Journal of Al-Azhar University Engineering Sector

Vol. 19, No. 73, October 2024, 1349 - 1368

ADAPTIVE CRUISE CONTROL WITH LANE KEEPING ASSIST USING REINFORCEMENT LEARNING

Taghreed Mamdouh*, Ayman Elshenawy, Hussein Tantawy

Systems and Computers Engineering Department, Faculty of Engineering, Al-Azhar University, Nasr City, Cairo, Egypt.

* **Correspondence**: taghreedmamdouh19@gmail.com

Citation:

T. Mamdouh, A. Elshenawy, H. Tantawy" Adaptive Cruise Control with Lane Keeping Assist Using Reinforcement Learning", Journal of Al-Azhar University Engineering Sector, vol. 19, No. 73, 2024, 1349-1368.

Received: 20 June 2024-06-

Revised: 22 July 2024

Accepted: 31 July 2024

DOI: 10.21608/auej.2024.298711.1676

ABSTRACT

Adaptive Cruise Control (ACC) is an autonomous driving technology that enables vehicles to maintain a preset velocity and keep a safe distance from the lead vehicle by modifying the longitudinal acceleration of the ego vehicle. Reinforcement learning (RL) is a machine learning approach that learns by interacting with environments through trial and error. It receives rewards from the environment to evaluate its actions and improve its policies through learning algorithms. In this paper, an intelligent ACC system is proposed to enhance the system's overall performance by utilizing two main sub-systems. First, an ACC is implemented using the Twin Delayed Deep Deterministic (TD3) policy gradient algorithm with a continuous action space to control the acceleration of a vehicle. Second, a Lane Keeping Assist (LKA)is implemented using the Deep Q-Network (DQN) algorithm with a discrete action space for steering angle control. The system architectures and vehicle dynamics of the two sub-systems are modelled and simulated using Simulink. The TD3 and DQN algorithms are trained to perform ACC with LKA through cooperative behavior. Many experiments are carried out to evaluate the performance of the proposed system. The obtained results demonstrate the system's ability to follow a preset velocity, keep a safe distance from a lead vehicle, maintain the ego vehicle centered in its lane, and mitigate the risk of collision that may arise from lane changes.

KEYWORDS : Advanced driver assistance systems, Adaptive Cruise Control, , Lane keeping Assist , Reinforcement Learning.

نظام تثبيت السرعة التكيفي مع مساعد الحفاظ على المسار باستخدام التعلم المعزز

تغريد ممدوح *، أيمن الشناوى، حسين طنطاوى

قسم هندسة النظم والحاسبات ، كلية الهندسة، جامعة الأزهر، القاهرة ، مصر

taghreedmamdouh19@gmail.com : *البريد الاليكتروني للباحث الرئيسي

الملخص

نظام تثبيت السرعة التكيفي (ACC) هوعبارة عن تقنية قيادة ذاتية تمكن المركبات من الحفاظ على سرعة محددة مسبقًا والحفاظ على مسافة آمنة من السيارة الأمامية عن طريق تعديل التسارع الطولي للمركبة الخلفية. التعلم المعزز (RL) هو نهج للتعلم الآلي فيه يتم التعلم من خلال التفاعل مع البيئات من خلال التجربة والخطأ. يتلقى مكافآت من البيئة لتقييم أفعاله وتحسين سياساته من خلال خوارزميات التعلم. في هذا البحث، تم اقتراح نظام (ACC) ذكي لتعزيز الأداء العام للنظام من خلال استخدام نظامين فرعيين رئيسيين. أولاً، يتم

تنفيذ نظام التحكم في السرعة(ACC) باستخدام (TD3) مع مساحة عمل مستمرة للتحكم في تسارع السيارة. ثانيًا، يتم تنفيذ مساعد الحفاظ على المسار(LKA) باستخدام خوارزمية (DQN) مع مساحة عمل منفصلة للتحكم في زاوية توجيه السيارة. تم تصميم ومحاكاة بنيات النظام وديناميكيات المركبات للنظامين الفرعيين باستخدام (Simulink) .تم تدريب خوارزميات (TD3) و(DQN) على أداء (ACC) مع (LKA) من خلال سلوك تعاوني. تم إجراء العديد من التجارب لتقييم أداء النظام المقترح. توضح النتائج التي تم الحصول عليها قدرة النظام على متابعة سرعة محددة مسبقًا، والحفاظ على مسافة آمنة من السيارة الأمامية، والحفاظ على المركبة الخلفية متمركزة في حارتها وتخفيف مخاطر الاصطدام التي قد تنشأ عن تغيير المسار.

الكلمات المفتاحية : أنظمة مساعدة السائق المتقدمة، نظام تثبيت السرعة التكيفي، مساعد الحفاظ على المسار، التعلم المعزز.

1. INTRODUCTION

Research has focused on Advanced Driver Assistance Systems (ADAS) in recent years to decrease driving errors and increase traffic efficiency. ADAS systems are designed with safe and effective human-machine interfaces to aid the driver in vehicle driving and are expected to enhance the safety of both the vehicle and the road. ADAS enhance driver safety by adding features like lane departure alert, ACC, and self-parking systems, which can reduce driver fatigue and make driving more enjoyable. The widespread adoption of these technologies has demonstrably decreased the number of accidents on the road ADAS collect data about the driving environment via a variety of sensors, such as cameras, radar, and lidar. This data is then processed by software, which uses algorithms to determine appropriate action. Then, the system communicates with the vehicle's actuators, such as brakes or steering wheel, to take corrective action [1].

In traditional Cruise Control (CC) there is no automatic way to decrease the ego vehicle velocity when the lead vehicle in front of it; manual control from the driver is required to prevent collision [2]. ACC is selected as a solution for the aforementioned problem. ACC is used in various vehicles to adjust the velocity of the ego vehicle at a constant preset velocity chosen by the driver to keep a safe distance from lead vehicles automatically [3]. It uses sensors to measure both the relative distance and the relative velocity between the ego and lead vehicles in the same lane. The development of the ACC system depends on the advancement of sensors and algorithms. Sensors may include lidar, radar, or camera sensors, but radar sensors are considered the best option because they are not affected by weather conditions such as fog or dust. This makes them ideal for use in areas with harsh weather conditions [4].

As shown in **Fig. 1**, the ACC system functions in two different scenarios: (i) Speed control scenario, in which the ego vehicle moves at a preset velocity chosen by the driver; and (ii) Spacing control scenario, in which the ego vehicle maintains a safe distance from the lead vehicle. Generally, the ACC system switches between these two scenarios based on measurements from the radar. As an illustration, if the lead vehicle is in close proximity, the ACC system chooses the speed control scenario, making the ego vehicle move at a constant speed chosen by the driver if it maintains a safe distance [5].

Over the decades, researchers have been continuously working on the development of ACC to address traffic congestion issues. In [6], an ACC system is implemented using stop-and-go maneuvers for vehicle modeling with a Proportional, Integral, and Differential (PID) controller . In [7], the authors presented an ACC system using PID and fine-tuned it using differential evolution. The author of [8] suggested an ACC system that could utilize fuzzy logic to identify speed signs. A comparison of fuzzy algorithms and artificial neural networks (ANNs) was used in [9] to develop an ACC system. The authors in [10] presented the system modelling and control strategy for an ACC system that uses linear quadratic regulation as an optimal control method to follow a lead vehicle. Model predictive controller for designing an ACC system was presented in [11]. Using the supervised actor-critic RL technique, the

author of [12] presented a novel ACC system that took into consideration variations in road conditions, such as wet or wintry conditions.

In [13], a human framework for a vehicle following strategy based on deep RL for multiple driving scenarios was proposed. [14] proposed a novel approach using ANNs and RL to describe ACC. The safety-aware intelligent ACC system was presented by the author in [15] in order to carry out an efficient evaluation of driving safety by adjusting the safety model's parameters in adaptation to changing traffic conditions using RL.

