Autonomous Driving Decision-Making Based on an Improved Actor-Critic Algorithm

Rong HU[1]*, Ping HUANG[2]

[1] School of Intelligence Technology, Geely University of China, No. 123, Sec. 2, Chengjian Avenue, Eastern New District, Chengdu, 641423, China
727749104@qq.com (**Corresponding author*)

[2] School of Mathematics and Computer Science, Panzhihua University, No. 10, North Section of Third Avenue, East District, Panzhihua, 617000, China
1269228282@qq.com

Abstract: Autonomous driving technology, as a new type of automotive driving technology, contributes to reducing the number of traffic accidents and to lowering the mortality rate for traffic accidents. However, due to the limited prior knowledge of designers, the current autonomous driving decision-making systems face difficulties in dealing with complex and ever-changing traffic scenarios. In view of this, this study proposes an autonomous driving decision-making model based on the Soft Actor-Critic algorithm and a long short-term memory network. The experimental results obtained in complex mixed traffic scenarios for the average collision frequency, the average lane change frequency, the average following distance, and the average distance from the lane centerline for the decision-making model based on the Soft Actor-Critic algorithm and a long short-term memory network were 0.9, 9.8, 19 metres, and 0.36 metres, respectively. Moreover, the average arrival time, root mean square of acceleration, and root mean square of the acceleration change rate obtained by the proposed model were 127 seconds, 1.1, and 1.3, respectively, these values being superior to the ones obtained by the other employed models. In addition, when facing a sudden pedestrian crossing, the collision time for the proposed decision-making model was the shortest at 8.3 seconds, which is 1.2 seconds higher than the values obtained by the other employed models. The obtained outcomes prove that the decision-making model based on the Soft Actor-Critic algorithm and a long short-term memory network proposed in this paper can cope with complex and changing traffic scenarios, and ensure the safety of both pedestrians and drivers.

Keywords: Autonomous driving, Decision model, Decision intelligent agents, Actor-Critic algorithm, Long short-term memory network.

Introduction

The autonomous driving system refers to the train operation system with fully automated and highly centralized control of the driver`s actions (Yuan et al., 2022). It makes use of advanced communication, computing, networking, and control technologies to enable the timely and continuous control of the vehicle. Autonomous driving (AD) technology utilizes video cameras, radar sensors, and laser rangefinders to understand surrounding traffic conditions and navigate the road ahead using maps (Wu et al., 2023). Volvo distinguishes four levels of driverless driving according to the automation level: driving assistance, partial automation, high and full automation. The Driving Assistance System, Partial Automation System, High Automation System and Full Automation System are the corresponding automated driving systems for these four levels (Huang et al., 2023). The current AD technology is in a highly automated stage. In the autonomous driving system of the car, the quality of the decision-making system often determines the performance of the entire driving system, which is the key for dealing with the traffic environment. The current AD decision-making algorithms mainly include finite state machines, decision trees, neural networks, and Q-learning (Liao et al., 2024). Finite state machines cannot represent concurrency and cannot describe asynchronous concurrent systems. Decision trees have a high degree of subjectivity in determining the probability of various options, which may lead to decision-making errors (Wang, W. et al., 2022). Neural networks are unable to explain their own reasoning process and reasoning basis, while the Q-learning algorithm cannot capture task structures well (Zhang, T. et al., 2023). The above methods face difficulties in dealing with complex and ever-changing traffic scenarios. The Actor-Critic (AC) algorithm fuses the merits of both value-based and policy-based algorithms, enabling one to obtain a simultaneous output along with the evaluation of decisions. Therefore, to improve the safety of AD decisions and cope with complex and ever-changing traffic environments, this paper presents an AD decision model based on the Soft Actor-Critic algorithm (SAC) and Long Short-Term Memory (LSTM) network. This model innovatively introduced the LSTM network into the SAC network structure, enabling the improved SAC algorithm to accurately represent environmental semantics. The contribution of this study is to propose an effective autonomous driving decision-making method that addresses the challenge of taking reliable and accurate decisions

https://doi.org/10.24846/v33i4y202404

related to autonomous driving in uncertain environments, thereby effectively improving the safety of autonomous driving and reducing the incidence of traffic accidents. At the same time, this research is expected to provide effective solutions for solving high-order autonomous driving decision-making problems and promote the evolution of automotive intelligence.

The remainder of this article is structured as follows. Section 2 is a literature review, which briefly described the current research results for AD decision-making and AC algorithms. Section 3 sets forth the proposed research method, analysing the AD decision model based on the AC algorithm and LSTM. Section 4 discusses the experimental findings, analysing the results of the decision-making process and the safety of the AD decision models in both single and mixed traffic scenarios. Finally, Section 5 concludes this paper.

