Conclusion

This study successfully developed a constrained-action DQN-based adaptive traffic signal control system using real-world IoT CCTV data and SUMO simulation, demonstrating measurable improvements over the fixed-time baseline. The manual phase rotation mechanism proved compatible with existing Indonesian traffic signal infrastructure, offering a deployable alternative to fully autonomous RL-based systems without requiring hardware modifications

While the fixed phase sequence constraint resulted in more modest performance gains compared to free-phase RL approaches, this trade-off is intentional and reflects the practical realities of urban traffic infrastructure deployment in Indonesia. Future work should focus on multi-intersection extension and real-world deployment validation. The proposed framework provides a technically viable foundation for adoption within the smart city agenda and offers actionable guidance for traffic authorities and local governments seeking data-driven, infrastructure-compatible solutions for urban congestion management in Indonesian cities.

References

[1] S. Damadam, M. Zourbakhsh, R. Javidan, and A. Faroughi, “An Intelligent IoT Based Traffic Light Management System: Deep Reinforcement Learning,” *Smart Cities*, vol. 5, no. 4, pp. 1293–1311, Sep 2022, doi: 10.3390/smartcities5040066.

[2] J. Zang, P. Jiao, S. Liu, X. Zhang, G. Song, and L. Yu, “Identifying Traffic Congestion Patterns of Urban Road Network Based on Traffic Performance Index,” *Sustainability*, vol. 15, no. 2, pp. 948, Jan 2023, doi: 10.3390/su15020948.

[3] Y. Wu, “Enhancing Urban Traffic Flow Through Fuzzy Logic-Based Signal Light Control Optimization:,” *International Journal of e-Collaboration*, vol. 20, no. 1, pp. 1–13, Nov 2024, doi: 10.4018/IJeC.358746.

[4] J. Moreno-Malo, J.-L. Posadas-Yagüe, J. C. Cano, C. T. Calafate, J. A. Conejero, and J.-L. Poza-Lujan, "Improving traffic light systems using Deep Q-networks," *Expert Systems with Applications*, vol. 252, pp. 124178, Oct 2024, doi: 10.1016/j.eswa.2024.124178.

[5] R. Al-qudah, Y. Khamayseh, M. Aldwairi, and S. Khan, "The Smart in Smart Cities: A Framework for Image Classification Using Deep Learning," *Sensors*, vol. 22, no. 12, pp. 4390, Jun 2022, doi: 10.3390/s22124390.

[6] Y. Tian *et al.*, "Traffic signal active control method for short-distance intersections," *PLoS ONE*, vol. 20, no. 3, pp. e0319804, Mar 2025, doi: 10.1371/journal.pone.0319804.

[7] S. Bouktif, A. Cheniki, A. Ouni, and H. El-Sayed, "Deep reinforcement learning for traffic signal control with consistent state and reward design approach," *Knowledge-Based Systems*, vol. 267, pp. 110440, May 2023, doi: 10.1016/j.knosys.2023.110440.

[8] R. Dai and Y. Li, "Double deep network-based traffic signal optimization method for isolated intersections," *DTS*, vol. 5, no. 1, pp. 42–52, 2026, doi: 10.48130/dts-0026-0004.

[9] Y. Lin, A. Xie, and X. Liu, "Autonomous Vehicle Decision and Control through Reinforcement Learning with Traffic Flow Randomization," *Machines*, vol. 12, no. 4, pp. 264, Apr 2024, doi: 10.3390/machines12040264.

[10] C. Cai and M. Wei, "Adaptive urban traffic signal control based on enhanced deep reinforcement learning," *Sci Rep*, vol. 14, no. 1, pp. 14116, Jun 2024, doi: 10.1038/s41598-024-64885-w.

[11] R. Muwardi *et al.*, "YOLO-LIO: A Real-Time Enhanced Detection and Integrated Traffic Monitoring System for Road Vehicles," *Algorithms*, vol. 19, no. 1, pp. 42, Jan 2026, doi: 10.3390/a19010042.

[12] S. Mansouryar, C. Colombaroni, N. Isaenko, and G. Fusco, "Deep Reinforcement Learning Approach for Traffic Light Control and Transit Priority," *Future Transportation*, vol. 5, no. 4, pp. 137, Oct 2025, doi: 10.3390/futuretransp5040137.

[13] Y. Kong, Y. Li, and C.-H. Hsia, "MALight: A Deep Reinforcement Learning Traffic Light Control Algorithm with Pressure and Attentive Experience Replay," *JIT*, vol. 25, no. 7, pp. 955–962, Dec 2024, doi: 10.70003/160792642024122507001.

[14] S. M. M. R. Swapno *et al.*, "A reinforcement learning approach for reducing traffic congestion using deep Q learning," *Sci Rep*, vol. 14, no. 1, pp. 30452, Dec 2024, doi: 10.1038/s41598-024-75638-0.

[15] I. A. Salenek, Y. A. Seliverstov, S. A. Seliverstov, and E. A. Sofronova, "Improving the quality of route generation in SUMO based on data from detectors using reinforcement learning," *CRM*, vol. 16, no. 1, pp. 137–146, Feb 2024, doi: 10.20537/2076-7633-2024-16-1-137-146.