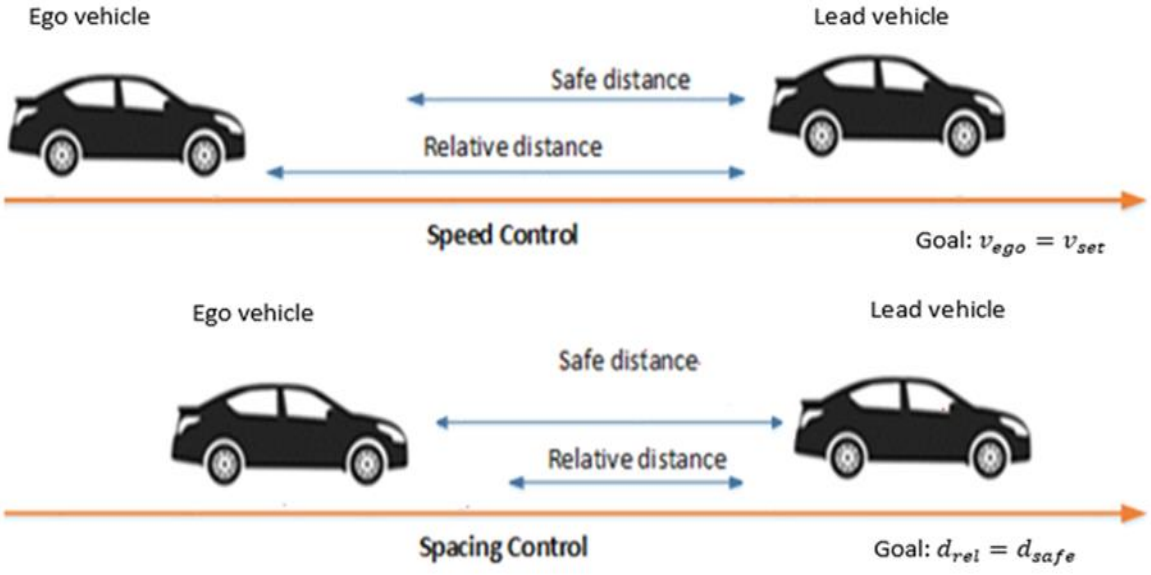


Fig. 1. shows ACC system operation modes

The author of [16] suggested an ACC technique based on asynchronous supervised deep RL to improve the driver's perception during vehicle following and braking. To create a driver-like cruise effect, actual driver cruising data was included in the supervision network. This helped to direct the control strategy, which uses the actor-critic network as a basic control unit. In [17], the author proposed an ACC to balance followability and comfortability through RL. In [18], the author proposed using the Twin Delayed Deep Deterministic Policy Gradient (TD3) method to design an ACC.

However, while previous researchers have achieved sufficient performance in following a lead vehicle, they have not adequately considered the safety factor that reduces vehicle crashes during lane changes. To mitigate this hazardous behavior, the ACC systems should be enhanced to consider not only followability but also safety. This can be accomplished by ensuring that the vehicle follows the lane's centerline whether the road is straight or curved.This is the main objective of this paper.

In this paper, an intelligent ACC is suggested to improve the overall performance of the system by utilizing two main sub-systems. The first sub-system is implemented using the TD3 policy gradient algorithm with a continuous action space to control vehicle acceleration. The second sub-system is a LKA implemented using the Deep Q-network (DQN) algorithm with a discrete action space for steering angle control. The ACC system architecture and vehicle dynamics are modeled and simulated using Simulink. The TD3 and DQN algorithms are trained to perform ACC with LKA via cooperative behavior and achieve satisfactory results.

This paper's main contribution can be summarized as follow: an intelligent ACC system with LKA is proposed, modeled, designed, and implemented using the TD3 and DQN algorithms, respectively. The system’s performance is evaluated and discussed.

The paper's structure is as follows: Section 2 addresses the formulation of the problem. The main control algorithms utilized in the system's development are shown in section 3. The proposed ACC with LKA is presented in section 4. Section 5 lists simulation results.Section 6 discusses the safety and robustness. Section 7 applies the proposed system in real world . Section 8 Compares traditional vs. proposed ACC. Section 9 concludes the paper.

2. FORMULATION OF The PROPLEM

The ACC problem can be described as follows: given a dynamic environment that contains the ego and the lead vehicles, respectively. ACC is used to control an ego vehicle and maintain a temporal interval (constant time gap) ($t_g = 1.4s$) between the two vehicles .

Temporal interval is defined as a constant time gap between the two vehicles is expressed as a reaction time that the driver needed to react in normal condition. Driver reaction times often fall into one of two ranges: 1.2 to 2.0 seconds. An average of 1.4 seconds is frequently employed in engineering models to examine driver behavior and traffic flow.

Additionally, ACC must keep the ego vehicle centered in its lane, regardless of whether the road is straight or curved. The dynamic environment of both the ego and lead vehicles can be modeled as follows:

2.1 Lead vehicle modelling

The lead vehicle's longitudinal dynamics can be calculated using its actual velocity v_{lead} and its actual position x_{lead} using a first order system described in equations (1) and (2).

$$v_{lead} = a_{lead} \times \frac{1}{s(0.6s+1)} + v_{lead0} \quad (1)$$

$$x_{lead} = \frac{v_{lead}}{s} + x_{lead0} \quad (2)$$

Where, a_{lead} , v_{lead0} , and x_{lead0} are the acceleration, initial velocity, and initial position of the lead vehicle respectively [18].

2.2 Ego vehicle modelling

A 3DOF model is used to describe the dynamics of the ego vehicle and determine its yaw, lateral, and longitudinal motions. The vehicle dynamics output (like longitudinal velocity V_x and lateral velocity V_y) are based on body fixed coordinates. To obtain the trajectory traversed by the vehicle, the following formulas are used to translate the body fixed coordinates into global coordinates in **Fig. 2**: (The two equations (3) ,and (4)are used to analyze the forces affecting the vehicles).

$$X' = V_x \cos(\Psi) - V_y \sin(\Psi) \quad (3)$$

$$Y' = V_x \sin(\Psi) + V_y \cos(\Psi) \quad (4)$$

Where is Ψ is the yaw angle.

The vehicle influenced by longitudinal traction force which is the force generated by the motor to propel the vehicle forward and lateral traction force which is the force due to tire friction affecting the vehicle's lateral motion. and the moment around the vehicle's vertical center of gravity.

Equations (5) , (6) and (7) are used to calculate these motions.

Assuming a small front wheel steering angle, the tire's lateral force is equal to the product of the tire's slip angle and cornering stiffness. By setting $v_y = y'$, the differential equation for dynamics can be obtained as follows:

$$v_y'' = \left(\frac{2\,c_{f+}2c_r}{m\,v_x} * y'\right) - \left(\frac{2c_f\,l_f - 2\,c_r\,l_r}{m\,v_x} - v_x\right) * \dot{\Psi} + \frac{2c_f}{m} * \delta) \tag{5}$$

$$\Psi^{\cdot\cdot} = (\frac{-2c_f\,l_f - 2c_r\,l_r}{I_z} * y' + \left(\frac{-2c_f l_f^2 + 2\,c_r\,l_r^2}{I_z\,v_x}\right) * \dot{\Psi} + \frac{2c_f\,l_f}{I_z} * \delta) \tag{6}$$

$$v_x' = v_y\,\dot{\Psi} + a_x \tag{7}$$

Where m is the ego vehicle mass, c_f and c_r are the cornering stiffness of the front and rear tires of the vehicle $(\mathrm{N/rad})$,and $l_f$and l_r are the longitudinal distances from the center of gravity to the front and rear tires(m), , v_x is the vehicle's longitudinal velocity (m/s) , I_z is the vehicle's yaw moment of inertia(mNs^2) around the Z axis, a_x is the desired acceleration. Ψ represents the vehicle's yaw angle, $\dot{\Psi}$ represents its yaw rate and δ represents its front steering angle.

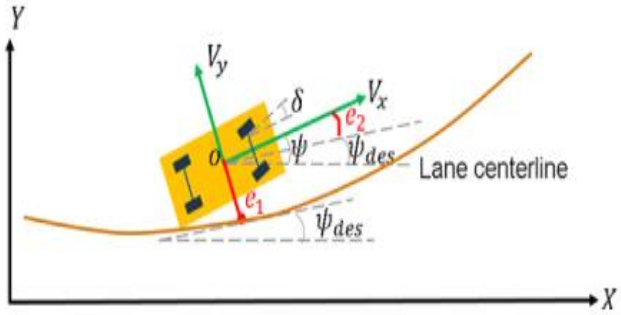


Fig. 2. Ego Vehicle Steering Errors

By regulating the front steering angle δ, the LKA function helps the ego vehicle stay in its lane and follow the curving route. As seen in **Fig. 2**, the ego vehicle will make mistakes while attempting to maintain the desired path. The yaw angle error e_2 and the lateral deviation error e_1 are driven to zero in order to achieve this goal [19].

Dynamics for lateral deviation:

$$e_1' = v_x e_2 + v_y \tag{8}$$

Where $\int e_1' = e_1 \quad lateral\ deviation$

Dynamics for relative yaw angle:

$$e_2 = \Psi - \Psi_{des} \tag{9}$$

Where the vehicle's desired yaw angle rate is shown by $\Psi_{des} = v_x/R$ (R refers to the road curvature's radius). $e_2' = \dot{\Psi}$ -$_Q$ v_x Where $\int e_2' = e_2 \quad relative\ yaw\ angle$ These equations can be used to depict the ego vehicle's longitudinal control in order to achieve ACC:

$$\mathrm{a}' = \frac{1}{\tau} * (\mathrm{a_x}) - \mathrm{a} \tag{10}$$

Where a is the actual acceleration of the vehicle, τ is the temporal interval and a_x is the desired acceleration of the vehicle.

$$x_{ego} = \frac{v_x}{s} + x_{ego0} \quad (11)$$

where x_{ego0} is the ego vehicle's starting position and x_{ego} is the ego vehicle's position at any given time. The required gap between the two vehicles is known as the safe distance (d_{safe}). The tracking velocity of the ego vehicle determines the safe distance, taking into account a default distance (d_{def}) in the process:

$$d_{safe} = v_x * t_g + d_{def} \quad (12)$$

Every second, the system calculates the safe distance between the lead and ego vehicles. If the relative distance is less than the safe distance, the ego vehicle follows the lower of the lead vehicle's velocity and the driver-set velocity. In this scenario, the ego vehicle maintains a specific distance from the lead vehicle. If the relative distance $d_{rel}$is more than the safe distance, the ego vehicle follows the driver-set velocity [20].

$$d_{rel} = x_{lead} - x_{ego} \quad (13)$$

Equation(13) calculates the relative distance d_{rel} between the lead vehicle and the ego vehicle by subtracting the position of the ego vehicle x_{ego} from the position of the lead vehicle x_{lead} .

$$d_{error} = d_{safe} - d_{rel} \quad (14)$$

Equation(14) calculates the distance error d_{error} by subtracting the relative distance d_{rel} from the safe distance d_{safe} .

