

Article

Not peer-reviewed version

---

# Adaptive Traffic Signal Control Using Deep Reinforcement Learning with Noise Injection

---

[Raul Alejandro Velasquez Ortiz](#)<sup>†</sup>, [María Elena Lárraga Ramírez](#)<sup>\*†</sup>, [Luis Alvarez-Icaza](#)<sup>†</sup>,  
[Héctor Alonso Guzmán-Gutiérrez](#)<sup>†</sup>

Posted Date: 12 February 2026

doi: 10.20944/preprints202602.0953.v1

Keywords: traffic congestion; adaptive traffic signal control (ATSC); deep reinforcement learning (DRL); convolutional neural networks (CNNs); noise injection (NI); advantage actor-critic (A2C)





Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a [Creative Commons CC BY 4.0 license](#), which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

# Adaptive Traffic Signal Control Using Deep Reinforcement Learning with Noise Injection

Raul Alejandro Velasquez Ortiz <sup>1,†</sup> , María Elena Lárraga Ramírez <sup>1,†,\*</sup> , Luis Alvarez-Icaza <sup>1,†</sup>   
and Héctor Alonso Guzmán-Gutiérrez <sup>2,†</sup> 

<sup>1</sup> Instituto de Ingeniería, Universidad Nacional Autónoma de México, Coyoacán, Ciudad de México 04510, México

<sup>2</sup> Escuela Nacional de Estudios Superiores Unidad Morelia, Universidad Nacional Autónoma de México, Morelia, Michoacán 59190, México

\* Correspondence: mlarragar@iingen.unam.mx

† These authors contributed equally to this work.

## Abstract

Adaptive Traffic Signal Control (ATSC) remains a critical challenge for urban mobility. In this direction, Deep Reinforcement Learning (DRL) has been widely investigated for ATSC, showing promising improvements in simulated environments. However, a noticeable gap remains between simulation-based results and practical implementations, due to reward formulations that do not address phase instability. Stochastic variations may trigger premature phase changes (“flickers”), affecting signal behavior and potentially limiting deployment in real scenarios. Although several works have examined delay, queues, and decentralized coordination, stability-focused variables remain comparatively less explored, particularly in single yet complex intersections. This study proposes a decentralized DRL model for ATSC with Noise Injection (ATSC-DRLNI) applied to a single intersection, introducing a stability-oriented reward function that integrates flickers, queue length, and Advantage Actor-Critic (A2C) learning feedback. The model is evaluated in Simulation of Urban MObility (SUMO) platform and compared against seven baseline methods, using real traffic data from a Mexican city for calibration and validation. Results suggest that penalizing flickers may contribute to more stable phase transitions, while reductions of up to 40% in queue length were observed in heavy-traffic scenarios. These findings indicate that incorporating stability-related variables into reward functions may help bridge the gap between DRL-based ATSC studies.

**Keywords:** traffic congestion; adaptive traffic signal control (ATSC); deep reinforcement learning (DRL); convolutional neural networks (CNNs); noise injection (NI); advantage actor-critic (A2C)

## 1. Introduction

Traffic congestion is a phenomenon with a large impact around the world [1–3]. In the United States alone, cities such as Chicago, Boston, and New York have recorded the highest levels of congestion worldwide, with consequences for public health, induced by pollutant emissions, stress, and the time spent by drivers in congestion [4–7]. Mexico City ranks fourth in traffic congestion and also faces the variety of social problems derived from this phenomenon [7–9]. Traffic congestion also generates significant economic losses [10], for example, in the USA, the cost of traffic congestion resulted in a loss of \$74 billion nationwide [7,11]. By 2030, this phenomenon is expected to be present in many more cities due to the growth of the population and the use of vehicles [12,13].

To reduce vehicle congestion at intersections, control of traffic lights is necessary [14,15]. However, the normal use of this technology, with fixed-time (FT) phases, cannot be very efficient because this strategy cannot respond to changing traffic situations [16]. The addition of sensors [17], allowed the design of Adaptive Traffic Signal Control (ATSC) systems [18]. During the early years of ATSC implementations, several techniques were proposed to optimize vehicular traffic. Among the most important are: Split Cycle Offset Optimization Technique (SCOOT) [19], Self-Organizing Traffic Lights

(SOTL) [20–22], Real-Time Hierarchical Optimized Distributed and Effective System (RHODES) [23], and Sydney Coordinated Adaptive Traffic System (SCATS) [24], which were attempts to set more efficiently the timing of traffic lights. Although these techniques were functional during the 1980s and 1990s, they are not adequate to deal with the more complex scenarios that arise with increasing vehicular traffic.

Artificial Intelligence (AI) has significantly enhanced the performance of ATSC. Since 2015, there are more than 2500 publications related to ATCS, including mathematical modeling and AI applications [25]. Techniques such as Fuzzy Logic (FL) or Genetic Algorithms (GAs) have helped to reduce traffic congestion by designing more intelligent phase-timing patterns of traffic lights [26,27]. Other new AI techniques based on Machine Learning (ML) are also used. In particular, a widely used subset is Deep Learning (DL), which has achieved important advances in the area of vehicular traffic [28]. Deep Reinforcement Learning (DRL) arises from the combination of DL with Reinforcement Learning (RL) algorithms that are used in the learning stage [29,30]. RL incorporates neural networks (NN) that, considering vehicle demand, identify characteristics of traffic behavior that are very helpful in phase-timing design [31–34].

In general, DRL techniques can be classified as centralized, where a set of traffic lights communicate and provide feedback between them for better decision making [35], and decentralized, where traffic lights act independently of each other [36]. There is also research that proposes hybrid solutions using both paradigms [37]. Due to the expected increase in the number of connected and automated vehicles (CAVs), which do have a different interaction with infrastructure or vehicular traffic, analysis of technologies to control intersections also allows to propose behaviors that would not be possible in regular scenarios [38].

The complexity of DRL techniques in ATSC is related to the size of the traffic network to be analyzed and the intrinsic difficulties of the algorithmic implementation of the chosen DRL model. Complexity is greater in centralized techniques because they manage several intersections simultaneously. The results obtained to date indicate that there are still pending issues, such as long execution times, convergence problems, and the large number of variables to be considered, that limit their application to real situations [31,33,39–41]. There are DRL techniques that have been successfully applied to a set of single intersections. However, the reported results correspond to cases where intersections and traffic conditions are similar [42,43].

Single intersection studies use variables such as queue length, waiting time, vehicle spacing, specific vehicle type, or number of vehicles approaching the intersection as reference information, and work with the complexity of traffic and the given choice of variables [31]. They usually take traffic lights at the intersection as a single global agent [44]. The research in [27,45,46], shows a number of works where optimization is solved with metaheuristic-based algorithms.

Although most ATSC studies with DRL treat intersections as exclusive global agents or apply decentralized strategies to multiple intersection systems [47–50], few of them deal with the agent "anxiety" to prematurely change phase with small stochastic variations. There are some studies that use a decentralized approach at single intersections, modeling each traffic light as an independent agent and including the number of traffic light changes (flickers) as a specific variable [51]. This approach could allow better local coordination and traffic stability by solving the convergence problems that constrain traditional models, such as excessive changes in traffic lights that can affect traffic safety and stability [52].

It is still necessary to apply DRL models in single intersections with the intention of correcting erratic decision making and improving the stability of vehicular phases by combining flickers, waiting times, or vehicular queue sizes, as has been mentioned in [53–55]. Works such as [54,56,57], also confirm that these combinations have not yet been sufficiently addressed and suggest that, despite the need to deal with scenarios with multiple intersections, problems with single intersections need to be solved by expanding their complexity scale. Recently [56], mentioned that the use of stable phase mechanisms using variables such as flickers, has not been fully explored. In summary, the use of DRL models and algorithms in simple intersections has open problems that still need to be addressed.

In this paper, we present a novel optimization method in which we involve three elements: vehicle queue size, flickers, and Advantage Actor-Critic (A2C) RL with Noise Injection (NI), using a decentralized approach that takes traffic lights as individual agents, generating communication between the different traffic lights installed at a single intersection. This type of combination of variables, agents and communication at a single intersection has not been used before and justifies exploring its performance.

The aim of this work is to develop and evaluate a decentralized DRL framework for traffic signal control at a single intersection, incorporating a novel adaptive reward function. Although focused on a single intersection, this study is highly relevant, as it provides a foundational testbed for complex traffic control strategies, demonstrates the practical benefits of integrating multiple traffic factors, and establishes principles that can be extended to multi-intersection networks.

This work differs from existing approaches by focusing on traffic signal control at a single intersection using a decentralized DRL framework, demonstrating that even in a single-node scenario, decentralization enhances local adaptability. Unlike most studies, the proposed model integrates into the reward function traffic related variables, flickers and queue length, together with A2C learning results, providing a more comprehensive assessment of traffic dynamics. The use of A2C improves learning stability and convergence compared to traditional DRL methods. In this way, each intersection operates as an independent agent adapting to local traffic conditions. In addition, the adaptive reward function optimizes key metrics such as queue lengths and traffic balance. Overall, this methodology enables autonomous and adaptive phase optimization, reduces congestion and delays, and is scalable to multi-intersection networks while maintaining decentralized control. Finally, by grounding the model in realistic traffic data and simulations, the approach offers practical applicability and a closer approximation to real-world traffic management, setting it apart from many theoretical or simplified studies.

The main contributions of the proposed study are summarized as follows:

1. A decentralized DRL framework with NI (ATSC-DRLNI) for single-intersection ATSC in which each traffic light operates as an independent agent adapting to local traffic conditions.
2. A stability-oriented reward function that integrates queue length, flickers, and A2C learning feedback in an effort to mitigate premature phase changes and support smoother convergence.
3. A comparative evaluation against seven baseline methods in Simulation of Urban MObility (SUMO) platform using real traffic data from a Mexican city for calibration and validation, suggesting improvements in queue reduction and phase stability.
4. An effort toward narrowing the gap between simulation-based ATSC research and potential real-world applications that emphasizes the role of stability-related variables in decentralized DRL settings.

The article is organized as follows: In Section 2, a brief overview of the ML and the techniques to be implemented will be given. The general problem formulation, as well as the details of the proposed model and algorithm, are presented in Section 3. Simulations details and the results of the experiments performed to verify the effectiveness of the present model are presented in Section 4. Next, is Section 5, where the analysis and discussion of the highlights of this research is established. Finally, Section 6 presents the conclusions and future work on this line of research.

