Scientific Journal of Silesian University of Technology. Series Transport
Zeszyty Naukowe Politechniki Śląskiej. Seria Transport

Volume 125 **2024**

p-ISSN: 0209-3324

e-ISSN: 2450-1549

DOI: https://doi.org/10.20858/sjsutst.2024.125.16

Journal homepage: http://sjsutst.polsl.pl

Article citation information:
Rizehvandi, A., Azadi, S. Developing a controller for an adaptive cruise control (ACC) system: utilizing deep reinforcement learning (DRL) approach. *Scientific Journal of Silesian University of Technology. Series Transport*. 2024, **125**, 243-257. ISSN: 0209-3324.
DOI: https://doi.org/10.20858/sjsutst.2024.125.16.

Ali RIZEHVANDI[1], Shahram AZADI[2]

DEVELOPING A CONTROLLER FOR AN ADAPTIVE CRUISE CONTROL (ACC) SYSTEM: UTILIZING DEEP REINFORCEMENT LEARNING (DRL) APPROACH

Summary. The addition of Adaptive Cruise Control (ACC) to vehicles enables automatic speed adjustments based on traffic conditions after the driver sets the maximum speed, freeing them to concentrate on steering. This study is dedicated to the development of a passenger car ACC system using Deep Reinforcement Learning (DRL). A critical aspect of this ACC system is its capability to regulate the distance between vehicles by taking into account preceding and following vehicle speeds. It considers three primary inputs: the memory-stored speed of the following vehicle, the lead time specified by the driver, and the radar-measured distance. By adapting speed in different traffic scenarios, the system contributes to averting potential accidents. This research delves into constructing a controller that utilizes the Deep Deterministic Policy Gradient (DDPG) algorithm and compares its outcomes with those from the DQN algorithm. The DDPG controller supervises the longitudinal control actions of a vehicle, enabling it to execute stopping and moving maneuvers safely and efficiently.

Keywords: adaptive cruise control (ACC) system, deep reinforcement learning (DRL) method, deep deterministic policy gradient (DDPG) algorithm

[1] Department of Mechanical Engineering, K.N. Toosi University of Technology, Tehran 15418-49611, Iran. Email: ali.rizehvandi@email.kntu.ac.ir. ORCID: https://orcid.org/0009-0007-9259-2101
[2] Department of Mechanical Engineering, K.N. Toosi University of Technology, Tehran 15418-49611, Iran. Email: aazadi@email.kntu.ac.ir. ORCID: https://orcid.org/0000-0002-2060-6988

1. INTRODUCTION

Integrating processors, controllers, and automation technologies in intelligent transportation systems improves safety, performance, and efficiency, while reducing environmental impacts and energy consumption. Smart vehicles are essential in these systems as they automate various driving tasks, including lane keeping, obstacle avoidance, following other vehicles, and identifying and evading dangerous situations. Developing smart cars with comfortable, safe, and efficient driving experiences is the main goal. Over the years, Adaptive Cruise Control (ACC) has advanced to improve driving safety, comfort, fuel efficiency, and traffic capacity. Modern ACC systems use sensors and intelligent algorithms to automatically adjust the vehicle's speed, ensuring a safe distance from vehicles ahead, thereby minimizing the chance of collisions and decreasing driver fatigue. Additionally, ACC's effective management of speed and distance helps in creating a more seamless flow of traffic, increasing the road's capacity, and reducing congestion. As ACC technology progresses, it has great potential to influence the future of car safety and transportation. Reinforcement learning (RL) has become a popular approach for creating ACC controllers in recent times. RL algorithms such as Deep Q-Networks (DQN), Proximal Policy Optimization (PPO), and Deep Deterministic Policy Gradient (DDPG) allow the learning of intricate control strategies from raw sensor data or state information. Through interaction with the environment, ACC controllers based on RL can adjust and refine their behavior, improving their performance across different driving situations. ACC systems can effectively manage various traffic conditions, road geometries, and driver preferences due to their flexibility, which is superior to traditional rule-based approaches. Furthermore, RL-based ACC controllers can potentially adapt and learn from changing traffic patterns and driver behaviors, leading to safer and more efficient driving experiences.