3. CONTROL ALGORITHMS

3.1. Reinforcement learning (RL)

RL is an approach to machine learning that utilizes stored experiences to learn in dynamic environments. During training, software agents and the environment perform trial-and-error interactions to collect their experiences. The RL agent develops the ability to behave in an environment to maximize its reward. The environment and the RL agent interact, as shown in **Fig. 3,** by interacting with the environment through actions and receiving observations (states)and rewards in return. A scalar function that measures the agent's performance in finishing the task acts as the reward. The agent, which consists of a policy and a learning algorithm, is the core component of the learning process [21].

The way the agent interacts is by choosing actions according to the policy and the environment's observations. Applied to Deep Neural Networks (DNNs) or lookup tables, the policy acts as an adjustable parameter approximation for functions. By continually adjusting the policy parameters in response to observations, actions, and rewards, the learning algorithm determines the best policy of action to maximize reward during the task. During training, the agent stores one or more parameterized function approximators, depending on the learning method employed. The RL agent can make use of approximators as actors, critics, or both.

RL agents can be classified into two categories based on the approximator that is being used:

i. Value-based agents: they choose actions based only on the perceptions of their critics and an indirect policy representation. They employ approximators to depict a value function or Q-value function. Despite their strong performance in discrete action spaces, they can become computationally expensive for continuous action spaces.
ii. Policy-based agents: they choose actions based only on the perceptions of their actors and use direct policy representation.

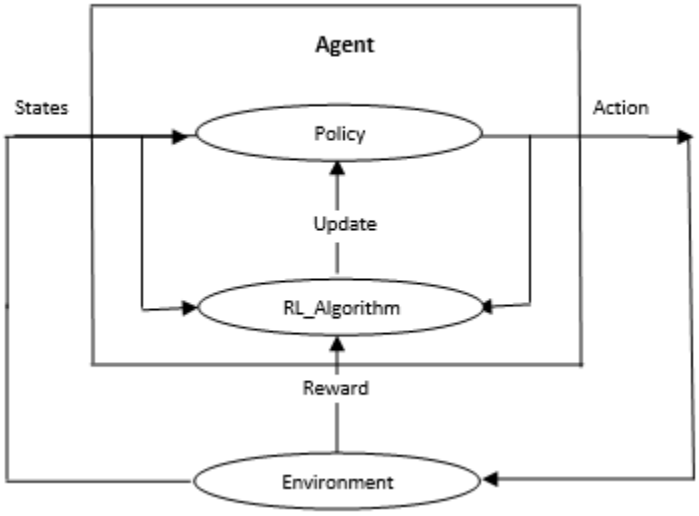


Fig. 3. Reinforcement learning framework

The policy may be stochastic or deterministic. These agents are capable of handling continuous action spaces and are simpler., although the training process may converge on local minima and be sensitive to noisy measurements. Actor-critic agents are agents that work with both actors and critics. Instead of using the reward instantly, the actor receives feedback from the critic throughout training to determine the optimal strategy for action. Simultaneously, the value function is acquired by the critic from the rewards in order to critique the actor properly. These agents are capable of operating in continuous and discrete action spaces [22].

3.2. Overview of Deep Q Network (DQN) algorithm

Algorithm DQN is a successor to the Q-learning algorithm, combining Q-learning with DNNs to enable agents to learn the optimal policy in complex environments. Q-learning is a tabular method that uses a Q-table to store state and action spaces, working well with small state space environments. However, creating and updating a Q-table for large environments is not efficient. To address this limitation, function approximators like DNNs are used to map state and action pairs to a Q-value.

Two neural networks, the Q-network and the target network, along with an element known as experience replay memory, make up the DQN. **Fig. 4 shows** The DQN Framework . The agent trained to generate the optimum state-action value is the Q-network. By making the target network and the Q-network the same, the optimization's stability is increased. Experience replay gathers previous experiences and interacts with the environment to produce training data. From the current state, it chooses an ε-greedy action, performs it in the environment, and receives the next state and reward. This observation is stored as a training data sample. Based on a mini-batch of experiences chosen randomly from the buffer, the agent updates the Q-network. DQN agents can be used in environments with discrete action spaces and continuous or discrete observations [23] . Listing 1 shows a pseudo code for the DQN algorithm.

The following is an illustrative example of how a DQN algorithm can be used for LKA in a vehicle. The LKA system recognizes when the vehicle shifts out of its lane and automatically modifies the steering to get it back into compliance whether the road is straight or curved. It uses sensors to monitor the vehicle's position relative to lane markings and provides corrective steering input if the vehicle begins to drift out of its lane without signaling. DQN is used to help the vehicle stay in the center of its

lane safely and efficiently. The vehicle starts in a random position within the lane, e.g., slightly to the left with a small yaw angle towards the right. By observing the current state, the system selects the most suitable action using DQN. The chosen action, such as a small left steer, is executed, and the resulting state and reward are observed. The performance is evaluated through the reward function which encourages the vehicle to stay in the center of the lane. It will be positive for staying close to the lane center and negative for moving away from the lane center.

Listing 1: DQN Algorithm

Initialize replay memory D to capacity N

Initialize random weights for action-value function Q.

For episode $= 1, M$ **do**

Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$

For $t = 1, T$ **do**

Select $\epsilon = p(a_t)$, where ϵ is the probability of random action a_t

Otherwise select $a_t = max_a Q^*(\phi(s_t), a\,;\,\theta)$

Perform the action a_t and compute the related reward r_t and image x_{t+1}

Set $s_{t+1} = s_t, a_t, r_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$

Insert transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D.

Select transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ of a random minibatch from D.

Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma max_{\grave{a}}\, Q(\phi_{j+1}, \grave{a}\,;\,\theta) & \text{for non terminal } \phi_{j+1} \end{cases}$

Compute a gradient descent step on $(y_j - Q(\phi_j, a_j\,;\,\theta))^2$ with respect to the network parameters θ

End For

End For

Listing . 1 .shows a pseudo code for the DQN algorithm

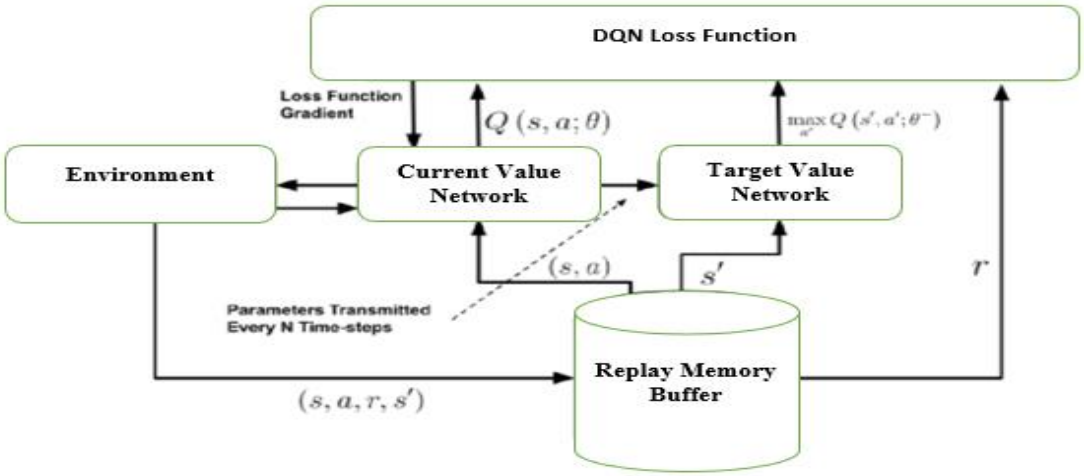


Fig. 4. The DQN Framework [24]

3.3. Overview of Twin Delayed Deep Deterministic Policy Gradient (TD3)

RL initially operated in environments with discrete action spaces, limiting its applicability to problems requiring continuous control. However, certain control problems, such as ACC, necessitate continuous actions. To get around this limit, the Deep Deterministic Policy Gradient (DDPG) method was created, allowing agents to work in environments with continuous action spaces. While DDPG offers advantages, it suffers from a significant drawback: Q-value overestimation. Q-values estimate the future reward an agent can expect from taking a specific action in each state. DDPG's tendency to overestimate Q-values can lead the agent to make suboptimal decisions.

