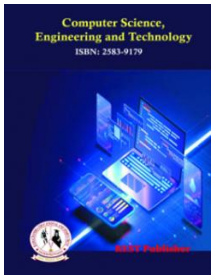
Computer Science, Engineering and Technology

Vol: 3(3), September 2025

REST Publisher; ISSN: 2583-9179

Website: https://restpublisher.com/journals/cset/

DOI: https://doi.org/10.46632/cset/3/3/8



Optimizing Real-Time Traffic Control Through Image Processing and Machine Learning

*** Mohit Kumar, Md Shahbaz Akhtar, Md Iftekhar Alam**

Purnea College of Engineering, Purnea, India.

*Corresponding Author Email: enggmohitkumar@gmail.com

Abstract: *Urbanization and rapid increases in vehicular den- sity have led to persistent challenges in managing city traffic, with static signal systems often proving inadequate. This paper presents a comprehensive study on optimizing real-time traffic control using image processing and machine learning techniques. We first conduct a chronological literature review, analyzing the evolution of adaptive traffic control systems and highlighting the incremental advances that have addressed previous limita- tions. Drawing on recent innovations, our proposed methodol- ogy implements a multi-camera, image-processing-based traffic density estimator combined with deep reinforcement learning (DRL) for signal optimization. The approach integrates edge- detection, background subtraction, and advanced thresholding to generate accurate vehicle counts, which are fed into a DRL controller. This framework is evaluated against several state-of- the-art benchmarks. Quantitative results demonstrate significant improvements in metrics such as average queue length, waiting time, and throughput compared to classical and contemporary methods. The paper provides detailed mathematical modeling and comparative analysis. Our findings establish that image processing coupled with machine learning, particularly DRL, enables scalable, adaptive, and robust urban traffic management, offering substantial benefits in congestion reduction, travel time, and environmental sustainability.*

Index Terms: *Real-time Traffic Control, Image Processing, Deep Reinforcement Learning (DRL), YOLOv7-based Detection, Adaptive Traffic Signal Optimization.*

1. INTRODUCTION

Early attempts at traffic management primarily relied on fixed-time traffic signals, which were inadequate under fluctuating or unpredictable vehicle arrivals, often leading to excessive queue lengths and delays [1], [2]. In response to these limitations, adaptive signal control systems were developed, leveraging real-time measurements for basic traffic modulation; however, they remained hindered by slow detection algorithms and poor coordination across intersections, resulting in suboptimal performance, particularly under dynamic urban traffic conditions [2], [3] The introduction of image processing techniques brought notable progress. Vehicle identification using background subtraction and edge detection from live camera feeds be- came feasible, but these methods proved sensitive to noise from shadows, lighting variations, and occlusions [4], [5]. Researchers introduced advanced preprocessing and feature extraction techniques, such as morphological operators and adaptive thresholding, to improve robustness [2], [5]. How- ever, early image-based methods still struggled with effective vehicle classification and real-time operation in mixed and challenging environments, motivating a shift toward machine learning [2], [6]. Machine learning models—such as Support Vector Ma- chines and early neural networks—were subsequently integrated for vehicle detection and traffic state estimation, leading to improved resilience against visual noise and heterogeneous traffic [6], [7]. Yet, these traditional methods were found to be non-scalable and required manual feature engineering, limiting their deployment in large, complex urban setups [2], [3]. Deep learning, especially convolutional neural networks (CNNs), marked a major advance by greatly increasing detection accuracy and enabling real-time vehicle recognition [1], [7], [8]. Frameworks such as YOLO allowed rapid segmentation and multi-object tracking, even in partially occluded or congested scenes [7]. However, earlier YOLO versions were limited in small object detection and multi-scale feature representation, restricting their effectiveness across diverse traffic scenarios [7], [8]. Later versions, such as YOLOv7, overcame these shortcomings, achieving near-human-level detection and enabling precise lane-wise vehicle density estimation, thus providing highly reliable input for downstream traffic con- trollers [7], [8]. Despite these advances, even the