## 2. Preliminaries on Machine Learning

In this section, we will briefly explain some RL and DL preliminaries, to show how the DRL and its algorithms work, their functions and categories, and to describe the actor-critic problem [29].

### 2.1. Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) are statistical models belonging to ML that identify data characteristics and, from them, formulate predictions for decision making [58,59]. ANNs have the following form:

$$\hat{y} = \left( \sum_{i=1}^n x_i \cdot w_i + b \right) \quad (1)$$

where the input estimates  $x$  are multiplied by a weight value  $w$  given for each input with a bias  $b$  added. The process is repeated for the input and outputs of each layer of the ANN.

A type of ANN, named Convolutional Neural Network (CNN), is commonly used for image-oriented processing and training, where images can be transformed into one-dimensional and grayscale vectors, which are then processed pixel by pixel. Typically, CNN networks consist of several layers: fully connected layers, pooling layers, convolution layers, and nonlinear layers. The fully connected layers and the convolutional layers have parameters, while the nonlinear layers and the pooling layers do not have parameters [60,61].

## 2.2. Reinforcement Learning (RL)

RL is a discrete time Markov Decision Process (MDP) that allows an agent to make future decisions based on current observation [62,63]. A MDP is defined by 5-tuples  $\{S, A, T, R, \gamma\}$ , where  $S$  represents the set of states contained in the environment,  $A$  is the set of actions that an agent can take within the environment,  $T$  is a transition function, which is the probability of movement between the states of the environment,  $R$  is the reward/penalty function that guides the learning, and  $\gamma \in [0, 1]$  is a discount factor that impacts the weight of rewards or penalties in RL-oriented policies. This process is usually executed by trial and error and depends on the amount of information that is given to the model, its size, and the learning time.

The ultimate goal of using RL is to obtain the highest cumulative sum of agent's learning, after an extended period of training [64]. Formally defined, the overall reward  $G_t$  is the maximization at time  $t$  of the sum of rewards  $R_t$  evaluated at different future times, according to

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} \dots + R_T, \quad (2)$$

where  $T$  is a previously established final time step. This type of maximization is known as reward maximization.

A variation in the RL approach consists in the use of a discount factor, in which agents weigh the rewards at future times using a discount factor.

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}. \quad (3)$$

There are other forms oriented to maximize the cumulative reward by maximizing the actual value of the reward  $R(t)$  for any given time  $t$ .

## 2.3. Q-Learning (DQN)

To obtain the optimal reward, Q-Learning is used that is based on evaluating the different agent's actions using  $\epsilon$ -greedy, a learning policy based on the exploration of the agent possible actions. At initial times, random actions are explored as a trial-and-error learning activity [65,66]; as time progresses, the best-performing rewards are exploited and the best actions are selected. A  $\epsilon$ -greedy algorithm is easy to understand and provides greater accuracy in learning, given a training time. The updating function  $QL_t$  is of form

$$QL_t(s, a) = QL_{t-1}(s, a) + \alpha \left[ R_t + \gamma \max_{a'} \{QL_t(s, a') - QL_{t-1}(s, a)\} \right], \quad (4)$$

where  $a$  are the actions,  $\alpha \in [0, 1]$  defines the learning rate or step size; if  $\alpha = 0$ , the agent will never learn from the new actions, conversely if  $\alpha \rightarrow 1$  the agent will favor recent information, as the value of  $QL_t(s, a)$  is modified more quickly. The innovation term depends on the reward  $R_t$  and the choice of

the action  $a$  that maximizes the growth of  $QL_t(s, a)$ , weighted by  $\gamma \in [0, 1]$ , a discount factor, which in Q-Learning determines the significance of the upcoming actions. If  $\gamma$  is set to 0, the agent discards future rewards entirely and only considers current rewards, on the other hand, if  $\gamma$  is set to 1, the agent will focus on getting rewards in the long run. A third parameter  $\epsilon$  can be used to introduce randomness in the search space of the  $\epsilon$ -greedy action selection procedure of the Q-learning algorithm. In the action selection phase, a given action is selected based on the Q-values available [67,68].

#### 2.4. Advantage Actor-Critic (A2C)

RL contains algorithms that are regularly divided into three groups: actor-only, critic-only and actor-critic, where the words actor and critic are synonyms for policy and function values [69–71]. Actor-only methods work with a set of parameterized policies that can be used directly in the optimization processes, while critic-only methods use time-difference (TD) learning, which has a smaller variance in expected estimates. This method is based on the choice of greedy actions, which suggests the choice of functions with the highest value in the expected performance. However, this process is computationally expensive when the action space is continuous. Advantage Actor-Critic (A2C) methods combine the advantages of actor-only and critic-only methods. This choice is capable of producing continuous actions and preserves the convergence properties of the policy gradient [70].

#### 2.5. Neural Networks in Reinforcement Learning

RL algorithms can be implemented without NN, using tabular and iterative methods, which make them more intuitive but generate scale and precision problems when large amounts of data and many variables are considered. In the case in our ATSC application, the use of ANNs, using standard weight-adjustment algorithms such as back-propagation, is appropriate.

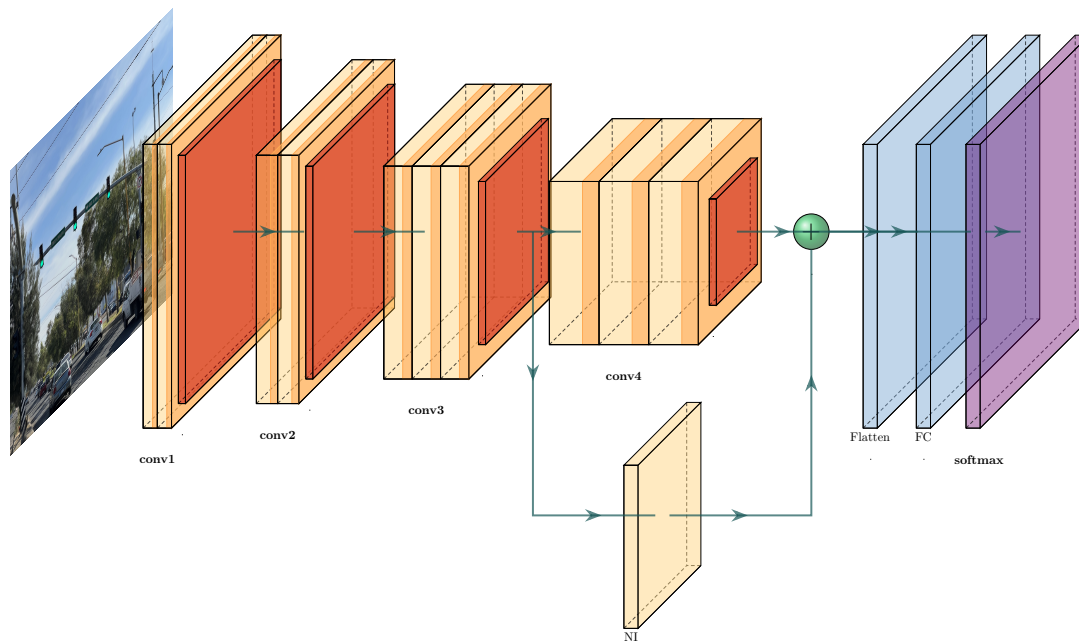
To obtain robustness to variations during training, Noise Injection (NI) is added, which consists of the addition of noise in the input data of a ANN during the training phase to improve performance [72,73]. This noise is represented by a random disturbance  $\epsilon$ , with a Gaussian distribution. The shape of the input with noise is represented as:

$$\tilde{x} = x + \epsilon, \quad \text{where } \epsilon \sim \mathcal{N}(0, \sigma^2) \quad (5)$$

where  $\tilde{x}$  consists of the sum of the original data input  $x$  plus the random noise obtained from a normal distribution with mean 0 and variance  $\sigma^2$ . The input to the ANN is then  $\tilde{x}$  instead of  $x$ .

### 3. Materials and Methods

In this section, we formulate the ATSC problem as a DRL task, making a clear definition of all elements, such as scenario behavior, reward function, actions, and states. We present the functions used, such as Experience Replay (ER) and DQN, and explain the training with NI and DRL. The general architecture is illustrated in Figure 1, where the model is named ATSC-DRLNI, while Table 1 contains the notation used in this proposal.



**Figure 1.** Proposed decentralized DRL framework (ATSC-DRLNI) architecture with of 4 Convolutional Neural Networks (CNNs) and a Noise Injection (NI) layer.

**Table 1.** Symbols used in the implementation of the decentralized DRL framework (ATSC-DRLNI) architecture.

Symbol	Meaning
$s$	State
$a$	Action
$R_t$	Reward
$E$	Environment
$A$	Agent
$N$	North
$S$	South
$E$	East
$W$	West
$QL$	Deep Q-Network (DQN)
$T$	Time frame
$G_l$	Green Light
$Y_l$	Yellow Light
$R_l$	Red Light
conv	Convolutional neural network

### 3.1. System Description

The ATSC-DRLNI model is based on the joint implementation of techniques belonging to DRL, and includes the CNN design, the data set (synthetic or real) processing, and the use of RL algorithms, in this case, ER and  $\epsilon$ -greedy policy. The general architecture is shown in Figure 1, and will be explained below and in the following sections.

