IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 10 September 2025, accepted 23 October 2025, date of publication 30 October 2025, date of current version 7 November 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3626936

RESEARCH ARTICLE

On Performance Improvement of Reinforcement Learning for Collision Avoidance in Autonomous Intersections

ÁRPÁD HUSZÁK[1,2], TAKAHITO YOSHIZAWA[3], ALIREZA AGHABAGHERLOO[3], (Student Member, IEEE), DAVE SINGELÉE[4], (Member, IEEE), AND BART PRENEEL[3], (Member, IEEE)

[1]Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics (BME), 1111 Budapest, Hungary
[2]Cloud Applications Research Group, HUN-REN-BME, 1052 Budapest, Hungary
[3]COSIC, KU Leuven, 3001 Leuven, Belgium
[4]DistriNet, Group-T, KU Leuven, 3000 Leuven, Belgium

Corresponding author: Takahito Yoshizawa (takahito.yoshizawa@esat.kuleuven.be)

This work was supported in part by the Flemish Government through the Cybersecurity Research Program under Grant VOEWICS02, in part by the Central Europe Leuven Strategic Alliance (CELSA) Project "Machine Learning-Based Cooperative Vehicle and Traffic Control Using Secure ML and Vehicle-to-Everything (V2X) Communications (ML-CCAM)," and in part by Belgian Federal Public Service Policy and Suppor: Beleid and Ondersteuning, Stratégie and Appui (BOSA) through the AIDE Project under the Program "Financing of Projects for the Development of Artificial Intelligence in Belgium" under Grant 06.40.32.33.00.10.

ABSTRACT Reinforcement Learning (RL) is a promising technology to make autonomous vehicles (AVs) a reality. The essential part of RL training is to derive an optimal policy to obtain the best performance in the intended environment. Here, the quality of the training and, consequently, the effectiveness of the subsequent policy depend on several factors, including algorithm selection, environment definition, and parameter values. In this paper, we assert that the training process itself is an equally important determining factor for effective training. We conducted multiple training experiments using two RL algorithms in a multi-agent environment and a number of various configuration settings, and verified that optimizing the amount of input data based on its relevance significantly improves the training effectiveness in AVs navigating an unsignalized intersection. Our approach is specifically effective in a complex environment, where a significantly large amount of observable data hinders the effective policy convergence and quality of the training, rather than helping it. By applying our design to identifying and eliminating data that are unimportant for the training objective, we reduced the number of observable inputs during the training process. As a result, we demonstrated a more effective training strategy in a highly complex and dynamic environment such as AVs on the road. Furthermore, we achieved a higher success rate by adopting one policy per agent in a multi-agent environment as opposed to a single policy for all agents. Our results show 99% and 55∼78% reduction in collision in vehicle trajectories with and without turning at the intersection compared to baseline training performance, respectively. In addition to the quantitative metrics, we analyzed the qualitative behaviors of the learned policies to understand how strategic actions contributed to collision reduction. Our positive test result sheds light on a new approach to the RL training: data reduction in multi-policy training. We believe this new approach is equally important to conventional approaches, such as algorithm selection and parameter tuning, toward effective RL training.

INDEX TERMS Autonomous vehicles, reinforcement learning, collision avoidance, deep-q network (DQN), proximal policy optimization (PPO), training optimization, observation filtering, multi-policy learning.

I. INTRODUCTION

Reinforcement learning (RL) is one of the three areas of machine learning (ML), along with supervised learning and

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

unsupervised learning. One of its main characteristics is the numerous repetitions of the learning process, called an *episode*, to gradually reach optimal behavior as an agent goes through its actions in a well-defined environment. This learning process is facilitated by achieving a good balance between *exploration* and *exploitation* with a gradual shift from the former to the latter during the training process. This shift facilitates the convergence to an optimal series of actions by increasingly relying on past experiences.

One promising application of RL is to train autonomous vehicles (AVs). SAE J3016 [1] defines the five stages of driving automation, with level 5 being fully automated driving. In level 5, AVs have complete control of the vehicle at all times under all circumstances to an extent that a human driver is no longer required. In this context, the RL model in the vehicle's On-Board Unit (OBU) is equipped with the RL *policy* which determines how the vehicle drives under various circumstances to be fully autonomous in its maneuver.

Due to various societal implications of AVs, such as the legal framework and liability aspects, AVs must be fully accepted by consumers and society before their mass deployment is realized [2]. As vehicles are increasingly computerized with communication functions, modern vehicles are often characterized as *computers on wheels* [3]. One important characteristic of AVs is their nature of being a cyber-physical system (CPS), i.e., consequences of unintended operations of vehicles' computer systems can result in physical damages to properties, injury, or even death. From the RL perspective, this implies that the training phase is necessarily done in the simulation setting, rather than on open roads in the real world. This training phase involves many factors and parameter choices, such as selecting a suitable learning algorithm, determining the reward value, discount factor, learning rate, etc. All of them influence the effectiveness of the outcome of the RL training: the final *policy*. As the number of these parameters increases, the possible value combinations also grow exponentially. Obtaining the best parameter combination requires many training runs and comparing and evaluating them. In this sense, obtaining the best *policy* is a time-consuming process involving many trial-and-error runs.

Another equally important factor for effective training is the training process itself. This is especially applicable in a highly complex and dynamic operating environment, such as AVs navigating roads. The complexity increases as the number of factors increases, such as the road topology, the number of vehicles on the road, and the presence of traffic signals and other road users such as pedestrians.

We used the multi-agent RL concept, which is a decentralized approach by default, as each vehicle represents an agent with its intention and goals to navigate its path toward its destination while coordinating its maneuver with the presence of other vehicles and observing traffic rules. Thus, each agent has its own policy that determines the action based on its own observations. In this situation, the number of observable data can significantly increase, resulting in a potential delay in decision-making due to the increasing level of processing and storage burden in vehicle OBUs. Furthermore, a single-policy approach may hamper the achievement of the optimal policy for each agent (vehicle). This is a concern as AV is a real-time system (RTS) that requires strict time-bound processing and decision-making to achieve road safety. In this sense, we consider that RL training can be improved to achieve higher efficiency and stable performance. The conventional approach in RL to improve performance is to rely on high-performance and high-capacity computing platforms, such as GPUs, to run the RL training. Contrary to this, our proposed approach makes the RL training more efficient and less time-consuming, with good convergence results with or without the use of such high-performance hardware. In other words, our approach is effective independent of the hardware environment the training is conducted.

Our contributions in this paper are the following:

- We address the scalability aspects of multi-agent RL training in a complex environment and propose multiple concrete approaches to optimize the training process, namely filtering out input data that do not contribute to making RL training effective and efficient.
- We apply these approaches in our simulation runs and demonstrate their effectiveness and quantify the level of improvement in training performance, such as the collision rate, compared to the baseline values.
- We used two different training algorithms, Proximal Policy Optimization (PPO) and Deep-Q Network (DQN), in our simulation and evaluated their training performance.
- We apply a multi-policy approach in a multi-agent environment to evaluate its effectiveness compared to a single-policy rule.
- In the spirit of open science, we make our code available at *https://github.com/al6ifj/ML-CCAM*

The rest of this paper is organized as follows: we first present related works in Sec. II, followed by a brief overview of RL in Sec. III. Starting from Sec. IV, we discuss in detail our simulation of our improved training approach. In Sec. V, we present our observations and results, followed by our conclusion in Sec. VI.

II. RELATED WORKS

There exist many papers that apply RL in AVs. The general theme is to investigate the use of RL to control traffic around intersections. It is generally called Autonomous Intersection Management (AIM). Almost all of the related papers discussed in this section address traffic control around intersections using RL from various perspectives.

Table 1 lists related papers in recent years. This table captures the key points of each related work, highlighting both its pros and cons. The number of papers on the performance comparison of different RL algorithms is more than that of multi-agent coordination. Furthermore, papers that focus on the combination of (1) multi-agent, (2) training

performance improvement, and (3) applied in AIM are even less. Out of fourteen (14) recent papers we listed, only five (5) of them fall into this category as shown in the third column in Table 1. In this sense, our paper offers additional insights into the area that this combination represents.

A. ALGORITHM PERFORMANCE COMPARISON

A survey of RL-based AIM decision-making schemes at an unsignalized intersection is presented in [4]. It summarized papers that discuss their simulation work based on intersection types, simulation and data collection methods, along with their view on each of these works, and compared performance in terms of intersection type, maneuver, success and collision rates.

Multiple papers compares performance of various algorithms and evaluate relative differences in simulation results: (1) between PPO and Advantage Actor-Critic (A2C) [5], (2) between PPO and Deep Deterministic Policy Gradient (DDPG) [9], [10], and (3) among DQN, DDQN, and Dueling Double DQN (D3DQN) [11]. The simulation softwares used in these simulations are: Scalable Multi-Agent Reinforcement Learning Training School for Autonomous Driving (SMARTS) [5], [10], TORC [9], and CARLA [11]. SMARTS is a relatively new simulator for vehicular communication developed by Huawei [6].[1]

The following observations are made as a result of these simulations. In [5], their evaluation indicated that PPO outperformed A2C across the board. In [9], their result showed that DDPG had better performance in higher reward and faster convergence than PPO, concluding that the former is more suitable for AVs for this reason. The results of [10] were rather inconclusive as their simulation model fell short of the state-of-the-art approaches due to several reasons, such as training failing to converge. In [11], among the three algorithms they tested, D3DQN's performance was the best with the fewest negative rewards. If we combine these three results, among PPO, A2C, and DDPG, the last appears to have the best performance result. Similarly, among DQN, DDQN, and D3DQN, the last one has the best performance.[2]

B. MULTI-AGENT COORDINATION

On the multi-agent coordination front, there are two notable works. First, reinforced AIM (RAIM) utilizes Multi-Agent Deep Reinforcement Learning (MADRL) to safely maneuver vehicles around the intersection [12]. In this scheme, up to 32 vehicles (including the ego vehicle) can be included to model how the ego vehicle determines its speed. MADRL uses DNN, which consists of four fully-connected layers to progressively reduce the output neurons, with the final neuron representing the prediction of the ego vehicle's speed. The algorithm to optimize this is called Twin Delayed Deep Deterministic Policy Gradient (TD3), which is a successor of DDPG. Their simulation results indicate that RAIM outperformed in all metrics against the benchmark.[3]

Second, the same authors proposed advanced RAIM (*adv.*RAIM) [13] which is an improved version of RAIM [12]. It is an *"ego-centric multi-agent system"* which uses Long Short Term Memory (LSTM) in the state/conflict encoder module. It periodically controls the vehicles' speed through an intersection in a cooperative and decentralized manner. Their simulation result showed reduced travel time, time loss, fuel consumption, and emission over RAIM.