A new adaptive leading cruise control approach for connected and automated vehicles (CAVs) is introduced in [1]. This approach takes into account its influence on following human-operated vehicles (HDVs) with varying driving habits in a unified optimization framework to improve overall energy efficiency. The driving behaviors of HDVs when they follow other vehicles are statistically adjusted based on data from the Next Generation Simulation dataset. When connected and automated vehicles (CAVs) and human-operated vehicles (HDVs) share the road, the longitudinal speed control of CAVs plays a crucial role in reducing the energy consumption of following HDVs by minimizing unnecessary acceleration and braking. Furthermore, the reward function of reinforcement learning incorporates the energy efficiencies of both CAVs and HDVs, as well as objectives concerning car-following safety and traffic efficiency. Huang and the team devised safety measures known as control barrier functions, allowing an RL-based Adaptive Cruise Control (ACC) controller to safely maneuver within a collision-safe zone while retaining the capability to explore. They tailored the control barrier functions for highly nonlinear systems with substantial inertia effects pertinent to commercial vehicles. They presented an algorithm to handle actuation saturation by utilizing these barrier functions. The performance ACC controller uses the Maximum A Posteriori Policy Optimization (MPO) algorithm, which integrates a hybrid action space to learn fuel-optimal gear selection and torque control policies, and incorporates these filters. This differs from an approach using reward-shaping RL [2, 3]. The proposed approach shows substantial enhancements in fuel efficiency over traditional ACC algorithms in evaluations across different driving situations. Furthermore, an autonomous car-following model called SECRM (Safe, Efficient, and Comfortable RL-based car-following Model) is presented, prioritizing maximizing traffic efficiency and minimizing jerk while maintaining a strict analytical safety constraint on acceleration. The reason for this restriction is the need for the following vehicle to maintain a safe distance to prevent a potential collision in case the lead vehicle abruptly slows down. Yang and colleagues have developed a system called Deep Reinforcement Learning-based adaptive cruise control (DRL-ACC) [4, 5]. This system generates reliable, flexible, and quick-responding car-following policy agents and suggests a distinct high-level action space for accelerating, decelerating, and maintaining the current speed. It also includes a multi-objective reward system that reflects safe, responsive, and logical traffic conduct. Additionally, in [6], the effectiveness of the proposed framework is compared with conventional ACC methods within the CoMoVe framework, with the aim of ensuring stability, comfort, and efficient traffic flow in various real-world driving situations. The DRL approach demonstrates strong road usage efficiency when the lead vehicle's speed varies, with very few instances of exceeding ideal following distances. Furthermore, Park and colleagues suggest a data-based application of Deep Reinforcement Learning (DRL) for Adaptive Cruise Control. This method takes into account various goals including safety, passenger comfort, and effective road capacity utilization. The method's performance is being compared to traditional ACC approaches in the CoMoVe framework, which accurately models communication, traffic, and vehicle dynamics. In this research, we are using the Deep Deterministic Policy Gradient (DDPG) algorithm to control the Adaptive Cruise Control (ACC) system, representing a significant advancement in autonomous driving technology. DDPG is a reinforcement learning algorithm designed for tasks with continuous action spaces, making it well-suited for applications like ACC, where smooth and precise control is crucial. By utilizing DDPG implementation, the ACC system has the capacity to grasp efficient control techniques through its interactions with the environment, consistently adjusting its behavior to suit various driving situations. This methodology shows potential for enhanced flexibility and resilience when compared to conventional rule-based or heuristic control approaches. Additionally, DDPG empowers the ACC system to derive knowledge from its encounters, possibly enhancing its effectiveness as it encounters a broader spectrum of real-world driving conditions.

The article is structured as follows: Section 2 provides an overview of the ACC system, Section 3 explores the DDPG algorithm and how it is applied to the ACC problem, and Sections 4 and 5 evaluate simulation and simulation results, respectively.

2. ADAPTIVE CRUISE CONTROL (ACC) SYSTEM

The adaptive cruise control system (ACC) is designed to keep a safe distance or time interval with the vehicle ahead while also keeping the car at a constant speed chosen by the driver. The safe distance is the minimum space needed for a driver to regain control and avoid an accident if the front vehicle suddenly stops. The time interval between two cars represents the time it takes for the following car to reach the current position of the front car. Figure 1 provides a visual representation of this time interval between the two vehicles. When driving, It is important for the driver to stay focused for extended periods and be prepared to react quickly to sudden changes, which often happen within fractions of a second. Additionally, maintaining a safe distance from other vehicles, as well as those in adjacent lanes and oncoming traffic, is crucial for overall safety. The Adaptive Cruise Control (ACC) system now assists with tasks that were traditionally the sole responsibility of drivers. Like the standard cruise control (CC) system, ACC keeps the car at the driver's chosen speed. Unlike regular cruise control, Adaptive Cruise Control (ACC) adjusts the vehicle's speed if it detects a slower-moving vehicle on the road, ensuring a safe distance is maintained.

Once the opposite route is clear again, the system restores the speed to the driver's desired optimal speed. This adaptive feature substantially improves driving safety and minimizes the likelihood of accidents [7].

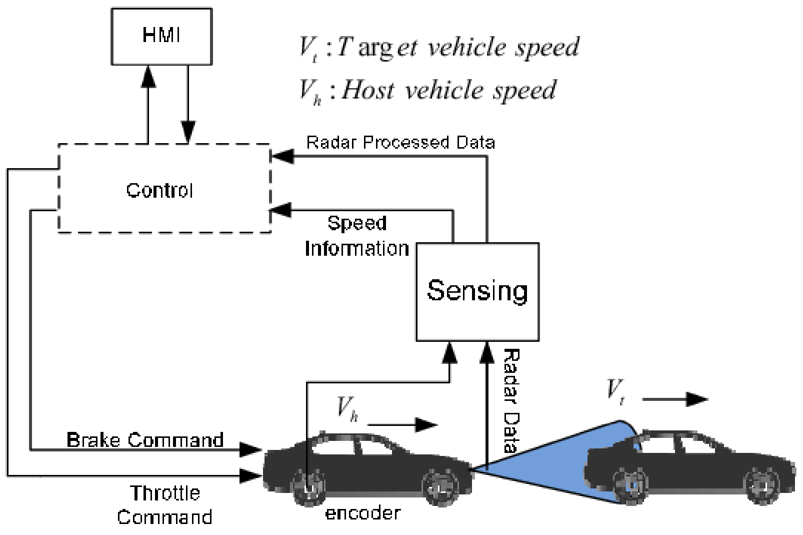

Fig. 1. Schematic of the adaptive cruise control (ACC) system [8]

The driver's decisions are always prioritized when operating the vehicle. When the gas pedal is pressed, the car's speed increases, and releasing the pedal allows the ACC system to bring the car's speed back to the desired value. The ACC system offers the advantage of maintaining a specific distance between vehicles on the highway, thereby regulating traffic flow in such conditions. The ACC system is categorized into two types: independent and dependent. In independent mode, the car functions based on its own gathered information, while in dependent mode, it relies on communication with adjacent vehicles and infrastructure. This study will focus mainly on the independent ACC system, covering topics such as technology for constructing automotive radar systems, controller design for the ACC system (including intelligent controllers like neural networks, fuzzy logic, and robust control methods such as the slip mode method, Adaptive Controller, or classic controllers), integration of the ACC system with other driver-assisted systems like the route detector system, and statistical assessment of the ACC system on driving safety and the reduction of road accidents. The primary objective of the system is to prevent accidents, detect obstacles around the car, determine its location, and then implement the necessary control commands. Tools such as radar, LIDAR, cameras, and ultrasonic sensors are utilized for obstacle detection.