The DL scheme is divided into two streams, one where the inputs are processed by four CNN layers connected in series, and another where the output of the third CNN is passed through a NI-layer, where gaussian noise is incorporated, and whose output is added to the output of the fourth CNN. The NI-layer will improve the behavior of the network, specifically with regularization, robustness, and performance. This NI-layer implements a rectified linear unit function (ReLU), that generates a flow with a discrete probability distribution. The final output vector is used in: i) the reward function, ii) to update the CNN

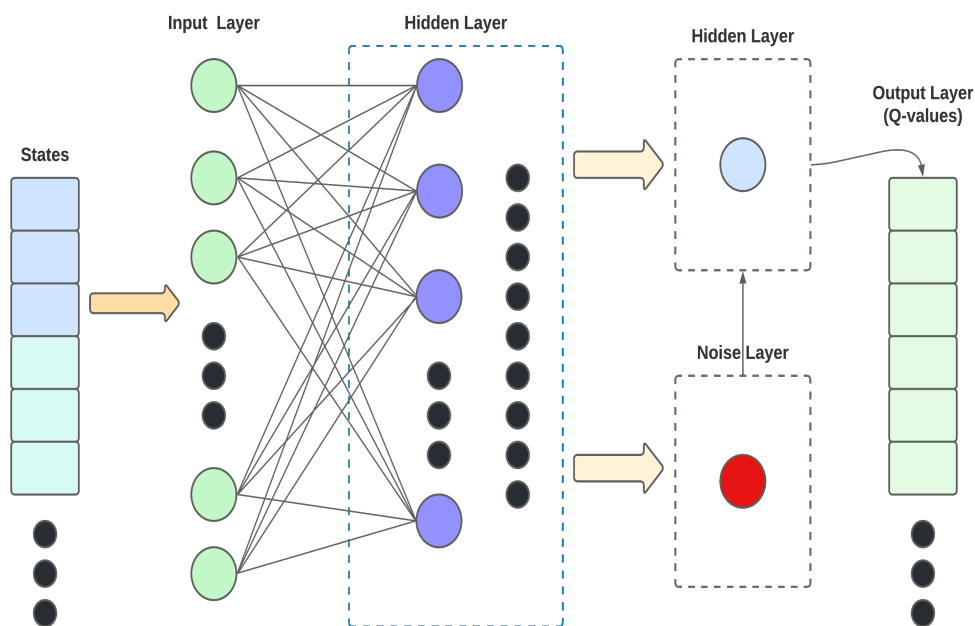
weights with back-propagation at each step using the DQN algorithm, and iii) to create the batches and buffers to be used for ER learning. Finally, the  $\epsilon$ -greedy algorithm selects the actions.

### 3.2. Actions

The set of feasible actions include all possible directions and paths that the traffic light can adopt and, therefore, that an agent is free to choose from. For example, suppose that a traffic light change may occur within a given time frame  $T$ , and the agent starts a cycle with the green phase of the traffic light ( $G_I$ ), then, depending on the state of traffic, the agent will move to the yellow phase ( $Y_I$ ), and finally to the red phase of the traffic light ( $R_I$ ). To maintain a stable vehicular flow, after a time for  $R_I$  the agent should switch back to the  $G_I$  repeat the cycle, noting that only one phase can be active at any time. Action ( $G_I, Y_I, R_I$ ) of agent  $A$  at time  $t$  is denoted by  $A_t$ .

### 3.3. States

The state at time  $t$ , where an action is taken, is formed as a vector whose elements contain information from the environment, which is given by the specific features obtained from the traffic light sensors in use (length of vehicles, vehicle density, the distance between stopped and moving vehicles, etc). In our case, we use as information the total number of vehicles at a distance  $D$  from the intersection under analysis and associate this number also with the actions of the traffic light signals, e.g.,  $G_I, Y_I$  and  $R_I$ , and their respective time frames. Figure 2 illustrates the data flow, from states to Q-values.



**Figure 2.** Data flow in the CNN structure in the ATSC-DRLNI model.

### 3.4. Reward

The reward function is an essential component of the RL algorithm design, since it is the feedback index that evaluates the quality of the agent's performance.

We implement a reward function  $R_t$  at time  $t$  based on the combination of the vehicle queue, flickers, and the agent action  $A_t$  as follows:

$$R_t = -\frac{1}{p(A_t)} \cdot \frac{Q_t}{Q_{norm}} - \frac{F_t}{F_{norm}}, \quad (6)$$

where  $p(A_t)$  is the probability of action  $A_t$ , decided in the A2C policy described below,  $Q_t$  the size of the vehicle's queue in the scenario considered,  $F_t$  the number of flickers at time  $t$ .  $Q_{norm}$  and  $F_{norm}$  are appropriate factors to scale the involved variables.

This reward function has an additional penalty included when there are no vehicles and a flicker is produced, in that case

$$R_t \leftarrow R_t - F_p \quad (7)$$

with  $F_p$  the chosen penalty. The proposed reward then weighs the length of the queue and the number of flickers. A positive change in the reward indicates an improvement in the intersection condition, and conversely, a negative change in the reward shows deterioration. The reward function is used by the agents, in this case the traffic lights involved in the scenario that will be explained later on, and the interaction with the environment determines the policy of each agent. This reward function uses the DQN function presented before. To detect changes in the reward, the following time difference is introduced

$$S_t = R_t - R_{t-1} \quad (8)$$

### 3.5. The Neural Network Structure

The structure of the proposed NN model is illustrated in Figure 2. This NN is designed in such a way that it can process any state and produce a proper action. The input received comes from video cameras installed at several points before the intersection that, once processed, provide information about the number of vehicles at a certain distance  $D$  of the intersection, for the considered scenario, as will be explained later on. There are four CNN hidden layers aiming to maintain an adequate performance both in the NN and in the pre-processing. These layers include a NI-layer with a ReLU activation function. The number of output nodes for obtaining the Q-values functions is equivalent to the number of actions, which, in turn, are the triggers for obtaining the best reward. The NI-layer provides a statistical probability distribution by means of GN, which is added to the result of the last CNN. The output layer vector is the set of Q-values used to minimize the updating function  $Q_t$  in Equation (4). Algorithm 1 describes the A2C policy optimization in the ATSC-DRLNI model.

---

#### Algorithm 1: ATSC-DRLNI policy optimization

---

**Input:** Images from intersection cameras  
**Output:** Set of Q-values  
**Data:** Testing set

- 1 Initialize policy (stochastic) // Estimated value
- 2  $S_t, A_t \leftarrow \Pi_t(s,a)$  // Elaborate policy
- 3  $S_t \leftarrow S_{t-1}, R_t$  // Time difference equation
- 4  $\delta_t \leftarrow R_t \gamma S_{t-1} - S_t$  // Adjust actor
- 5 Update actor such that **if**  $\delta_t > 0$  **then**
- 6  $\lfloor p(A_t) \uparrow$  // Positive time difference
- 7 **else**
- 8  $\lfloor p(A_t) \downarrow$  // Negative time difference
- 9 Update critic such that,  $S_t \leftarrow S_{t-1} + \alpha \delta_t$  // Adjust estimated value  $S_t$
- 10  $S_t = S_{t-1}$  // Repeat algorithm from step 2

---

The policy  $\Pi_0$  is initialized with a random stochastic value to calculate the function  $S_0$ . Then, given a state  $S_t$  and an action  $A_t$  carried out, the policy for time  $t$ ,  $\Pi_t$  is determined. The error term from one iteration to the next is computed in step 3 of the algorithm, where the goal is to adjust  $S_t$  using  $R_t$ , the current value of the reward, and the previous value  $S_{t-1}$ .

We update the actor by adjusting the action probabilities for state  $S_t$  based on the error term. If  $\delta_t$  is positive, we must raise the probability of taking action  $A_t$ , because it increases the reward, conversely, if

it is negative, we must lower the probability of taking action  $A_t$ , because choosing it results in a poorer performance in the objective function. This is done by the actor who knows the actions.

Updating the critic will adjust the estimated value of  $S_t$  using  $S_{t-1}$  and  $\alpha$ , the learning ratio of the critic, not the actor. Finally, the new state must be set and the algorithm is repeated from step 2.

## 4. Results

In this section, the scenarios and dataset for the simulation are established. To verify the performance of our proposal, simulation results comparing them with other well established methods available in the open literature are presented. A summary of the statistics of the general behavior obtained is also included.

### 4.1. Setup

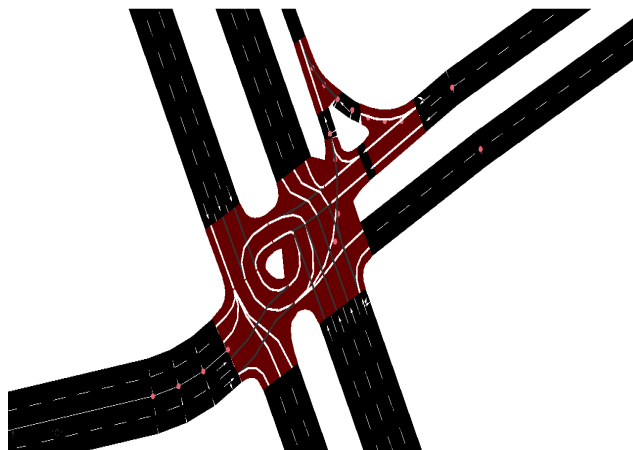
To validate the functionality of the methodology, several simulations were carried out for different behaviors of vehicular traffic. The simulations were coded with the Simulation of Urban MObility (SUMO) sumo-gui version 1.10.0, Python 3.8.8, and OpenIA Gym version 0.7.4 and executed in a cluster with the following technical data: MSI Intel(R) Core(TM) i7- 10870H CPU @ 2.20GHz, 16 GB DDRA RAM, VRAM 8GB DDR6 NVIDIA RTX3600, and 3060 NVIDIA CUDA Core. Relevant intersection data used for the simulations are shown in Table 2.

**Table 2.** Intersection parameters.

Parameter	Value
Number of lanes	10
Traffic Flow	Random trips
Simulation platform	SUMO
Yellow phase duration	5 s
Initial Green phase duration	30 s
Distance $D$ to intersection to detect vehicles	800 m
Value of $Q_{norm}$	5,000 vehicles
Value of $F_{norm}$	100 flicker
Value of $F_p$	0.2 flicker

### 4.2. Simulation Environment and Dataset

The traffic intersection depicted in Figure 3 is the environment used to train the RL algorithm. It represents an intersection in Aguascalientes, a city in Mexico, whose data are provided by the local *Dirección de Tránsito y Vialidad* and were obtained by an intelligent traffic system provided with cameras and sensors. Vehicle's directions in the intersection are  $N, S, E, W$ , indicated as  $N \leftrightarrow S$  and  $E \leftrightarrow W$ , implying that both directions are valid at the same time. Vehicle directions also includes turns, as vehicles can go from  $S$  to  $W$  ( $S \rightarrow W$ ) and from  $N$  to  $E$  ( $N \rightarrow E$ ).



**Figure 3.** Intersection used for training of ATSC-DRLNI .

Training simulations were executed in SUMO platform. The intersection has four phases with the following directions: Phase 1 (north-south) and (south-north); Phase 2 (east-west) and (west-east); Phase 3 (east-south) and (north-east); Phase 4 (east-north) and (west-north). As mentioned, all the input data information are from an intersection in Aguascalientes City, Mexico. Sensor information was available every 30 s, and data for a complete week were used, and training was performed using 4 hour subsets. For a complete 4-phase cycle (green-yellow-red), the average vehicle flow is 400. The vehicles generated by the simulator are probabilistically defined and matched to the data-set provided by the *Dirección de Tránsito y Vialidad*.