C. RL ARCHITECTURE VARIANTS

There are several approaches to hierarchical RL models as architectural variations of the IAM. A two-layered hierarchical RL architecture is presented for learning-based decision-making for left-turn at unsignalized intersections [14]. It combines Soft Actor-Critic (SAC) principles and Model Predictive Control (MPC) to execute the left-turn maneuver. Their use of the adaptive entropy regularization technique results in faster convergence, higher mean rewards, and lower collision rates. Their simulation result showed that their framework has improved training and navigation.

Another hierarchical control framework applies DRL in a highway scenario in which an AV overtakes other vehicles efficiently and safely [15]. This framework uses Dueling DQN (DDQN) algorithm to control the AV to achieve this objective. The paper presents its effectiveness based on convergence rate and control performance.

Yet another hierarchical model, called Hierarchical Option MDP (HOMDP), addresses the scalability issue associated with Partially Observable MDP (POMDP) [16]. It is useful in scenarios that involve environmental uncertainties in the decision-making process. The HOMDP scheme solves the POMDP problem by efficiently storing the most important information to determine the vehicles' maneuver. Their simulation result showed that the HOMDP scheme improves the vehicles' performance in a 4-way intersection.

Finally, a hybrid architecture to determine vehicles' movement in the intersection is presented in [17]. This DRL-based solution consists of three components: (1) operative level (driving command generation), (2) strategy level (trajectory definition), and (3) tactical level (high-level decision execution). Their simulation was done using three intersection scenarios: with traffic lights, with traffic signs, and uncontrolled. Their simulation results showed that the PPO algorithm could infer the ego vehicle's behavior for different intersection scenarios based only on the behavior of adversarial vehicles.

[1] The authors of [6] claim that their implementation is the most advanced and suitable simulation tool for multi-agent RL (MARL) research compared to other existing road traffic simulations such as CARLA [7] and SUMO [8].

[2] We note that other factors, such as the simulator used, specific objectives of the simulations, various simulation settings, and other configurations, should also be taken into consideration as they would likely affect the outcomes of the simulations.

[3] The benchmark includes travel time, waiting time, time loss, speed, and electric power consumption.

TABLE 1. Related work.

Ref.	Year	MA TO in IAM (*1)	RL Algo.	Contribution	Pros/Cons
			Algorithm Performance Comparison		
[4]	2024	Yes	DQN, DDQN, DDPG, MEDDPG, TD3, T-TD3, DQ-GAT, MTD-GPT, and others	Presents comprehensive comparison of RL-based decision-making schemes at unsignalized intersection.	**Pros:** Extracts essential observations from analysis of referenced papers. **Cons:** A survey of others' papers with limited performance discussion.
[5]	2023	No	PPO, A2C	Clearly demonstrated PPO outperforms A2C according to their evaluation metric.	**Pros:** Clear comparison of two representative algorithms. **Cons:** Evaluation metrics and state/action space definitions appear limited in scope.
[9]	2023	No	PPO, DDPG	Comparison of PPO and DDPG performance on lane keeping and collision avoidance.	**Pros:** Clear demonstration of DDPG showing better performance than PPO in the AV context. **Cons:** —
[10]	2023	Yes	PPO, DDPG	Multi-agent RL training using PPO and DDPG.	**Pros:** Combination of multi-agent and algorithm performance compaison. **Cons:** The simulation did not reach an optimal result.
[11]	2023	No	DQN, DDQN, D3DQN	Performance comparison of three DQN-based prediction algorithms to detect road uncertainties applied in the AIM context.	**Pros:** Clear conclusion that D3DQN demonstrated the best performance. **Cons:** —
			Multi-Agent Coordination		
[12]	2020	Yes	TD3	Use of a 4-layer DNN to determine vehicle's speed in multi-agent AIM environment.	**Pros:** Propose TD3, which is a successor of DDPG. The result shows improvement in all metrics. **Cons:** —
[13]	2022	Yes	LSTM	Further improvement of [12] with the use of LSTM.	**Pros:** The improved version reduces travel time, time loss, and fuel consumption over [12]. **Cons:** —
			RL Architectural Variants		
[14]	2023	Yes	SAC	Introduces two-layered left-turn navigation decision mechanism using SAC and MPC principles.	**Pros:** Demonstration of improved training and navigation. **Cons:** —
[15]	2020	No	DQN, DDQN	Scenario of a vehicle overtaking in highway using DDQN to control AVs.	**Pros:** Demonstrates the effectiveness of DDQN. **Cons:** —
[16]	2018	No	HOMDP	Proposes HOMDP to addresses the scalability issue of POMDP to predict uncertainties.	**Pros:** Demonstrates the improvement in prediction performace in the AIM environment. **Cons:** —
[17]	2022	No	PPO	Introduces 3-levels of vehicle maneuvering in AIM - operative, strategy, and tactical level.	**Pros:** Demonstrates the ego vehicle's ability to control its maneuver based on the behavior of adversarial vehicles. **Cons:** —
			Centralized Traffic Control		
[18]	2022	No	Q-learning	Proposes centralized traffic coordination at an intersection based on a polling mechanism.	**Pros:** Centralized control approach ensures predictable service. **Cons:** Opposite of multi-agent autonomous AV.
[19]	2022	No	Q-learning	Introduces DRL-based centralized intersection management based on request-permission mechanism.	**Pros:** Centralized control approach ensures predictable service. **Cons:** Communication-based solution to simply replace traffic lights.
[20]	2021	No	DQN	Introduces smart traffic control using DRL and SR.	**Pros:** Demonstrates improved performance and efficiency. **Cons:** Opposite of multi-agent autonomous AV.

Note (1): "Yes" if it satisfies 3 factors: (1) multi-agent (MA), (2) training optimization (TO) in (3) intersection maneuvering; "No" otherwise.

D. CENTRALIZED TRAFFIC CONTROL

Several papers focus on the use of RL in the centralized traffic control entity at the intersection to optimize traffic flow.[4]

A polling-based solution coordinates traffic flow at intersections [18]. In this scheme, vehicles in each lane form a queue, and this polling station at the intersection serves vehicles in these queues in such a way that no collision occurs as vehicles cross the intersection. Another centralized control mechanism, called Deep Reinforcement Learning-based Autonomous Intersection Management (DRLAIM), coordinates vehicle movement at the intersection using Deep Reinforcement Learning (DRL) technique [19]. In this scheme, vehicles approaching the intersection send requests to the controller, which uses DRL to determine the vehicles' priorities.

A two-stage traffic flow management model in an intelligent traffic signal control [20] employs a combination of DRL and Smart Re-routing (SR) to control traffic flow to avoid congestion around the intersection. They used a five-layer DNN model to optimize the model's performance. Their simulation results indicated improved performance and efficiency over the no-re-routing situation.

III. REINFORCEMENT LEARNING PRIMER

In this section, we give an overview of RL and its two most notable algorithms, Deep Q-Network (DQN) and Proximal Policy Optimization (PPO). There are numerous references and sources on this subject; some of them focus more on theoretical background such as [21], and others which cover more practical programming aspects, such as [22]. A survey on various DRL algorithms and their descriptions is given in [23]. We keep this section at a minimal level by focusing on algorithms relevant to our work. Interested readers are advised to refer to these sources for further details.

A. REINFORCEMENT LEARNING

Reinforcement Learning (RL) emulates the way human acquires new knowledge through multiple iterations of trial-and-error experiments without any prior knowledge. RL operates in a well-defined *environment* in which an *agent* observes the current *state* ($s \in S$, where S is a set of all states) and takes an *action* ($a \in A$, where A is a set of all actions) based on a transition probability Eq. 1:

$$P_a(s, s') = Pr(S_{t+1} = s' \mid S_t = s, A_t = a); \tag{1}$$

where S_t and A_t represent state and action at time t and S_{t+1} is the next state at time $t + 1$ as a result of taking the action A_t as illustrated in Figure 1. Depending on the *action* taken, the *agent* receives a positive or negative reward R_t at time t. The *environment* is expressed in the Markov Decision Process (MDP), which represents the concept that the decision process is solely dependent on the current state [21]. An *episode* is an iteration of an experiment that terminates when either the objective is reached or times out without achieving the objective. Throughout multiple iterations of *episodes*, this probability at all states s converges and achieves the optimal set of actions to satisfy the predefined objective.

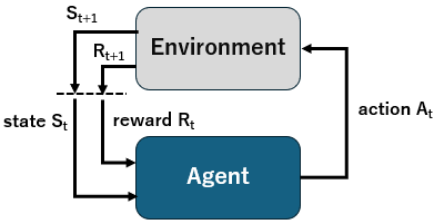


FIGURE 1. Reinforcement learning concept.

The objective of the RL is to determine the optimal *action* in each *state* in order to maximize the total accumulated reward over time for achieving the stated objective. By repeating *episodes* multiple times, the *agent* converges to an optimal action a in each state s. This converged set of state/action pairs is called a *policy* (denoted as π). Stated in another way, a policy π represents an optimal mapping from state s to action a that maximizes the overall return G which is expressed as shown in Eq. 2:

$$G = \sum_{k=0}^{\infty} \gamma^k R_{k+1} \tag{2}$$

where γ is a discount factor ($0 < \gamma \leq 1$).

For a successful convergence to a *policy* over a series of *episodes*, the concept of *exploration and exploitation* is employed. This is a balancing act between relying on testing out random actions and utilizing the positively-rewarded actions from experience. Gradually shifting from the former to the latter is the key to converging toward a policy as the *agent* repeats the *episode*. It has been shown that Q-learning converges under all circumstances [25].

B. DRL ALGORITHMS OVERVIEW

There are numerous algorithms for DRL. They are classified into three categories: value-based, policy-based, and actor-critic algorithms [23]. They all have distinct characteristics with relative advantages and disadvantages, making each of them appropriate and suitable for different circumstances and environments [26], [27]. Table 2 summarizes these key characteristics of these algorithm categories.