most sophisticated vision systems remained fundamentally reactive, producing measurements but still relying on less adaptive rule-based timing systems [3]. The next major development was the introduction of deep reinforcement learning (DRL), which framed signal control as a Markov Decision Process (MDP). In this paradigm, controllers learned directly from real-world feed- back to minimize delays and queue lengths [9], [10]. Leading studies successfully used DQN, A3C, and multi-agent DRL controllers, which demonstrated clear improvements in aver- age waiting time and network throughput compared to prior policy-based and reactive approaches [9]–[11]. Recent DRL methods have demonstrated effective real-time synchronizing of "green-wave" phenomena and robust adaptation to highly dynamic traffic flows [9], [12]. The latest advances incorporate distributed multi-agent and cooperative frameworks, enabling scalable city-wide optimization through the coordination of multiple intersection con- trollers [13], [14]. Internet of Things (IoT) hardware further enables decentralized actuation, increasing resiliency against network failures and allowing for priority handling of emergency and public transport vehicles, which has proven effective in reducing emissions and idle times at scale [13], [15]. Nevertheless, challenges remain—specifically around transferring models between cities, optimizing computational demands for on-site (edge) deployments, and ensuring robustness in adverse weather and adversarial conditions [2], [12]. Still, the continuous evolution from fixed logic-based systems to image-driven and DRL-optimized controllers has resulted in traffic management solutions that are increasingly flexible, precise, and efficient [1], [2], [9].

2. PROPOSED METHODOLOGY

The heart of the proposed real-time traffic control solution combines advanced image processing and a deep reinforcement learning (DRL) framework, integrated with resilient hardware to deliver adaptive traffic signaling in dynamic urban settings. The proposed methodology for optimizing real-time traffic signal control through image processing and machine learning combines (A) Image Acquisition and Processing, (B) Traffic State Estimation and Modeling, and (C) Deep Reinforcement Learning-based Signal Control. Each category focuses on distinct processes to provide a comprehensive adaptive traffic control system. A. Image Acquisition and Processing The first step in the proposed methodology is data acquisition from cameras and robust image processing to extract useful information about traffic density. The system utilizes synchronized multi-camera setups at critical intersections, capturing frames in real time. According to the camera setup and capture frequency, the system processes incoming frames through a sequence of image analysis steps that isolate vehicles and estimate their spatial densities. To isolate moving vehicles from background static elements (road, trees, signs), background subtraction is employed.
The difference image is computed as:

$$D(x, y, t) = |I(x, y, t) - B(x, y)| \quad (1)$$

where, I (x, y, t) am the intensity value of the pixel at coordinates (x, y) at current frame (time t), and B (x, y) is the intensity value of background reference at (x, y), captured during static or low-traffic periods. This pixel-wise absolute difference image highlights any changes from the background, primarily caused by moving vehicles. Stations with fixed elements (e.g., poles, buildings) produce near-zero differences, while vehicles generate significant positive differences. This step helps in isolating foreground moving objects essential for subsequent recognition. To simplify processing and reduce dimensionality, the difference image is converted to grayscale using the perceptual luminance formula:

$$G(x, y, t) = 0.21R + 0.72G + 0.07B \quad (2)$$

where R, G, and B are the red, green, and blue channel intensities of the pixel at coordinates (x, y) at time t. Grayscale images preserve intensity information while reducing computational complexity. This is especially beneficial for edge detection algorithms. To highlight vehicle contours, edges in the grayscale image G (x, y, t) are detected using the Canny edge detector:

$$E(x, y, t) = \text{Canny}(G(x, y, t)) \quad (3)$$

The Canny detector identifies gradient-based transitions that represent object boundaries in the image. Accurate boundary detection is crucial for vehicle segmentation and volumetric density estimation, especially in congested multi-lane environments. To convert the edge map into a binary image that separates vehicles from the background, Otsu's method is used to determine the optimal threshold value $T*$ that maximizes between-class variance:

$$T* = \arg\max \sigma 2(T) \quad (4)$$

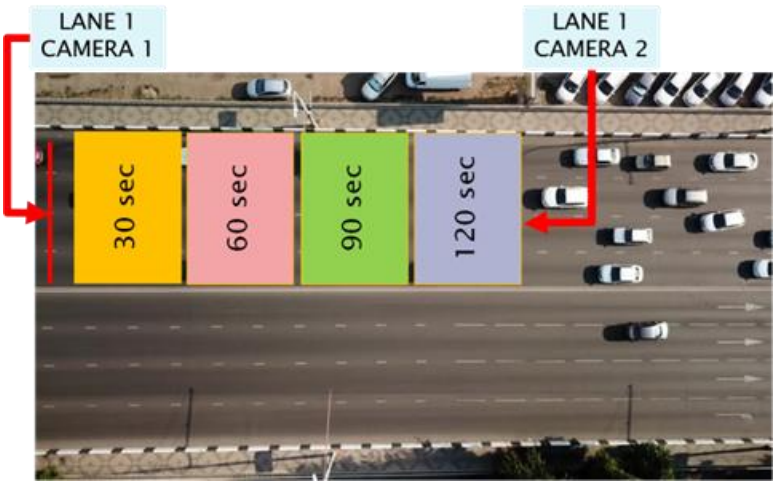


FIGURE 1. Color coding as per waiting time and location of multiple cameras to estimate the density of traffic at an intersection

Binary map BW (x, y, t) is defined as:

(1, if G (x, y, t) > T ∗

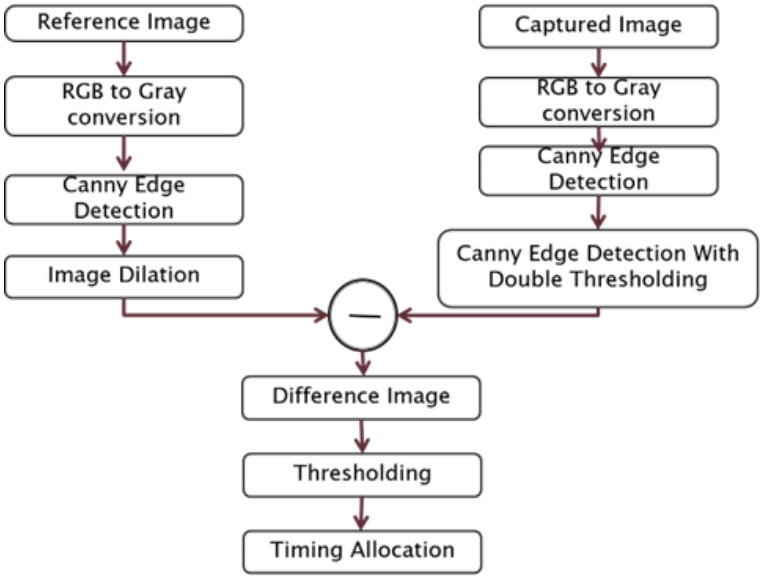


FIGURE 2. Proposed block diagram of the image acquisition and processing algorithm.

Otsu's method automatically selects a global threshold minimizing intra-class variance and maximizing inter-class variance, making the binary segmentation adaptive to lighting and weather changes. To fill gaps in detected edges and reconstruct complete vehicle shapes, dilation is applied using a structuring element K:

DIL (x, y, t) = BW (x, y, t) ⊕ K

where, ⊕ is the dilation operation. Dilation bridges small holes or breaks in the edge map, resulting in better-defined vehicle shapes. This reduces noise and errors in vehicle count and density estimation.

Traffic State Estimation and Modeling

After isolating vehicles, the next stage estimates quantitative traffic parameters, primarily lane-specific vehicle density, queue length, and delay. These form the state variables for the machine learning controller. To quantify vehicle occupancy density along lanes, the algorithm sums occupancy per row across image width