There are two vehicle traffic conditions, one corresponding to normal situations and another, for specific times of the day, to the case where demand is higher and congestion is to be expected. A combination of data including both conditions was used for system training in all simulations. Two scenarios were simulated: in the first one, phases 1 and 2 are optimized to give preference to straight traffic, while the second optimization is based on phases 3 and 4, which favor more complex maneuvers (turns to right and left) that are, in general, more prone to cause congestion. Special configurations for the right and left lanes, where vehicles execute turns, were also defined with the intention of representing real situations at urban intersections, where turns are a critical issue in traffic handling. The distance  $D = 800\text{ m}$  selected to detect incoming vehicles at the intersection, represents a typical urban environment with relatively short distances between intersections and poses an additional challenge for the synchronization of traffic lights.

**Table 3.** Intersection scenario 1 parameters

Parameter	Value
Number of lanes	10
Traffic Flow	Random trips
Simulation platform	SUMO
Yellow phase duration	5 s
Initial Green phase duration	30 s
Distance $D$ to intersection to detect vehicles	800 m
Value of $Q_{norm}$	5,000 vehicles
Value of $F_{norm}$	100 flicker
Value of $F_p$	0.2 flicker

**Table 4.** Intersection scenario 2 parameters.

Parameter	Value
Number of lanes	14
Traffic Flow	Random trips
Simulation platform	SUMO
Yellow phase duration	5 s
Initial Green phase duration	30 s
Distance $D$ to intersection to detect vehicles	800 m
Value of $Q_{norm}$	6,500 vehicles
Value of $F_{norm}$	100 flicker
Value of $F_p$	0.2 flicker

Once trained, the ATSC-DRLNI model was applied to the two environments shown in Figure 4, where the same 4 phases, traffic conditions and scenarios used for the training still apply. All data were obtained from the same source. The results shown below are obtained from simulations of these two environments and correspond to scenario 1, where phases 1 and 2 are optimized to give preference to straight traffic, as better results were obtained with this scenario.

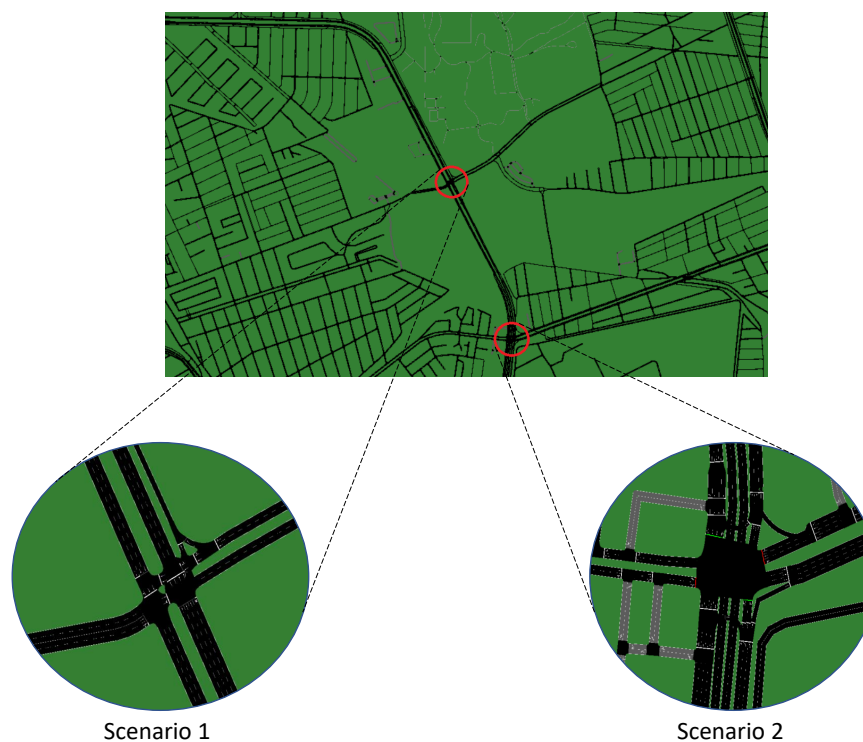


Figure 4. Intersections used to test ATSC-DRLNI model.

#### 4.3. Selected Methods for Comparison

Seven other traffic control methods were selected from the open literature to use for comparison purposes with the proposal of this work, which are listed below. In all of them, the same scenarios and reward function, where applicable, were used.

- Fixed-Time (FT): This is a classical non-adaptive method used for traffic light control that serves as a baseline for comparisons.
- Multi-Agent Deep Reinforcement Learning (MADRL): Multi-agent algorithm based on DRL. It uses gradient policies and trains agents in a centralized manner, where collective decision-making is based on real-time information feedback to agents. There are also decentralized versions, where agents are autonomous with decision-making not depending on other agents.
- Multi-Agent Recurrent Deep Deterministic Policy Gradient (MARDDPG): decentralized algorithm for agent training where each agent shares policies and information with other agents. Individual agents are capable to execute decision independently.
- Single-Agent Based Algorithm (SABA): DQN based decentralized algorithm where agents do not relate between them, although they try to predict what other agents may decide.
- Longest Queue First (LQF): This method favors lane selection where the green light is already active and gives priority to vehicles close to the intersection, a situation that may produce inconsistent phase cycles.
- Q-Learning (DQN): RL-based method for action learning that uses a predefined number of Q-values that are related to the number of actions. Actions are stored in a table that contains elements, such as vehicles queue, waiting time of vehicles in lanes close to the intersection and the current phase.
- Self-Organizing Traffic Lights (SOTL): This method uses a self organizing algorithm to determine the phases of the traffic lights.

#### 4.4. Simulation Results

Simulations use the parameters shown in Table 5, that are the values that yielded the best results in terms of the final value of the metrics, convergence, and stability are included in Table 6.

**Table 5.** Description of model parameter used in simulations.

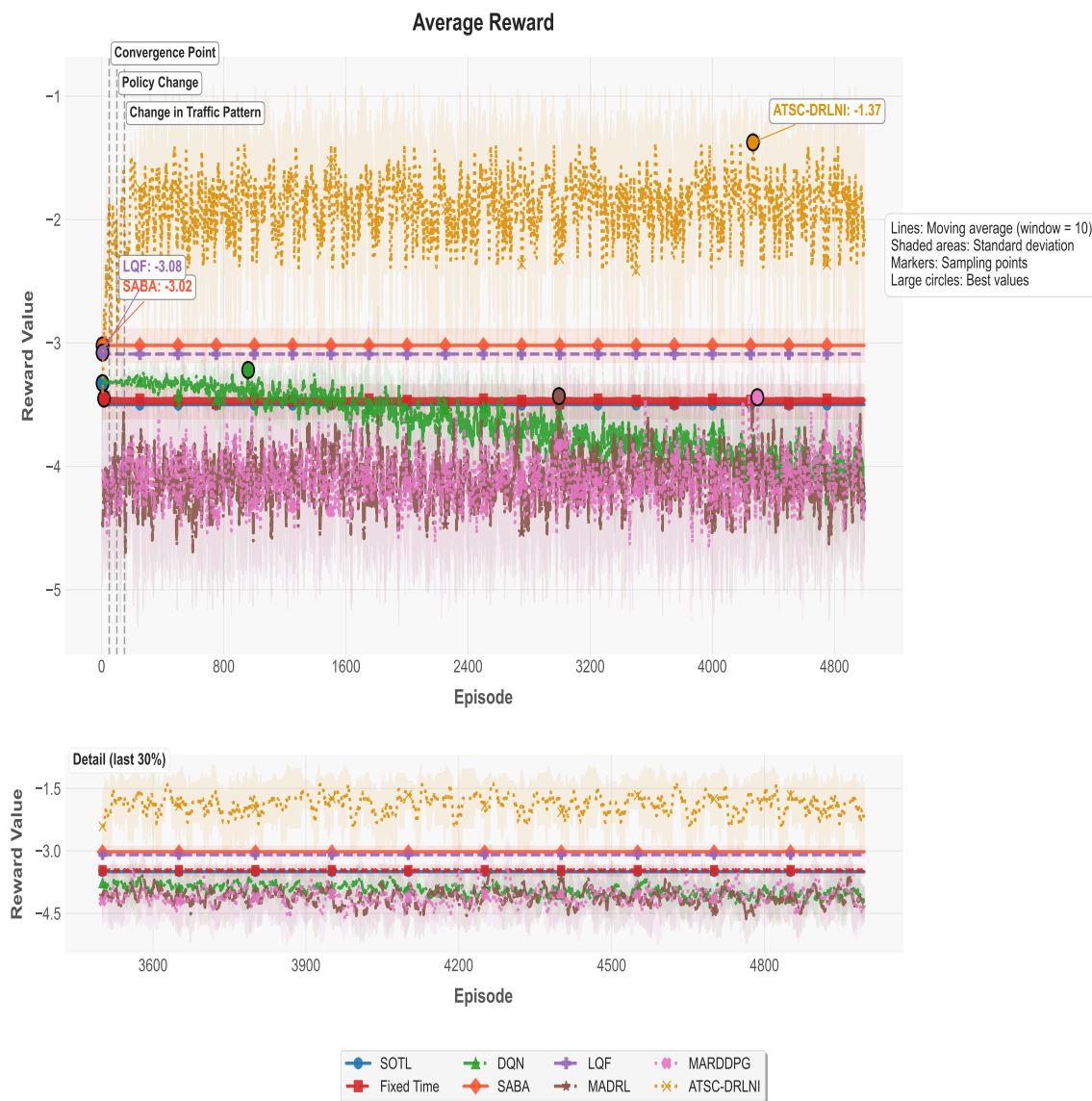
Parameter	Description
Batch size	Number of samples used at each iteration to determine the NN error.
Neural network layers	Number of ANN layer to identify patterns or characteristics of the input data.
Hidden layers	Number of intermidate layer to process the information between input and output.
Noise injection layer	Layer to introduce gaussian noise to improve model generalization.
Activation function	Non linear function that determines the output of each neuron. ReLU is the most used because of its high computational efficiency and capacity to mitigate the gradient vanishing problem.
Learning rate	Rate controlling the size of weight adjustment during NN training, based of previous reward values.
Discount factor	Factor that determines the importance of future rewards in comparison with past rewards.
Optimizer	Algorithm to adjust NN parameters and to control convergence speed during training.
Episode	Total number of simulations episodes to be executed during training.
Epsilon starting	Initial value of exploration parameter $\epsilon$ for the $\epsilon$ -greedy strategy.
Epsilon ending	Final value of exploration parameter $\epsilon$ for the $\epsilon$ -greedy strategy, after exponential decay.