The performance of the resulting trained model is highly sensitive to the specific environment and scenario. This paper aims to demonstrate that the proposed extensions and features can significantly improve the efficiency of the trained model. For this reason, we selected DQN and PPO as two representative algorithms from value-based and policy-based categories, respectively, for our experiments.

[4] As these solutions are centralized control mechanisms, they are not multi-agent autonomous navigation decisions by AVs. Semantically, we consider this approach to be equivalent to conventional traffic lights, which is also central traffic flow control, except for the use of RL and DNN. However, we include this category here for a comprehensive coverage of this subject.

These algorithms are well-researched, mature, and widely used. Furthermore, multiple RL libraries (e.g., RLlib, Stable-Baselines3) provide optimized implementations of these algorithms, making them solid baselines for our purpose. In our experiments, we use Rainbow DQN which combines several improvements of this algorithm [28], [29], [30]. In the remainder of this section, we give a brief overview of these two algorithms.

C. DEEP Q NETWORK (DQN)

Q-learning was first introduced by Watkins in his PhD thesis in 1989 [31]. The Q-function is a state-action value function. Its goal is to determine the *action a* that maximizes the expected total reward over all successive steps from the current state *s*. This is shown in Eq. 3.

$$Q(S_t, A_t) \leftarrow (1-\alpha)Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a)) \tag{3}$$

where α and γ are the learning rate ($0 < \alpha \leq 1$) and discount factor, respectively, and R_{t+1} is the reward for moving from state S_t to S_{t+1}. The set of best *actions* at all *states* ($\{(a*, s)\}\forall s$) yields the best *policy* at the end of the training process. "Q" for Q-learning signifies the *quality* of an action taken at a state.

The downside of this approach is that it is unscalable. Because this calculation requires all action/state values (Q values) in all states to be maintained in a table called *Q table*, in practical applications where the action and state space become large, it becomes untenable to maintain the Q-table. An alternative approach is to estimate the Q value by using Deep Neural Network (DNN) instead of relying on a discrete table to maintain action/state values. DQN is an off-policy algorithm. Additional benefit of utilizing DNN is that DQN can process non-discrete and high-dimensional data such as images or sensor data [44].

D. PROXIMAL POLICY OPTIMIZATION (PPO)

PPO is an on-policy algorithm; it was introduced by OpenAI engineers in 2017 [38]. It is a type of policy gradient method that alternates *"between sampling data through interaction with the environment, and optimizing a 'surrogate' objective function using stochastic gradient ascent"* [38]. PPO is an improvement over Trust Region Policy Optimization (TRPO) in that the former possesses better characteristics and improved performance than the latter, such as simpler implementation, more general applicability, and better sample complexity.

Given the current state, PPO outputs action probabilities or specific actions and a value estimate for those actions by using a clipping mechanism, which prevents large policy changes. The key idea of PPO is expressed by Eq. 4:

$$L^{clip}(\theta) = \hat{\mathbb{E}}_t \left[min(r_t(\theta)\hat{\mathbb{A}}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{\mathbb{A}}_t) \right] \tag{4}$$

where $\hat{\mathbb{E}}_t[\ldots]$ and $\hat{\mathbb{A}}_t$ are an expectation (empirical average) and an estimation of the *advantage function* at time t, respectively. Ratio function $r_t(\theta) = \frac{P_a(s,s'|\pi=\pi_{\theta_t})}{P_a(s,s'|\pi=\pi_{\theta_{t-1}})}$ represents the change of probability of selecting action a in state s under the current policy π_t over the old policy π_{t-1}. Thus, $r_t(\theta) > 1$ represents a more likely selection in the current policy, while $0 \leq r_t(\theta) < 1$ represents a less likely selection in the current policy, respectively [38], [45]. The *advantage* $\hat{\mathbb{A}}$ indicates the effect of the selected action. A positive ($\hat{\mathbb{A}}_t > 0$) or negative ($\hat{\mathbb{A}}_t < 0$) value indicates that the action has a positive or a negative effect on the outcome, respectively.

This equation expresses the expectation of a minimum of two terms. The first term ($r_t(\theta)\hat{\mathbb{A}}_t$) represents the standard policy gradient objective, and the second term ($clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{\mathbb{A}}_t$)) is the modified probability ratio by clipping the r_t to keep the policy update within the boundary of $[1-\epsilon, 1+\epsilon]$ (ϵ is a hyperparameter). This clipping prevents taking a step too far away from the current policy. As a result, PPO enforces a conservative policy change over time by limiting the amount of changes from one policy to the next, leading to a stable optimal policy.

E. COMPARISON OF DQN AND PPO

Kozlica et al. [45] conducted a study to compare and evaluate the performance of DQN and PPO. The setting was a machine that sorts and assembles products using both of these algorithms. The authors tested these algorithms under two different reward functions and evaluated their performance using multiple criteria – average percentage of correctly assembled and sorted products, average episode length, and percentage of successful episodes. The result indicates that PPO outperforms DQN in all of these criteria in both reward functions. This experiment demonstrated that the optimal reward function significantly influences the performance and effectiveness of the algorithm being used. In this sense, this experiment showed the importance of selecting the right algorithm and defining the associated parameters for the intended task.

IV. OBSERVATION FILTERING METHODS FOR IMPROVED TRAINING

In an unsignalized autonomous intersection, the ego vehicle collects information about the environment, especially surrounding vehicles, to avoid collisions and safely cross the intersection. Based on these observations, the ego vehicle (agent) performs an action from the action space. The goal is to maximize the cumulative reward. The complexity of the observation space directly impacts the efficiency of RL training in several ways, including computational cost, sample efficiency, convergence time, and stability.

The motivation of the proposed approaches is that high-dimensional observation spaces require the agent to process a large amount of information. This increases the computational burden and memory requirements, slowing down learning. The agent may need deeper or more

TABLE 2. DRL algorithm classifications [24].

Classification	Value-based	Policy-based	Actor-Critic
Algorithms	• Q-Learning ([31] 1989) • Action-Reward-State-Action (SARSA) ([32] 1994) • Deep Q-Network (DQN) ([33] 2013) • Double DQN ([34] 2016) • Dueling DQN ([35] 2015)	• Vanilla Policy Gradient (VPG) ([36] 1992) • Trust Region Policy Optimization (TRPO) ([37] 2015) • Proximal Policy Optimization (PPO) ([38] 2017)	• Advantage Actor-Critic (A2C) ([39] 1983) (note 1) • Asynchronous Advantage Actor–Critic (A3C) ([40] 2016) • Deep Deterministic Policy Gradient (DDPG) ([41] 2015) • Twin Delayed DDPG (TD3) ([42] 2018) • Soft Actor-Critic (SAC) ([43] 2018)
Characteristics	• Value function estimates the expected return from a state-action pair. • The agent follows the policy selection with highest estimated value.	• Policy is parameterized and maps from state to action. • Policy parameters are adjusted to maximize the expected cumulative reward.	• Combines value-based and policy-based approaches (actor represents policy and critic evaluates the action).
Advantages	• Simple to implement. • Performs well in finite set of actions.	• Supports continuous and high-dimension action space. • Learns stochastic policies, suitable for exploration and uncertainty. • Policy is explicitly represented.	• More stable and efficient learning due to combining both value-based and policy-based approaches. • Achieves higher performance in complex environments.
Disadvantages	• "Curse of dimensionality". • Policy is not explicitly represented.	• Requires more data to converge than value-added method. • Training can be unstable due to high variance in policy gradient estimates. • More complex to implement and tune.	• Complexity and computational cost due to maintaining two networks. • Agility of policy updates is dependent on the critic's value estimate performance. • Susceptibility to overestimation or underestimation and inefficiencies in learning if critics are poorly trained.
Suitable for	• Discrete and manageable action space. • Under limited computational resources. • Simple solution is desirable.	• Action space is continuous or high-dimensional. • Requires stochastic policies. • Modeling of complex behaviors.	• When more stable and efficient learning is required. • When higher performance is needed in complex environments.

Note 1: Our search indicates that no single definitive introduction paper for this algorithm exists, but its key components evolved from prior work in [39].

complex neural networks to handle such data, which further increases training time and resource usage. Additionally, high-dimensional spaces often suffer from the *curse of dimensionality*, where the data becomes sparse, making it harder for the agent to generalize and learn meaningful patterns.

Moreover, there can be a significant number of steps and states in the examined autonomous intersection environment when not all possible actions are equally good. While such scenarios might seem harmless at first glance, they introduce several challenges that can slow down learning, cause slow convergence, inefficient exploration, high variance in updates, and policy instability. The lack of meaningful feedback for exploration makes exploration inefficient, as the agent cannot determine which actions might lead to better long-term results. As a consequence, the agent may take longer to discover optimal policies, and its exploration efforts may be wasted on actions that do not contribute to learning. This inefficiency is particularly problematic in environments where exploration is already challenging due to high dimensionality or complex state spaces.

In policy gradient methods, such as PPO or TRPO, the lack of meaningful feedback can cause the policy to stagnate. If all actions yield similar rewards, the gradient updates become less effective because the agent cannot distinguish between better and worse actions. This stagnation can lead to increased variance in gradient estimates, making training less stable and efficient. Similarly, in value-based methods like DQN, if all actions have similar Q-values, the agent may struggle to learn a meaningful policy. The Q-values may converge slowly or fail to reflect the true value of actions, delaying the discovery of optimal policies.

In multi-agent environments, the challenges are amplified. If all actions are equally good for an individual agent, the agent may fail to coordinate with others or adapt to their strategies. This inefficiency can lead to longer training times and increased complexity, as the agent must navigate not only its actions but also the actions of other agents in the environment. Thus, our primary goal was to investigate new approaches that filter unnecessary actions to improve the efficiency of the training.

We also analyzed the performance of single- and multi-policy approaches. Using a single policy approach (all agents share the same policy) or a multi-policy approach (different groups of agents have distinct policies) also has significant implications for learning dynamics and scalability. A single policy approach is simpler and usually more efficient, as there is only one set of parameters to optimize. This reduces computational complexity and memory requirements, making it easier to scale to environments with a large number of agents. On the other hand, a multi-policy approach is particularly beneficial in environments requiring heterogeneity and specialization.