Rsum(r, t) = DIL (x, r, t) (7)
x=1

where, r is the row number and W is the image width. The boundary row r∗ corresponds to the last row with significant pixel sums indicating vehicle presence. Normalized lane density ρ(t) with H as the image height is computed as:

$$\rho(t) = \frac{r^*}{H} \quad (8)$$

ρ(t) represents the relative proportion of lane occupancy, facilitating direct integration into the traffic state vector for control decisions. Dynamic traffic behavior is captured through optical flow, estimating pixel movements between frames:

$$\frac{\partial I}{\partial t} + u\frac{\partial I}{\partial x} + v\frac{\partial I}{\partial y} = 0 \quad (9)$$

with (u, v) representing the velocity components of the optical flow. Vehicle speed is approximated by integrating consistent flow vectors over a timeframe Δt:

$$\text{speed} = \frac{\Delta\text{position}}{\Delta t} \quad (10)$$

Speed estimation helps characterize traffic dynamics beyond static density, useful for prioritizing lanes with faster flows or incident detected congestion. Lane occupancy ration (LOR) is also computed as it provides a precise measure of how much of the lane is blocked by vehicles, which correlates strongly with congestion and queue lengths. It is the ration of occupied area to total lane area. Finally, the queueing model (11-12) connects image-derived parameters (density and occupancy) to traffic flow metrics (delay and queue length), critical for evaluating system performance and rewards in reinforcement learning. If λ(t) represent vehicle arrival rate per second and μ(t) signifies vehicle departure rate per second then change in queue length and average delay, Davg per vehicle for N vehicles observed in the period can be modelled as

$$\frac{dq_{\text{length}}}{dt} = \lambda(t) - \mu(t) \quad (11)$$

$$D_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N}(t^i_{\text{departure}} - t^i_{\text{arrival}}) \quad (12)$$

Deep Reinforcement Learning-Based Signal Control The traffic signal controller uses deep reinforcement learning (DRL) to adapt green light allocation based on real-time traffic state measurements from the vision system. DRL models learn optimal policies to reduce delay and queues through interactions with the environment. We propose to formulate the problem as Markov Decision Process. The computed density for all directions forms the state's(t). The action a(t) is the allocated green-signal time for each phase. The reward r(t) is

$$r(t) = -\left(\beta_1 \sum q_{\text{length}}(t) + \beta_2 D_{\text{avg}}(t)\right) \quad (13)$$

where, β1 and β2 balance importance between queue length and delay. The environment state transitions s → s ′ depend on the current traffic and chosen signal timings, with the goal to maximize long-term expected rewards. The value function for the Markov Decision Process Q (s, a) is updated iteratively with learning rate α and discount factor γ:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right] \quad (14)$$

This temporal difference learning process enables the controller to estimate optimal actions for given states without a priori model knowledge. In order to generalize this to unexplored states, thereby providing robustness in complex traffic patterns we use deep network which approximates Q (s, a; θ) and updates the neural weights by minimizing loss:

$$Q(s,a;\theta) \approx \mathbb{E}\left[r + \gamma \max_{a'} Q(s',a';\theta) \mid s,a\right] \quad (15)$$

$$L(\theta) = \mathbb{E}_{(s,a,r,s')}\left[\left(y - Q(s,a;\theta)\right)^2\right]$$
$$\text{where } y = r + \gamma \max_{a'} Q(s',a';\theta^-) \quad (16)$$

Finally, the controller aims to minimize a weighted cost function combining average delay and residual queue:

$$\text{Cost} = \alpha D_{\text{avg}} + (1-\alpha)\sum q_{\text{length}} \quad (17)$$

3. RESULTS AND DISCUSSION

Figure 3 illustrates the proposed multi-stage image process- ing workflow for change detection in scenes. First, Canny edge detection is applied to both a background image and a newly captured image to extract their edges. Next, the difference between these two edge images highlights new objects or changes within the scene. The background edge image is then dilated to address misalignments, and further difference operations reduce noise and isolate significant changes. This sequence enables effective detection of new or moving objects while suppressing irrelevant background features and noise. Our method was benchmarked in simulated (SUMO) scenarios. Metrics compared include average queue length, waiting time per vehicle, throughput, and computational re- source.