**Table 6.** Simulation parameters for traffic light control simulations.

DQN Model Simulation Parametes	
Parameter	Value
Batch size	32
Neural network layers	4
Number of Hidden Layers	3
Noise injection layers (Gaussian)	1
Activation function	ReLU
Learning rate	$1 \times 10^{-4}$
Discount factor	0.99 MSE
Optimizer	Adam
Episodes	5,000
Epsilon starting	1.0
Epsilon ending	0.05

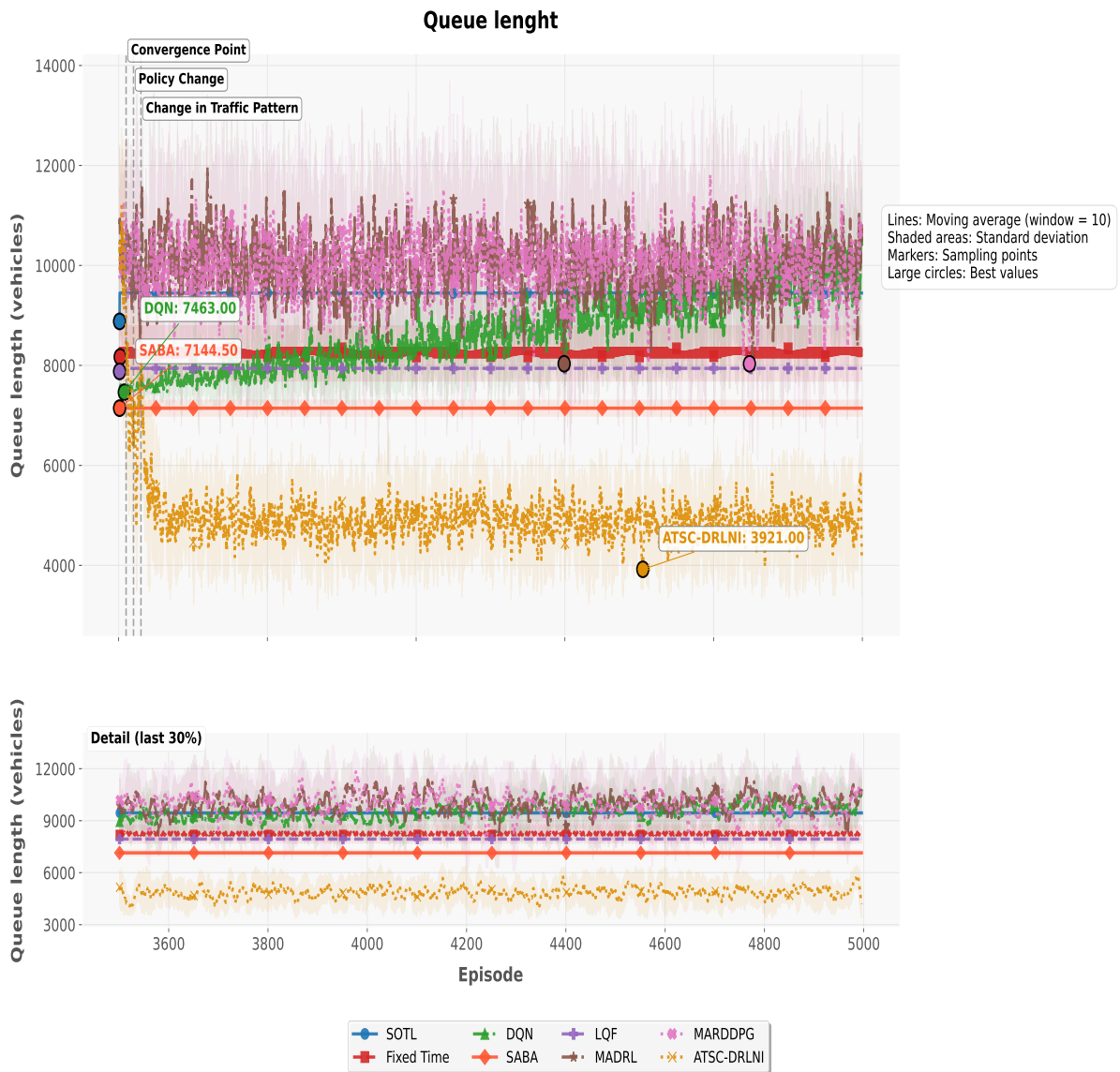
The results of applying the model are represented using different metrics: Average Waiting Time (AWT), Average Queue Length (AQL), Average Reward (AR), and traffic light changes (flickers), this one divided by  $F_{norm}$ . Figures 5-7 show an example of one simulation of 5,000 episodes, where the AR, AQL, and flickers are plotted against the number of episodes for all the eight models considered. In all cases, they are plotted using a moving average with a window of 10 episodes and a shadow area representing the standard deviation.

In Figure 5 the final value of the AR obtained with ATSC-DRLNI is higher than that of any other model; it is 38% better than the closest model, SABA. The AR value begins at -3.18 and ends at -1.80, showing a large variation in the first 300 episodes. SABA and LQF, non RL models, achieve a much faster convergence, however, their final AR values are -3.02 and -3.08, respectively. In the case of DQN, it has acceptable initial AR values, however, it shows a poor learning capacity, finishing with a value of -4.3, similar to those obtained with MARDDPG and MADRL.



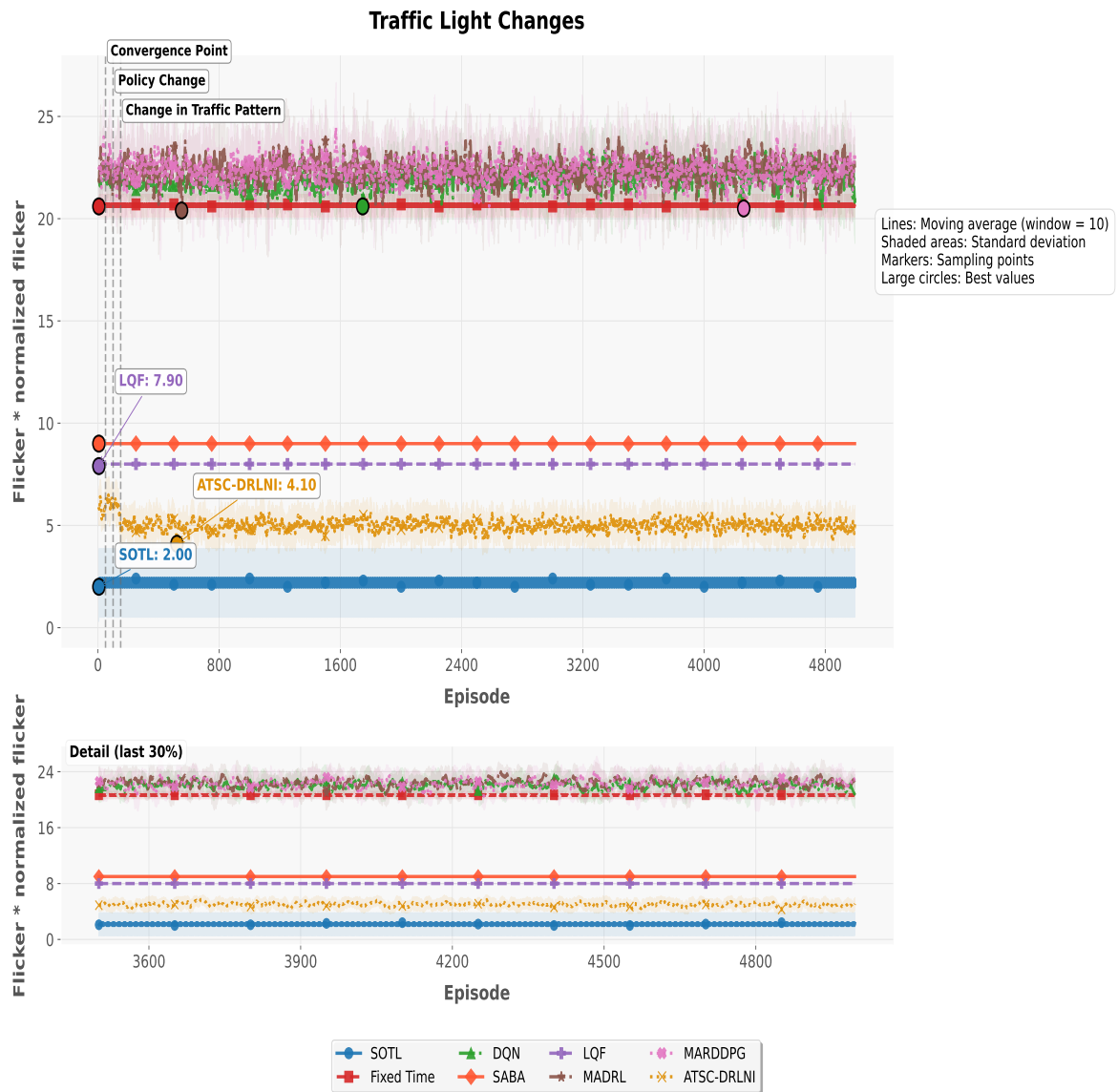
**Figure 5.** Simulation results: Average Reward (AR).

The queue length for the eight models is presented in Figure 6 expressed in seconds/hour. For ATSC-DRLNI, the value converges to 2,426 vehicles. This value is significantly smaller (40%) than those achieved by SABA, SOTL, LQF and FT, which attain intermediate queue length results. The other models, MARDDPG, MADRL and DQN have the worst queue length results.



**Figure 6.** Simulation results: Average Queue Length (AQL).

With respect to the adimensional number of flickers in Figure 7 the best results were obtained by SOTL, followed by ATSC-DRLNI, which compares favorable with the other six models, where LQF and SABA have a performance only slightly worse than ATSC-DRLNI.



**Figure 7.** Simulation results: adimensional number of traffic light changes (flicker).

A summary of the results for the metrics considered is included in Table 7, which includes mean values, final values, and standard deviation during the last episode.

Table 7. Complete results for the 8 tested models.