The TD3 algorithm was developed to overcome DDPG's overestimation problem. It shares a similar structure to DDPG but utilizes two critic networks instead of one. This "twin" strategy reduces overestimation of the Q-value and stabilizes the learning process. TD3, an off-policy algorithm, does not interact with the environment directly; instead, it learns from past experiences that are saved in a replay buffer. This allows TD3 to learn more efficiently and avoid potential instability problems. To improve learning stability, TD3 also uses target networks to update the actor and critic networks.

Compared to Q-functions, TD3 updates the policy and targets less frequently. The TD3 agent introduces noise to the target action during policy updates, reducing the possibility that the policy would exploit actions with high Q-value estimations. At every time step in the training process, a TD3 agent updates the networks of actors and critics. A mini batch of experiences chosen at random from the replay buffer is used to update the actor and critic. The replay buffer is used to store past experiences. The tuple of (S_t, a_t, r_t, S_{t+1}) contains the agent experience. The agent reacts with the environment at each time step t, performs action a_t to transfer the current state to the following state S_{t+1}, and is rewarded with value r_t each time step t. Because TD3 involves two active critic networks, the smallest valued Q-values are selected as the target. **Fig.5** shows the frame work of TD3 algorithm.TD3 employs the Bellman error loss function during the learning process:

$$y \leftarrow r + \gamma \, min_{i=1,2} \, Q_{\theta_i} N^{-1} \Sigma (y - Q_{\theta_i'}, \bar{a}) \tag{15}$$

$$\theta_i \leftarrow argmin_{\theta i} N^{-1} \Sigma (y - Q_{\theta_i}(s,a))^2 \tag{16}$$

where N indicates the mini-batch size, θ_i indicates the parameters, and S indicates the current state.

$$\tilde{a} \leftarrow \pi'_{\phi}(S') + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim clip\,(N\,(0, \bar{\sigma}), -c, c) \tag{17}$$

Using a stochastic noise model, the TD3 agent modifies the action selected by the policy at each training step. π'_{ϕ} represents the target actor network, while ϵ represents the noise that is added to the actual action. [25,26]. Listing 2 shows a pseudo code for the TD3 algorithm.

The following is an illustrative example of how a TD3 algorithm can be used for ACC in a vehicle. The ego vehicle is equipped with a radar sensor and a TD3-based ACC system. Here's a breakdown of how it might work: The TD3 agent receives information from the vehicle's sensors, such as the distance to the lead vehicle, the relative velocity between the ego vehicle and the lead vehicle, and the ego vehicle's velocity. Based on the received information, the agent outputs an acceleration value for the ego vehicle, which may be positive or negative acceleration. The TD3 agent evaluates the state and receives a reward based on its performance, taking into consideration safety, comfort, and efficiency. The TD3 agent is trained in a virtual environment with various traffic scenarios (different velocities, sudden braking, lane changes). Over time, through trial and error, the TD3 agent learns the optimal acceleration strategy to maintain a safe distance, provide a smooth ride, and achieve the desired velocity in various traffic conditions. Once trained, the TD3 policy is transferred to the real vehicle's control system. The vehicle's sensors continuously provide real-time state information (distance, velocity). The TD3 policy

recommends the most suitable acceleration based on the current state, enabling smooth and efficient ACC.

Listing 2: TD3 Algorithm

Setup critic networks Q_{θ_1} , Q_{θ_2} and actor network π_ϕ with initial parameters θ_1 , θ_2 ,ϕ

Initializing target networks $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$, $\phi' \leftarrow \phi$, replay buffer β

for $t = 1$ to T **do**

Find action with exploration noise $a \sim \pi_\phi(s) + \epsilon$

$\epsilon \sim N(0, \sigma)$ and compute the related reward r and new state's $'$

Insert transition tuple (s, a, r, s′) of mini batch from β

Select N transitions (s, a, r, s′) from β

$\tilde{a} \leftarrow \pi_\phi'(S') + \epsilon$, $\epsilon \sim$ clip $(N(0, \bar{\sigma}), -c, c)$

$y \leftarrow r + \gamma \min_{i=1,2} Q_{\theta_i} N^{-1} \Sigma (y - Q_{\theta_i'}, \bar{a})$

Modify critics $\theta_i \leftarrow \text{argmin}_{\theta i} N^{-1} \Sigma (y - Q_{\theta_i}(s, a))^2$

if t mod d **then**

Modify ϕ by the deterministic policy gradient:

$$\nabla_p h\, i\, J(\phi) = N^{-1} \Sigma \nabla_a Q_{\theta_1}(s, a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$$

Modify target network:

$$\theta_i' \leftarrow \mathcal{T}_{\theta_i} + (1 - \mathcal{T})\theta_j' \,, \phi' \leftarrow \mathcal{T}_\phi + (1 - \mathcal{T})\phi$$

end if

end for

Listing . 2. shows a pseudo code for the TD3 algorithm

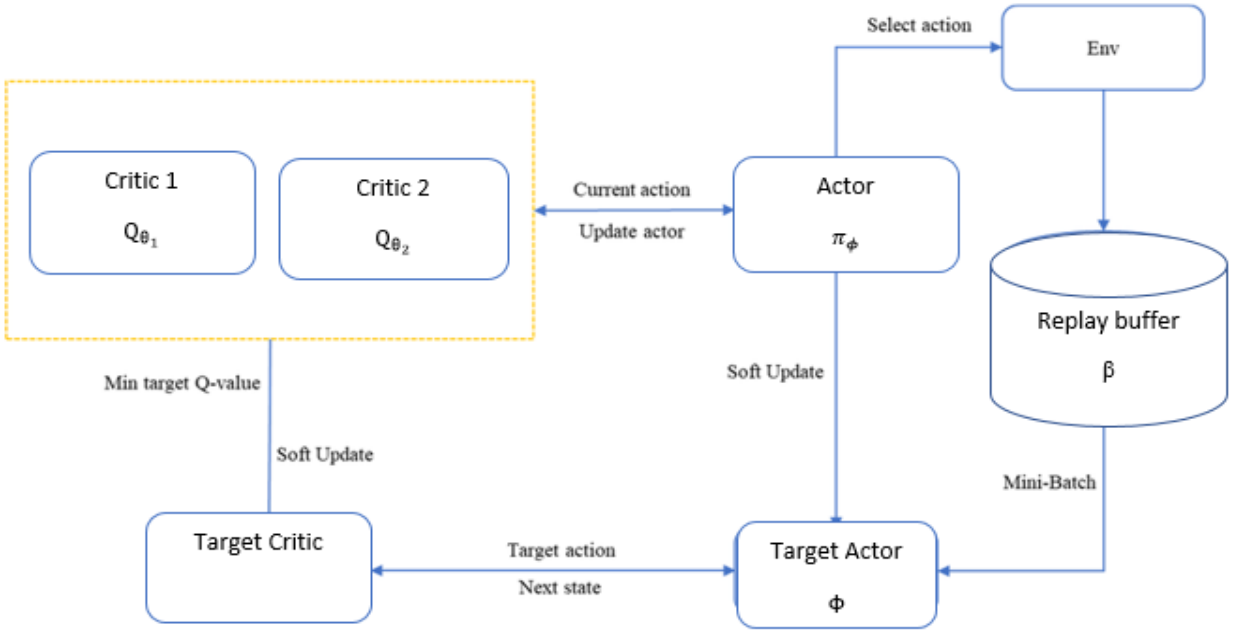


Fig.5 . The TD3 algorithm Framework [27]

4. THE PROPOSED ACC SYSTEM WITH LANE KEEPING ASSIST

4.1. System design

In this section, the proposed ACC with LKA is presented, designed, and implemented. **Fig. 6** illustrates the system's suggested structure. The environment is based on a simple longitudinal model for the lead vehicle and a basic bicycle model for the ego vehicle. The purpose of ACC with LKA is to assist the ego vehicle in travelling at a set velocity while managing longitudinal acceleration and braking to keep a safe distance behind a leading vehicle and employing front steering angle control to maintain the vehicle centered in its lane.

The TD3 agent is regarded as a longitudinal controller; it interacts with the driving environment by sending an action signal consisting of continuous acceleration and receiving observations that contain longitudinal measurements, such as the ego vehicle's longitudinal velocityv_x, its integral $\int e_v$, and the velocity error.

$$e_v = v_{ref} - v_x \tag{18}$$

The ego vehicle's reference velocity, v_{ref}, is specified as follows. The ego vehicle tracks the lower of the lead vehicle's velocity and the driver-set velocity if $d_{rel} < d_s$, this way, the ego vehicle keeps some distance from the lead vehicle. If $d_{rel} > d_s$, the ego vehicle follows the driver-set velocity.

The safe distance in this paper, is defined as $d_{safe} = v_x \ * t_g + d_{def}$ which is a linear function of the ego vehicle longitudinal velocity v_x , The ego vehicle's tracking velocity is determined by the safe distance.