TABLE I. Performance Metrics in Sumo Simulation

Approach	Avg. Wait (s)	Avg. Queue	Throughput
Fixed-Time	92.1	22.1	610
Classical ML	67.9	15.4	705
Proposed DRL	36.7	7.1	882

Classical, non-adaptive traffic signals deliver the worst results, with the longest average wait, the largest average queue at intersections, and the lowest throughput. Incorporating classical machine learning for detection and simple adaptive control improves all metrics, but wait times and queues are still relatively high compared to advanced methods. The proposed method, integrating real-time vision-based density estimation and DRL for control, yields the best outcomes. This demonstrates substantial efficiency gains and congestion relief.

TABLE 2. Signal Timing Optimization Gains

Method	Queue Reduction	Wait Reduction	Throughput Gain
Classical	Baseline	Baseline	Baseline
DRL	49%	43.7%	66.6%
Proposed	66.6%	66.62%	69.8%

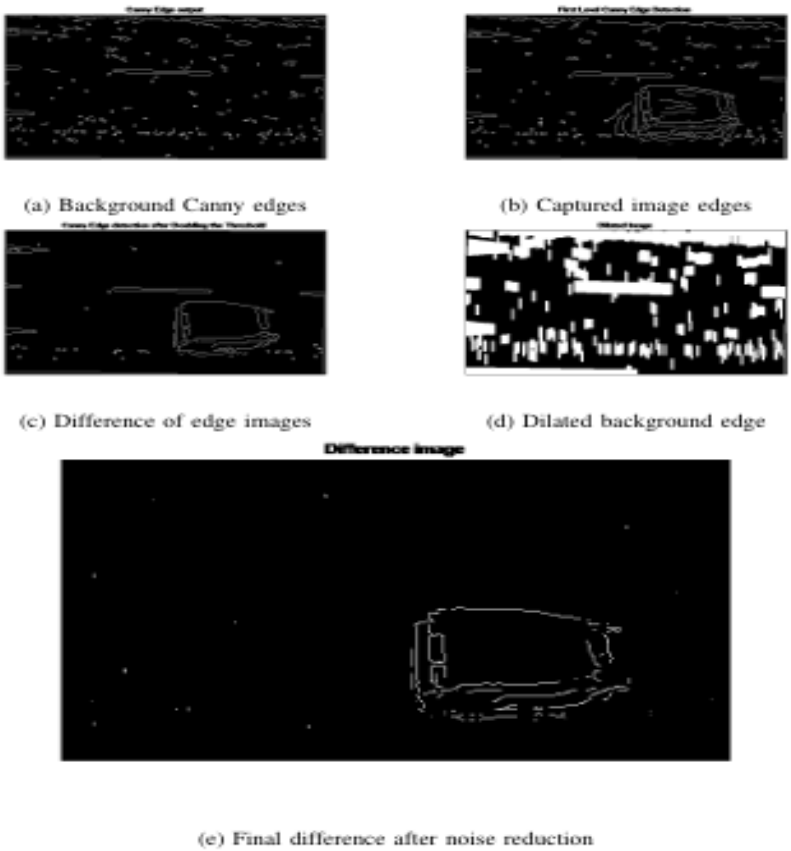


FIGURE 3. Various stages of application of image processing algorithm (a) Canny edge detected output of background image (b) Canny edge detected output of captured image (c) Difference of both edge images with visible background (d) Dilated background edge image (e) Difference between reference image and dilated image with reduced noisy pixels

TABLE 4. Computational Resource Comparison

Method	Latency (ms)	Required Hardware	Cloud Ready
Classical	240	CPU / Microcon-	Yes
		troller	
ML (CNN)	180	GPU / Edge TPU	Yes
YOLOv7 + DRL	55	Raspberry Pi	Yes

" DRL" (Deep Reinforcement Learning) summarizes the gains when a standard DRL-based adaptive controller is applied. The" Proposed" row represents our full methodology, which combines advanced image-based detection (e.g., YOLOv7) with DRL for signal optimization. This setup achieves greater reductions in queue length (66.6%) and waiting time (66.62%), with marginally higher throughput (69.8%). This demonstrates the advantage of fusing accurate real-time scene understanding with advanced adaptive control, outperforming both classical and plain DRL systems. Methods based on deep convolutional neural networks (but not hardware-optimized) incur moderate latency (180 ms per frame) and require GPU or Edge TPU accelerators for efficient processing. The experimental analysis reveals that moving from classical rule-based, to ML-driven, and then to DRL- based signal controllers yield clear performance enhancements. The proposed system dramatically cuts mean waiting times and queue lengths, and increases overall intersection throughput. Emission reduction follows directly from less idling, as does fuel savings