Metrics	Methods							
	vanilla_dqn	saba	marddpq	madrl	lqf	fixed_time	sofl	ATSC-DRLNI
AQL_mean	4341.726	3572.250	4994.691	5004.292	3970.439	4117.582	4723.433	2497.479
AQL_std	596.119	164.266	983.944	993.988	6.435	242.440	55.359	675.804
AQL_final	4708.643	3572.250	4992.671	4986.901	3970.500	4117.548	4724.000	2426.285
AWT_mean	2170.863	1786.125	2497.345	2502.146	1985.220	2058.791	2361.716	1248.739
AWT_std	298.059	82.133	491.972	496.994	3.217	121.220	27.680	337.902
Reward_mean	-3.661	-3.020	-4.103	-4.112	-3.090	-3.472	-3.500	-1.902
Reward_std	0.368	0.130	0.577	0.583	0.002	0.118	0.025	0.648
Reward_final	-3.912	-3.020	-4.100	-4.101	-3.090	-3.472	-3.500	-1.874
Flicker_mean	21.885	9.000	22.328	22.340	8.000	20.666	0.001	5.024
Flicker_std	1.199	0.000	1.738	1.757	0.014	0.472	0.042	0.832

Three different metrics to show learning stability, traffic balance, and system stability were also obtained based on the following definitions:

$$Learning\_stability = 1 - \min\left(1, \frac{\sigma_{R_{late}}}{|\mu_{R_{late}}| + \epsilon}\right) \quad (9)$$

with

$$R_{late} \{r_i : i \geq 3,000\} \text{ reward in final episodes} \quad (10)$$

$$\mu_{R_{late}} = \frac{1}{|R_{late}|} \sum_{r \in R_{late}} r \quad (11)$$

$$\sigma_{R_{late}} = \sqrt{\frac{1}{|R_{late}| - 1} \sum_{r \in R_{late}} (r - \mu_{R_{late}})^2} \quad (12)$$

and  $\epsilon = 0.1$  a regularization factor.

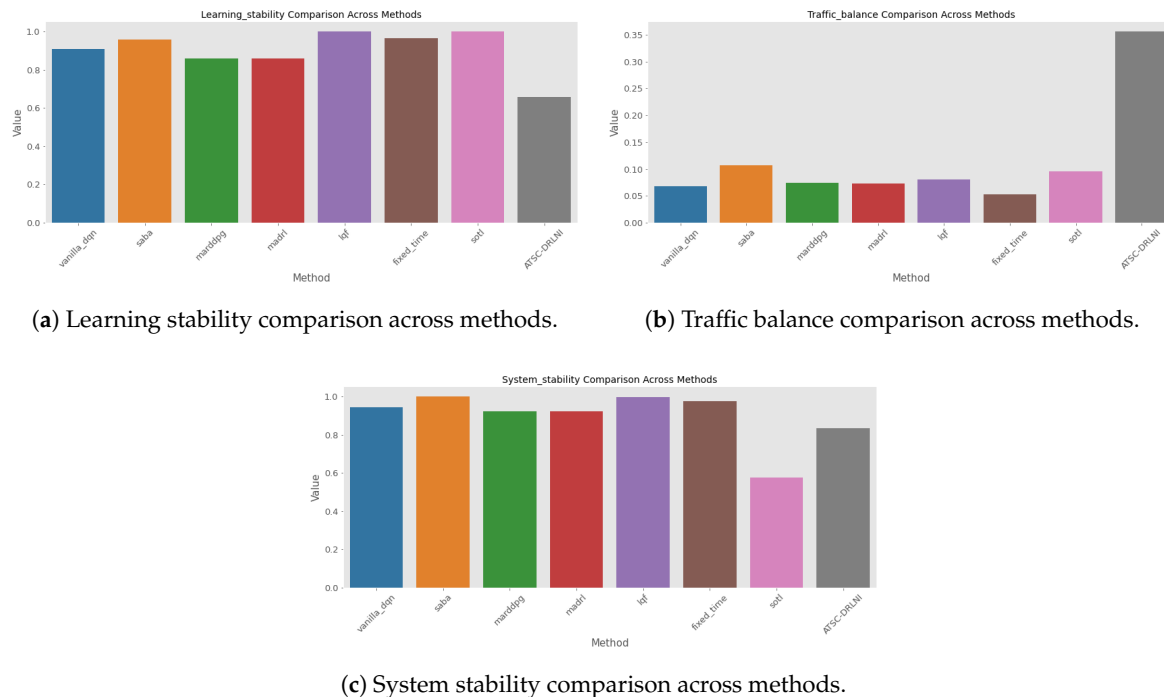
$$Traffic\_balance = 1 - \frac{|\mu_{Q_{NS}} - \mu_{Q_{EW}}|}{\max(0.1, \mu_{Q_{NS}} + \mu_{Q_{EW}})} \quad (13)$$

with  $\mu_{Q_{NS}}$  medium size of vehicle queue in the North-South direction and  $\mu_{Q_{EW}}$  medium size of vehicle queue in the East-West direction.

$$System\_stability = 1 - \frac{\sigma_{flicker}}{\max(0.1, \mu_{flicker})} \quad (14)$$

where  $\mu_{flicker}$  is the mean value of the flickers, and  $\sigma_{flicker}$  its standard deviation.

The values for these three metrics are shown in Figure 8 where, in all cases, the closer the value to 1, the better the metric. It can be noted that the ATSC-DRLNI performs best in terms of traffic balance and worse in terms of learning stability, an issue that will be commented below. Its performance in terms of System Stability is similar to the other methods.



**Figure 8.** Learning stability, traffic balance and system stability of the 8 simulated methods.

## 5. Discussion

Findings show a consistent positive correlation between the waiting time and the size of the queue length, similar to that shown in other studies [53,55], indicating that the inclusion of only one of two variables in the reward function is correct. The use of the number of flickers in the reward function showed a positive overall impact in the obtained results, in contrast to other methods for single intersections [47–49,53], where the reward function focuses only on waiting time and/or queue length, without considering this variable.

The results presented suggest that optimization using the ATSC-DRLNI model can be a viable option to handle traffic lights and alleviate traffic congestion in real scenarios. From a theoretical point of view, results also align with those in [53–57], as our simulation results indicate the convenience of using a combined reward function in single intersections.

There is research that has also used flickers [53,55], in a DRL context with multiobjective models, showing reductions up to 18% in waiting time. However, these studies do not combine queue length and flickers in the reward function.

The good performance of the ATSC-DRLNI model in balancing traffic (Figure 8-b), and in queue reduction (Figure 6) are a direct consequence of the NI layer that, by introducing controlled disturbances during training, forces agents to discriminate between spurious fluctuations in demand and real tendencies in congestion. This, in turn, prevents "over-reactions" in the control policy to small queue increments, a behavior that in other models (such as standard DQN) produces unstable phase changes. System stability, as shown in Fig. 8-c, corroborates that agents have learned a more robust control policy, that avoids overfitting with the data training set.

It is worth mentioning that the ATSC-DRLNI model has some limitations that must be addressed. First, simulations take longer than the other models considered in the comparison. Second, there are some environmental factors, such as pedestrians, special vehicles (ambulances, police cars, or autonomous vehicles), that are not included. Third, evaluation of multiple intersections and more complex scenarios is still missing.

Future research must focus on scenarios with multiple intersections, trying to model the ATSC at each intersection as a decentralized agent. The use of MADRL may be useful in this situation [74,75]. Finally, field testing is necessary to obtain a real assessment of ATSC strategies [31,47,74].

## 6. Conclusions

This research investigates the application of DRL to optimize ATSC and emphasizes the trade-off between minimizing queue lengths and suppressing signal instability at complex intersections. We developed and simulated a training environment on the SUMO platform, allowing the proposed ATSC-DRLNI model to learn signal timing policies through an A2C framework. Experimental results suggest that the agent skillfully balances queue size and phase flickers, outperforming seven benchmarks—including multiagent approaches such as MADRL and MARDDPG—with a 40% reduction in queue length and sustained stability. However, although these results highlight the considerable potential of decentralized traffic management, they are limited to the simulated contexts described here.

A key contribution of this study is the analysis of how the A2C algorithm improves communication between agents and learning stability, enabling simultaneous optimization of queue lengths and signal flashes under congested traffic conditions. However, the disparity between the optimizations derived from simulation and practical implementation remains a major obstacle, especially with regard to physical constraints and unmodeled stochastic elements. To address this disparity, future research focused on validating these control strategies in increasingly realistic simulations or hardware-in-the-loop (HIL) is needed.

In addition, the methodology proposed in this paper could bridge the gap between theoretical autonomous traffic systems and their practical implementation, although it requires rigorous safety evaluations before real-world deployment. Future work will focus on creating multi-objective reward functions that explicitly penalize signal flickering without sacrificing wait time minimization, as well as integrating vulnerable road users, such as pedestrians, into state representations. Overall, this study posits deep reinforcement learning as a powerful tool for designing scalable traffic management paradigms, while highlighting the need for further empirical corroboration for the transition from simulations to reality.

**Author Contributions:** Conceptualization, R.A.V.O., M.E.L.R. and L.A.; methodology, R.A.V.O., L.A. and M.E.L.R.; software, R.A.V.O.; validation, R.A.V.O., M.E.L.R., L.A. and H.G.; formal analysis, R.A.V.O.; investigation, R.A.V.O., M.E.L.R., L.A. and H.G.; resources, R.A.V.O.; data curation, R.A.V.O.; writing—original draft preparation, R.A.V.O.; writing—review and editing, L.A, M.E.L.R. and H.G.; visualization, R.A.V.O.; supervision, M.E.L.R., L.A. and H.G.; project administration, R.A.V.O.; funding acquisition, M.E.L.R. and L.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was partially funded by the Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT), DGAPA-UNAM, through projects IN101922 and IT100126.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is available upon request due to ethical and privacy restrictions of the Direction of Mobility of the city of Aguascalientes, Mexico, which requires authorization from the person in charge. The dataset is available upon request for privacy reasons however the code is available at: [https://github.com/RaulVelasquez/cnnn\\_ver2](https://github.com/RaulVelasquez/cnnn_ver2).