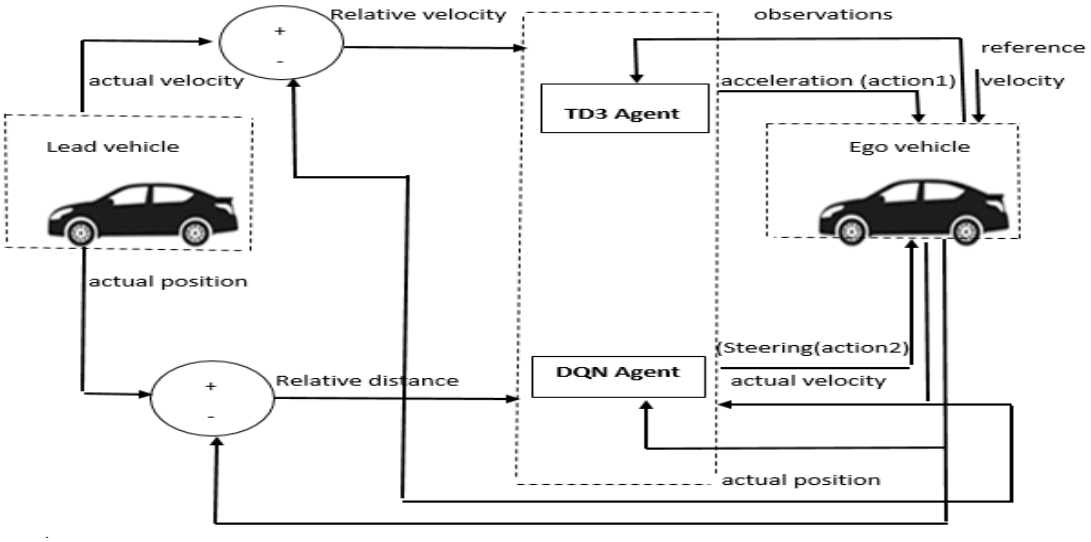
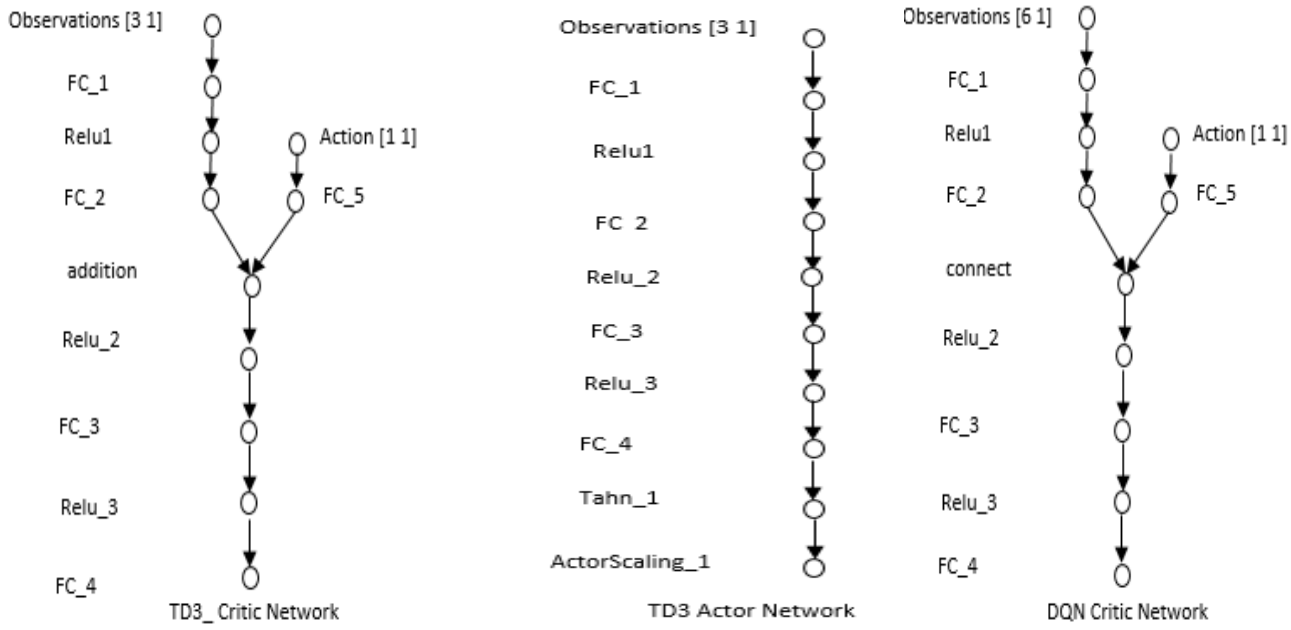


Fig. 6. The proposed ACC with LKA architecture

The TD3 agent's acceleration command signal has a 2 m/s^2upper limit and a -3 m/s^2 lower limit. The DQN agent acts as a lateral controller, interacting with the driving environment by receiving observations that include lateral measurements, such as the relative yaw angle e_2, the lateral deviation e_1, its derivatives, $\dot{e}_1$and $\dot{e}_2$,and its integrals, $\int e_1$ and $\int e_2$. The DQN agent's steering command signal is composed of discrete steering angle actions that take values in steps of 1° (0.0175 rad) from -15° (-0.2618 rad) to 15°.

4.2. Environmental Setting

As illustrated in **Fig. 7**, the TD3 agent is implemented using three networks: one network for the actor to choose which action to perform given an observation **(Fig. 7-b)** and two critic networks of the same design for approximating the long-term reward depending on observations and action **(Fig. 7-a)**. To update the actor network, the TD3 agent uses the output of the two critic networks with the lowest value function. The TD3 agent is able to overcome the overestimation issue with the DDPG algorithm due to these two critic networks. As shown in **(Fig. 7-c)** the DQN agent uses one critic for approximating the long-term reward.



a. Critic of TD3 b. Actor of TD3 c. Critic of DQN

Fig. 7. Actor and critic networks of both TD3 and DQN algorithms

Table 1 : The hyper-parameters utilized to simulate the actor and critic networks

Parameter name value	Agent 1	Agent 2
Observations dimension	[3 1]	[6 1]
Action dimension	[1 1]	[1 1]
Learning rate for actor	0.001	--
Learning rate for critic	0.0001	0.0001
Optimizer type	Gaussian	--
Experience buffer length	1000000	1000000
Batch size	64	64
Discount Factor	0.99	0.99
Policy Update Frequency	2	1
Target Update Frequency	2	1
Smooth factor	0.001	0.001
Sample time	0.1	0.1

4.3. Hyper parameter tuning and its impact on the performance of the algorithms.

Observations Dimension: The input dimensions should match the nature of the environment and the ACC with LKA problem. The observation dimensions provide more information to the model. (2) Action Dimension: The action dimensions should align with the controllable variables in the environment. Increasing action dimensions enhances the model's ability to perform complex actions. (3) Learning Rate for Actor: A high learning rate can lead to faster updates but cause instability in learning. A low learning rate leads to increase stability of learning but makes training slower. (4) Learning Rate for Critic: Similar to the actor learning rate. The balance between the actor and critic learning rates is crucial for stable learning. (5) Optimizer Type: affects the speed and stability of convergence. The TD3 algorithm uses a deterministic policy, and to encourage exploration during training, Gaussian noise is typically added to the actions taken by the policy. This helps the agent explore the action space more effectively. (6) Experience Buffer Length: The buffer length affects the diversity of data available for learning. A larger buffer provides more diversity but requires more memory. (7) Batch Size: A larger batch size leads to more stable updates but requires more memory and reduces update diversity. A smaller batch size introduces more noise in updates but helps capture finer patterns. (8) Discount Factor: Determines the importance of future rewards. A high discount factor makes the model focus on long-term rewards, while a low discount factor makes it focus on short-term rewards. (9) Policy Update Frequency: The frequency of policy updates affects how quickly the model learns. Increasing the frequency speeds up learning but may introduce more fluctuations. (10) Target Update Frequency: Increasing the target update frequency enhances learning stability but slows down training. (11) Smooth Factor: Used for target policy smoothing. Increasing the smooth factor can reduce update fluctuations and improve learning stability. (12) Sample Time: Affects interaction with the environment and data recording. Smaller sample times provide more precise data.

4.4. The TD3 and DQN reward functions:

The TD3 agent's reward function can be described as follows: At every time step t the environment sends a reward signal.$\mathrm{r_{t\_ACC}}$:

$$r_{t\_ACC} = -(0.01e_v^2 + 0.1a_{t-1}^2) - 10Z_t + N_t \tag{19}$$

$$Z_t = \begin{cases} 1 & in\ the\ event\ that\ the\ simulation\ ends \\ 0 & else \end{cases} \tag{20}$$

$$N_t = \begin{cases} 1 & e_v^2<1 \\ 0 & else \end{cases} \tag{21}$$

The reward function is used to penalize or encourage the TD3 agent based on factors such as velocity error $\mathrm{e_v}$, control input $\mathrm{a_{t-1}}$ (which is the acceleration), and additional parameters $(N_t, \mathrm{Z_t})$ It is designed to punish the agent if the simulation ends early or to promote it to minimize the velocity error.

