УПРАВЛІННЯ У ТЕХНІЧНИХ СИСТЕМАХ

CONTROL IN TECHNICAL SYSTEMS

UDC 004.94

DEEP REINFORCEMENT LEARNING OPTIMIZATION METHODS FOR TRAFFIC LIGHTS AT SIGNALIZED INTERSECTIONS

Boyko N. I. – PhD, Associate Professor, Associate Professor of the Department of Artificial Intelligence Systems, Lviv Polytechnic National University, Lviv, Ukraine.

Mokryk Y. L. – Post-graduate student, Department of Artificial Intelligence Systems, Lviv Polytechnic National University, Lviv, Ukraine.

ABSTRACT

Context. Intersections are the most critical areas of a road network, where the largest number of collisions and the longest waiting times are observed. The development of optimal methods for traffic light control at signalized intersections is necessary for improving the flow of traffic at existing urban intersections, reducing the chance of traffic collisions, the time it takes to cross the intersection, and increasing the safety for drivers and pedestrians. Developing such an algorithm requires simulating and comparing the work of different approaches in a simulated environment.

Objective. The aim of this study is to develop an effective deep reinforcement-learning model aimed at optimizing traffic light control at intersections.

Method. A custom simulation environment is designed, which is compatible with the OpenAI Gym framework, and two types of algorithms are compared: Deep Q-Networks and Proximal Policy Optimization. The algorithms are tested on a range of scenarios, involving ones with continuous and discrete action spaces, where the set of actions the agent may take are represented either by different states of the traffic lights, or by the length of traffic light signal phases. During training, various hyperparameters were also tuned, and different reward metrics were considered for the models: average wait time and average queue length. The developed environment rewards the agent during training according to one of the metrics chosen, while also penalizing it for any traffic rule violations.

Results. A detailed analysis of the test results of deep Q network and Proximal Policy Optimization algorithms was provided. In general, the Proximal Policy Optimization algorithms show more consistent improvement during training, while deep Q network algorithms suffer more from the problem of catastrophic forgetting. Changing the reward function allows the algorithms to minimize different metrics during training. The developed simulation environment can be used in the future for testing other types of algorithms on the same task, and it is much less computationally expensive compared to existing solutions. The results underline the need to study other methods of traffic light control that may be integrated with real-life traffic light systems for a more optimal and safer traffic flow.

Conclusions. The study has provided a valuable comparison of different methods of traffic light control in a signalized urban intersection, tested different ways of rewarding models during training and reviewed the effects this has on the traffic flow. The developed environment was sufficiently simple for the purposes of the research, which is valuable due to the large computational requirements of the models themselves, but can be improved in the future by expanding it with more complex simulation features, such as various types of intersections that aren't urban, creating a road network of intersections that would all be connected to each other, adding pedestrian crossings, etc. Future work may be done to refine the simulation environment, expand the range of considered algorithms, consider the use of models for vehicle control in addition to traffic light control.

KEYWORDS: reinforcement learning, signalized intersection, traffic control, proximal policy optimization, deep Q-learning.

ABBREVIATIONS

DRL is a deep reinforcement learning;
SUMO is a Simulation of Urban MObility;
SOTL is a self-organizing traffic light;
DQN is a deep Q network;
VRCIS is a Vehicle-Road-Cloud Integration System;
CAV is a connected autonomous vehicle;
HDV is a Human-driven vehicle;
RL is a reinforcement learning;
AWT is an average waiting time;
AQL is an average queue length;
PPO is a Proximal Policy Optimization;
TRPO is a Trust Region Policy Optimization;
V2X is a Vehicle-to-everything;
V2I is a Vehicle-to-infrastructure.

NOMENCLATURE

S is a state space of the environment;

DOI 10.15588/1607-3274-2025-4-21

A is an action space;

$P(s_{t+1} \mid s_t, a_t)$ is a transition probability function;

$R(s_t, a_t)$ is a reward function;

α, β, λ are weight coefficients;

γ is a discount factor;

a_t are actions;

s_{t+1} are transitions;

$r_t(\theta)$ is a probability ratio;

$\hat{A}_t$ is an aestimated advantage function;

$Q(S_t, A_t)$ represents the Q-values in the S_t state if action A_t was chosen;

α parameter represents the learning rate;

γ is a discount factor.

INTRODUCTION

As cities expand and the number of vehicles on the roads continues to climb, the challenge of managing traffic efficiently has become more urgent than ever. Ensuring that road networks are not only safe but also capable of handling increasing traffic volumes is a top priority for modern traffic management authorities. In this context, smart solutions for optimizing traffic flow – especially at intersections where congestion and accidents are most common – are gaining critical importance.

Statistics show that intersections are the most accident-prone areas of any road network, with a higher frequency of collisions compared to other road segments. [1] Notably, intersections without traffic lights experience more accidents than signalized ones. Traffic congestion in cities is an especially acute problem in the morning and evening when people commute to and from work. [2] Traffic signals help to regulate the flow and reduce the likelihood of human error, contributing to safer traffic environments.

Object of the study: the phases of traffic lights and their effect on the flow of traffic at signalized urban intersections.

Subject of the study: deep reinforcement learning-based algorithms for controlling the phases of traffic lights according to the state of the road network around an intersection.

The aim of this study is to develop a reinforcement learning agent that is capable of controlling the phases of traffic lights at a signalized intersection to optimize a chosen evaluation metric, such as the average wait time of the vehicles at the intersection, or their average queue length. This algorithm needs to also minimize the danger to the vehicles on the road by avoiding collisions between them. The goal is for these findings to contribute to the robustness of existing urban infrastructure, to make it safer and more efficient.

Tasks of the research:

– Analyze existing methods and algorithms for traffic light control optimization at a signalized intersection.

– Develop or select a suitable simulation environment that would simulate the flow of traffic, providing the agents with rewards and penalties.

– Create deep learning algorithms that are capable of controlling traffic lights in the simulated environment.

– Analyze the results and compare them with existing traffic light control methods.

1 PROBLEM STATEMENT

The primary goal of this study is to develop an effective DRL model aimed at optimizing traffic light control at intersections. To achieve this, a simulated environment will be created to replicate vehicular movement through intersections. Various DRL algorithms will be implemented and evaluated based on key performance metrics related to traffic efficiency and safety.

The rationale behind this research lies in the increasing demand for intelligent traffic control systems that can manage traffic flows dynamically and adaptively, particularly in large urban areas. Such systems have the potential to reduce congestion, improve traffic throughput, and minimize accidents resulting from human mistakes.

To fulfill this objective of designing a method for traffic light timing optimization, several tasks are required:

1. Analyze existing machine learning tools for traffic light optimization at intersections.

2. Develop an environment capable of simulating traffic flows at signalized intersections.

3. Design and implement a reinforcement learning algorithm to optimize traffic light timing within the simulation.

4. Tune the parameters of the optimization algorithm and compare its performance to that of traditional or existing control strategies.

Before a simulation environment or algorithm can be developed however, we need to discuss several issues that plague the development of such models:

– One major challenge is creating a realistic and scalable virtual environment for training agents that would control traffic lights using reinforcement learning. This environment must support extensive simulations and provide different performance metrics. A key difficulty is the automated creation of rare but critical traffic scenarios, as manually modeling all possible events is computationally infeasible.

– Building an algorithm that will be able to efficiently control the movement of traffic through an intersection, while avoiding any traffic collisions or other traffic rule violations, and at the same time minimize the waiting time of passengers. While traffic rules can be programmed, replicating nuanced human decision-making remains difficult.