**Acknowledgments:** We thank the Direction of Mobility of the city of Aguascalientes, Mexico for providing access to the required information under the research project SSP/DM/CIV/S.010/22.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

IA	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
DRL	Deep Reinforcement Learning
MDP	Markov Decision Process
ANN	Artificial Neural Network
NN	Neural Network
CNN	Convolutional Neural Network
DQN	Deep Q-Network
ER	Experience Replay
ReLU	Rectified Linear Unit
GN	Gaussian Noise
NI	Noise Injection
GAs	Genetic Algorithms
cGAs	Cellular Genetic Algorithms
FL	Fuzzy Logic
TD	Time Difference
A2C	Advantage Actor Critic
SUMO	Simulation of Urban Mobility
ATSC	Adaptive Traffic Signal Control
FT	Fixed-Time
SCOOT	Split Cycle Offset Optimization Technique
SOLT	Self-Organizing Traffic Lights
RHODES	RealTime, Hierarchical, Optimized, Distributed, and Effective System
SCATS	Sydney Coordinated Adaptive Traffic System
MARDDPG	Multi-Agent Recurrent Deep Deterministic Policy Gradient
MARL	Multi-Agent Reinforcement Learning
MADRL	Multi-Agent Deep Reinforcement Learning
SABA	Single-Agent Based Algorithm
LQF	Longest Queue First
cGAs	Cellular Genetic Algorithms
CI	Continuous Integration
IoT	Internet of Things
CAVs	Connected and Automated Vehicles
AR	Average Reward
AQL	Average Queue Length
AR	Average Reward
Flicker	Traffic Light Changes
HIL	Hardware-In-Loop

## References

1. Thomson, I.; Bull, A. Urban traffic congestion: its economic and social causes and consequences. *Cepal review* **2006**, *2002*, 105–116. <https://doi.org/10.18356/f43aec5d-en>.
2. Bull, A.; for Latin America, U.N.E.C.; the Caribbean.; für Technische Zusammenarbeit, D.G. *Traffic Congestion: The Problem and how to Deal with it*; Cuadernos de la CEPAL, United Nations, Economic Commission for Latin America and the Caribbean, 2003.
3. González-Aliste, P.; Derpich, I.; López, M. Reducing Urban Traffic Congestion via Charging Price. *Sustainability* **2023**, *15*. <https://doi.org/10.3390/su15032086>.
4. Herath Bandara, S.J.; Thilakarathne, N. Economic and Public Health Impacts of Transportation-Driven Air Pollution in South Asia. *Sustainability* **2025**, *17*. <https://doi.org/10.3390/su17052306>.

5. Krzyzanowski, M.; Kuna-Dibbert, B.; Schneider, J. *Health effects of transport-related air pollution*; WHO Regional Office Europe, 2005.
6. Zhang, K.; Batterman, S. Air pollution and health risks due to vehicle traffic. *Science of the Total Environment* **2013**, *450-451*, 307 – 316. <https://doi.org/10.1016/j.scitotenv.2013.01.074>.
7. Inrix. 2025 INRIX Global Traffic Scorecard, 2025.
8. Cravioto, J.; Yamasue, E.; Okumura, H.; Ishihara, K.N. Road transport externalities in Mexico: Estimates and international comparisons. *Transport Policy* **2013**, *30*, 63 – 76. <https://doi.org/10.1016/j.tranpol.2013.08.004>.
9. Salazar-Carrillo, J.; Torres-Ruiz, M.; Davis, C.A.; Quintero, R.; Moreno-Ibarra, M.; Guzmán, G. Traffic Congestion Analysis Based on a Web-GIS and Data Mining of Traffic Events from Twitter. *Sensors* **2021**, *21*. <https://doi.org/10.3390/s21092964>.
10. Thorne, E.; Chong Ling, E.; Phillips, W. Assessment of the economic costs of vehicle traffic congestion in the Caribbean: a case study of Trinidad and Tobago. Technical report, Naciones Unidas Comisión Económica para América Latina y el Caribe (CEPAL), 2024.
11. Fleming, S. Traffic congestion cost the US economy nearly \$87 billion in 2018, 2019. Accessed: 2025-11-15.
12. Pop, M.D.; Pandey, J.; Ramasamy, V. Future Networks 2030: Challenges in Intelligent Transportation Systems. In Proceedings of the 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2020, p. 898 – 902. <https://doi.org/10.1109/ICRITO48877.2020.9197951>.
13. Goetz, A.R. Transport challenges in rapidly growing cities: is there a magic bullet? *Transport Reviews* **2019**, *39*, 701 – 705. <https://doi.org/10.1080/01441647.2019.1654201>.
14. Congestion, M.U.T. *Managing urban traffic congestion*; OECD Publishing: Paris, France, 2007.
15. Eom, M.; Kim, B.I. The traffic signal control problem for intersections: a review. *European Transport Research Review* **2020**, *12*. Cited by: 131; All Open Access, Gold Open Access, <https://doi.org/10.1186/s12544-020-00440-8>.
16. Thunig, T.; Scheffler, R.; Strehler, M.; Nagel, K. Optimization and simulation of fixed-time traffic signal control in real-world applications. In Proceedings of the Procedia Computer Science, 2019, Vol. 151, p. 826 – 833. <https://doi.org/10.1016/j.procs.2019.04.113>.
17. Koonce, P.; Rodegerdts, L. Traffic signal timing manual. Technical report, United States. Federal Highway Administration, 2008.
18. Qadri, S.S.S.M.; Gökçe, M.A.; Öner, E. State-of-art review of traffic signal control methods: challenges and opportunities. *European Transport Research Review* **2020**, *12*. <https://doi.org/10.1186/s12544-020-00439-1>.
19. Serban, A.A.; Frunzete, M. Analysis and comparison between urban traffic control systems. In Proceedings of the 2022 IEEE 28th International Symposium for Design and Technology in Electronic Packaging (SIITME), 2022, p. 92 – 95. <https://doi.org/10.1109/SIITME56728.2022.9988444>.
20. Cools, S.B.; Gershenson, C.; D'Hooghe, B., Self-Organizing Traffic Lights: A Realistic Simulation. In *Advances in Applied Self-Organizing Systems*; Springer London: London, 2013; pp. 45–55. [https://doi.org/10.1007/978-1-4471-5113-5\\_3](https://doi.org/10.1007/978-1-4471-5113-5_3).
21. Cools, S.B.; Gershenson, C.; D'Hooghe, B. Self-Organizing Traffic Lights: A Realistic Simulation. *Advanced Information and Knowledge Processing* **2008**, p. 41 – 50. [https://doi.org/10.1007/978-1-84628-982-8\\_3](https://doi.org/10.1007/978-1-84628-982-8_3).
22. Ferreira, M.; Fernandes, R.; Conceição, H.; Viriyasitavat, W.; Tonguz, O.K. Self-organized traffic control. In Proceedings of the Proceedings of the Seventh ACM International Workshop on Vehicular InterNetworking, New York, NY, USA, 2010; VANET '10, p. 85–90. <https://doi.org/10.1145/1860058.1860077>.
23. Mirchandani, P.; Wang, F.Y. RHODES to intelligent transportation systems. *IEEE Intelligent Systems* **2005**, *20*, 10 – 15. <https://doi.org/10.1109/MIS.2005.15>.
24. Lowrie, P. SCATS, Sydney Co-ordinated Adaptive Traffic System: A traffic responsive method of controlling urban traffic. , Roads and Traffic Authority of New South Wales. Traffic Control Section, 1990.
25. Wang, T.; Cao, J.; Hussain, A. Adaptive Traffic Signal Control for large-scale scenario with Cooperative Group-based Multi-agent reinforcement learning. *Transportation Research Part C: Emerging Technologies* **2021**, *125*. <https://doi.org/10.1016/j.trc.2021.103046>.
26. Tunc, I.; Yesilyurt, A.Y.; Soylemez, M.T. Different Fuzzy Logic Control Strategies for Traffic Signal Timing Control with State Inputs. In Proceedings of the IFAC-PapersOnLine, 2021, Vol. 54, p. 265 – 270. <https://doi.org/10.1016/j.ifacol.2021.06.032>.
27. Shaikh, P.W.; El-Abd, M.; Khanafer, M.; Gao, K. A Review on Swarm Intelligence and Evolutionary Algorithms for Solving the Traffic Signal Control Problem. *IEEE Transactions on Intelligent Transportation Systems* **2022**, *23*, 48 – 63. <https://doi.org/10.1109/TITS.2022.3014296>.