2. Literature Review

In the field of AD, facing complex interactive scenes, how to achieve an accurate and efficient decision-making with regard to vehicle movement has become a current research hotspot. Wang, H. et al. (2022) presented a probabilistic reconstruction-based learning method for recognizing the internal state of multi-vehicle sequential interactions to enhance the accuracy of AD decisions. This approach considered the sequential decision-making for merging tasks as a dynamic stochastic process, and integrated the internal states of multi-vehicle sequential interactions into a Gaussian mixture regression model with a hidden Markov model, and utilized the expectation maximization algorithm to estimate the model parameters. The experimental findings indicated that the extracted internal states can semantically represent the dynamic decision-making process and make accurate predictions (Wang, H. et al., 2022). Nan et al. (2022) proposed an intention prediction method and a mixed strategy Nash equilibrium theory-based framework for decision-making problems in uncontrolled intersections. This framework used a combination algorithm which joined the Hidden Markov Model and SVM to predict the driving intention of the target vehicle at intersections, and the Bessel curve was adopted to fit the predicted trajectory of the target vehicle. At the same time, the S-T graph was used to determine whether there were spatiotemporal conflict points between the target vehicle and the self-driving vehicle. The experimental outcomes showed that this decision-making framework can enable vehicles to safely and comfortably pass through intersections (Nan et al., 2022). Xu et al. (2022) proposed an integrated decision-making framework for autonomous vehicles on highways to solve the comprehensive decision-making problem related to AD on highways. This framework first used reinforcement learning to learn the optimal state-action pairs for specific scenarios, and used imitation learning to memorize experience pools through deep neural networks. The experimental outcomes denoted that the framework can efficiently drive the vehicle to the predetermined state while ensuring its safety (Xu et al., 2022). For safety and personalized driving of autonomous vehicles, Huang et al. (2021) proposed a decision-making framework with integrated trajectory planning and tracking control algorithms. This framework utilized dynamic potential fields to express the interactions between vehicles, and planned and tracked trajectories through artificial potential fields and constrained Delaunay triangulation. The experimental findings illustrated that this framework can make safe and personalized decisions, and more effectively execute AD motion control in dynamic environments (Huang et al., 2021). Zhang, Y. et al. (2023) proposed a new algorithm based on 3D LIDAR point cloud data to identify vehicles in the environment for the problem of vehicle recognition in automatic driving. The point cloud compression method based on nearest neighbour points and octree voxel center point boundary extraction is applied to the point cloud data, and then the vehicle point cloud recognition algorithm based on image mapping is used for vehicle recognition. The experimental results show that this algorithm effectively improves the accuracy for vehicle recognition (Zhang, Y. et al, 2023).

The AC algorithm effectively compensates for the shortcomings of both value-based and policy-based algorithms. It can output the policies directly and evaluate the quality of the current policies in real time by means of value functions. Therefore, it is widely used in various fields. Chen et al. (2023) suggested a formation control method based on a radial basis function neural network and an AC algorithm for unmanned surface vehicles with collision avoidance and predetermined performance. In the context of this method, the radial basis function neural network was utilized to approximate the modeling uncertainty, and

the optimality of formation control was ensured by the AC algorithm. Experimental findings demonstrated that this method was significantly superior to other methods (Chen et al., 2023). Wu et al. (2022) put forward a tracking control method for robotic knee joint prostheses based on AC to address the issue of configuring impedance parameters for robotic prosthetics. The experimental findings showed that this method can enable the robot to walk on flat ground, on different terrains, and at different speeds. Zhang & Xue (2022) proposed a decision framework based on convolutional neural networks and AC algorithms for the decision-making problem of artificial intelligence commanders in tactical warfare games. This framework utilized convolutional neural networks and AC algorithms to express the situation of the battlefield, and it attempted different tactical strategies through reinforcement learning. The experimental findings showed that the decisions made by this framework can lead to achieving higher scores in tactical warfare games. Madan & Bhatia (2021) proposed a deep network crawling model with the asynchronous advantage AC algorithm to address the issue of website information crawling. This model comprised multiple agents, each of which was capable of learning in different environments, updating the local gradient of the coordinator, and generating a more stable system. The experimental findings revealed that the proposed network crawling model outperforms other models. Dey & Xu (2023) put forth a control model based on a hierarchical game-based algorithm and an AC algorithm for the distributed adaptive formation control problem of large-scale multi-agent systems. This model employed cooperative games to formulate distributed inter-group formation control for leaders and solved the optimal distributed formation control problem through the AC algorithm. The experimental findings proved that this method can effectively achieve the control of large-scale multi-agent systems.