3. METHOD

This study employs the deep reinforcement learning (DRL) methodology to manage the Adaptive Cruise Control (ACC) system, utilizing the robust Deep Deterministic Policy Gradient (DDPG) algorithm. Originating from the actor-critic framework, this algorithm guarantees the ACC's safe and efficient operation across diverse traffic conditions.

The section commences with an overview of reinforcement learning (RL) and subsequently explores specific algorithms within the DRL domain. It elucidates the interaction between the agent and the environment in RL, followed by an assessment of both value-based and policy-based algorithms. The discussion includes the Deep Q-learning (DQL) algorithm, which integrates neural networks with Q-learning, alongside the SARSA algorithm. Furthermore, it investigates policy-based algorithms such as Trust Region Policy Optimization (TRPO) and Proximal Policy Optimization (PPO). In conclusion, it highlights the importance of the DDPG algorithm in the context of this research.

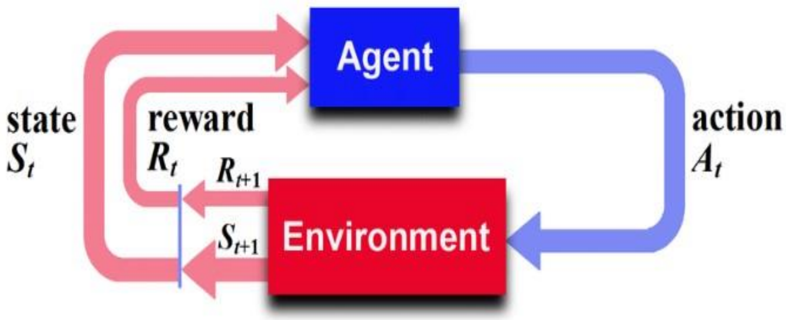

Fig. 2. Reinforcement learning (RL) model [9]

3.1. Reinforcement Learning (RL) method

The reinforcement learning (RL) methodology exemplifies the interaction between an intelligent agent and its environment. This approach is not only highly effective, but also crucial for tackling challenges related to sequential decision-making. The agent's objective is to determine the optimal series of control actions by utilizing feedback from its surroundings. Due to its characteristics of self-evaluation and continuous improvement, RL is widely applied across numerous research domains, as highlighted by citations in references [10-12]. In the realm of highway decision-making, the agent is representative of the leading vehicle, while the other vehicles constitute the environment. This interaction is aptly modeled through Markov Decision Processes (MDP), where the subsequent state variable is contingent solely upon the current state and function [13]. The Markov property is vital for resolving sequential decision-making issues in autonomous driving contexts. The corresponding Markov Decision Process (MDP) frequently collaborates with Reinforcement Learning (RL) in the form of a tuple (S, A, P, R, γ), where S and A denote the sets of states and actions, respectively. P and R are critical elements within the RL framework, representing transition and reward modeling, respectively, as illustrated in Figure 2.

In the RL approach, the current action influences both immediate and future rewards. Thus, the discount factor γ acts as a fixed coefficient to reconcile these two aspects.

To denote the total of future rewards, we define the cumulative reward R_t as:

$$\boldsymbol{R_t = \sum_t^{\infty} \gamma^t . r_t} \tag{1}$$

In order to evaluate the value of states during the process of taking actions and measuring the resultant rewards, we define two functions: the value function and the Q-function, which are described as follows.

$$\boldsymbol{V^{\pi}(s_t) = E_{\pi}[R_t | S_t, \pi]} \tag{2}$$

$$\boldsymbol{Q^{\pi}(s_t, a_t) = E_{\pi}[R_t | S_t, a_t, \pi]} \tag{3}$$

The strategy for control actions is denoted by π, the value function is represented by V, and the state-action function, commonly referred to as the Q-table, is indicated by Q. For the purpose of updating, the state-action function is typically redefined in the following manner:

$$\boldsymbol{Q^{\pi}(s_t, a_t) = E_{\pi}[r_t + \gamma\, maxQ^{\pi}(s_{t+1}, a_{t+1})]} \tag{4}$$

Ultimately, the ideal control action is established through the control policy that is derived from the state-action function, as demonstrated in the following illustration.

$$\boldsymbol{\pi\,(s_t) = arg\,max\,Q(s_t, a_t)} \tag{5}$$

Consequently, the various algorithms associated with the reinforcement learning (RL) method alter the state-action function $Q(s, a)$ through distinct mechanisms. When categorized according to their update methodologies, RL algorithms can be divided into several groups, with the two primary categories being policy-based and value-based approaches [14].

3.2. Value-based methods

To determine the best policy, the state-action function can be represented as equation (6):

$$\boldsymbol{Q_{\pi*}(s, a) = max_{\pi}(Q_{\pi}(s, a))} \tag{6}$$

The optimal policy can be expressed as equation (7):

$$\boldsymbol{\pi * (s) = arg\,max_a(Q_{\pi*}(s, a))} \tag{7}$$

Q-learning is a method employed to tackle reinforcement learning (RL) challenges, focusing on determining the optimal value-action function. This approach is characterized as off-policy and model-free. The values associated with specific states and actions are known as Q-values. In the Q-learning framework, these Q-values are modified based on the information gathered by the agent through its interactions with the environment, following the principles outlined in the Bellman optimization equation (Equation 4). To facilitate Q function learning, value-based strategies implement the ε-greedy policy. In this strategy, the agent selects a new action based on the Q-value with a probability of 1 - ε (where ε is a value ranging from 0 to 1), while opting for a random action with a probability of ε. Value-based methods focus solely on training the agent to enhance the value function or Q function, independent of the input policy.