– High-precision vehicle detection and movement prediction. Autonomous systems require highly accurate sensing and prediction capabilities, far exceeding those in other industries. These systems must function reliably under varying conditions, including poor weather and low visibility.

DOI 10.15588/1607-3274-2025-4-21

– Scalability and data demands: deep learning models used for traffic control are computationally intensive and require vast amounts of diverse training data. To ensure robustness across all possible driving scenarios, these models must be trained on extremely large, representative datasets – often measured in petabytes.

All of the above points make it clear that creating a hand-crafted algorithm for controlling traffic lights is challenging. This is the reason why machine learning approaches have often been considered for this task instead. However, as noted, the method of using neural networks or any other machine learning tool is also not without issues. Particularly when it comes to modelling the necessary training data and rewards.

The problem of optimizing traffic light control using reinforcement learning can be formally described as a Markov Decision Process (MDP) defined by the tuple:

$$M = (S, A, P, R, \gamma), \quad (1)$$

1. A state $s_t \in S$ includes:
– the current traffic light phases;
– the number of vehicles approaching the intersection;
– the number of vehicles that have already passed;
– queue lengths on each approach to the intersection.
2. A includes:
– Discrete: predefined combinations of traffic light phases (e.g., {*green-red*, *yellow-red*, ...});
– Continuous: the duration of the current traffic light phase $a_t \in [\underline{a}_{\min}, a_{\max}] \subset \mathrm{R}$.
3. $P(s_{t+1} \mid s_t, a_t)$ defines the probability of moving to state s_{t+1} after taking action a_t in state s_t. This is modeled by the simulator.
4. $R(s_t, a_t) \in R$ defined as:

$$R(s_t,a_t) = -\alpha\, AWT(s_t,a_t) - \beta \cdot AQL(s_t,a_t) - \lambda \times \times penalties(s_t,a_t). \quad (2)$$

The goal is to find an optimal policy $\pi^*: S \to A$ that maximizes the expected cumulative discounted reward:

$$\pi^* = arg \max_{\pi} \mathrm{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R\left(s_t, a_t\right)\right]. \quad (3)$$

Methods for Finding the Optimal Policy:
– Deep Q-Network: approximates the Q-function $Q(s, a) \approx Q_\theta(s, a)$ using a neural network and updates it using the Bellman equation:

$$Q\left(s_t, a_t\right) \leftarrow (1-\alpha) Q\left(s_t, a_t\right) + \alpha\left[R_t + \gamma \max_{a'} Q\left(s_{t+1}, a'\right)\right]. \quad (4)$$

– PPO: a policy gradient method that optimizes the clipped surrogate objective:

$$L^{CLIP}(\theta) = E_t\left[\min\left(r_t(\theta)\hat{A}^t, clip\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}^t\right)\right]. \quad (5)$$

To solve this optimization problem, modern DRL algorithms such as DQN or PPO are employed. In particular, PPO is used for both discrete and continuous action spaces, while DQN is suitable for discrete cases.

2 LITERATURE REVIEW

Before choosing or designing our own simulation environment and reinforcement learning algorithm for training the traffic light control agents, an analysis of existing literature on the topic was performed.

This section has been split into two parts: one in which we analyze existing environments for running traffic simulations, comparing their advantages and disadvantages. In the second part, we looked at various other methods that have been proposed for vehicle and traffic light management using machine learning techniques.

One of the first tasks in building an effective algorithm for traffic light control is to build a virtual simulated environment for the training and validation of the developed algorithms. To begin with, it is worth considering existing solutions to this problem. Such an environment should allow for a large number of simulations to be run quickly to allow for training many iterations of the reinforcement learning algorithm. Also, the environment should provide certain metrics by which to evaluate and compare the performance of different algorithms.

There are a number of such environments that have been created for simulating road traffic. It is worth evaluating these environments according to several criteria: speed, ease of implementation of the machine learning algorithm in the environment, and simulation accuracy.

CARLA Simulator [3] is a simulator for training and validation of autonomous driving systems. CARLA provides open digital data (city plans, buildings, vehicles) that have been created for this purpose and can be freely used (Figure 1). The modeling platform supports flexible specification of sensor sets, environmental conditions, full control of static and dynamic agents, mapping, and much more. The disadvantages of the environment are the same as in the previous environment: large system requirements that can slow down training and difficulty in integrating with the machine learning model development environment. Many studies have been done using the CARLA Simulator related to autonomous traffic control [4, 5].

Flow [6] is a traffic simulation platform designed to evaluate driving algorithms for autonomous vehicles in various critical scenarios. It allows integration with machine learning software development environments. The main problem can be too much simulation complexity, which will have a bad effect on the length of training.

DOI 10.15588/1607-3274-2025-4-21

Figure 1 – Visualization of the CARLA Simulator environment

Eclipse SUMO [7] is another open source environment. This environment focuses on traffic simulation tasks with many vehicles to plan and predict different traffic situations (Figure 2). One of the key disadvantages of this environment is the high requirements for the system on which the simulation takes place. Training many agents will be much more difficult due to the high complexity of the simulator itself. In addition, it is difficult to connect this simulator to a Python-based machine learning model development environment, such as Jupyter.

Figure 2 – Visualization of the Eclipse SUMO environment

DriverGym [8] is an open-source reinforcement learning model training environment that allows you to train models to learn driving rules on real data (Figure 3). The algorithm can access semantic maps to control it. Other agents can be modeled from existing data or controlled by a special behavioral model previously trained on real-world data. This environment provides an extensive evaluation system with easily customizable metrics to evaluate trained algorithms.

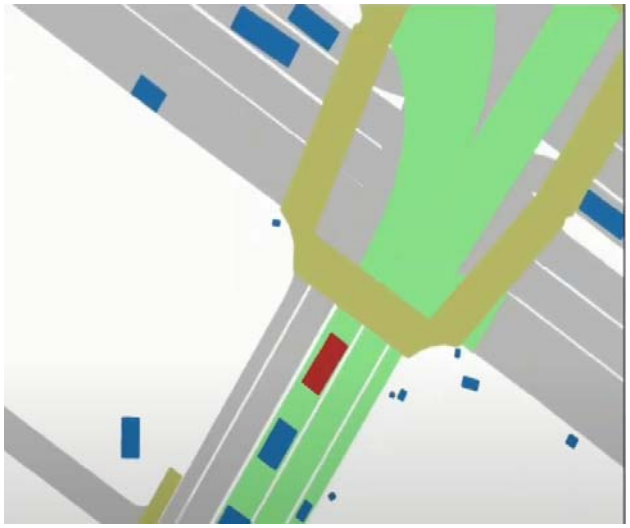

Figure 3 – Visualization of the DriverGym environment

There are also some simulators that are customized for the OpenAI Gym platform [9], such as DeepCars, gym_trafficlight, and gym_graph_traffic. Such environments are usually very easy to integrate with the Python development environment and do not require large computing resources, but they are often too simple. For example, some of them only simulate movements in discrete steps, or have no way to evaluate the algorithm's performance using metrics.

After looking at existing applications for traffic simulation, we have decided to implement our own environment. The advantages of this approach include:

The ability to easily connect it to a machine learning algorithm development environment.

The ability to define any simulation parameters and any metrics to evaluate the performance of each algorithm

The ability to fully customize performance and optimize the simulation for the tasks we need during development

Existing Optimization Approaches.