A. ENVIRONMENT

The environment represents a 4-way, multi-lane unsignalized autonomous intersection ecosystem with which the agents interact, serving as the source of all perceptual inputs and the recipient of all agent actions. In practice, the environment is defined by the observation space ($\mathcal{S}$), action space ($\mathcal{A}$), and reward function ($\mathcal{R}$). In the examined multi-agent environment, each vehicle is an individual agent, also referenced as *ego vehicle*. The concept of multi-agent learning allows the agents to find a near-optimal series of actions simultaneously to maximize their reward function in a changing environment. These actions change the state of the environment, so the action of one agent impacts the future steps of the other agents.

1) OBSERVATION SPACE

Similar to other related works, we also included the most relevant features of the ego vehicle and surrounding vehicles in the observation space, such as position (x, y), speed (v), and destination edge (Ω). We introduced four different observation spaces to compare their effectiveness:

- Absolute coordinates (Eq. 5):

$$\mathcal{S}_{\text{abs_coord}} = \{x_e, y_e, v_e\} \cup \{(x_i, y_i, v_i)\}_{i=1}^{N} \tag{5}$$

  where N is the number of surrounding vehicles, while the e index stands for the ego vehicle.
- Relative coordinates:
  Each ego vehicle has its own coordinate system, where $(x, y) = (0, 0)$ is the position of the ego vehicle.
- Relative coordinates with destination edges (Eq. 6):
  Destination edges are also included, which shows the way the vehicle will follow. Thus, the ego vehicle has information on whether a surrounding vehicle will go left, right, or straight.

$$\mathcal{S}_{\text{rel_coord_dest}} = \{x_e, y_e, v_e\} \cup \{(x_i, y_i, v_i, \Omega_i)\}_{i=1}^{N} \tag{6}$$

- Polar coordinates with destination edges (Eq. 7):
  Relative coordinates of other vehicles are transformed into distance and angle.

$$\mathcal{S}_{\text{pol_coord_dest}} = \{(d_i, \alpha_i, v_i, \Omega_i)\}_{i=1}^{N} \tag{7}$$

  where d and α represent the distance and angle, respectively.

2) ACTION SPACE

We assumed the vehicles have predefined path trajectories, so the agent is responsible only for adjusting the speed of the ego vehicle to avoid collision and successfully pass through the intersection. The agent can perform two actions: accelerate and decelerate for the environment step duration as formalized in Eq. 8.

$$\mathcal{A} = \{a, -a\} \tag{8}$$

3) REWARD FUNCTION

The reward function is composed of a static- and distance-related dynamic component, $\mathcal{R} = \mathcal{R}_{\text{stat}} + \mathcal{R}_{\text{dyn}}$. The static part is given by Eq. 9:

$$\mathcal{R}_{\text{stat}} = \begin{cases} 10, & \text{if the ego vehicle crossed the intersection} \\ -10, & \text{in case of collision} \\ -0.1, & \text{weak negative reward in each timestep} \\ & \text{in order to avoid vehicle to stop} \end{cases} \tag{9}$$

The dynamic part is responsible for encouraging the ego vehicle to keep its distance from the closest vehicle. The penalty grows linearly as the vehicles get closer, and the distance (d) between the two vehicles is less than the penalty distance (d_p) as shown in Eq. 10.

$$\mathcal{R}_{\text{dyn}} = \begin{cases} -\dfrac{(d_p - d)}{d_p}, & \text{if } d < d_p \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

The concept of $\mathcal{R}_{\text{dyn}}$ was motivated by [18], but instead of the quantized values, we used continuous interpolation to give more precise rewards as feedback.

B. METHODOLOGY

Our proposed methods aim to filter out actions that will not provide meaningful feedback, slowing down the training convergence and causing policy instability. Actions that have a negligible effect on the obtained reward should be filtered out because, in such cases, the agent cannot determine which actions might lead to better long-term results. Filtering out vehicles that pose no collision risk is a crucial optimization that mirrors how human drivers intuitively process traffic. When crossing a busy intersection, experienced drivers naturally focus their attention on vehicles that could potentially intersect with their path and ignore those that are too far away, moving in non-conflicting directions, or have already cleared the danger zone. In practice, in a multi-agent environment, an action is determined for an agent only if it receives observation in the previous step. Without observation, no action is defined. Therefore, action filtering is closely related to observation filtering.

The proposed filtering methods can also be considered as pre-processing tasks to strengthen the weight of important parameters, ignore non-significant ones, and perform actions only if needed. Therefore, the rule-based filtering techniques can not be used without RL, they are only part of the workflow.

1) OBSERVATION RANGE FILTERING

Vehicles that are too far from the ego vehicle have a very low impact on the collision risk ratio and the received reward. We defined a filter range distance (d_f) to:

1) reset the observation values to zero for all vehicles having distance from the ego vehicle $d > d_f$,



FIGURE 2. Observation filtering based on range distance.

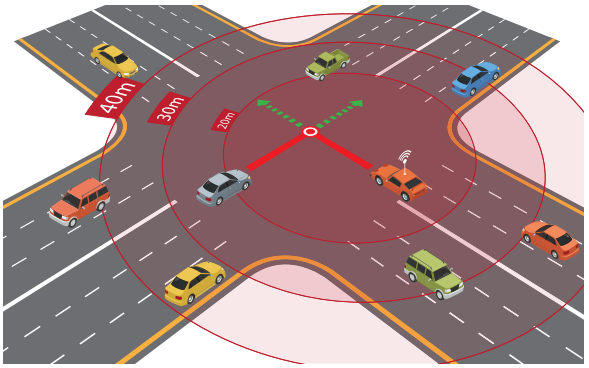


FIGURE 3. Action Movement Vector Filtering (AMVF) aims to ignore vehicles with no common conflict point with the ego vehicle based on movement vector.

2) if there are no vehicles within the d_f range, we do not create an observation for the agent so that it will not receive action for the next environment step.

Filtering of the observation range at the 4-way / 3-lane intersection is illustrated in Figure 2.

2) OBSERVATION SPACE ORDERING AND SIZE REDUCTION

The observation space is composed of information received from surrounding vehicles. As a complex observation space can lead to slower training convergence, we aimed to keep it as simple as possible. Instead of including all surrounding vehicles in the observations, we limited their number to N. It is important to note that we first sort the vehicles by distance in descending order from the ego vehicle, then we keep only the first N closest vehicles in the reduced observation space ($\mathcal{S}_N$), while further vehicles are discarded, as formalized in Eq. 11 and Eq. 12.

$$\begin{aligned} \mathcal{S}_{\text{sorted}} &= \left[P_{(1)}, P_{(2)}, \ldots, P_{(K)}\right] \\ &\text{where} \quad d(P_{(1)}) \leq d(P_{(2)}) \leq \cdots \leq d(P_{(K)}) \end{aligned} \tag{11}$$

Extract the first N elements of $\mathcal{S}_{\text{sorted}}$.

$$\mathcal{S}_N = \{P_{(1)}, P_{(2)}, \ldots, P_{(N)}\} \subseteq R \tag{12}$$

3) ACTION MOVEMENT VECTOR FILTERING (AMVF)

As the ego vehicle observes the surrounding vehicles' position and their movement angles, we can determine whether their paths intersect with the ego vehicle's trajectory. In other words, we can decide whether there is a risk of collision. If yes, the information about the vehicle posing a risk should be included in the observation; otherwise, it can be neglected. The concept of AMVF is illustrated in Figure 3.

To solve the conflict point existence problem, we made the following formalization (Eq. 13):

$$\begin{bmatrix} \cos\theta_e & -\cos\theta_v \\ \sin\theta_e & -\sin\theta_v \end{bmatrix} \begin{bmatrix} t_e \\ t_v \end{bmatrix} = \begin{bmatrix} x_v - x_e \\ y_v - y_e \end{bmatrix} \tag{13}$$

where θ_e and θ_v are the movement angles of the ego vehicle and the observed vehicle, respectively. Similarly, x, y are the coordinates, and t is the time needed to reach the intersection (conflict) point. For the paths to intersect, there must exist $t_e, t_v \geq 0$. Eq. 14 has a solution if the determinant is non-zero.

$$D = \det \begin{bmatrix} \cos\theta_e & -\cos\theta_v \\ \sin\theta_e & -\sin\theta_v \end{bmatrix} = \sin(\theta_v - \theta_e) \neq 0 \tag{14}$$

Compute t_e, t_v using *Cramer's rule* used as an explicit formula for the solution of a system of a linear equation shown in Eq. 15.

$$\begin{aligned} t_e &= \frac{(x_v - x_e)(-\sin\theta_v) - (y_v - y_e)(-\cos\theta_v)}{D} \\ t_v &= \frac{\cos\theta_e(y_v - y_e) - \sin\theta_1(x_v - x_e)}{D} \end{aligned} \tag{15}$$

If both $t_e \geq 0$ and $t_v \geq 0$, the paths intersect in the future. If one is negative, the paths crossed in the past. Obviously, if $\theta_e = \theta_v$ there is no unique solution, as the paths are parallel.

If the ego vehicle finds there are no conflict points with the surrounding vehicle, the agent does not need action in the next step because there is no risk of collision. The AMVF algorithm is universal for turning and non-turning scenarios. In a non-turning case, the θ_e and θ_v movement angles are constant, while in a turning case, they change during the turn. While the aim of the AMVF is to give a true/false trigger about future collision risk, the algorithm works in the same way for both turning and non-turning scenarios.

4) ACTION CONFLICT FILTERING (ACF)

At a conventional intersection, conflict points represent the critical locations where vehicle paths cross, merge, or diverge, creating potential collision risks. In a 4-way intersection, there are 32 total conflict points: 16 crossing conflicts (where vehicles move perpendicular to each other, posing the highest risk for severe angle collisions), 8 merging conflicts (where vehicles combine into a single lane, such as when two lanes merge into one), and 8 diverging conflicts (where vehicles split into separate paths, typically lower risk but still relevant for sideswipes or rear-end collisions). Crossing conflicts are particularly dangerous because they involve vehicles moving at right angles, increasing the likelihood of high-impact crashes.

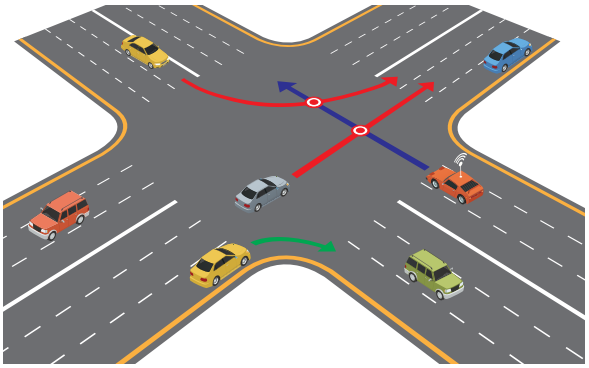


FIGURE 4. Action Conflict Filtering (ACF) aims to ignore vehicles with no common conflict point with the ego vehicle based on source and destination edge.