As previously mentioned, the Deep Reinforcement Learning (DRL) methodology utilizes neural networks to estimate the Q function. In DRL-based decision-making frameworks, an agent engages with a dynamic environment by choosing a series of actions over several time steps and learns from the feedback received, specifically rewards, to optimize the total reward. The Deep Q Network (DQN) is recognized as one of the leading algorithms in the domain of deep reinforcement learning. Originally designed for playing Atari games, DQN integrates the strengths of deep learning (neural networks) and Q-learning to create an innovative value-state function. In Q-learning, the update equation is expressed as follows:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \tag{8}$$

The learning rate, denoted as α, determines the extent to which previous and current experiences from the environment are integrated. Furthermore, both the state and action are considered in a subsequent phase. Nevertheless, Q-learning faces challenges when addressing issues with a large state variable space, as it requires considerable time to develop the Q table. To overcome this limitation, the DQN approach employs a neural network to approximate the Q table, referred to as *Q*(*s*, *a*; *θ*). In DQN, the neural network takes arrays of state variables and control actions as inputs and generates the state-value function as its output. To measure the difference between the estimated and actual Q tables in DQN, a loss function is introduced as follows:

$$\boldsymbol{L(\theta) = E[\textstyle\sum_{t=1}^{N}(y_t - Q(s, a\,; \theta))^2]} \tag{9}$$

$$y_t = r_t + \gamma \max_{a'} Q(s',a';\theta')$$

In DQN, there are two distinct sets of parameters, denoted as θ and θ', which correspond to two separate networks: the prediction network and the target network. The primary function of the prediction network is to estimate the current control action, while the target network is utilized to compute the target value. Notably, the target network periodically synchronizes its parameters with those of the prediction network after a predetermined number of time steps. This synchronization process facilitates the convergence of the target Q table, thereby mitigating the gradual instability associated with the network. In DQN, the online neural network is updated through the application of gradient descent, as detailed below:

$$\nabla_\theta L(\theta) = E[(y_i - Q(s,a;\theta))\nabla_\theta Q(s,a;\theta)] \tag{11}$$

This operation effectively transforms the DQN into a value-based algorithm. States and rewards are acquired based on specific criteria, contributing to the overall learning process [15].

3.3. Policy-based methods

Value-based approaches focus on optimizing the value function (Q) and subsequently deriving the optimal policy from this function. In contrast, policy-based methods typically aim to directly enhance a policy by learning from the cumulative rewards obtained, as represented in equation (12).

$$\boldsymbol{max\,J(\theta) = maxv_{\pi_\theta} = maxE_{\pi_\theta}[G_t|s_t = s]} \tag{12}$$

Here, *θ* represents the policy parameters, which are updated using the policy gradient as described in equation (13).

$$\boldsymbol{\nabla J(\theta) = E_\pi|G_t \nabla ln\pi_\theta(a_t|s_t)} \tag{13}$$

In policy-based methods, updates to parameters enhance the probability of actions executed by the agent in a logarithmic manner. The objective of policy gradient methods is to optimize efficiency by modifying the policy according to the steepest ascent. Prominent algorithms employed in policy-based approaches include TRPO and PPO. The TRPO algorithm adjusts network parameters while adhering to a policy update constraint that employs the Kullback-Leibler (KL) divergence, thereby ensuring that the revised policy remains closely aligned with the existing policy. In contrast, the PPO algorithm introduces a constraint within the optimization objective in the form of a distinct surrogate objective function. In the case of PPO, if the newly proposed policy deviates significantly from the prior one, the objective function incurs a penalty, leading to a negative reward. Although the Q-learning algorithm (DQN) is proficient in addressing challenges associated with high-dimensional state spaces, it faces limitations when dealing with discrete action spaces. Nonetheless, numerous real-world problems, particularly those involving physical control tasks, require continuous action spaces, rendering the discrete nature of DQN a limitation.

3.4. Deep Deterministic Policy Gradient (DDPG) algorithm

The Deep Deterministic Policy Gradient (DDPG) approach addresses this limitation by functioning as an off-policy algorithm suitable for continuous action spaces. DDPG consists of two main elements: the enhancement of the Q function executed by the critic network and the development of the policy carried out by the actor policy network. The Bellman equation relevant to the DDPG algorithm is represented by equation (14).

$$\boldsymbol{L(\emptyset,\theta) = E_D\left[(Q_{\emptyset}(s,a) - \left(r + \gamma\,(1-d) max Q_{\emptyset}(\acute{s},\acute{a})\right))^2\right]} \tag{14}$$

In the DDPG algorithm, the critic network is denoted as Q(s, a) and is characterized by parameters $\emptyset$. As indicated in equation 14, the policy network aims to develop a deterministic policy $u_{\varphi}(s)$ that enables the agent to select actions that optimize the Q-function $Q*(s, a)$. To mitigate the issue of network instability inherent in the DDPG algorithm, a replay buffer and target networks are employed. These target networks, which include a target critic network and a target policy network, share the same architecture as their primary counterparts; however, their parameters φ are updated at a slower rate. The parameters of the target networks are adjusted through the Polyak averaging method, which facilitates a gradual alignment with the main networks, thus enhancing overall stability.

The operation of the DDPG algorithm is illustrated in Figure 3.

4. SIMULATION

In this study, the MATLAB R2022a software was utilized to model the control of the Adaptive Cruise Control (ACC) system through the Deep Reinforcement Learning (DRL) approach. The reinforcement learning framework developed herein incorporates the longitudinal dynamics of both the lead and target vehicles. The primary objective is to regulate the speed of the lead vehicle to a specified target by managing acceleration and longitudinal braking via the ACC, while ensuring a safe distance is preserved from the target vehicle. The parameters for the action space, state space, and reward function are delineated as follows.