Aside from the simulation environment, we also need to choose the algorithms for training agents for the task of controlling traffic lights. There are a few things to consider here, including how these algorithms are known to perform at similar tasks, how well they scale and whether they support discrete or continuous actions spaces and observation spaces. Because of this, a few studies have been chosen for analysis to determine which algorithms are the most suitable for the task. The results of this analysis are summarized in Table 1. A further detailed description of each paper is given after the table.

DOI 10.15588/1607-3274-2025-4-21

Table 1 – Summary of existing research into traffic and traffic signal control with deep reinforcement learning

Name	Task	Tools	Year	Ref.
Deep Reinforcement Learning for Traffic Light Timing Optimization	Controlling traffic light timing	Q-learning, MaxPressure, Self-organizing traffic lights	2022	[10]
Deep Q-network-based traffic signal control models	Controlling traffic signals	Q-learning, Synchro model	2021	[11]
An Efficiency Enhancing Methodology for Multiple Autonomous Vehicles in an Urban Network Adopting Deep Reinforcement Learning	Controlling a network of autonomous vehicles at an unsignalized intersection	Reinforcement learning, Proximal policy optimization	2021	[1]
A Control Method with Reinforcement Learning for Urban Un-Signalized Intersection in Hybrid Traffic Environment	Controlling autonomous vehicles at an unsignalized intersection	Reinforcement learning, Proximal policy optimization, VRCIS, V2X, V2I	2022	[12]

In the first paper [10], the authors propose a traffic signal synchronization optimization method based on a dual dueling deep Q-network, MaxPressure, and SOTL, namely EP-D3QN, which controls traffic flows by dynamically adjusting the duration of traffic lights in a cycle and whether the phase is switched based on predefined rules and lane pressure. In EP-D3QN, each intersection corresponds to an agent, and the road entering the intersection is divided into grids, each grid stores the speed and position of the vehicle, thus forming an information matrix of the vehicle as well as the state of the agent. An agent action is a set of traffic light phases in a signal cycle that has four values. The effective duration of the traffic light is 0–60 seconds, and the switching of the traffic light phases depends on its pressing and the established rules. The agent's reward is the difference between the sum of the accumulated waiting times of all vehicles in two consecutive signal cycles. SUMO is used to model two traffic scenarios. Two types of evaluation indicators are selected, and four methods are compared to verify the effectiveness of EP-D3QN. The experimental results show that EP-D3QN has excellent performance in light and heavy traffic flow scenarios, which can reduce the waiting time and travel time of vehicles and improve the traffic efficiency of the intersection.

The authors of the next study [11] investigated a solution-finding method using artificial intelligence, which has recently gained widespread attention because it can solve complex problems such as traffic signal control. In this study, two traffic signal control models were developed using reinforcement learning and microscopic simulation-based estimation for an isolated intersection and two coordinated intersections. To develop these models, a DQN was used, which is a promising reinforcement learning algorithm. The performance was evaluated by comparing the developed traffic signal control models in this study with a fixed time signal optimized by Synchro, which is a traffic signal optimization model. The evaluation showed that the developed traffic signal control model of the isolated intersection was validated, and the intersection coordination was better than that of the fixed-time signal control method.

The next paper [1] is devoted to advanced deep reinforcement learning. It considers the impact of leading autonomous vehicles on the urban network in mixed traffic. A set of hyperparameters is proposed to achieve better performance. First, a set of hyperparameters for reinforcement learning agents is selected. Second, experiments are conducted with an autonomous vehicle in an urban network with different levels of autonomous vehicle penetration. Third, the superiority of leading autonomous vehicles is evaluated using fully manual vehicles and leading manual vehicle experiments. Finally, proximal behavior optimization with a pruned objective is compared to proximal behavior optimization with an adaptive Kullback-Leibler penalty to test the superiority of the proposed hyperparameter. It is shown that the fully automated traffic increased the average speed by a factor of 1.27 compared to the entire experiment with a manual vehicle. The proposed method becomes significantly more efficient at higher penetration rates of autonomous vehicles. In addition, leading autonomous vehicles can help reduce congestion.

The last research we analyzed [12] is devoted to the method of coordinated control with optimization of proximal behavior in the VRCIS, where this control problem is formulated as a reinforcement learning problem. In this system, vehicles that are connected to the entire environment V2X were used to maintain communication between vehicles, and vehicle wireless communication technology can detect vehicles that use vehicle-to-infrastructure wireless communication V2I, thus achieving a cost-effective method. Then, the CAV defined in VRCIS was trained to adapt to HDV safely through intersections using reinforcement learning. A valid, scalable RL framework is developed that can convey topologies that may be dynamic traffic. Then, the state, action, and reward of RL are designed according to the problem of an urban intersection without traffic lights. Finally, how to deploy within the RL framework was described and several experiments were conducted with this framework to verify the effectiveness of the proposed method.

3 MATERIALS AND METHODS

As explained previously, a custom simulation environment has been developed. This will allow us to

DOI 10.15588/1607-3274-2025-4-21

better customize the simulation, make it more efficient and more easily integrate it with the tools for training RL agents.

Two algorithms have been chosen for training RL agents: Deep Q-Networks and Proximal Policy Optimization. This is because they have shown good results in the studies examined previously, and they are relatively lightweight in terms of computation, so many iterations can be trained without significant time requirements. Furthermore, such algorithms can be trained on a discrete or continuous action space (particularly PPO), which will be utilized in the experimental setup.

The simulated environment consists of an intersection where four roads meet. Each direction has a traffic light that allows traffic to move from that direction to any other direction. For example, the upper traffic light allows traffic from the lower street in any direction.

Traffic lights switch two at a time. At the beginning of the simulation, the left and right traffic lights are red. Accordingly, the upper and lower traffic lights are green. After switching, the upper and lower traffic lights first switch to yellow, then to red. After that, the left and right traffic lights switch to yellow and then to green. Figure 4 shows a simple visualization of what the environment would look like if we display all the vehicles and roads to scale.

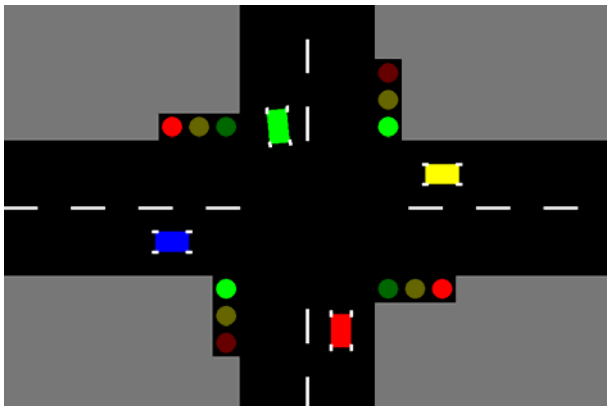

Figure 4 – Visualization of one of the scenarios in the custom simulated environment of a signalized intersection

Traffic light control is implemented using one of two approaches: discrete action spaces or continuous action spaces.

In the discrete case, traffic lights can occupy one of four predefined states, corresponding to signal phases for horizontal and vertical directions: green-red, yellow-red, red–yellow, and red-green. Every 1,000 simulation steps, the agent selects one of these possible states, determining the next phase of traffic flow. Figure 5 depicts how the discrete action space algorithm is applied during training for agents that control traffic light states.