The proposed ACF approach reduces noise in the input data, ensuring that the agent does not waste computational resources or training cycles processing irrelevant information. For example, a vehicle that does not cross the ego vehicle's path has no bearing on the immediate decision-making process, as illustrated in Figure 4.

In this approach, we used the source-destination information to determine whether conflict points exist for two vehicles or not.

The proposed AMVF and ACF techniques differ fundamentally from attention-based methods in both mechanism and operational paradigm. While AMVF/ACF approaches employ deterministic geometric rules (conflict-point analysis, edge-based filtering) that require no training and provide explicit interpretability, attention-based methods, such as [47], use learned weight distributions to implicitly prioritize observations, requiring extensive training data and potentially missing edge cases (e.g., rare conflict geometries). The advantage of AMVF/ACF methods is that they generalize across intersection topologies by relying on fundamental geometric relationships, whereas attention methods may require retraining for new layouts. Moreover, the computational resource requirements of AMVF/ACF are also lower, enabling real-time execution in resource-constrained OBUs.

In practice, both the AMVF and ACF approaches are closely related to observation filtering. In RL frameworks (e.g., RLlib), the way to avoid an agent taking an action is not to give the observation state in the previous step.

5) MULTI-POLICY APPROACH

In tasks where agents need to perform different roles, distinct policies allow each agent or group of agents to learn specialized behaviors tailored to their specific tasks, leading to improved overall performance. Although the agents look similar in the examined autonomous intersection environment, we can categorize them based on the source edge the ego vehicle originates its trajectory. In this case, instead of a single policy, where all situations are handled by one trained model, we divide the observable situations and assign the policy to the agent according to its originating point.

We define a policy mapping function $\psi : \mathcal{E} \rightarrow \Pi$, where:

- $\mathcal{E} = \{N, E, S, W\}$ is the set of source edges
- $\Pi = \{\pi_N, \pi_E, \pi_S, \pi_W\}$ is the set of policies (one per edge)

For a vehicle i entering from edge $e_i \in \mathcal{E}$, its policy is determined by Eq. 16:

$$\pi_i = \psi(e_i) \tag{16}$$

We used the same observation and action spaces in each policy.

TABLE 3. SUMO configuration.

Parameter	Value
Traffic volume	1.25 veh/sec (4500 veh/hour)
Simulation step	0.1 sec
Default speed	13.89 m/s (50 km/h)
Speed change in one simulation step	$\pm$ 2 m/s or no change

V. SIMULATION SETUP AND EVALUATION

We defined a 4-way and 3-lane autonomous intersection environment using the SUMO microscopic traffic simulation tool [8]. The simulator enables precise control over traffic demand, individual vehicle dynamics, road geometry, and driving rules. In our experiments, we used *randomTrips.py* SUMO tool to generate random traffic whenever an episode starts. By default, the turning rate is random with equal probabilities in scenarios with turns enabled. The traffic density was controlled by the traffic volume parameter that specifies the number of vehicles entering the simulation. With the settings applied, a vehicle entered the simulation every 0.8 sec. The SUMO simulation step duration was reduced to 0.1 sec. from the default 1 sec., to have an acceptable time resolution that also determines the action frequency of the RL model. Outside the intersection zone, the vehicles are controlled by the SUMO algorithm with a default speed limit of 13.89 m/s. Thus, in most cases, vehicles enter the intersection zone at this speed. After entering the intersection zone, the vehicles are controlled by the RL agents. The agent can choose from three actions: increase or decrease the speed by 2 m/s, or keep the current speed. The SUMO-related configuration parameters are collected in Table 3, all other simulation parameters are SUMO defaults.

We control the assigned RL agent's speed through the TraCI API after the agent vehicle enters the intersection zone. The open-source RLlib library [28] for reinforcement learning is an ideal choice for the multi-policy, 4-way autonomous intersection environment, as it natively supports multi-agent, multi-policy training without hacks unlike generic RL frameworks. As our aim is to investigate the improvement in our proposed methods, we chose DQN (Double Dueling DQN) and PPO with an LSTM layer for our experiments, which are the two most popular RL

TABLE 4. DQN and PPO algorithms parameter configuration.

Parameter	DQN	PPO
Batch size	4096	8192
Fragment size	512	1024
Learning rate	0.0001	0.0001
Gamma	0.9	0.9
Clip	n/a	0.3
Lambda	n/a	0.99
LSTM size	n/a	32

TABLE 5. Best-Performing Configuration Used As The Initial Setup.

Parameter	Value
Observation space	rel_coord_dest
Obervation range (r)	20 m
Number of vehicles in observation space (N)	1
AMVF	on
ACF	on

algorithms used nowadays. We configured the algorithms using parameter values as shown in Table 4.

As most of the works in this field assume a 4-way intersection environment with or without turn enabled, we also separately conducted our experiments in this manner. To train and evaluate the model, we used one node of the Komondor HPC with 4 cores of AMD EPYC 7763 CPU and 8 GB RAM. We configured the model training to run for 1 million agent steps in non-turning scenarios and 3 million agent steps in turning scenarios. Training a model took approximately 10 and 30 hours, respectively.

In reinforcement learning, an episode represents a complete sequence of interactions between an agent and its environment, from an initial state to a terminal state. In our case, we terminated an episode when any of the vehicles collided.

Our experiments can be divided into two main parts. First, we introduce how the proposed approaches affect the training phase, followed by the evaluation of the trained model. To make the obtained results comparable, we used the best-performing configuration as the initial setup (Table 5), and only the observed parameter was changed in a series of measurements.

A. TRAINING PHASE

The training phase in reinforcement learning acts as a diagnostic dashboard, revealing how effectively the model learns and adapts to the environment. In our 4-way autonomous intersection environment, we analyzed two scenarios, (*i.*) turns disabled: vehicles move in a straight line, (*ii.*) turns allowed: vehicles can drive straight ahead and turn right and left.

Training metrics provide crucial information on whether edge-specific policies (e.g. π_N for northbound vehicles) are developing safe and efficient navigation strategies. The success of the training process can be recognized through key metrics: climbing episode rewards signal improving decision making, while decreasing collision rates indicate better safety awareness.

In the first experiment with turns disabled, the four different observation spaces were compared from the episode reward mean of the $\pi_N, \pi_E, \pi_S, \pi_W$ policies and collision ratio point of view. The results presented in Figure 5 for both DQN and PPO algorithms show that using relative coordinates improves training efficiency. The destination edge information enables the agent to build an intuitive understanding of the vehicle's future trajectories. However, this information is valuable only if turns are allowed.

We can also conclude that PPO significantly outperforms DQN. The reason for lower cumulative rewards and higher collision rates can be due to rigid ϵ-greedy exploration (e.g., random actions in critical moments) or suboptimal rewards if Q-values are poorly estimated, which is common in complex multi-agent settings. On the other hand, PPO maintains lower collision rates due to smoother policy updates and adaptive exploration.

The observation range filtering approach proposes to include only the closest vehicles within a defined range r in the observation space of the agents. It acts as a cognitive filter for autonomous agents to sharpen their focus while cutting the noise of irrelevant data. By discarding information about vehicles beyond the critical radius, the observation space sheds unnecessary dimensions and sharpens the focus on vehicles with high collision risk, while eliminating the noise of irrelevant data. By excluding distant vehicles, we prevent the policy from wasting capacity on irrelevant correlations. Moreover, in all RL frameworks, the agent always processes the same observation dimensions. The proposed observation range filtering approach eliminates the need for complex padding schemes or dynamic architectures. The obtained results shown in Figure 6 justify our assumptions, as $r = 20$ m produced the highest episode rewards and lowest collision ratios for both DQN and PPO. Note that setting a high range value ($r = 40$ m) actually prevented the model from training.

Reducing the dimension of the observation space is also possible by limiting the number of vehicles included in the observation. Limiting the observation space to N closest vehicles offers several compelling advantages. First, vehicles in close proximity pose the highest risk of collisions. By filtering out distant vehicles, the agent concentrates computational resources on evaluating and responding to imminent dangers. Second, keeping only N vehicles ensures the observation space has a consistent dimensionality, while smaller input vectors reduce memory bandwidth and accelerate the training. The results show (Figure 7) that including all vehicles (within a defined range r) leads to the highest collision ratio, but the differences are not so significant as in the case of the observation range.

Among the proposed filtering techniques, AMVF has the highest impact on the training efficiency and, thus, the trained model performance. Vehicles that have either passed all

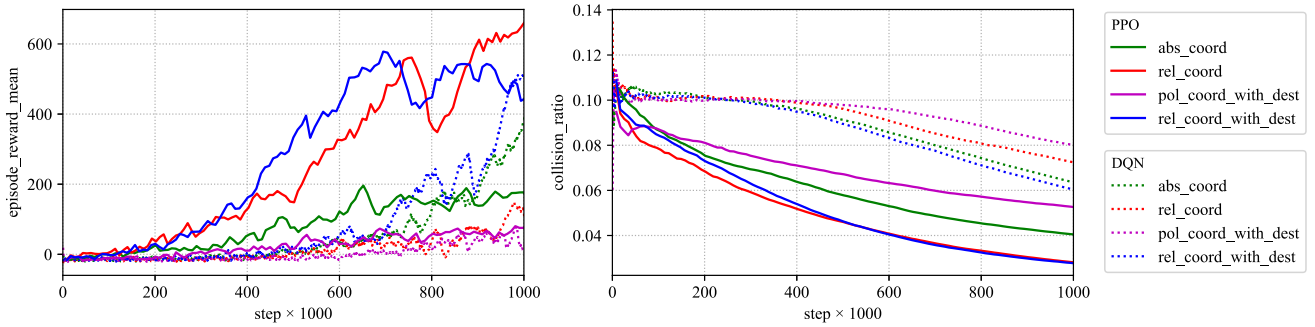


FIGURE 5. Comparison of the training process for four different observation space configurations in a non-turning scenario, evaluated using the DQN and PPO reinforcement learning algorithms. The graphs plot the mean episode reward (left) and the collision ratio (right) over the course of training. The results demonstrate that observation spaces incorporating relative coordinates and destination edge information lead to higher rewards and fewer collisions, indicating more efficient learning and safer navigation policies for autonomous vehicles at a 4-way intersection.