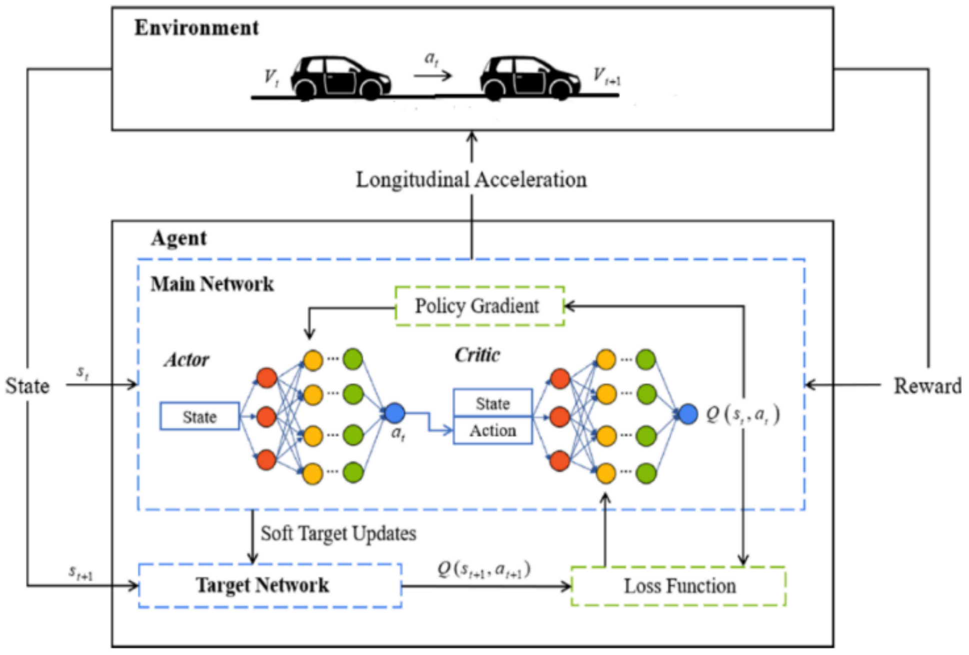

Fig. 3. DDPG algorithm mechanism in the adaptive cruise control (ACC) system [16]

4.1. Action-space

The action space, representing all potential actions that the learner can execute in the environment, encompasses the operational parameters accessible to the agent. In this study, the action space comprises the vehicle's acceleration, which spans from -3 to 2 meters per second and is controlled by the agent.

The reference speed for the preceding vehicle is determined as follows: if the relative distance is below the safe distance, the preceding vehicle maintains the minimum speed between its initial speed and the speed set by the driver. If the relative distance exceeds the safe distance, the preceding vehicle aligns its speed with that set by the driver.

The safe distance is computed as a linear function of the longitudinal velocity of the preceding vehicle, expressed as:

$$\boldsymbol{D_{safe} = V * t_{gap} + D_{defult}} \tag{15}$$

The safety distance, calculated from equation 15, determines the reference tracking speed for the preceding vehicle.

4.2. State-space

In reinforcement learning problems, the state represents the environment. Put differently, the state space encompasses all feasible states within a particular environment. In the current investigation, the state space encompasses the following: The speed error value $e = V_{ref} - V_{ego}$, its integral, and the longitudinal velocity of the preceding vehicle or V_{ego}. The simulation concludes under two conditions, either when the longitudinal velocity of the preceding vehicle is less than zero or when the relative distance between the original vehicle and the preceding vehicle is less than zero.

4.3. Reward function

The reward function plays an essential role in the reinforcement learning (RL) framework, as it establishes the goals and offers feedback to the agent. Nevertheless, creating a well-structured reward function that is in harmony with the objectives of the policy gradient can be a complex task that necessitates thorough deliberation.
This article defines the reward function as follows:

$$\boldsymbol{r_t = 0.1 * e_{2t} + u_{2t-1} + M_t} \tag{16}$$

where:

- u_{t-1} is the input of the previous time stage control;
- e_{2t} is the value of the velocity error;
- Mt is a binary indicator, taking a value between 0 and 1000.

The simulation described earlier is performed utilizing MATLAB and Simulink software, specifically version 2022. Within the Simulink environment, a block diagram representing the problem is developed, as demonstrated in Figure 4.

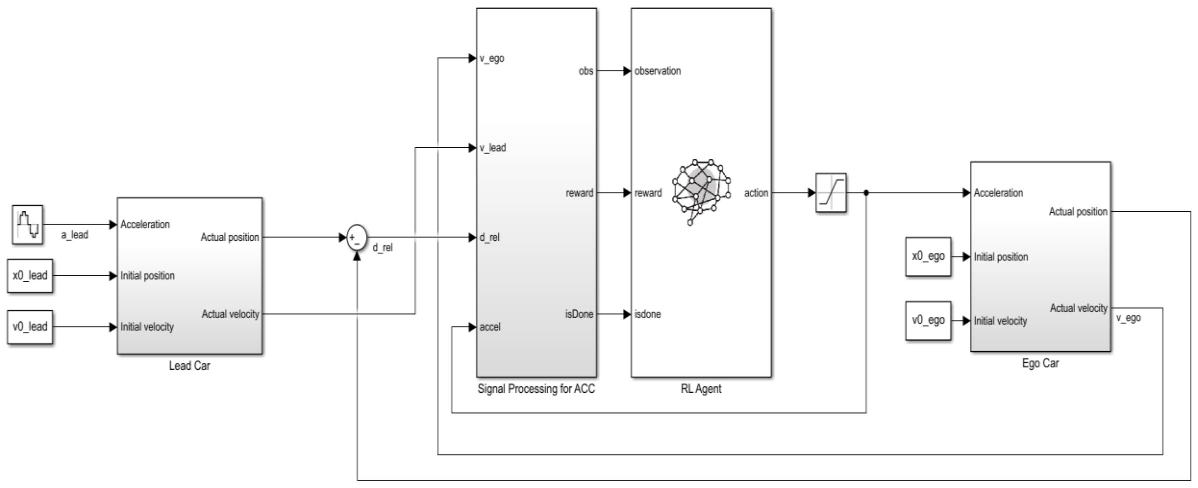

Fig. 4. Block diagram of DDPG algorithm for ACC system

Currently, we are evaluating the efficacy of DDPG and DQN, both utilizing a hierarchical control framework. However, there are distinctions in the upper levels. The default parameters for both DDPG and DQN are the same. Figures 5 and 6 illustrate the average rewards and episode rewards achieved by the DDPG and DQN agents, respectively. As shown in Figures 5 and 6, a greater reward indicates driving in the desired lane with more effective maneuvering.