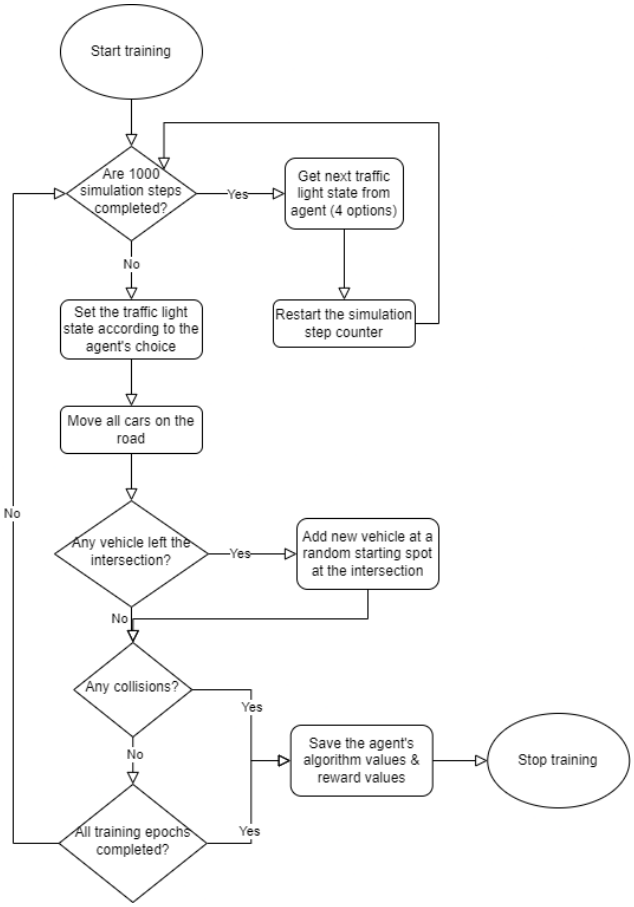

Figure 5 – Diagram of the discrete action space traffic light agent training algorithm

The continuous approach (Figure 6) allows for more dynamic control, where the agent selects the duration of each traffic light phase from a continuous range of values. Once a phase ends, the duration of the next phase is determined by the output of the reinforcement learning model, enabling finer adjustments based on traffic conditions.

The movement of cars is predetermined in this model, each car moves forward until it either stops at a red or yellow traffic light at the intersection or proceeds to cross at the intersection if the traffic light is green. While crossing, each car has a randomly determined direction that it needs to get to. The direction can be either straight ahead, left or right of the car's starting location. The car will either proceed straight or turn left or right depending on its current position, the state of the traffic light and the randomly chosen target direction.

The observation space for the traffic light control agent includes:

– The current state of the traffic signals.

– The proportion of vehicles currently on the road approaching the intersection from different directions.

– The proportion of vehicles that have already passed through the intersection from each direction.

– The proportion of vehicles still waiting in queues, i.e., the relative queue length on each side of the intersection.

DOI 10.15588/1607-3274-2025-4-21

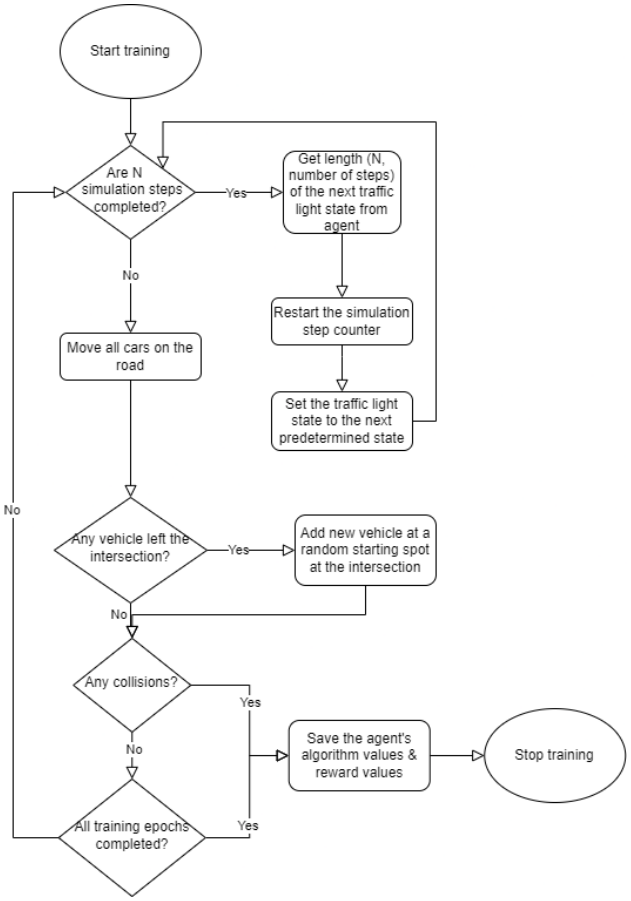

Figure 6 – Diagram of the continuous action space traffic light agent training algorithm

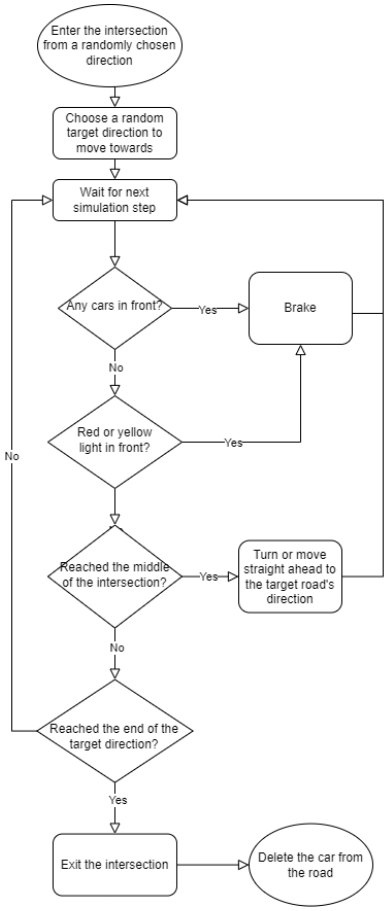

Figure 7 – Diagram of a car's movement pattern during a training episode

During training, it is possible to randomly vary traffic flow patterns by adjusting the distribution of vehicles across the intersection's lanes. Parameters control the rate at which non-agent-controlled vehicles enter the intersection from each direction. The rates can be thought of as a percentage chance that a new vehicle will come in from the top, bottom, left or right direction on the intersection. These rates are randomized at the start of each training episode, simulating asymmetric or uneven traffic conditions and adding variability to the environment. The agent is provided with these vehicle distribution ratios during training, helping it adapt its traffic light control to varying scenarios. Figure 7 shows the simple algorithm for the movement of randomized traffic on the road.

The reward function for traffic light control algorithms is based on two key performance metrics:

– Average Waiting Time (6) – the average number of simulation steps that each vehicle spends waiting at a red light before crossing the intersection.

– Average Queue Length (7) – the average number of vehicles queued at red lights from each direction at any given time.

These metrics are evaluated using the following formulas:

$$AWT = \frac{1}{n}\sum_{i=1}^{n} w_i, \tag{6}$$

where n is the total number of vehicles that have been at the intersection during the training or validation episode, w_i is the time that the vehicle spend standing still due to a red light or other vehicles ahead.

$$AQL = \frac{1}{m}\sum_{i=1}^{m} q_i, \tag{7}$$

where m is the number of directions around the intersection (in this case, always 4) and q_i is the number of vehicles that are on that road at the current step of the simulation.

To summarize, the goal of the traffic light control agents is to minimize either the AWT or the AQL metric, chosen before training for each experiment. During validation, all algorithms can be tested on their performance with both metrics, to compare the impact of the choice of training metric on the algorithm's performance.