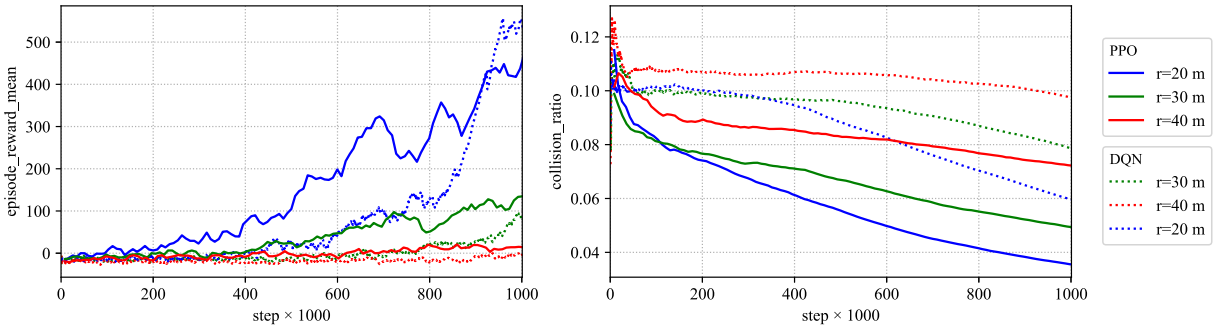


FIGURE 6. Effect of observation range (r) on the training performance. Comparison of training performance for different observation ranges (r), measured in meters. The graphs show the mean episode reward (left) and collision ratio (right) over training steps for both DQN and PPO algorithms. A range of $r = 20$ m results in the highest rewards and lowest collision rates, indicating the agent focuses best on immediately relevant vehicles. A larger range of $r = 40$ m introduces too much noise and prevents effective learning.

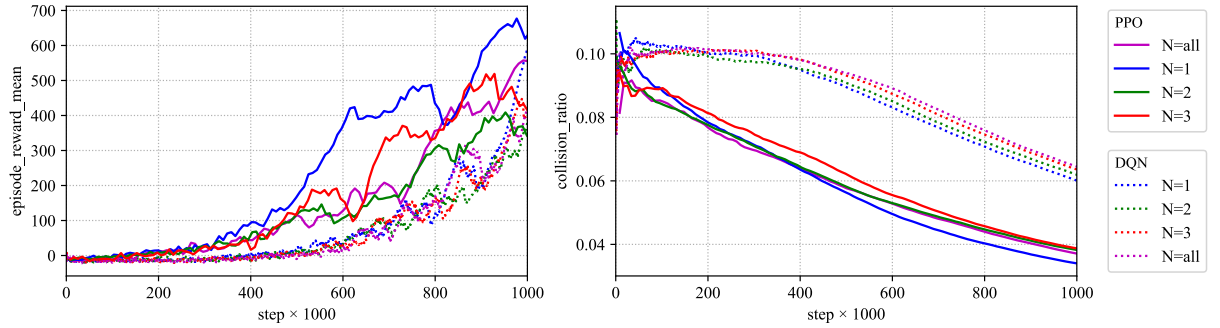


FIGURE 7. Number (N) of closest vehicles included in the observation space. Training performance by limiting the observation space to the N closest vehicles within a defined range. Metrics include mean episode reward and collision ratio over training steps for DQN and PPO. Including all vehicles (N=All) leads to the highest collision rate. Limiting N simplifies the observation space and improves focus, though the performance differences are less dramatic than when adjusting the physical range r.

conflict points or whose trajectories show no intersection with the ego vehicle's path contribute negligible mutual information to the immediate decision-making process and lead to "*ghost updates*" policy adjustments based on irrelevant inputs. Our experiments with intersection navigation tasks demonstrate that this selective observation reduces the number of agent steps that do not provide meaningful feedback for the training process and improves the policy performance due to the elimination of distracting noise variables. The mean results obtained from the reward for the episode in Figure 8 show that without AMVF, the model was not even able to learn. The reason is that the agent fundamentally reshapes the action-feedback loop during training, eliminating a pervasive source of degenerate policy

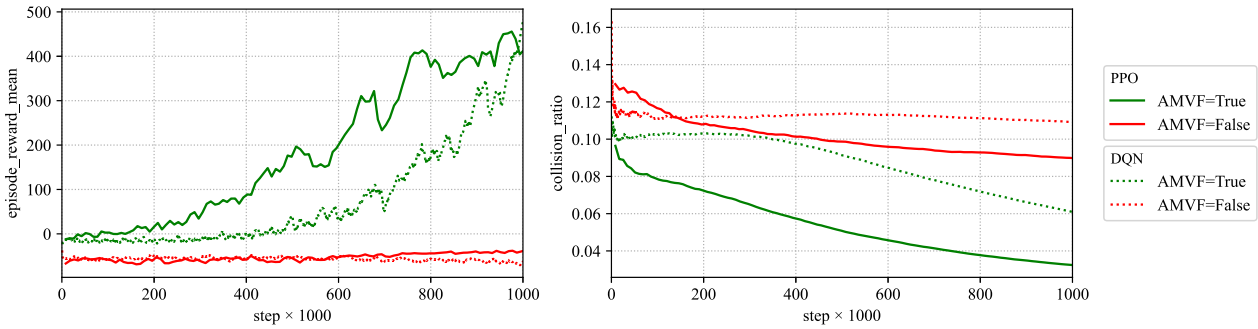


FIGURE 8. Impact of Action Movement Vector Filtering (AMVF) on training efficiency. AMVF removes vehicles from the observation space that pose no immediate conflict (e.g., they have passed all conflict points). The results show that without AMVF, the model fails to learn, as evidenced by near-zero reward. With AMVF enabled, the policy converges effectively, achieving high rewards and demonstrating that filtering irrelevant data is critical for successful learning.

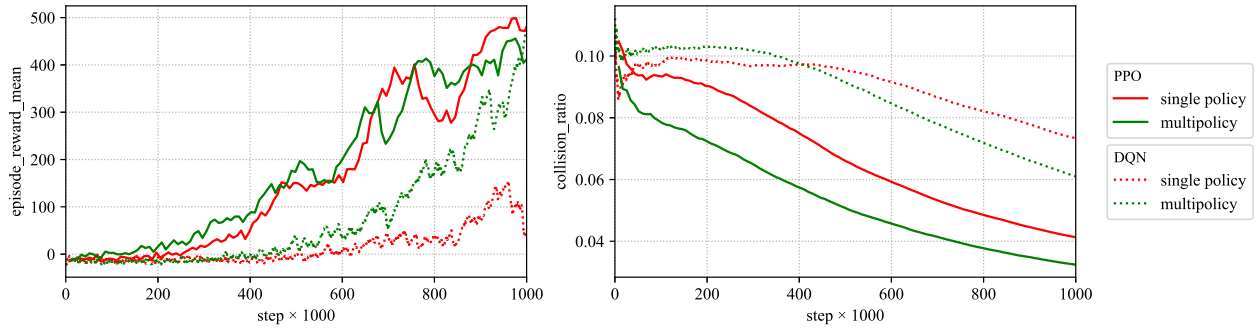


FIGURE 9. Training performance comparison between a single shared policy for all vehicles and a multi-policy setup with four distinct policies ($\pi_N, \pi_E, \pi_S, \pi_W$), each specialized for vehicles originating from a specific direction (North, East, South, West). The multi-policy approach achieves a higher mean episode reward and lower collision ratio, as task decomposition allows for specialized, more stable learning compared to a single policy that must average across all scenarios.

updates. Without AMVF, the policy cannot discern whether a successful outcome (e.g., smooth traversal) stemmed from valid actions (e.g., yielding to an oncoming car) or spurious ones (e.g., reacting to a distant vehicle). This ambiguity slows convergence.

ACF is closely related to AMVF because it aims to identify possible conflict points and filter irrelevant ones. The difference with AMVF is that ACF uses vehicle trajectories based on source and destination edges, instead of the movement vector. The drawback of ACF is that it does not take into account whether the vehicle has already passed the conflict point or not yet. For example, if a vehicle has already passed the conflict point, its route is still considered as a conflicting route. Thus, this vehicle is not removed from the ego vehicle's observation space, contrary to AMVF approach that has more strict filtering criteria. Therefore, ACF impacts the training efficiency only if AMVF is not used.

Finally, we analyzed how the policy configuration affects the training performance. By assigning distinct policies based on source edges ($\pi_N, \pi_E, \pi_S, \pi_W$), the system effectively decomposes the complex navigation task into specialized sub-problems, each with its own optimal solution space. Contrary to this, the single-policy baseline forces a monolithic solution to approximate traffic conditions, inevitably settling for policy parameters that represent a compromised average across all edge conditions. The reason the multi-policy approach performs better is that when all agents share a single policy, the environment becomes highly volatile from any individual agent's perspective because each policy update simultaneously alters the behavior of all agents, creating a moving target problem. The multi-policy architecture also demonstrates better sample efficiency. By decomposing the universal state space into edge-specific partitions, each policy trains on a more coherent and compact distribution of states. Our results presented in Figure 9 confirm that the multi-policy architecture preserves the benefits of shared learning while avoiding the performance penalties of complete policy uniformity.

B. EVALUATING THE TRAINED MODEL

Training efficiency, while important for practical development, represents merely the first stage of policy validation. A comprehensive evaluation of the trained model ensures that the learned behaviors not only solve the training environment but also generalize to the infinite state complexity of actual deployment conditions. Loading and evaluating a trained

TABLE 6. Collision ratios [%] during evaluation using the trained models.