The DDPG method exhibits enhanced training stability and accelerated learning compared to the DQN approach, resulting in consistently greater rewards after around 1000 episodes. The key element that underpins the superiority of DDPG is its implementation of the actor-critic network. This type of network is adept at evaluating the value of the selected action at each step, enabling the ego vehicle to swiftly recognize a more efficient decision-making strategy.

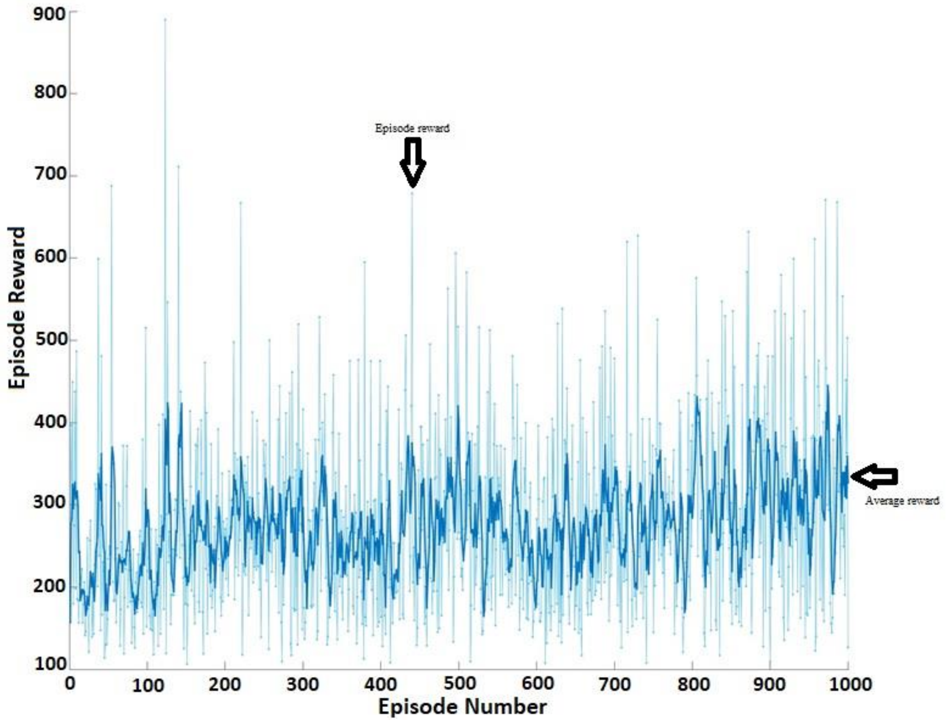

Fig 5. DDPG agent rewards

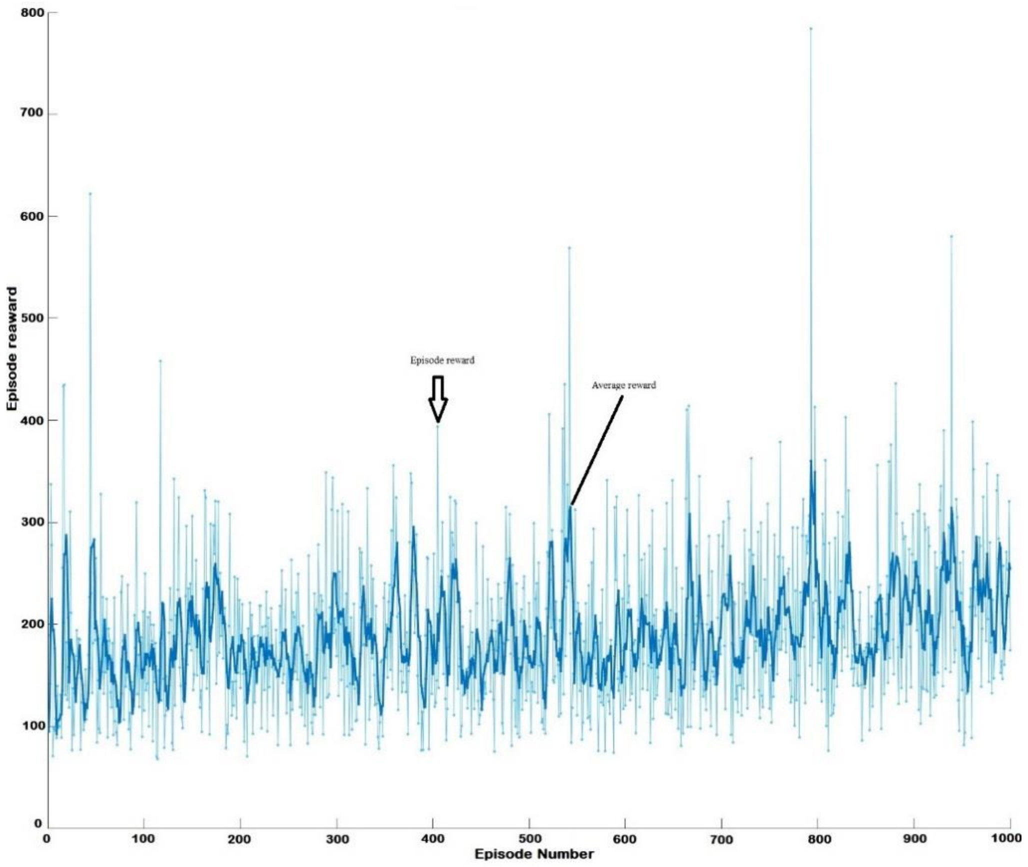

Fig 6. DQN agent rewards

5. RESULTS AND DISCUSSION

Figures 7, 8, and 9 depict the acceleration, distance, and velocity of the ego vehicle equipped with the ACC system for both DDPG and DQN agents, respectively.