DOI 10.15588/1607-3274-2025-4-21

4 EXPERIMENTS

Both DQN and PPO models were trained on the discrete action space scenario, while PPO models were additionally trained on the continuous action space scenario. The models were trained using both AWT and AQL metrics for the reward function, one at a time. The number of vehicles at the intersection and the directions from which they came was randomly varied with a uniform distribution.

We consider DQN models built from two neural networks: A Q-network and a target network with the same architecture. The architecture of the networks consists of two hidden layers, 64 neurons each, and an output layer, where the number of neurons corresponds to the size of the action space. We consider DQN models with different values of the learning rate hyperparameter, DQN with a decreasing learning rate hyperparameter, and PPO with different values of the entropy coefficient hyperparameter. The networks used in the PPO algorithms have the same architecture as those used in the DQN models. All algorithms were trained for 8 thousand steps, the constant learning rate was set to 0.01 and 0.001, and the learning rate decreased from LR 1.0 or 0.1, which decreased linearly to zero throughout the training process. The values of the entropy coefficient of the PPO models were 0.1, 0.01, 0.001, and 0.0001.

At the core of RL is an agent's policy, a mapping from states to actions that guides behavior. The agent must balance exploration (trying new actions to discover rewards) and exploitation (choosing known actions that yield good outcomes), a trade-off fundamental to RL.

One key algorithm for learning such behavior is Q-learning, a model-free RL method. It does not require prior knowledge of the environment and learns by estimating Q-values, the expected cumulative reward of taking an action in each state and following the policy thereafter. These Q-values are stored in a Q-table, and the agent selects actions based on the highest Q-value for its current state. The Bellman equation (8) lies at the heart of Q-learning, since it allows updating the Q-values iteratively [13].

$$Q(S_t,A_t)=(1-\alpha)\,Q(S_t,A_t)+\alpha(R_t+\gamma\max_{\alpha}Q(S_{t+1},\alpha)). \quad (8)$$

However, Q-learning struggles with large or continuous state spaces, as storing and updating Q-values in a table becomes impractical. This challenge is addressed by DQN, which replaces the Q-table with a neural network that approximates the Q-function. This allows the agent to generalize from limited experience to unseen states, making it suitable for complex tasks.

DQN also uses experience replay, where past experiences are stored and randomly sampled during training. This reduces correlations between consecutive actions and stabilizes learning. Together, neural function approximation and experience replay enable DQN to scale Q-learning to real-world applications, such as traffic signal control, game playing, and robotics.

Policy gradient methods are fundamental among the recent breakthroughs in the use of deep neural networks to control agents in video games, 3D environment and even go and chess. However, getting optimal results with strategy gradient methods is challenging because they are very sensitive to the choice of hyperparameters, such as the size of the training step. Too small a step makes the learning extremely slow, and too large a step oversaturates the signal with noise, which can also cause a catastrophic drop in the learning results. Very often, such algorithms are inefficient in the use of the training set, meaning that they require millions or billions of steps to learn how to perform the simplest tasks.

The problem with strategy search methods is that they can converge slowly if the information they operate on is noisy. For example, this happens when trajectories are long and the variance of returns is large in episodic problems. As mentioned above, value function-based methods that rely on time differences can help in this case. In recent years, several actor-critic algorithms have been proposed that follow this idea and have been shown to perform well on a variety of tasks.

Recently, machine learning researchers have been looking for a way to avoid these problems with algorithms such as TRPO and ACER by limiting or otherwise optimizing the size of the strategy update step. Such methods have their own drawbacks: ACER is much more complicated to implement than PPO, requiring the addition of code to adjust the optimal strategy and the repetition buffer, and these algorithms have only a slight advantage over PPO. TRPO, while useful for problems with a continuous action space, is not easy to combine with algorithms that share parameters between the strategy and the loss function, such as PPO.

PPO is a family of model-free reinforcement learning algorithms developed by OpenAI in 2017. Such algorithms are examples of strategy gradient methods, meaning they search in the strategy space rather than assigning values to specific actions in the action space.

PPO algorithms have some advantages over TRPO methods, but their implementation is relatively simpler, and they generally use training samples more efficiently than TRPO [14]. These results are achieved by changing the objective function in PPO algorithms [15].

Various metrics can be used to evaluate vehicle and traffic signal control algorithms at an intersection, some of which have already been mentioned in the previous sections.

First, it is worth noting that the neural network loss function, which approximates the Q-function for a DQN model or the reward function for a PPO model, is not a significant metric. The problem with such an indicator is that the loss function in these models is constantly changing, because the target data for network training is not stable. Usually, the loss function in DQN and PPO models is constantly growing or fluctuating, despite the improvement of the model results in practice. That is why it is necessary to look at the results of the algorithm in a

DOI 10.15588/1607-3274-2025-4-21

validation environment, and at the metrics that can be obtained from it.

One such metric is the average reward function, which is defined by the training or validation environment. In this environment, which was described earlier in this paper, the reward function takes several important factors into account: traffic rules, distance to the target, and successful completion of the episode. The developed reward function gives high rewards when cars avoid colliding with each other, avoiding crossing into the oncoming lane, crossing the intersection during a red light at a traffic light, waiting during a green light, and crossing the edge of the road. The positive factors are getting closer to the goal and successfully reaching the right lane on the right side of the intersection. The fewer steps required to reach the goal, the higher the reward value.

Another factor that can be used as a metric is the AWT. In practice, this metric can be roughly estimated using the average length of episodes, because in the environment developed, a training or validation episode ends after successfully reaching the target intersection. Thus, if the episode is completed successfully, the average number of cards, seconds, or steps in such episodes can be taken into account to find out how optimally cars move through the intersection.

Another metric for evaluating traffic signals is the average queue length at an intersection. This metric measures the average queue length of vehicles that have not yet crossed the intersection, for example, because they are waiting for the traffic light to change, or because they are waiting for the car in front of them to move.

In addition to metrics of the quality of the algorithms in the validation environment, we can also consider metrics of the training of the algorithms themselves. While the loss function metric does not provide valuable results when training DQN models, it is also possible to obtain an estimate of the training time of the models depending on their hyperparameters, and the number of episodes or iterations or steps that are required to achieve satisfactory results. Such metrics can help compare algorithms in terms of training performance, which can be valuable for selecting the fastest or cheapest algorithm to train on much more complex environments.

5 RESULTS

Two different training approaches were used for the algorithms: one where the reward function depended on the AWT of vehicles at a red light, and another where it instead depended on the AQL of vehicles at a red light. Additionally, the PPO algorithm was trained using two distinct traffic light control methods: a discrete approach, where the algorithm selects the traffic light phase every 100 steps, and a continuous approach, where the algorithm determines the duration (in steps) of each traffic light phase. The DQN algorithm was trained only with a discrete action space, as it tends to yield better results in such settings and because most DQN implementations support only discrete action spaces.

The models were trained on these environments for tens of thousands of iterations, where each iteration represents a set number of steps. After every few iterations the parameters of the models were saved to a file, which allows comparing them individually for validation in case the final models have a larger error compared to the earlier ones. This is a common problem with reinforcement learning algorithms (especially DQN), known as catastrophic forgetting. This validation step also compares the performance of all best models together (chosen by the reward metric), to compare the AWT and AQL metrics achieved by each of them.