		No Turns		Turns	
		PPO	DQN	PPO	DQN
Observation space	abs_coord	2.32%	0.51%	3.48%	4.15%
	rel_coord	0.13%	3.64%	3.35%	3.35%
	rel_coord_dest	0.13%	0.13%	2.01%	2.41%
	pol_coord_dest	2.44%	2.42%	4.55%	4.02%
r	20	0.13%	0.13%	2.01%	2.41%
	30	2.84%	2.81%	2.81%	4.29%
	40	4.52%	5.75%	4.69%	5.36%
N	1	0.13%	0.13%	2.01%	2.41%
	2	0.59%	1.13%	2.54%	3.35%
	3	0.79%	1.28%	2.41%	3.48%
	all	0.86%	1.09%	2.41%	4.82%
AMVF	off	13.96%	11.79%	8.85%	5.23%
	on	0.13%	0.13%	2.01%	2.41%
ACF	off	n/a	n/a	3.70%	3.86%
	on	n/a	n/a	2.01%	2.41%
Policy	single policy	0.68%	1.35%	2.94%	5.09%
	multipolicy	0.13%	0.13%	2.01%	2.41%
Baseline	keep speed	10.08%		6.12%	

model separate from analyzing training efficiency is critical because doing so provides a true measure of the policy's generalization capability, robustness, and performance. Running a trained RL model on new observations to produce an action is orders of magnitude less computationally expensive than training. Our trained model consumes less than 10 MB of storage space.

We evaluated all proposed filtering, observation space reduction, and policy-related methods with different parameter setups to see exactly how effective our techniques are. The primary performance metric in the autonomous intersection environment is the collision rate. The obtained results of 10,000 step-long experiments are summarized in Table 6. To make the table easier to read, we highlighted the best configuration for each configuration category.

Using the PPO algorithm with an LSTM layer and with the best configuration setup, we were able to achieve 0.13% collision ratio in the scenario when the turns were disabled. With turns allowed, the collision ratio increased to 2.01%. Both results are significant improvements compared to the corresponding results of related works collected in Table 7. It is worth highlighting that all of the proposed methods with the appropriate settings improved the performance of the model, but the most significant collision rate reduction was achieved by the AMVF method. Using this method reduced the collision rate from 13.96% (DQN) and 11.79% (PPO) to 0.13% in the no-turns scenario, which is a very impressive 99% collision event reduction. In the case of the turns-allowed scenario, the number of collisions was reduced by 55-78%.

In addition to the quantitative metrics, we analyzed the qualitative behaviors of the learned policies to understand how strategic actions contributed to collision reduction. The policies exhibited several key behaviors that improved safety and efficiency:

- *Proactive speed adjustment*: The ego vehicle learned to modulate its speed proactively when approaching other vehicles detected in conflicting paths. It is noticeable that the vehicle closer to the conflict point accelerates, and the one further away reduces its speed to avoid collision.
- *Robustness to edge cases*: The policies handled edge cases such as near-simultaneous arrival at conflict points by favoring conservative actions, e.g., slight deceleration of both conflicting ego vehicles when uncertainty was high. This behavior emerged naturally from the PPO algorithm's clipping mechanism, which discourages overly aggressive updates.
- *Conflict-point awareness*: The AMVF-enabled policies demonstrated a heightened awareness of conflict points, allowing the ego vehicle to prioritize interactions with vehicles on collision courses while ignoring irrelevant ones. For example, vehicles moving parallel or away from the intersection were correctly filtered out, reducing unnecessary deceleration.
- *Edge-specific strategies*: In the multi-policy approach, vehicles from different entry points (north, south, east, west) developed specialized strategies. Multi-policy decomposes the problem into smaller, edge-specific observation subspaces, simplifying the learning task for each policy, thus making it more efficient.
- *Dynamic distance management*: The dynamic reward component ($\mathcal{R}_{\text{dyn}}$) encouraged the ego vehicle to maintain an adaptive "safety bubble" around itself.

From our analysis of the experiment results, the single most important insight is to filter unnecessary agent actions that do not provide meaningful feedback for the training process. Furthermore, reducing the observation space complexity and categorizing agents using distinct policies is critically important to improve the performance of agent actions.

Although the main goal of the paper is to investigate new RL approaches for autonomous intersection in order to minimize the collision rate and improve the model training efficiency, we also examined multiple traffic-related performance indicators of autonomous intersection. The primary metrics used to verify the performance of traffic control approaches are travel time, waiting time, and congestion-induced time loss. Waiting time was defined as periods when vehicle speed dropped to 0.1 m/s or lower, which can result from various causes, including traffic congestion. While travel time and congestion-related delays are closely linked, both were included to contextualize the impact of congestion relative to overall journey duration. Additionally, secondary metrics such as emissions (CO, CO_2, NOx) and fuel consumption were assessed to highlight the potential environmental advantages of these systems, beyond just improvements in travel efficiency and traffic reduction. We used the default HBEFA3/PC_G_EU4 [51] emission and consumption model of SUMO, specified for gasoline-powered Euro 4 norm passenger cars. The obtained results

TABLE 7. Summary of autonomous driving studies at unsignalized intersections (4-Way, Without Traffic Sign).

Ref.	Year	Brief Summary	Collision Rate (%)		
			Straight	L. Turn	R. Turn
[16]	2018	RL-based approach using hierarchical options.	1.7%	2.6%	0.2%
[17]	2022	A hybrid architecture consisting of three components to control vehicles' maneuver using DRL.	5%	n/a	n/a
[15]	2020	DRL in highway scenario in which an AV overtakes other vehicles efficiently and safely.	>10%	n/a	n/a
[46]	2018	DRL using automatic generation of curriculum for training enhancement.	13.5%	n/a	n/a
[47]	2021	Attention-based deep reinforcement learning.	3%	5%	2%
[48]	2021	Hierarchical reinforcement learning with hybrid reward mechanism.	4.2%	n/a	n/a
[49]	2022	DQL and DDQ for decision-making in unsignalized intersections.	6%	10%	7.5%
[50]	2022	Safe and rule-aware deep reinforcement learning.	1.4%	n/a	n/a

TABLE 8. Efficiency and Fuel Consumption of Traffic Flow (turns allowed).

Control Type	Travel time [s]	Waiting time [s]	Time loss [s]	CO_2 emiss. [g]	CO emiss. [g]	NOx emiss. [mg]	Fuel cons. [g]
Right-hand priority	62.82	41.82	49.03	191.42	8.54	84.63	61.05
Traffic light	50.01	24.83	36.01	154.03	6.12	67.45	49.13
RL with AMVF	17.09	0.02	3.64	82.52	1.37	33.97	26.32

TABLE 9. Efficiency and Fuel Consumption of Traffic Flow (turns not allowed).

Control Type	Travel time [s]	Waiting time [s]	Time loss [s]	CO_2 emiss. [g]	CO emiss. [g]	NOx emiss. [mg]	Fuel cons. [g]
Right-hand priority	90.31	68.75	76.01	254.01	12.82	113.04	81.02
Traffic light	32.76	10.73	18.70	109.71	3.41	46.88	34.99
RL with AMVF	18.2	0.01	4.07	88.08	1.42	36.14	28.09

are summarized in Table 8 and Table 9 for turns allowed and without vehicle turns scenarios, respectively.

The proposed RL-based approach extended with AMVF proves significantly more efficient in terms of delays, emissions, and fuel usage, while maintaining robustness in varied traffic conditions. Its ability to minimize stopping and optimize flow highlights its potential as a superior solution for autonomous intersections.

VI. CONCLUSION AND FUTURE WORK

In this paper, we introduced new approaches to improve the training performance by filtering irrelevant actions and observations in reinforcement learning (RL) for autonomous intersection control. By systematically filtering irrelevant observations and actions, we have demonstrated how to dramatically improve both training efficiency and trained policy performance. Contrary to the conventional approach to rely on high-performance hardware platform, such as GPU, our approach is effective independent of the hardware platform. In this sense, our approach can be characterized as *software-based acceleration*. If we combine our approach with high-performance hardware platform, the level of acceleration will be even higher.

The key innovation lies in our action movement vector filtering (AMVF) approach, which eliminates noisy updates from non-conflicting vehicles, significantly improving training efficiency, policy performance, and collision avoidance. By implementing intelligent filters (distance thresholds, nearest-neighbor selection, conflict-point analysis), we enabled policies to focus computational resources on genuinely impactful interactions.

The superiority of PPO over DQN in this domain underscores the importance of algorithm selection for safety-critical applications. The PPO's policy gradient approach, enhanced with LSTM memory, consistently delivered safer and more adaptable behaviors. The performance of the proposed multi-policy architecture was also convincing, where edge-specific specialization allows each policy to develop nuanced strategies for its unique traffic challenges.

The presented results have far-reaching implications beyond traffic control. The principles we have applied, such as strategic observation filtering, action relevance assessment, and role-specific policy architectures, could make autonomous systems increasingly prevalent in safety-critical environments.

Regarding the future research, our results open several options, particularly in scaling the solution to city-wide deployments. Future work should investigate extendable or reusable RL models capable of optimizing traffic flow on a city scale while maintaining computational efficiency. Another critical direction is improving the robustness of the system against real-world uncertainties, such as perturbations by adversaries, sensor errors, and communication failures. Autonomous intersection control must remain reliable even with the presence of adversaries, noisy or missing data, which necessitates the development of fault-tolerant RL

algorithms. Future research could explore techniques such as adversarial training, uncertainty-aware RL, or hybrid models that combine learning with rule-based fallback mechanisms.

ACKNOWLEDGMENT

The authors acknowledge Digital Government Development and Project Management Ltd., for awarding us access to the Komondor HPC facility based in Hungary.

REFERENCES

[1] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, document SAE J3016, SAE International, 2021.

[2] R. Trypuz, P. Kulicki, and M. Sopek, "Ontology of autonomous driving based on the SAE J3016 standard," *Semantic Web*, vol. 15, no. 5, pp. 1837–1862, Oct. 2024.

[3] M. S. Blumenthal and R. Csernatoni, "Computers on wheels: Automated vehicles and cybersecurity risks in Europe," EU Cyber Direct, Tech. Rep., 2022. Accessed: Oct. 10, 2025. [Online]. Available: https://eucd.s3.eu-central-1.amazonaws.com/eucd/assets/nXN2Iy4l/computers-on-wheels-automated-vehicles-and-cybersecurity-risks-in-europe.pdf

[4] M. Al-Sharman, L. Edes, B. Sun, V. Jayakumar, M. A. Daoud, D. Rayside, and W. Melek, "Autonomous driving at unsignalized intersections: A review of decision-making challenges and reinforcement learning-based solutions," 2024, *arXiv:2409.13144*.

[5] Y. Zhou and Y. Chen, "Learning to drive in the NGSIM simulator using proximal policy optimization," *J. Adv. Transp.*, vol. 2023, pp. 1–12, Apr. 2023.

[6] M. Zhou et al., "SMARTS: Scalable multi-agent reinforcement learning training school for autonomous driving," 2020, *arXiv:2010.09776*.