The DQN agent's reward function can be described as follows: At every time step t the environment sends a reward signal $\mathrm{r_{t\_LKA}}$:

$$r_{t\_LKA} = -(0.1e_1^2 + 0.5u_{t-1}^2) - 10Z_t + 2B_t \tag{22}$$

$$B_t = \begin{cases} 1 & e_1^2<0.01 \\ 0 & otherwise \end{cases} \tag{23}$$

The reward function used to penalize or encourage the DQN agent considers the lateral error $\mathrm{e_1}$, the control element steering angle $\mathrm{u_{t-1}}$, and additional parameters (Z_t, B_t) to penalize the agent if the simulation ends early or encourage it to minimize the lateral error.

5. SIMULATION RESULTS

The proposed ACC with LKA model is implemented in MATLAB 2020b and the Simulink toolbox to simulate the behavior of this model. The system utilizes two RL agents trained collaboratively to perform ACC with LKA for a vehicle. For longitudinal vehicle control, the first RL agent (TD3) provides continuous acceleration commands. To control the position of the vehicle within its lane, the second RL agent (DQN) outputs discrete values for the steering angle. The block diagram for the Simulink model is displayed in **Fig. 8** Two agents, along with their signal processing blocks, are also shown in the block diagram. The model for calculating and propagating rewards is contained in the signal processing block of the upper agent to calculate the velocity error and control longitudinal motion for ACC functions. For the purpose of providing lane-keeping assist, the lower agent controls lateral motion. Its signal processing block computes the lateral error and includes the model for determining and sending rewards.

Table 2: shows episode manager training information

Training Results	RLAgent1 (ACC-basedTD3)	RLAgent2 (LKA-based DQN)
Episode Steps	600	600
Total Number Of Steps	60925	60529
Episode Reward	484.4112	1197.8865
Average Reward	486.6473	1191.4207
Episode Q_0	-26.2736	224.3921
Training Stopped by	Average Reward	Average Reward
Training Stopped at Value	480	1195

Table 2 shows episode manager training information for two agents**. Fig. 9** displays the system statistics using the reward functions that were designed. The duration takes 2 hours, 36minutes,27seconds. A maximum of 5000episode steps was used for each training episode. The DQN and TD3 agents achieved an average reward of more than 1195 and 480, respectively. When one agent reached its stopping criteria, it simulated its own policy, while the other agent continued with its training.

The developed algorithms were evaluated in a situation where the initial relative distance between the two vehicles was 70 meters. **Fig. 10** illustrates the results that were obtained. The driver set velocity (velocity chosen by the driver) is 28m/s. Since the relative distance is higher than the safe distance, the ego vehicle stays at the set velocity for the first 36 seconds **(Fig. 10-a)**. The acceleration is typically nonnegative to accelerate and reach the set velocity **(Fig. 10-c).** The relative distance is mostly less than the safe distance between 36 and 42 seconds **(Fig. 10-a)**, so the ego vehicle follows the minimum of the lead velocity and set velocity. Acceleration becomes nonzero as the lead velocity is smaller than the specified velocity (velocity plot a) to track the lead velocity **(Fig. 8-c).** The ego vehicle maintains a constant velocity **(Fig. 10-b)** and zero acceleration **(Fig. 10-c)** between 42 and 60 seconds.

The lateral deviation and relative yaw angle of the ego vehicle are both driven close to zero, as shown in Figures **(10-d)** and **(10-e)**. The vehicle has a nonzero yaw angle error (0.1 rad) and is initially off the centerline (–0.4 m). The ego vehicle follows the centerline due to the LKA feature. The steering angle **(Fig. 10-c)** indicates that, the DQN controller reaches a steady state in approximately 1 second. The lateral deviation is significantly reduced in just one second, as seen in **(Fig. 10-d)**, and it stays almost zero.

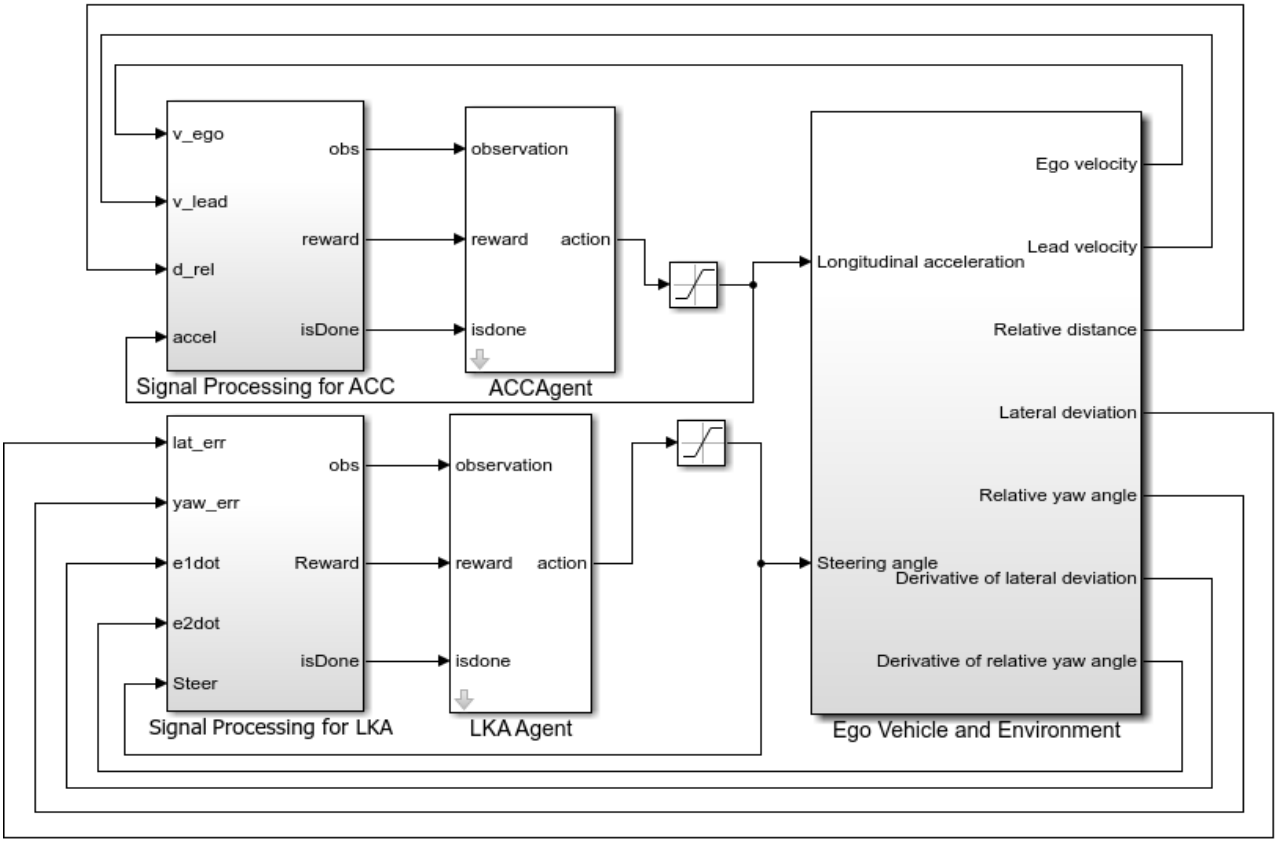
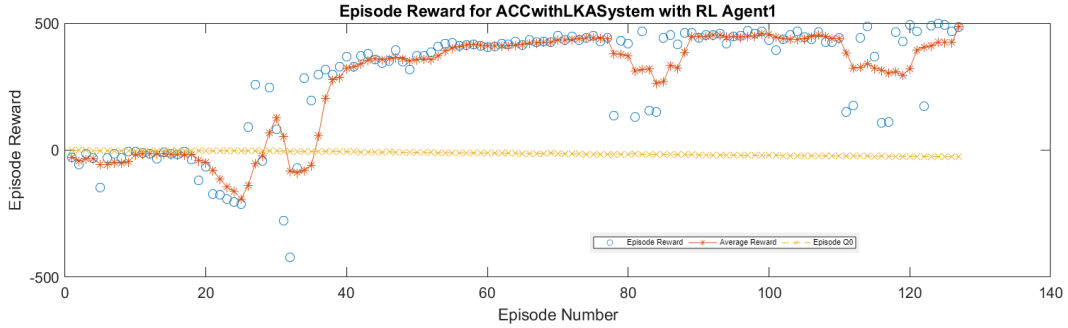
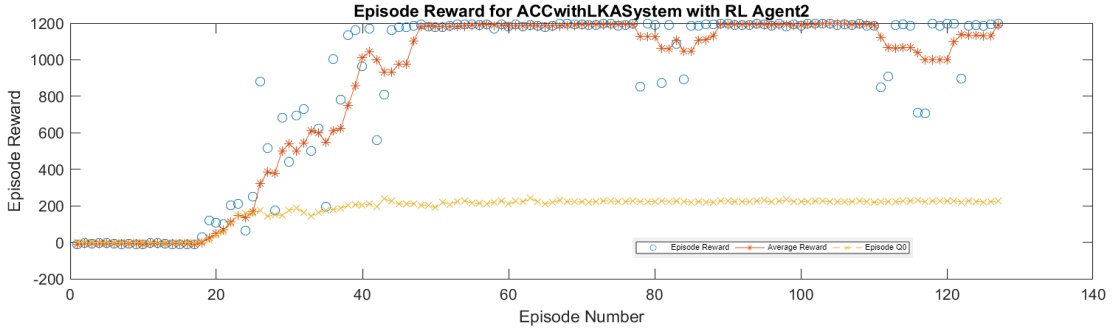