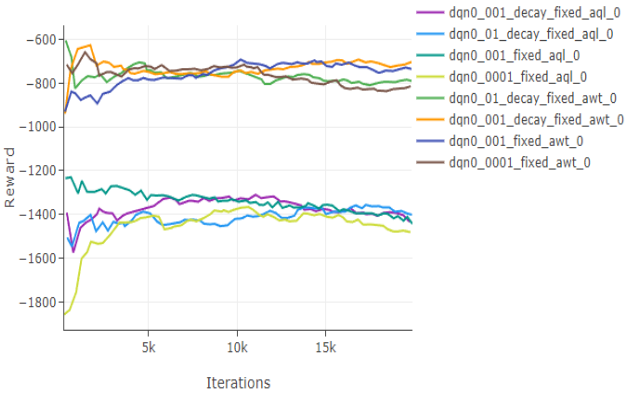

Figure 8 – Model training rewards with a discrete action space (DQN)

The first set of experiments was performed in a discrete action space where the DQN models could choose one of the available states for the traffic lights, for 20,000 iterations. As a result, Figure 8 shows that the DQN models, which were trained on the AWT metric, achieved higher rewards, but this value on its own is not meaningful, so it will be later compared during validation for a better understanding of the model's performance. Notably, the models with a learning rate of 0.001 achieved the best results, while the ones with smaller learning rates were more difficult to train. There is not a significant difference between models with a decaying learning rate and a constant learning rate. Although the ones with learning rate decay, perform slightly better on average, at least in this scenario.

The next graph demonstrates how PPO models with various learning rates performed on the same scenarios. It's important to note that on Figure 6 models with a fixed learning rate of 0.01 performed the best, regardless of whether they were using AQL or AWT as the reward metric. Once again, models trained on the AWT metric received slightly higher rewards, but this time the difference is much smaller than among the equivalent DQN models.

Finally, PPO models trained on a continuous action space show much faster improvement compared to the ones that were trained on a discrete action space. Allowing the algorithms to vary the length of the traffic light phases had a positive effect on training. This time, the learning rates did not have as significant of an impact on training, however algorithms trained on the AWT metric improve slightly less in 20,000 iterations than equivalent algorithms trained on the AQL metric.

DOI 10.15588/1607-3274-2025-4-21

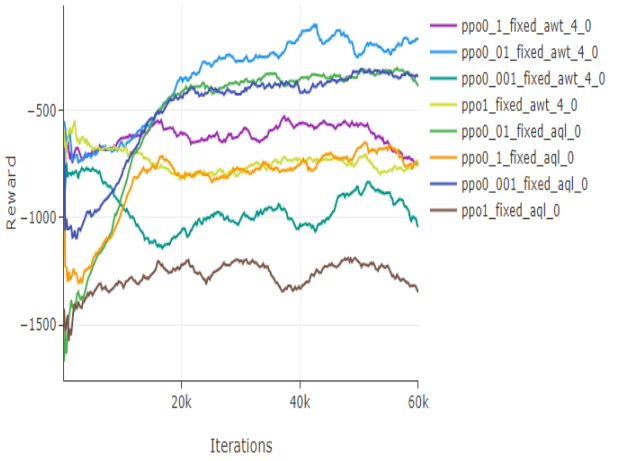

Figure 9 – Model training rewards with a discrete action space (PPO)

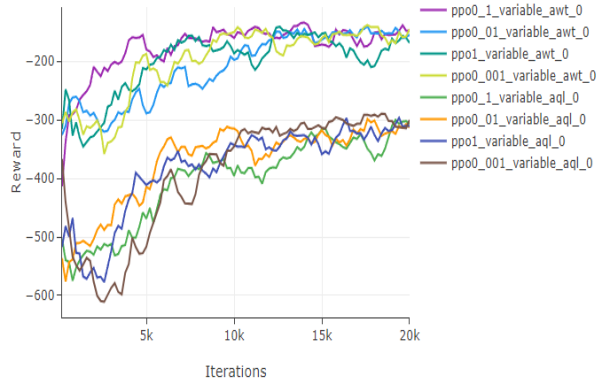

Figure 10 – Model training rewards with a continuous action space (PPO)

As illustrated in the graphs Fig. 9 and Fig. 10, PPO algorithms exhibit significantly better training performance in this scenario, with reward functions showing a more stable upward trend. This suggests that in the context of traffic light control, DQN algorithms suffer more from catastrophic forgetting, hindering their performance.

In Fig. 8 we can also observe how the chosen reward metric (AWT or AQL) impacted model performance during validation. The models were validated over 10,000 simulation steps, during which AWT and AQL were recorded. The models chosen here were the ones, which had the highest, reward value among the ones trained previously, since they were saved every few iterations, as mentioned before. The Fig. 11 shows that algorithms trained with different reward functions performed distinctly: AWT-based reward functions led to lower average waiting times, while AQL-based reward functions minimized queue lengths, as expected.

6 DISCUSSION

The study was aimed at developing an algorithm to optimize the control of traffic light phases at an intersection. The existing body of work in the subject area was analyzed, especially the papers that focused on deep reinforcement learning methods, as well as the existing frameworks for traffic simulation.

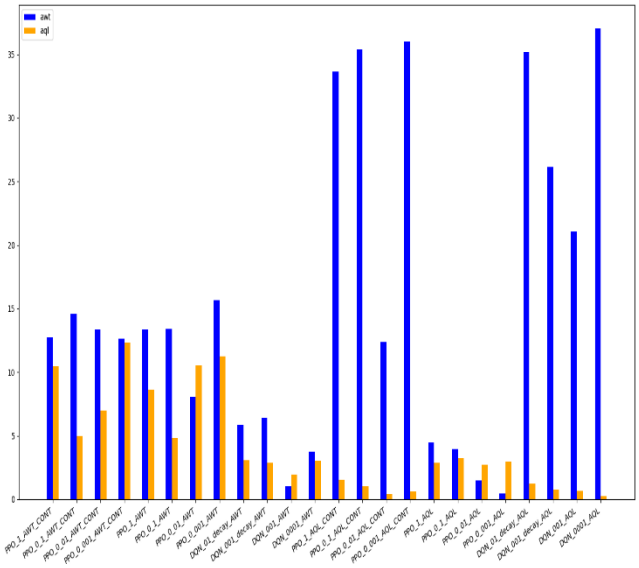

Figure 11 – Average queue length and average wait time compared for different traffic light deep learning algorithms with different reward functions

The research has underlined a few key challenges in creating an optimal traffic light control system, such as the development of a suitable training and testing environment, working with imprecise data about a vehicle's position and velocity, scaling the system to a larger real-life sized environment. To address these challenges, a novel simulation environment was created that allowed several approaches to be tested and compared.

The results indicate that the dependence of the reward function on the AWT and AQL metrics significantly impacted the results of algorithm training. Specifically, algorithms trained with the AWT metric in mind yielded lower AWT values for cars at red lights, and vice versa for those trained with the AQL metric. This highlights the importance of selecting appropriate reward metrics tailored to specific traffic management goals.

PPO algorithms trained on the traffic light control scenario demonstrated more stable reward function growth compared to DQN, indicating more consistent learning. This suggests that PPO is better suited for traffic light control tasks where training stability is critical. In this context, DQN algorithms showed more pronounced issues with catastrophic forgetting, further emphasizing the advantages of PPO for this application.

Furthermore, PPO algorithms that operated in a continuous action space, by controlling the duration of traffic light phases, performed significantly better than those using a discrete action space, where traffic light decisions occurred at fixed intervals (every 200 simulation frames). This ability to fine-tune phase durations in real-time contributes to improved traffic flow and highlights the benefits of using continuous control in traffic signal optimization.