28. Guo, Y.; Liu, Y.; Oerlemans, A.; Lao, S.; Wu, S.; Lew, M.S. Deep learning for visual understanding: A review. *Neurocomputing* **2016**, *187*, 27–48. Recent Developments on Deep Big Vision, <https://doi.org/10.1016/j.neucom.2015.09.116>.
29. Kumar, R.; Sharma, N.V.K.; Chaurasiya, V.K. Adaptive traffic light control using deep reinforcement learning technique. *Multimedia Tools and Applications* **2024**, *83*, 13851 – 13872. <https://doi.org/10.1007/s11042-023-16112-3>.
30. Bálint, K.; Tamás, T.; Tamás, B. Deep Reinforcement Learning based approach for Traffic Signal Control. In Proceedings of the Transportation Research Procedia, 2022, Vol. 62, p. 278 – 285. <https://doi.org/10.1016/j.trpro.2022.02.035>.
31. Tan, J.; Yuan, Q.; Guo, W.; Xie, N.; Liu, F.; Wei, J.; Zhang, X. Deep Reinforcement Learning for Traffic Signal Control Model and Adaptation Study. *Sensors* **2022**, *22*. <https://doi.org/10.3390/s22228732>.
32. Damadam, S.; Zourbakhsh, M.; Javidan, R.; Faroughi, A. An Intelligent IoT Based Traffic Light Management System: Deep Reinforcement Learning. *Smart Cities* **2022**, *5*, 1293–1311. <https://doi.org/10.3390/smartcities5040066>.
33. Ibrokhimov, B.; Kim, Y.J.; Kang, S. Biased Pressure: Cyclic Reinforcement Learning Model for Intelligent Traffic Signal Control. *Sensors* **2022**, *22*. <https://doi.org/10.3390/s22072818>.
34. Wu, J.; Ran, Y.; Lou, Y.; Shu, L. CenLight: Centralized traffic grid signal optimization via action and state decomposition. *IET Intelligent Transport Systems* **2023**, *17*, 1247 – 1261. <https://doi.org/10.1049/itr2.12354>.
35. Abdelghaffar, H.M.; Rakha, H.A. A Novel Decentralized Game-Theoretic Adaptive Traffic Signal Controller: Large-Scale Testing. *Sensors* **2019**, *19*. <https://doi.org/10.3390/s19102282>.
36. Mo, Z.; Li, W.; Fu, Y.; Ruan, K.; Di, X. CVLight: Decentralized learning for adaptive traffic signal control with connected vehicles. *Transportation Research Part C: Emerging Technologies* **2022**, *141*. <https://doi.org/10.1016/j.trc.2022.103728>.
37. Chow, A.H.; Sha, R.; Li, S. Centralised and decentralised signal timing optimisation approaches for network traffic control. *Transportation Research Part C: Emerging Technologies* **2020**, *113*, 108 – 123. <https://doi.org/10.1016/j.trc.2019.05.007>.
38. Guo, Q.; Li, L.; (Jeff) Ban, X. Urban traffic signal control with connected and automated vehicles: A survey. *Transportation Research Part C: Emerging Technologies* **2019**, *101*, 313 – 334. <https://doi.org/10.1016/j.trc.2019.01.026>.
39. Chu, T.; Wang, J.; Codeca, L.; Li, Z. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems* **2020**, *21*, 1086 – 1095. <https://doi.org/10.1109/TITS.2019.2901791>.
40. Zhao, H.; Dong, C.; Cao, J.; Chen, Q. A survey on deep reinforcement learning approaches for traffic signal control. *Engineering Applications of Artificial Intelligence* **2024**, *133*. <https://doi.org/10.1016/j.engappai.2024.108100>.
41. Aoki, S.; Rajkumar, R. Cyber Traffic Light: Safe Cooperation for Autonomous Vehicles at Dynamic Intersections. *IEEE Transactions on Intelligent Transportation Systems* **2022**, *23*, 22519 – 22534. <https://doi.org/10.1109/TITS.2022.3146457>.
42. Wang, B.; He, Z.; Sheng, J.; Chen, Y. Deep Reinforcement Learning for Traffic Light Timing Optimization. *Processes* **2022**, *10*. <https://doi.org/10.3390/pr10112458>.
43. El-Qoraychy, F.Z.; Dridi, M.; Creput, J.C. Deep Reinforcement Learning for Vehicle Intersection Management in High-Dimensional Action Spaces. In Proceedings of the Proceedings of the 2024 7th International Conference on Machine Learning and Machine Intelligence (MLMI), New York, NY, USA, 2024; MLMI '24, p. 39–45. <https://doi.org/10.1145/3696271.3696278>.
44. Shi, Y.; Wang, Z.; LaClair, T.J.; Wang, C.R.; Shao, Y.; Yuan, J. A Novel Deep Reinforcement Learning Approach to Traffic Signal Control with Connected Vehicles. *Applied Sciences* **2023**, *13*. <https://doi.org/10.3390/app13042750>.
45. Miletić, M.; Ivanjko, E.; Gregurić, M.; Kušić, K. A review of reinforcement learning applications in adaptive traffic signal control. *IET Intelligent Transport Systems* **2022**, *16*, 1269 – 1285. <https://doi.org/10.1049/itr2.12208>.
46. Majstorović, Ž.; Tišljarić, L.; Ivanjko, E.; Carić, T. Urban Traffic Signal Control under Mixed Traffic Flows: Literature Review. *Applied Sciences* **2023**, *13*. <https://doi.org/10.3390/app13074484>.
47. Zhong, D.; Boukerche, A. Traffic Signal Control Using Deep Reinforcement Learning with Multiple Resources of Rewards. In Proceedings of the Proceedings of the 16th ACM International Symposium on Performance

- Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks, New York, NY, USA, 2019; PE-WASUN '19, p. 23–28. <https://doi.org/10.1145/3345860.3361522>.
48. Li, D.; Wu, J.; Xu, M.; Wang, Z.; Hu, K. Adaptive Traffic Signal Control Model on Intersections Based on Deep Reinforcement Learning. *Journal of Advanced Transportation* **2020**, *2020*. <https://doi.org/10.1155/2020/6505893>.
  49. Pan, T. Traffic Light Control with Reinforcement Learning, 2023, [arXiv:cs.AI/2308.14295]. <https://doi.org/10.48550/arXiv.2308.14295>.
  50. Agrawal, A.; Paulus, R. Intelligent traffic light design and control in smart cities: A survey on techniques and methodologies. *International Journal of Vehicle Information and Communication Systems* **2020**, *5*, 436 – 481. <https://doi.org/10.1504/IJVICS.2020.111456>.
  51. Li, Z.; Xu, C.; Zhang, G. A Deep Reinforcement Learning Approach for Traffic Signal Control Optimization, 2021, [arXiv:eess.SP/2107.06115]. <https://doi.org/10.48550/arXiv.2107.06115>.
  52. Nawar, M.; Fares, A.; Al-Sammak, A. Rainbow Deep Reinforcement Learning Agent for Improved Solution of the Traffic Congestion. In Proceedings of the 2019 7th International Japan-Africa Conference on Electronics, Communications, and Computations, (JAC-ECC), 2019, p. 80 – 83. <https://doi.org/10.1109/JAC-ECC48896.2019.9051262>.
  53. Mirbakhsh, S.; Azizi, M. Adaptive traffic signal safety and efficiency improvement by multi objective deep reinforcement learning approach, 2024, [arXiv:cs.LG/2408.00814]. <https://doi.org/10.48550/arXiv.2408.00814>.
  54. Shao, J.; Zheng, C.; Chen, Y.; Huang, Y.; Zhang, R. MoveLight: Enhancing Traffic Signal Control through Movement-Centric Deep Reinforcement Learning. *CoRR* **2024**, *abs/2407.17303*, [2407.17303]. <https://doi.org/10.48550/ARXIV.2407.17303>.
  55. Sattarzadeh, A.R.; Pathirana, P.N. Enhancing Adaptive Traffic Control Systems with Deep Reinforcement Learning and Graphical Models. In Proceedings of the 2024 IEEE International Conference on Future Machine Learning and Data Science (FMLDS), 2024, p. 39 – 44. <https://doi.org/10.1109/FMLDS63805.2024.00017>.
  56. Dhashyanth, N.; Hemchand, R.; Priyanga, R.; Soorya, S.; Sudheesh, P. Adaptive Traffic Control Using Deep Reinforcement Learning. In Proceedings of the 2024 IEEE Pune Section International Conference (PuneCon), 2024, pp. 1–8. <https://doi.org/10.1109/PuneCon63413.2024.10895492>.
  57. Ma, D.; Zhou, B.; Song, X.; Dai, H. A Deep Reinforcement Learning Approach to Traffic Signal Control With Temporal Traffic Pattern Mining. *IEEE Transactions on Intelligent Transportation Systems* **2022**, *23*, 11789 – 11800. <https://doi.org/10.1109/TITS.2021.3107258>.
  58. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **1998**, *86*, 2278–2324. <https://doi.org/10.1109/5.726791>.
  59. Zhang, A.; Lipton, Z.C.; Li, M.; Smola, A.J. *Dive into Deep Learning*; Cambridge University Press, 2023. <https://D2L.ai>.
  60. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*. <https://doi.org/10.1016/j.heliyon.2018.e00938>.
  61. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press, 2016. <http://www.deeplearningbook.org>.
  62. François-Lavet, V.; Henderson, P.; Islam, R.; Bellemare, M.G.; Pineau, J. An introduction to deep reinforcement learning. *Foundations and Trends in Machine Learning* **2018**, *11*, 219 – 354. <https://doi.org/10.1561/220000071>.
  63. Nian, R.; Liu, J.; Huang, B. A review On reinforcement learning: Introduction and applications in industrial process control. *Computers & Chemical Engineering* **2020**, *139*, 106886. <https://doi.org/10.1016/j.compchemeng.2020.106886>.
  64. Mahadevan, S. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine learning* **1996**, *22*, 159–195. <https://doi.org/10.1007/BF00114727>.
  65. Naeem, M.; Rizvi, S.T.H.; Coronato, A. A Gentle Introduction to Reinforcement Learning and its Application in Different Fields. *IEEE Access* **2020**, *8*, 209320–209344. <https://doi.org/10.1109/ACCESS.2020.3038605>.
  66. Aubret, A.; Matignon, L.; Hassas, S. An Information-Theoretic Perspective on Intrinsic Motivation in Reinforcement Learning: A Survey. *Entropy* **2023**, *25*. <https://doi.org/10.3390/e25020327>.
  67. Hariharan, N.; Anand, P.G. A Brief Study of Deep Reinforcement Learning with Epsilon-Greedy Exploration. *International Journal of Computing and Digital Systems* **2022**, *11*, 541 – 551. <https://doi.org/10.12785/ijcds/110144>.
  68. Dabney, W.; Ostrovski, G.; Barreto, A. Temporally-Extended  $\epsilon$ -Greedy Exploration. In Proceedings of the International Conference on Learning Representations, 2021. <https://doi.org/10.48550/arXiv.2006.01782>.

69. Konda, V.; Tsitsiklis, J. Actor-Critic Algorithms. In Proceedings of the Advances in Neural Information Processing Systems; Solla, S.; Leen, T.; Müller, K., Eds. MIT Press, 1999, Vol. 12. <https://doi.org/10.1137/S0363012901385691>.
70. Grondman, I.; Busoniu, L.; Lopes, G.A.D.; Babuška, R. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* **2012**, *42*, 1291 – 1307. <https://doi.org/10.1109/TSMCC.2012.2218595>.
71. Wang, K.; Liu, A.; Lin, B. Single-Loop Deep Actor-Critic for Constrained Reinforcement Learning With Provable Convergence. *IEEE Transactions on Signal Processing* **2024**, *72*, 4871–4887. <https://doi.org/10.1109/TSP.2024.3461963>.
72. Zur, R.M.; Jiang, Y.; Pesce, L.L.; Drukker, K. Noise injection for training artificial neural networks: A comparison with weight decay and early stopping. *Medical Physics* **2009**, *36*, 4810 – 4818. <https://doi.org/10.1118/1.3213517>.
73. Bishop, C.M. Training with Noise is Equivalent to Tikhonov Regularization. *Neural Computation* **1995**, *7*, 108–116. <https://doi.org/10.1162/neco.1995.7.1.108>.
74. Yang, T.; Fan, W. Enhancing Robustness of Deep Reinforcement Learning Based Adaptive Traffic Signal Controllers in Mixed Traffic Environments Through Data Fusion and Multi-Discrete Actions. *IEEE Transactions on Intelligent Transportation Systems* **2024**, *25*, 14196–14208. <https://doi.org/10.1109/TITS.2024.3399066>.
75. Friesen, M.; Tan, T.; Jasperneite, J.; Wang, J. Application of Multi-Agent Deep Reinforcement Learning to Optimize Real-World Traffic Signal Controls. *Authorea Preprints* **2021**. <https://doi.org/10.36227/techrxiv.16974493.v1>.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.