Fig. 8. Block diagram of the Simulink model of the proposed system

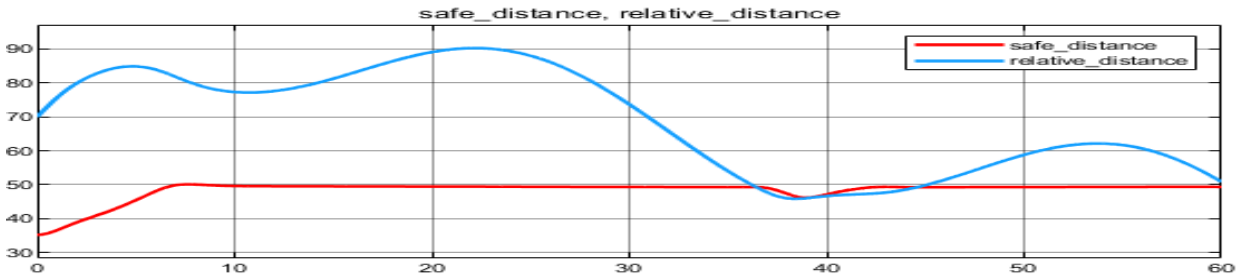


(a) Training plot of TD3 agent

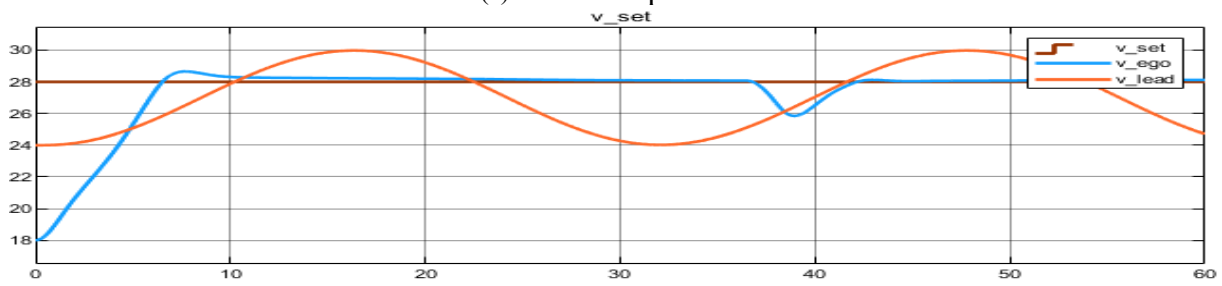


(b) Training plot of DQN agent

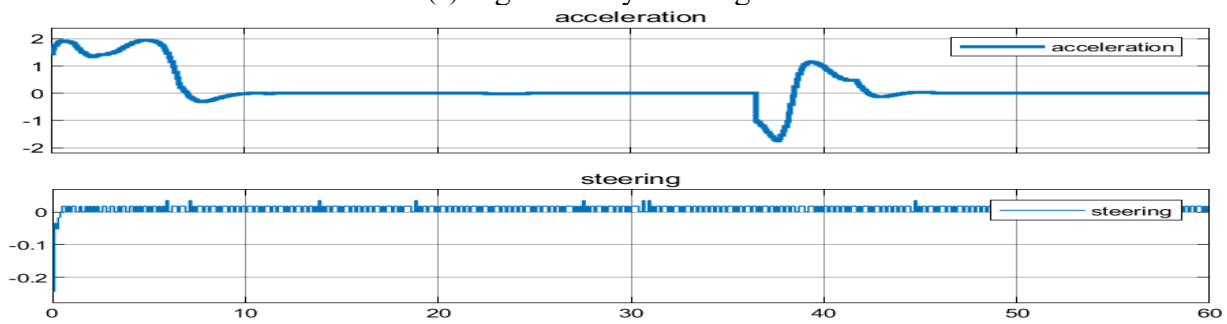
Fig. 9. Training plot for both TD3 and DQN



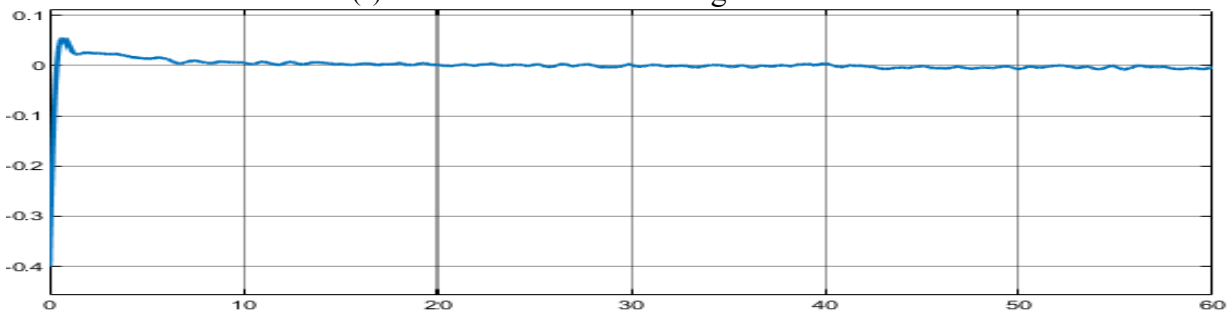
(a) Distance plot



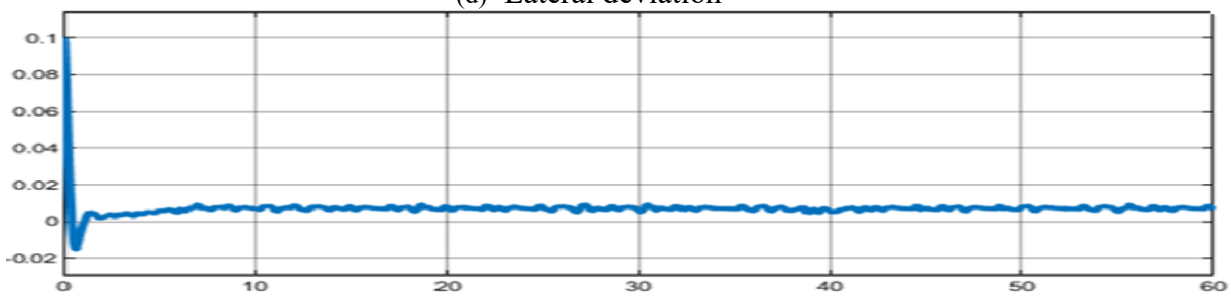
(b) Ego velocity tracking



(c) Acceleration and steering actions



(d) Lateral deviation



(e) Yaw angle

Fig. 10. simulation results of the proposed ACC

6. SAFETY AND ROBUSTNESS

The proposed system offers a combination of comfort and safety for drivers, enhancing the overall driving experience. (i) The proposed system can adapt vehicle speeds of surrounding vehicles, ensuring safe distances between vehicles and reducing collision risks. (ii) Also, the system can quickly respond to sudden changes. (iii) the system helps maintain lane position accurately by automatically steering the vehicle back on track if it unintentionally drifts out of the lane. (iv) The system reduces driver fatigue by automatically controlling speed and steering, allowing the driver to relax and stay focused. (v) The system contributes to maintaining lane position and reducing deviations, especially on curved roads, decreasing the likelihood of accidents due to vehicle drifting. In addition to safety, the system is considered as a robust system as it decreases the number of incidents by improving the interaction with the surrounding environment, and reduce the effort required by the driver to manage speeds and maintain lane positions. Hence the proposed system offers a combination of comfort and safety for drivers, enhancing the overall driving experience.

7. APLYING THE PROPOSED SYSTEM IN REAL WORLD

In this section, potential real-world applications that can be improved using the proposed system are discussed, along with the related challenges and suggested solutions, Additionally, a plan for testing the system is suggested:

i. Enhancing smooth driving: the ACC can be designed to prioritize smooth driving, resulting in less jerky acceleration and deceleration for passengers. (iv) Reducing driver fatigue by automating the task of maintaining speed and distance, which can reduce driver fatigue on long journeys. While ACC with LKA offers promising benefits for real-world applications, there are some challenges that make these applications difficult to implement. These challenges are: (i) Safety Assurance: This requires extensive testing and fail-safe mechanisms to guarantee the system prioritizes safety in all situations. (ii) Sensor Accuracy and Reliability: Sensor malfunctions or limitations can lead to unpredictable behavior. (iii) Data Collection and Training: Training ACC agents requires a vast amount of data covering diverse traffic scenarios. Collecting real-world data can be expensive and time-consuming, while purely simulated environments might not capture all real-world complexities. (iv) Computational Cost: Implementing RL algorithms on resource-constrained vehicles computers can be challenging. (v) Regulatory Considerations, Regulatory bodies need to be convinced of the safety and reliability of the ACC system before deployment. Explain ability of the decision-making process is crucial.
ii. ACC with LKA real-world applications: ACC with LKA can be implemented in several real-world applications such as: (i) Learning and adjusting to various traffic scenarios, such as dense traffic and sudden braking.
iii. Improving fuel economy by potentially minimizing unnecessary braking and acceleration.
iv. Feasibility of implementing the proposed system: While significant challenges exist, the potential benefits of ACC RL-based ACC with lane keeping assist, here are some possibilities for increased feasibility: (i) Focus on specific functionalities. (ii) Combining RL with traditional rule-based control systems could leverage the strengths of both for enhanced safety and adaptability. (iii) Developing other AI techniques to understand decision-making that must be handled during the system operation.
v. Field testing plan for the proposed system

vi. Here's a breakdown of a potential plan for field testing and validation of the proposed system in real-world driving conditions: (i) Controlled environment testing: In this phase, the system's basic functionality, response to a specific situations and the performance of the proposed system are tested. (ii) Real-world pilot testing: In this phase the system's performance in real-world traffic is evaluated with human supervision, and potential limitations are identified for improvement. (iii) Field testing phase: In this phase the system's performance across a broader range of real-world scenarios is tested, and feedback from other drivers is used to refine the proposed system.