DOI 10.15588/1607-3274-2025-4-21

Comparing these findings to prior studies, earlier research using DQN-based traffic light control showed improvements over fixed-time signal systems. However, this study builds on those results by demonstrating that PPO models using continuous action spaces can further improve performance in terms of AWT and queue length. The novel traffic simulation environment developed in this study enables detailed evaluation and comparison of such algorithms, providing a valuable tool for future research and potential real-world applications aimed at optimizing traffic signal operations.

CONCLUSIONS

During the training of both DQN and PPO models for traffic light control, a comparative analysis of AWT and AQL metrics was conducted when calculating reward functions. It was found that the reward function's dependence on AWT or AQL significantly influences training outcomes. Specifically, algorithms trained with AWT-based rewards achieved lower average waiting times, and vice versa for AQL-based rewards.

PPO algorithms consistently performed better in each traffic light control scenario, with more stable reward progression. Again, this confirms that DQN algorithms are more susceptible to catastrophic forgetting in this context. Moreover, PPO algorithms trained with a continuous action space, i.e., controlling the phase duration of traffic lights, outperformed all discrete control algorithms, which changed traffic light phases every 200 simulation frames.

These research results can be compared with previous studies on this topic. In the first section, we reviewed several works that utilized DQN or other Q-learning-based algorithms for traffic management at intersections. For instance, one study [12] examined DQN for traffic light control at a signalized intersection, developing a coordinated traffic signal control model that outperformed a static signal optimized using the Synchro method. In this thesis, similar DQN-based traffic light control methods were compared with PPO-based methods, including an analysis of discrete versus continuous action spaces in reinforcement learning agents for this scenario. It was found that the PPO algorithm developed by OpenAI [15], with appropriate hyperparameters, delivers superior results compared to DQN.

It is important to note that the findings presented here are limited in scope. Each algorithm was trained only once per set of hyperparameters over a fixed number of iterations. While the trends in reward functions and average episode lengths have been analyzed, these results are not definitive. To draw more robust conclusions, further studies involving more extensive experimentation and more complex scenarios are necessary.

Additionally, the simulated environment in this study focused solely on the intersection and did not account for traffic signs, pedestrians, or adjacent road segments. These simplifications represent key limitations, which future research can address.

The developed simulation environment offers a flexible platform for future research, allowing for the training and evaluation of various machine learning algorithms beyond those explored in this study

Scientific Novelty: The research contributes to the existing body of work on the subject of traffic control by creating a novel computationally cheap environment for testing several various traffic flow scenarios at a signalized intersection. It also gives insights into how the performance of PPO and DQN algorithms varies depending on the model's reward and the nature of the action space.

Practical Significance: The results of this research are relevant for transportation authorities, because they provide insight into the way signalized urban intersections can be optimized to allow for safer and more efficient traffic flow.

Prospects for Further Research: Future work could investigate alternative reinforcement learning approaches or entirely different methodologies to enhance model performance. Additionally, the simulated environment in this study focused solely on the intersection and did not account for traffic signs, pedestrians, or adjacent road segments. These simplifications represent key limitations, which future research can address.

Recommendations for Further Research: Expanding the scope to more realistic and challenging driving scenarios, including the presence of pedestrians, traffic signals, and complex road networks would provide valuable insights. Moreover, exploring enhanced versions of Q-learning or policy gradient methods tailored to these environments could lead to more effective and adaptable traffic control solutions.

ACKNOWLEDGEMENTS

The study was created within research topic "Methods and means of artificial intelligence to prevent the spread of tuberculosis in wartime" (№0124U000660), which is carried out at the Department of Artificial Intelligence Systems of the Institute of Computer Sciences and Information of technologies of the National University "Lviv Polytechnic".

REFERENCES

1. Tran Q.-D., Bae S.-H. An Efficiency Enhancing Methodology for Multiple Autonomous Vehicles in an Urban Network Adopting Deep Reinforcement Learning, *Appl. Sci.*, 2021, Vol. 11(4), P. 1514, Mode of access: https://doi.org/10.3390/app11041514
2. Vyklyuk Ya., Nevinskyi D., Boyko N. GeoCity – a New Dynamic-Spatial Model of Urban Ecosystem, *J. Geogr. Inst. Cvijic*, 2023, Vol. 73(2), P. 187–203, Mode of access: https://doi.org/10.2298/IJGI2302187V.
3. Team C. CARLA Simulator [Electronic resource]. Access mode: URL: https://carla.org/ (access date: 01.04.2025). Title from the screen.
4. Gutiérrez-Moreno R., Barea R., López-Guillén E., Araluce J., Bergasa L.M. Reinforcement Learning-Based Autonomous Driving at Intersections in CARLA Simulator,

DOI 10.15588/1607-3274-2025-4-21

Sensors, 2022, Vol. 22(21), P. 8373, Mode of access: https://doi.org/10.3390/s22218373.
5. Činčurak D., Grbić R., Vranješ M., Vranješ D. Autonomous Vehicle Control in CARLA Simulator Using Reinforcement Learning, *Int. Symp. ELMAR,* 2024, pp. 311–316, Mode of access: https://doi.org/10.1109/ELMAR62909.2024.10694632.
6. Wu C., Parvate K., Vinitsky E., Bayen A. M. Flow: A Modular Learning Framework for Mixed Autonomy Traffic, *IEEE Trans. Robot.*, 2022, Vol. 38.2, pp. 1270–1286.
7. Eclipse SUMO – Simulation of Urban Mobility [Electronic resource]. Access mode: URL: https://eclipse.dev/sumo/ (access date: 01.04.2025). Title from the screen.
8. Kothari P., Perone C., Bergamini L., Alahi A., Ondruska P. DriverGym: Democratising Reinforcement Learning for Autonomous Driving, *ArXiv,* 2021, Mode of access: https://doi.org/10.48550/arXiv.2111.06889.
9. OpenAI, Gym Beta [Electronic resource]. Access mode: URL: https://openai.com/research/openai-gym-beta (access date: 05.04.2025). Title from the screen.
10. Wang B., He Z., Sheng J., Chen Y. Deep Reinforcement Learning for Traffic Light Timing Optimization, *Processes* 2022, Vol. 10(11), P. 2458, Mode of access: https://doi.org/10.3390/pr10112458.
11. Park S., Han E., Park S., Jeong H., Yun I. Deep Q-network-based traffic signal control models, *PLOS ONE*, 2021, Vol. 16.9, Mode of access: https://doi.org/10.1371/journal.pone.0256405
12. Shi Y., Liu Y., Qi Y., Han Q. A Control Method with Reinforcement Learning for Urban Un-Signalized Intersection in Hybrid Traffic Environment, *Sensors*, 2022, Vol. 22, Mode of access: https://doi.org/doi:10.3390/s22030779.
13. Sutton R. S., Barto A. G. Reinforcement Learning: an Introduction, second edition: MIT Press, 2018, P. 552.
14. Schulman J., Wolski F., Dhariwal P., Radford A., Klimov O. Proximal Policy Optimization Algorithms, *ArXiv*, 2017, Mode of access: https://arxiv.org/abs/1707.06347.
15. OpenAI, Proximal Policy Optimization [Electronic resource]. Access mode: URL: https://openai.com/research/openai-baselines-ppo (access date: 11.04.2025). Title from the screen.

Received 14.07.2025.
Accepted 24.09.2025.