[7] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and K. Vladlen, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, Apr. 2017, pp. 1–16.

[8] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic traffic simulation using SUMO," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2575–2582. [Online]. Available: https://elib.dlr.de/124092/

[9] S. Siboo, A. Bhattacharyya, R. N. Raj, and S. H. Ashwin, "An empirical study of DDPG and PPO-based reinforcement learning algorithms for autonomous driving," *IEEE Access*, vol. 11, pp. 125094–125108, 2023.

[10] A. Mittal and A. Malte, "On multi-agent deep deterministic policy gradients and their explainability for SMARTS environment," 2023, *arXiv:2301.09420*.

[11] N. S. Manikandan, G. Kaliyaperumal, and Y. Wang, "Ad hoc-obstacle avoidance-based navigation system using deep reinforcement learning for self-driving vehicles," *IEEE Access*, vol. 11, pp. 92285–92297, 2023.

[12] A. Guillen-Perez and M. D. Cano, "RAIM: Reinforced autonomous intersection management—AIM based on MADRL," in *Proc. 34th NeurIPS Conf.*, 2020. [Online]. Available: https://www.researchgate.net/publication/357957238_RAIM_Reinforced_Autonomous_Intersection_Management_-_AIM_based_on_MADRL

[13] G.-P. Antonio and C. Maria-Dolores, "Multi-agent deep reinforcement learning to manage connected autonomous vehicles at tomorrow's intersections," *IEEE Trans. Veh. Technol.*, vol. 71, no. 7, pp. 7033–7043, Jul. 2022.

[14] M. Al-Sharman, R. Dempster, M. A. Daoud, M. Nasr, D. Rayside, and W. Melek, "Self-learned autonomous driving at unsignalized intersections: A hierarchical reinforced learning approach for feasible decision-making," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 12345–12356, Nov. 2023.

[15] J. Liao, T. Liu, X. Tang, X. Mu, B. Huang, and D. Cao, "Decision-making strategy on highway for autonomous vehicles using deep reinforcement learning," *IEEE Access*, vol. 8, pp. 177804–177814, 2020.

[16] Z. Qiao, K. Muelling, J. Dolan, P. Palanisamy, and P. Mudalige, "POMDP and hierarchical options MDP with continuous actions for autonomous driving at intersections," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2377–2382.

[17] R. Gutiérrez-Moreno, R. Barea, E. López-Guillén, J. Araluce, and L. M. Bergasa, "Reinforcement learning-based autonomous driving at intersections in CARLA simulator," *Sensors*, vol. 22, no. 21, p. 8373, Nov. 2022.

[18] U. Gunarathna, S. Karunasekera, R. Borovica-Gajic, and E. Tanin, "Real-time intelligent autonomous intersection management using reinforcement learning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2022, pp. 135–144.

[19] P. Karthikeyan, W.-L. Chen, and P.-A. Hsiung, "Autonomous intersection management by using reinforcement learning," *Algorithms*, vol. 15, no. 9, p. 326, Sep. 2022.

[20] A. Mushtaq, I. U. Haq, M. U. Imtiaz, A. Khan, and O. Shafiq, "Traffic flow management of autonomous vehicles using deep reinforcement learning and smart rerouting," *IEEE Access*, vol. 9, pp. 51005–51019, 2021.

[21] R. S. Sutton and G. Barto, *Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 2018.

[22] E. Bilgin, *Mastering Reinforcement Learning With Python*. Cambridge, U.K.: Cambridge Univ. Press, 2020.

[23] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, "A survey on deep reinforcement learning algorithms for robotic manipulation," *Sensors*, vol. 23, no. 7, p. 3762, Apr. 2023.

[24] M. Sewak, *Deep Reinforcement Learning*. Cham, Switzerland: Springer, 2019.

[25] M. T. Regehr and A. Ayoub, "An elementary proof that Q-learning converges almost surely," 2021, *arXiv:2108.02827*.

[26] (2024). *Value-Based Vs Policy-Based Reinforcement Learning*. [Online]. Available: https://papers-100-lines.medium.com/value-based-vs-policy-based-reinforcement-learning-92da766696fd

[27] Y. Liu, J. Yang, L. Chen, T. Guo, and Y. Jiang, "Overview of reinforcement learning based on value and policy," in *Proc. Chin. Control Decision Conf. (CCDC)*, 2020, pp. 598–603.

[28] (Feb. 15, 2025). *Algorithms*. [Online]. Available: https://docs.ray.io/en/latest/rllib/rllib-algorithms.html

[29] (2025). *Rainbow DQN*. [Online]. Available: https://paperswithcode.com/method/rainbow-dqn

[30] (2025). *Rainbow DQN—The Best Reinforcement Learning Has to Offer?*. [Online]. Available: https://readmedium.com/rainbow-dqn-the-best-reinforcement-learning-has-to-offer-166cb8ed2f86

[31] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. thesis, King's College, Cambridge, U.K., 1989.

[32] G. A. Rummery and M. Niranjan, "On-line Q-learning using connectionist," Dept. Eng., Univ. Cambridge, Cambridge, U.K., Tech. Rep., 1994, vol. 37.

[33] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing ATARI with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[34] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016, pp. 2094–2100.

[35] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, May 2016, pp. 1995–2003.

[36] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, May 1992.

[37] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.

[38] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[39] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vols. SMC–13, no. 5, pp. 834–846, Sep. 1983.

[40] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, May 2016, pp. 1928–1937.

[41] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[42] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.

[43] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, May 2018, pp. 1861–1870.
[44] (2025). *Deep Q-Learning in Reinforcement Learning*. [Online]. Available: https://www.geeksforgeeks.org/deep-q-learning/
[45] R. Kozlica, S. Wegenkittl, and S. Hiränder, "Deep Q-learning versus proximal policy optimization: Performance comparison in a material sorting task," in *Proc. IEEE 32nd Int. Symp. Ind. Electron. (ISIE)*, Jun. 2023, pp. 1–6.
[46] Z. Qiao, K. Muelling, J. M. Dolan, P. Palanisamy, and P. Mudalige, "Automatically generated curriculum based reinforcement learning for autonomous vehicles in urban environment," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1233–1238.
[47] H. Seong, C. Jung, S. Lee, and D. H. Shim, "Learning to drive at unsignalized intersections using attention-based deep reinforcement learning," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 559–566.
[48] Z. Qiao, J. Schneider, and J. M. Dolan, "Behavior planning at urban intersections through hierarchical reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 2667–2673.
[49] H. Shu, T. Liu, X. Mu, and D. Cao, "Driving tasks transfer using deep reinforcement learning for decision-making of autonomous vehicles in unsignalized intersection," *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 41–52, Jan. 2022.
[50] C. Zhang, K. Kacem, G. Hinz, and A. Knoll, "Safe and rule-aware deep reinforcement learning for autonomous driving at intersections," in *Proc. IEEE 25th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2022, pp. 2708–2715.
[51] D. Krajzewicz, S. Hausberger, P. Wagner, M. Behrisch, and M. Krumnow, "Second generation of pollutant emission models for SUMO," in *Proc. 2nd SUMO Conf. Model. Mobility Open Data*, Mar. 2014, pp. 89–101.



ÁRPÁD HUSZÁK received the Ph.D. degree in electrical and computer engineering from Budapest University of Technology and Economics (BME), in 2010. He is currently an Associate Professor with the Department of Networked Systems and Services. Previously, he joined the Mobile Innovation Center, Hungary, until 2010. He has done research on mobility management and 3D video streaming. His current research interests include network communications, vehicular networks, and machine learning.



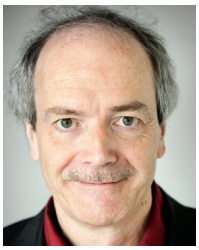
TAKAHITO YOSHIZAWA received the B.S. degree in information and computer science from Georgia Institute of Technology, in 1992, the M.S. degree in telecommunication from Southern Methodist University, in 2002, and the Ph.D. degree from Katholieke Universiteit Leuven (KU Leuven), Belgium, in 2023. As a Postdoctoral Researcher, he is currently continuing his research on security of V2X communication and reinforcement learning (RL) with the COSIC Group, KU Leuven. He has over 30 years of industry experience in mobile communication systems. He has contributed to all phases of development lifecycle of mobile infrastructure systems from 2G onward. He has participated and actively contributed to standardization organizations, such as 3GPP, and holds approximately 20 granted patents on communication systems.



ALIREZA AGHABAGHERLOO (Student Member, IEEE) received the B.Sc. degree in electrical engineering from the University of Tabriz, Tabriz, Iran, in 2018, and the M.Sc. degree in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 2020. He is currently pursuing the Ph.D. degree with the COSIC Research Group, KU Leuven, Leuven, Belgium. His research interests include the application of artificial intelligence (AI) to increase the reliability and performance of intelligent digital systems, enhancing the resilience of AI models, such as deep neural networks (DNNs), reinforcement learning (RL), and federated learning (FL), and the design and evaluation of communication protocols in connected and intelligent transportation systems.



DAVE SINGELÉE (Member, IEEE) received the master's degree in electrical engineering and the Ph.D. degree in applied sciences from KU Leuven, Belgium, in 2002 and 2008, respectively. He was the Industrial Research Manager (IOF) with the Research Group COSIC in KU Leuven until August 2025. Since September 2025, he is an Associate Professor at DistriNet Group-T of KU Leuven. His main research interests include cryptography, security and privacy of wireless communication networks, key management, distance bounding, cryptographic authentication protocols, and security solutions for medical devices.



BART PRENEEL (Member, IEEE) is currently a Full Professor with KU Leuven, Leuven, Belgium, where he heads the COSIC Research Group. He has authored over 450 scientific publications and is the inventor of five patents. His main research interests include cryptography, information security, and privacy. He is a member of the Advisory Group of ENISA and the Academia Europaea. He is also a member of the Knowledge Center of Belgian Data Protection Authority. In 2015, he was elected as a fellow of IACR. He received the RSA Award for Excellence in the Field of Mathematics, in 2014; the IFIP TC11 Kristian Beckman Award, in 2015; and the ESORICS Outstanding Research Award, in 2017. He was the program chair of more than 20 international conferences. He has been an invited speaker at over 120 conferences in 50 countries. He was the President of the International Association for Cryptologic Research (IACR) and is the Co-Founder and the Chairperson of the Board of the Information Security Cluster LSEC.