8. TRADITIONAL ACC SYSTEMS VS.THE PROPOSED SYSTEM COMPARISON

Directly comparing this system to others can be challenging due to a lack of standardized metrics. Each system implementation approach and underlying framework is unique. Therefore, a quantitative comparison is not feasible. Instead, we present a qualitative comparison between the proposed system's performance and recently implemented ACC systems. This comparison considers the methodologies employed in each system. Traditional ACC systems often use approaches like PID control, fuzzy logic, or model predictive control. The results of this qualitative comparison are presented in **Table 3**. As shown in **Table 3**, the proposed system may have a high safety factor compared to traditional ACC systems. It also offers high adaptability, includes lane keeping functionality, and is smoother than other traditional ACC systems making it more follow able than them.

Table 3. shows Traditional ACC systems vs. proposed system Comparison

Feature	Traditional ACC	Proposed system
Safety Factor	Moderate	High Potential (Adapts, needs rigorous testing)
Adaptability	Limited (Pre-programmed rules)	High (Learns from diverse scenarios)
Lane Keeping	May require separate lane Departure Warning system.	Included
Followability	Good (Maintains set distance)	Potentially smoother (Prioritizes smooth acceleration/deceleration)

CONCLUSION AND FUTURE WORK

This paper introduces a multi-objective ACC system designed to enhance traditional ACC capabilities. An intelligent ACC with LKA is proposed, modeled, and developed using TD3 and DQN algorithms. These algorithms collaboratively learn to perform ACC with LKA, yielding satisfactory results. System performance evaluation demonstrates its ability to follow lane centerlines, maintain safe distances from lead vehicles, and adhere to driver-set velocities. The proposed system significantly improves ACC vehicle capabilities, preventing collisions by preventing lane changes on straight or curved roads and providing a robust multi-objective ACC solution. The integration of ACC and LKA functions through cooperative behavior significantly enhances overall vehicle autonomy and safety.

While the proposed system exhibits promising results, there are avenues for further improvement and exploration: (i) the simulation environment must be enlarged to include more complex traffic conditions, such as heavy traffic, merging vehicles, and adverse weather, to assess the system's robustness. (ii) the proposed system must be implemented and tested on a real vehicle to validate its performance in real-world conditions and address potential challenges. (iii) more advanced RL

algorithms such as Actor-Critic methods or Hierarchical RL must be explored to potentially improve learning efficiency and performance. (iii) the integration of the proposed ACC system into cooperative multi-vehicle systems must be investigated to enhance traffic flow and safety. By addressing these areas, future research can contribute to the development of even more sophisticated and reliable autonomous driving technologies.

REFERENCES

[1] Tigadi, A., Gujanatti, R., Gonchi, A., & Klemsscet, B. (2016). Advanced driver assistance systems. International Journal of Engineering Research and General Science, 4(3), 151-158.
[2] Yadav, A. K., & Szpytko, J. (2017). Safety problems in vehicles with adaptive cruise control system. Journal of KONBiN, 42(1), 389-398.
[3] Kılıç, Islam, et al. "Intelligent adaptive cruise control system design and implementation." 2015 10th System of Systems Engineering Conference (SoSE). IEEE, 2015.
[4] Paul, A., Chauhan, R., Srivastava, R., & Baruah, M. (2016). Advanced driver assistance systems (No. 2016-28-0223). SAE Technical Paper.
[5] Bharadwaj, S. S. M., & Dattawadkar, S. (2015). Design of autonomous cruise control unit for intelligent vehicles. International Journal of Emerging Technology and Advanced Engineering, 5(6).
[6] Sivaji, V. V., and M. Sailaja. "Adaptive cruise control systems for vehicle modeling using stop and go manoeuvres." International Journal of Engineering Research and Applications 3.4 (2013): 2453-2456.
[7] Dalip, I. A., and J. D. Jiya. "Optimal Design of Pid-Controller For Adaptive Cruise Control Using Differencial Evolution."
[8] Rizvi, Raazi, et al. "Fuzzy adaptive cruise control system with speed sign detection capability." 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). IEEE, 2014.
[9] Singh, Ankita, C. S. Satsangi, and P. Panse. "Adaptive cruise control using fuzzy logic." Int J Digital Appl Contemp Res 3.8 (2015): 1-7.
[10] Park, Changwoo & Lee, Hyeongcheol. (2017). A Study of Adaptive Cruise Control System to Improve Fuel Efficiency. International Journal of Environmental Pollution and Remediation. 10.11159/ijepr.2017.002.
[11] E. Kural and B. Aksun Güvenç, "Model Predictive Adaptive Cruise Control," 2010 IEEE International Conference on Systems, Man and Cybernetics, Istanbul, Turkey, 2010, pp. 1455-1461, doi: 10.1109/ICSMC.2010.5642478.
[12] B. Wang, D. Zhao, C. Li and Y. Dai, "Design and implementation of an adaptive cruise control system based on supervised actor-critic learning," 2015 5th International Conference on Information Science and Technology.
[13] Zhu, Meixin, Xuesong Wang, and Yinhai Wang. "Human-like autonomous car-following model with deep reinforcement learning." Transportation research part C: emerging technologies 97 (2018): 348-368.
[14] Yarom, Or Aviv, et al. "Systematic Model-based Design of a Reinforcement Learning-based Neural Adaptive Cruise Control System." ICAART (3). 2022.
[15] Das, Lokesh Chandra, and Myounggyu Won. "Saint-acc: Safety-aware intelligent adaptive cruise control for autonomous vehicles using deep reinforcement learning." International Conference on Machine Learning. PMLR, 2021.
[16] Han, Zhonghai, et al. "Research on vehicle cruise control based on online asynchronous supervised reinforcement learning." Journal of Physics: Conference Series. Vol. 1873. No. 1. IOP Publishing, 2021.

[17] Maruyama, Nagayasu, and Hiroshi Mouri. "A proposal for adaptive cruise control balancing followability and comfortability through reinforcement learning." ROBOMECH Journal 9.1 (2022): 22.
[18] Bishen, Himanshu Kumar, K. V. Shihabudheen, and PP Muhammed Shanir. "Adaptive Cruise Control Using Twin Delayed Deep Deterministic Policy Gradient." 2023 5th International Conference on Energy, Power, and Environment: Towards Flexible Green Energy Technologies (ICEPE). IEEE, 2023.
[19] Nie, Zifei, and Hooman Farzaneh. "Adaptive cruise control for eco-driving based on model predictive control algorithm." Applied Sciences 10.15 (2020): 5271.
[20] Kılıç, Islam, et al. "Intelligent adaptive cruise control system design and implementation." 2015 10th System of Systems Engineering Conference (SoSE). IEEE, 2015.
[21] Busoniu, Lucian & de Bruin, Tim & Tolić, Domagoj & Kober, Jens & Palunko, Ivana. (2018). Reinforcement learning for control: Performance, stability, and deep approximators. Annual Reviews in Control. 46. 10.1016/j.arcontrol.2018.09.005.
[22] Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.
[23] Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. “Playing Atari with Deep Reinforcement Learning.” ArXiv:1312.5602 [Cs], December 19, 2013.
[24] Hu, Mingzhe & Zhang, Jiahan & Matkovic, Luke & Liu, Tian & Yang, Xiaofeng. (2022). Reinforcement Learning in Medical Image Analysis: Concepts, Applications, Challenges, and Future Directions. 10.48550/arXiv.2206.14302.
[25] Fujimoto, Scott, Herke van Hoof, and David Meger. "Addressing Function Approximation Error in Actor-Critic Methods". ArXiv:1802.09477 [Cs, Stat], 22 October 2018. https://arxiv.org/abs/1802.09477.
[26] Yang, K., Guler, S. I., & Menendez, M. (2016). Isolated intersection control for various levels of vehicle technology: Conventional, connected, and automated vehicles. Transportation Research Part C: Emerging Technologies, 72, 109-129.
[27] Jiang, Caiyu & Wang, Jianhua. (2022). A Portfolio Model with Risk Control Policy Based on Deep Reinforcement Learning. Mathematics. 11. 19. 10.3390/math11010019.