[16] B. Muyombo and H. Vadapalli, "Enhancing Urban Traffic Flow: An AI-Driven Traffic Light Management System," *Procedia Computer Science*, vol. 270, pp. 3848–3859, 2025, doi: 10.1016/j.procs.2025.09.510.

[17] S. Baeva, N. Hinov, and P. Nakov, "Comparative Analysis of Some Methods and Algorithms for Traffic Optimization in Urban Environments Based on Maximum Flow and Deep Reinforcement Learning," *Mathematics*, vol. 13, no. 14, pp. 2296, Jul 2025, doi: 10.3390/math13142296.

[18] K. Khalid, A. Bouayad, and K. Fardousse, "Adaptive Traffic Lights Control System Using Deep Q-Network and Salp Swarm Optimization," *IEEE Access*, vol. 13, pp. 114199–114207, 2025, doi: 10.1109/ACCESS.2025.3583656.

[19] W. Zai and D. Yang, "Improved Deep Reinforcement Learning for Intelligent Traffic Signal Control Using ECA_LSTM Network," *Sustainability*, vol. 15, no. 18, pp. 13668, Sep 2023, doi: 10.3390/su151813668.

[20] P. Michailidis, I. Michailidis, C. R. Lazaridis, and E. Kosmatopoulos, "Traffic Signal Control via Reinforcement Learning: A Review on Applications and Innovations," *Infrastructures*, vol. 10, no. 5, pp. 114, May 2025, doi: 10.3390/infrastructures10050114.

[21] H. Zhang, Z. Fang, Y. Chen, H. Dai, Q. Jiang, and X. Zeng, "Traffic signal optimization control method based on attention mechanism updated weights double deep Q network," *Complex Intell. Syst.*, vol. 11, no. 5, pp. 217, May 2025, doi: 10.1007/s40747-025-01841-9.

[22] H. Xue, P. Liu, and Z. Mu, "Research on Traffic Signal Control Algorithm Based on Deep Reinforcement Learning," *International Journal of Advanced Network, Monitoring and Controls*, vol. 10, no. 3, pp. 72–80, Sep 2025, doi: 10.2478/ijanmc-2025-0027.

[23] K. Cao *et al.*, "Optimization Control of Adaptive Traffic Signal with Deep Reinforcement Learning," *Electronics*, vol. 13, no. 1, pp. 198, Jan 2024, doi: 10.3390/electronics13010198.

[24] R. A. Velasquez Ortiz, M. E. Lárraga Ramírez, L. A. Alvarez-Icaza, and H. A. Guzmán Gutiérrez, "Adaptive Traffic Signal Control Using Deep Reinforcement Learning with Noise Injection," *Applied Sciences*, vol. 16, no. 8, pp. 3833, Apr 2026, doi: 10.3390/app16083833.

[25] H. Meess, J. Gerner, D. Hein, S. Schmidtner, G. Elger, and K. Bogenberger, "First steps towards real-world traffic signal control optimisation by reinforcement learning," *Journal of Simulation*, vol. 18, no. 6, pp. 957–972, Nov 2024, doi: 10.1080/17477778.2024.2364715.

[26] J. Lv, Z. Wang, and J. Ma, "Carbon Emission Reduction in Traffic Control: A Signal Timing Optimization Method Based on Rainbow DQN," *Applied Sciences*, vol. 15, no. 3, pp. 1101, Jan 2025, doi: 10.3390/app15031101.

[27] Z. Sun *et al.*, "Joint control of traffic signal phase sequence and timing: a deep reinforcement learning method," *Digital Transportation and Safety*, vol. 4, no. 2, pp. 118–126, 2025, doi: 10.48130/dts-0025-0008.

[28] S. Sun and M. Liu, "Optimisation of Lane-Level Dynamic Traffic Control Strategy Based on Bidirectional Adaptive Gated Graph Convolutional Network and Deep Reinforcement Learning," *IET Intelligent Trans Sys*, vol. 19, no. 1, pp. e70055, Jan 2025, doi: 10.1049/itr2.70055.

[29] S. R. M. Savithramma and R. Sumathi, "Intelligent traffic signal controller for heterogeneous traffic using reinforcement learning," *Green Energy and Intelligent Transportation*, vol. 2, no. 6, pp. 100124, Dec 2023, doi: 10.1016/j.geits.2023.100124.

[30] W. Bouzi, S. Bentaieb, and A. Ouamri, "Optimizing Traffic Light Control using Enhanced DQN: Minimizing Waiting Time for Regular and Emergency Vehicles," *Transport and Telecommunication Journal*, vol. 26, no. 3, pp. 266–275, Apr 2025, doi: 10.2478/ttj-2025-0020.

[31] C. Wu, I. Kim, and Z. Ma, "Deep Reinforcement Learning Based Traffic Signal Control: A Comparative Analysis," *Procedia Computer Science*, vol. 220, pp. 275–282, 2023, doi: 10.1016/j.procs.2023.03.036.

[32] D. C. Khrisne, "Optimizing traffic lights at unbalanced intersections using deep reinforcement learning," *IAES International Journal of Artificial Intelligence*, vol. 14, no. 4, pp. 2991–3002, Aug. 2025.

[33] H.-Y. Lin and M.-Y. Tsai, "Traffic light detection for nighttime driving with log-polar transform incorporated learning," *Results in Engineering*, vol. 25, pp. 103783, Mar 2025, doi: 10.1016/j.rineng.2024.103783.