УДК 004.94

МЕТОДИ ОПТИМІЗАЦІЇ РОБОТИ СВІТЛОФОРІВ НА РЕГУЛЬОВАНИХ ПЕРЕХРЕСТЯХ ЗА ДОПОМОГОЮ ГЛИБИННОГО НАВЧАННЯ З ПІДКРІПЛЕННЯМ

Бойко Н. І. – канд. економ. наук, доцент, доцент кафедри Системи штучного інтелекту, Національний університет «Львівська політехніка», Львів, Україна.
Мокрик Я. Л. – аспірант кафедри Системи штучного інтелекту, Національний університет «Львівська політехніка», Львів, Україна.

АНОТАЦІЯ

Актуальність. Перехрестя є найбільш критичною ділянкою дорожньої мережі, де спостерігається найбільша кількість зіткнень та найдовший час очікування. Розробка оптимальних методів керування світлофорами на регульованих перехрестях необхідна для покращення руху транспортного потоку на існуючих міських перехрестях, зменшення ймовірності зіткнень, часу, необхідного для перетину перехрестя, та підвищення безпеки для водіїв і пішоходів. Розробка такого алгоритму вимагає моделювання руху транспорту та порівняння роботи різних підходів у змодельованому середовищі.

Мета роботи є розробка ефективної моделі глибинного навчання з підкріпленням (DRL), спрямованої на оптимізацію керування фазами світлофорів на перехрестях.

Метод. Розроблено власне симуляційне середовище, сумісне з OpenAI Gym, та проведено порівняння двох типів алгоритмів: глибинні Q-мережі та метод оптимізації близьких стратегій. Алгоритми протестовано на низці сценаріїв, включаючи сценарії з неперервним та дискретним просторами дій, де набір дій, які може виконати агент, представлений або різними станами світлофора, або тривалістю фаз сигналу світлофора. Під час навчання також налаштовувалися різні гіперпараметри та розглядалися різні метрики винагороди для моделей: середній час очікування та середня довжина черги. Розроблене середовище винагороджує агента під час навчання відповідно до однієї з обраних метрик, а також штрафує його за порушення правил дорожнього руху.

Результати. Проведено детальний аналіз результатів тестування алгоритмів DQN та PPO. Загалом, алгоритми PPO демонструють більш стабільне покращення під час навчання, тоді як алгоритми DQN більше страждають від проблеми катастрофічного забування. Зміна функції винагороди дозволяє алгоритмам мінімізувати різні метрики під час навчання. Розроблене моделююче середовище може бути використане в майбутньому для тестування інших типів алгоритмів на тій самій задачі, і воно є значно менш затратним в обчислювальному плані порівняно з існуючими рішеннями. Отримані результати підкреслюють необхідність дослідження інших методів керування світлофорами, які можуть бути інтегровані з реальними світлофорними системами для більш оптимального та безпечного руху транспортних потоків.

Висновки. Дослідження надало порівняння різних методів управління світлофорами на регульованому міському перехресті, протестувало різні способи заохочення моделей під час навчання та проаналізувало вплив, який це має на транспортний потік. Розроблене середовище було досить простим для цілей дослідження, що є цінним через великі обчислювальні вимоги самих моделей, але в майбутньому його можна вдосконалити, розширивши його більш складними функціями моделювання, такими як різні типи перехресть, які не є міськими, створення дорожньої мережі перехресть, які були б з'єднані між собою, додавання пішохідних переходів тощо. У майбутньому планується вдосконалити середовище моделювання, розширити спектр розглянутих алгоритмів, розглянути можливість використання моделей для керування транспортними засобами на додаток до керування світлофорами.

КЛЮЧОВІ СЛОВА: навчання з підкріпленням, сигналізовані перехрестя, керування транспортом, метод оптимізації близьких стратегій, глибинне Q-навчання.

DOI 10.15588/1607-3274-2025-4-21

ЛІТЕРАТУРА

1. Tran Q.-D. An Efficiency Enhancing Methodology for Multiple Autonomous Vehicles in an Urban Network Adopting Deep Reinforcement Learning,/ Q.-D. Tran, S. -H. Bae // Appl. Sci. – 2021. – Vol. 11(4). – P. 1514, Mode of access: https://doi.org/10.3390/app11041514
2. Vyklyuk Ya. GeoCity – a New Dynamic-Spatial Model of Urban Ecosystem / Ya. Vyklyuk, D. Nevinskyi, N. Boyko // J. Geogr. Inst. Cvijic. – 2023. – Vol. 73(2). – P. 187–203, Mode of access: https://doi.org/10.2298/IJGI2302187V.
3. Team C. CARLA Simulator [Electronic resource]. Access mode: URL: https://carla.org/ (access date: 01.04.2025). Title from the screen.
4. Reinforcement Learning-Based Autonomous Driving at Intersections in CARLA Simulator / R. Gutiérrez-Moreno, R. Barea, E. López-Guillén et al.] // Sensors. – 2022. – Vol. 22(21), – P. 8373, Mode of access: https://doi.org/10.3390/s22218373.
5. Autonomous Vehicle Control in CARLA Simulator Using Reinforcement Learning / [D. Činčurak, R. Grbić, M. Vranješ, D. Vranješ] // Int. Symp. ELMAR. – 2024. – P. 311–316, Mode of access: https://doi.org/10.1109/ELMAR62909.2024.10694632.
6. Flow: A Modular Learning Framework for Mixed Autonomy Traffic / C. Wu, K. Parvate, E. Vinitsky, A. M. Bayen // IEEE Trans. Robot. – 2022. – Vol. 38.2. – P. 1270–1286.
7. Eclipse SUMO – Simulation of Urban Mobility [Electronic resource]. Access mode: URL: https://eclipse.dev/sumo/ (access date: 01.04.2025). Title from the screen.
8. DriverGym: Democratising Reinforcement Learning for Autonomous Driving / [P. Kothari, C. Perone, L. Bergamini et al.] // ArXiv. – 2021, Mode of access: https://doi.org/10.48550/arXiv.2111.06889.
9. OpenAI, Gym Beta [Electronic resource]. Access mode: URL: https://openai.com/research/openai-gym-beta (access date: 05.04.2025). Title from the screen.
10. Deep Reinforcement Learning for Traffic Light Timing Optimization / [B. Wang, Z. He, J. Sheng, Y. Chen] // Processes. – 2022. – Vol. 10(11). – P. 2458, Mode of access: https://doi.org/10.3390/pr10112458.
11. Deep Q-network-based traffic signal control models / [S. Park, E. Han, S. Park et al.] // PLOS ONE. – 2021. – Vol. 16.9, Mode of access: https://doi.org/10.1371/journal.pone.0256405
12. A Control Method with Reinforcement Learning for Urban Un-Signalized Intersection in Hybrid Traffic Environment / [Y. Shi, Y. Liu, Y. Qi, Q. Han] //Sensors. – 2022. – Vol. 22, Mode of access: https://doi.org/doi:10.3390/s22030779.
13. Sutton R. S. Reinforcement Learning: an Introduction, second edition / R. S. Sutton, A. G. Barto. – MIT Press, 2018. – P. 552.
14. Proximal Policy Optimization Algorithms / [J. Schulman, F. Wolski, P. Dhariwal et al.] // ArXiv. – 2017, Mode of access: https://arxiv.org/abs/1707.06347.
15. OpenAI, Proximal Policy Optimization [Electronic resource]. Access mode: URL: https://openai.com/research/openai-baselines-ppo (access date: 11.04.2025). Title from the screen.

DOI 10.15588/1607-3274-2025-4-21