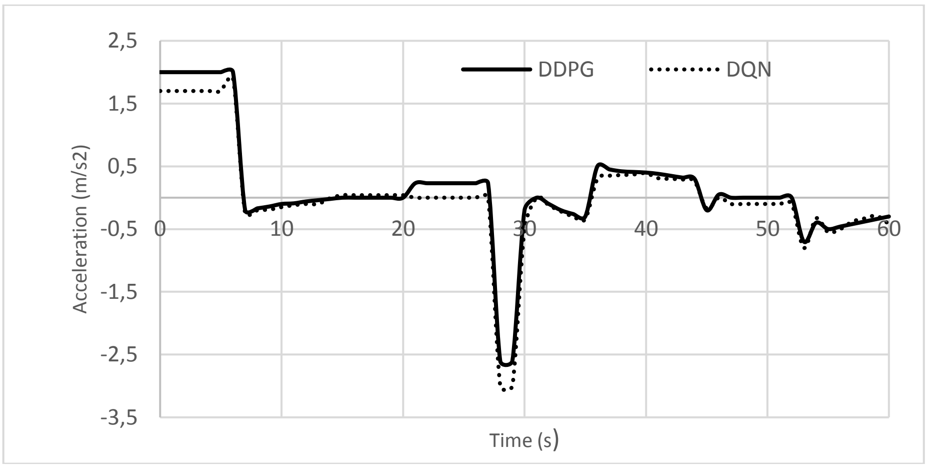

Fig. 7. Acceleration of the ego vehicle (with adaptive cruise control (ACC) system)

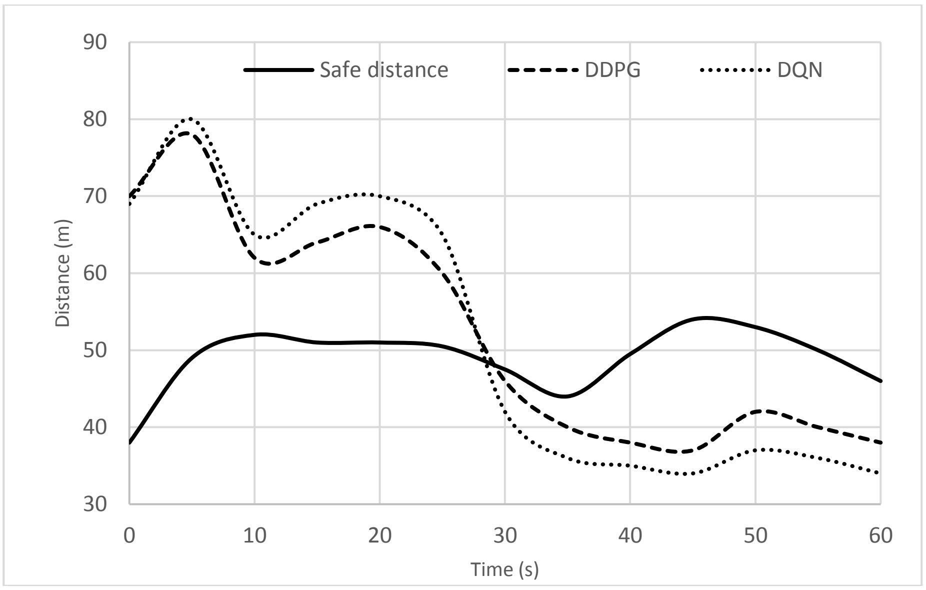

Fig. 8. Distance traveled by the ego vehicle (with adaptive cruise control (ACC) system)

These findings indicate that the DDPG algorithm, utilized in the ACC system, has exhibited robust performance in tracking both the safe distance and the velocity of the lead vehicle (v_{lead}). In essence, the DDPG algorithm's capability to precisely monitor the safe distance between the ego vehicle and the lead vehicle underscores its effectiveness in sustaining a safer driving environment compared to DQN. This is critical for averting collisions and upholding overall road safety.

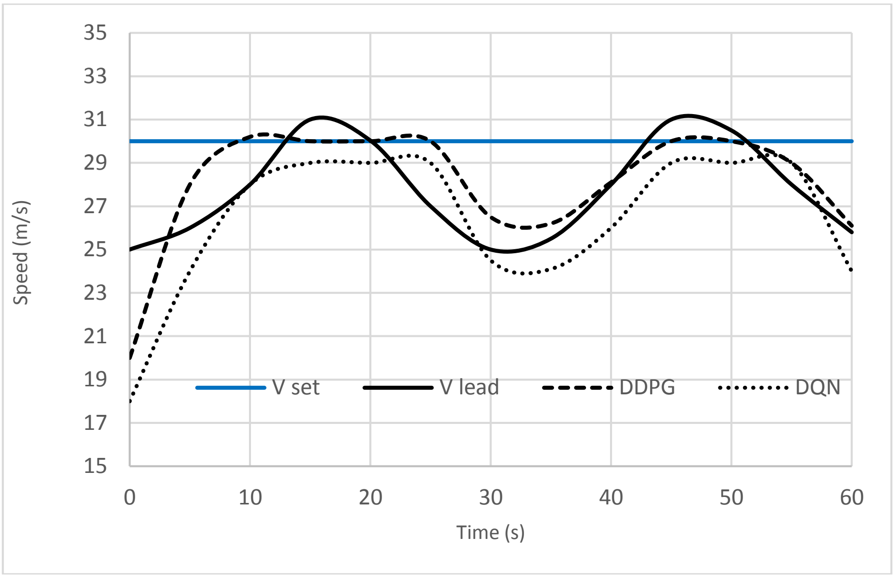

Fig. 9. Speed of target and ego vehicles (with adaptive cruise control (ACC) system)

Moreover, by accurately tracking the velocity of the lead vehicle, the DDPG algorithm empowers the ego vehicle to uphold an appropriate following distance while adapting its speed accordingly. This not only improves safety but also fosters smoother and more efficient traffic flow.