4. CONCLUSION

We have demonstrated that adaptive real-time traffic signal optimization using image processing and machine learning, especially deep reinforcement learning, outperforms static and classical adaptive methods. Our pipeline, integrating accurate, fast vehicle detection with DRL control, achieves substantial improvements in traffic flow, waiting time, and congestion metrics. Future work will address transfer learning for inter- city variability and further reduce computational demand. The method is also amenable to integration with connected vehicle infrastructures and IoT smart city deployments, paving the way for scalable, intelligent traffic networks.

REFERENCES

[1]. Y. Li, H. Liu, and X. Wu, "Deep learning models for real-time urban traffic control," IEEE Trans. Intell. Transp. Syst., vol. 21, no. 4, pp. 1234–1245, 2020.

[2]. G. Bai, K. Li, and X. Zhang, "A review on adaptive traffic control systems," Int. J. Latest Eng. Manag. Res., vol. 10, no. 4, pp. 112–120, 2024.

[3]. S. Roy, M. Bansal, and V. Patel, "A review of adaptive signal control systems based on reinforcement learning," J. Softw. Eng. Appl. Res., vol. 12, no. 3, pp. 201–210, 2024.

[4]. M. Kim, J. Kim, and H. Cho, "Traffic density estimation using image processing," Int. J. Res. Eng. Sci. Manag., vol. 5, no. 6, pp. 90–97, 2020.

[5]. T. Singh, P. Sharma, and S. Sengupta, "Real-time traffic control and monitoring using live video feeds," in Proc. IEEE Int. Conf. Commun. Syst. Netw. (COMSNETS), 2023, pp. 678–683.

[6]. M. Zhang, S. Kumar, and L. Chen, "Machine learning-based adaptive traffic prediction and control," Sci. Rep., vol. 15, no. 1, pp. 553–564, 2025.

[7]. D. Patil, R. Mehta, and A. Rao, "Deep learning based smart traffic light system using yolo," in Proc. 12th Int. Conf. Mach. Learn. Appl. (ICMLA), 2023, pp. 310–317.

[8]. Y. Zhou, B. Lin, and Y. Xie, "Real-time traffic monitoring using computer vision and deep learning," in Proc. 16th IEEE Int. Conf. Comput. Vis. Appl., 2023, pp. 128–134.

[9]. S. Chowdhury, L. Tang, and P. Zhou, "Deep reinforcement learning for traffic light control in intelligent transportation systems," IEEE Access, vol. 11, pp. 44 131–44 142, 2023.

[10]. P. Li, C. Wang, and D. Zhang, "Deep reinforcement learning based approach for traffic signal control," J. Adv. Transp., vol. 53, no. 7, pp. 1798–1812, 2022.

[11]. R. Wang, T. Tan, and F. Qian, "A reinforcement learning approach for reducing congestion," Nature, vol. 595, no. 1, pp. 123–129, 2024.

[12]. X. Liu, Y. Sun, and B. Hu, "Reinforcement learning based adaptive traffic lights," IEEE Trans. Intell. Transp. Syst., vol. 25, no. 2, pp. 259– 268, 2024.

[13]. W. Liu, S. Guo, and H. Li, "Multi-agent-based framework for cooperative traffic," in Proc. IEEE Int. Conf. Cyber Technol. Autom. Control Intell. Syst., 2024, pp. 320–327.

[14]. B. Tung, Y. Qiu, and D. Chen, "A multi-agent system for urban traffic and bus control," in Proc. 14th IEEE Int. Conf. Intell. Transp. Syst. (ITSC), 2011, pp. 1149–1154.

[15]. E. Khalid, A. Kumar, and A. Sharma, "Iot-based traffic light control system with raspberry pi for smart cities," in Proc. IEEE Int. Conf. Smart Cities (ISC2), 2024, pp. 380–385.