Figures 8 and 9 illustrate the simulation outcomes spanning 60 seconds, with the target vehicle positioned 70 meters ahead of the preceding vehicle. Initially, from the start until 28 seconds, the relative distance surpasses the safe distance, prompting adjustments to the acceleration of the preceding vehicle (indicated by midpoints). Acceleration increases to enhance speed and achieve the set speed, resulting in a peak. Subsequently, between 28 and 60 seconds, the relative distance drops below the safe distance, prompting the preceding car to maintain the minimum steering speed and set speed. Additionally, from 28 to 36 seconds, the target vehicle's speed is lower than the set speed (midpoints), leading the preceding vehicle to decelerate and synchronize with the original vehicle's speed. Acceleration becomes negative during this period (as depicted in the chart above). Between 36 and 60 seconds, the preceding vehicle adjusts its acceleration to closely follow the reference speed (depicted in the middle graph). During this period, the preceding vehicle matches the set speed from 43 to 52 seconds and tracks the speed of the target vehicle between 36 to 43 seconds and 52 to 60 seconds. This accomplishment underscores the capability of the DDPG algorithm to effectively regulate the speed and distance of the ego vehicle to the lead vehicle, thus showcasing its suitability for real-world ACC applications.

6. CONCLUSION

According to the results and evaluations in this research, incorporating ACC in the automotive industry leads to lower fuel usage. The reduction in energy wastage in inertial components is due to the decrease in speed fluctuations of vehicles. Additionally, ACC systems improve upon traditional CC systems' restrictions and enhance safety by autonomously managing the vehicle in crucial situations. This article presents the DDPG algorithm, a learning method that includes actor and critic networks. The purpose of the critic network is to enhance the policy set by the actor network by estimating upcoming total reward values. The study utilized the DDPG method to address the ACC issue. The findings show that the DDPG algorithm is more successful than DQN in controlling the ACC system, highlighting its effectiveness and efficiency. As a result, it can be concluded that the suggested DDPG algorithm is both feasible and successful in addressing ACC problems than previous study like [4].

References

1. Li J., H. Zhang, H. Zhang, Y. Yu. 2021. "Deep reinforcement learning for adaptive cruise control in mixed traffic scenarios". *Transportation Research Part C: Emerging Technologies* 124: 102908.
2. Huang W., Y. Qiu, L. Zhang, Y. Liu. 2022. "Reinforcement learning-based adaptive cruise control with safe exploration strategies". *IEEE Robotics and Automation Letters* 7(2): 4562-4569.
3. Wang L., Z. Chen, Y. Zheng. 2021. "Improving safety and efficiency in adaptive cruise control through deep reinforcement learning". *Transportation Research Part B: Methodological* 148: 193-215.
4. Yang T., Y. Liu, Y. Li. 2022. "Deep reinforcement learning for adaptive cruise control under uncertain traffic conditions". *IEEE Access* 10: 5678-5689.
5. Zhu Q., R. Chen, Z. Wang, L. Zhang. 2021. "Deep reinforcement learning for adaptive cruise control with model-free uncertainty estimation". *IEEE Transactions on Cybernetics* 51(5): 2341-2350.
6. Park J., J. Lee, S. Kim. 2022. "Enhancing adaptive cruise control using deep reinforcement learning with temporal difference learning". *Expert Systems with Applications* 189: 116028.
7. Liu T., B. Tian, Y. Ai, L. Chen, F. Liu, D. Cao. 2019. "Dynamic states prediction in autonomous vehicles: Comparison of three different methods". *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*: 3750-3755.
8. Sutton R.S., A.G. Barto. 2018. *Reinforcement Learning: An Introduction* (2nd ed.). Cambridge, MA, USA: MIT Press.
9. Li G., L. Yang, S. Li, X. Luo, X. Qu, G. Paul. 2022. "Human-like decision-making of artificial drivers in intelligent transportation systems: An end-to-end driving behavior prediction approach". *IEEE Intelligent Transportation Systems Magazine* 14(1): 45-56. DOI: https://doi.org/10.1109/MITS.2022.3085986.
10. Zhang Q., J. Lin, Q. Sha, B. He, G. Li. 2020. "Deep interactive reinforcement learning for path following of autonomous underwater vehicle". *IEEE Access* 8: 24258-24268.
11. Zhang D., N.L. Azad, S. Fischmeister, S. Marksteiner. 2023. "Zeroth-Order Optimization Attacks on Deep Reinforcement Learning-Based Lane Changing Algorithms for Autonomous Vehicles". *Proceedings of the International Conference on Informatics in Control, Automation and Robotics* 1: 665-673. DOI: 10.5220/0012187700003543.
12. Lv K., X. Pei, C. Chen, J. Xu. 2022. "A safe and efficient lane change decision-making strategy of autonomous driving based on deep reinforcement learning". *Mathematics* 10(9): article 1551: 1-14.
14. Zhang, Q., J. Lin, Q. Sha, B. He, G. Li. 2020. "Deep interactive reinforcement learning for path following of autonomous underwater vehicle". *IEEE Access* 8: 24258-24268.

15. Zhang D., N.L. Azad, S. Fischmeister, S. Marksteiner. 2023. "Zeroth-Order Optimization Attacks on Deep Reinforcement Learning-Based Lane Changing Algorithms for Autonomous Vehicles. *Proceedings of the International Conference on Informatics in Control, Automation and Robotics* 1: 665-673. DOI: 10.5220/0012187700003543.
16. Research Gate. Available at: https://www.researchgate.net/figure/The-structure-of-the-DDPG model-6_fig5_356879301.

Received 06.09.2024; accepted in revised form 07.11.2024