In summary, current research on intelligent driving systems has been quite effective, but most AD decision systems face difficulties with regard to decision-making in complex traffic scenarios. The AC algorithm has the merit of being able to make decisions and evaluate decisions. Therefore, to enable AD decision-making in complex scenarios and improve the safety of decision-making, this paper proposes an AD decision-making model based on an improved AC algorithm.

3. Research Methodology

To implement AD decision-making for vehicles, an improved SAC-based decision method is proposed, which first constructs an intelligent AD decision agent and a decision execution strategy based on the SAC algorithm. Meanwhile, LSTM is introduced to improve the effectiveness of decision agents in handling complex tasks.

3.1 Design and Improvement of the Autonomous Driving Decision-making Intelligent Agent Based on SAC

Although AD technology brings convenience to humanity, there are also some safety risks. The AC algorithm combines value-based and policy-based algorithms (Zhang et al., 2024), making it easy to select proper actions in a continuous action space. In the AC algorithm, the Actor network (AN) is used to predict the probability of actions (Du et al., 2023), while the Critic network (CN) is used to guide the updating of the AN (Li, X. et al., 2023). Nevertheless, due to the poor convergence of the AC algorithm, it is easy for the algorithm to fall into local optima. The SAC algorithm introduces a maximum entropy-based reinforcement learning strategy in the AC algorithm, which can raise the learning speed of the algorithm and avoid its falling into local optima. The SAC algorithm framework is shown in Figure 1.

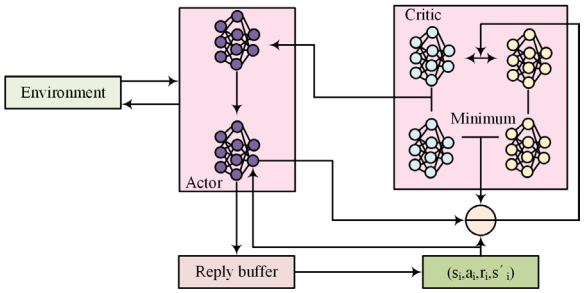


Figure 1. Framework of the SAC algorithm

As it is shown in Figure 1, the SAC algorithm will train four Q networks and the AN simultaneously. The Q network is divided into an evaluation Q network and a target Q network, and the purpose of setting up two corresponding Q networks is to avoid the problem of high estimation of Q values (Sun & Si, 2023). When training a network, if the deviation of the state estimation values is too large, it can lead to divergence problems (Zha et al., 2023). By setting up two value networks, this problem can be effectively avoided (Chen et

al., 2022). The optimization objective of the SAC algorithm is shown in equation (1):

$$J(\pi)=\sum_{t=0}^{T}E_{(s_t,a_t)\sim\rho^{\pi}}\left[r(s_t,a_t)+\lambda H(\pi(\cdot|s_t))\right] \quad (1)$$

In equation (1), $J(\pi)$ denotes the optimization objective. T represents the number of time steps taken by the intelligent agent to interact with the environment in each round. E represents the mathematical expectation. s_t represents the state of the intelligent agent at the time t. a_t represents the action of the intelligent agent at the t moment. ρ^{π} represents the trajectory distribution under the employed strategy. $r(s_t, a_t)$ represents the reward obtained by the intelligent agent for executing various actions in the corresponding state. λ represents the regularization coefficient. $H(.)$ represents the entropy value used to determine the strategy. The strategy function can be obtained by optimizing the Kullback-Leibler divergence, and its calculation is shown in equation (2):

$$J_{\pi}(\theta)=E_{s_t\sim D,a_t\sim\pi_{\theta}}\left[\lambda\log(\pi_{\theta}(a_t|s_t))-Q_{\beta}(s_t,a_t)\right] \quad (2)$$

In equation (2), $J(\theta)$ denotes the policy function. π^{θ} represents the AN. θ denotes the parameters of the AN. Q_{β} represents the CN. β represents the parameters of the CN. D represents the experience replay pool for storing training samples. When constructing AD decision-making intelligent agents, it is also necessary to design the state space (Khan et al., 2023). Considering that the state space is the foundation of decision agents, it directly affects the convergence of algorithms and the quality of decisions. Therefore, when constructing the state space, it is critical to ensure that the state space is beneficial for the training speed of the model (Belattar et al., 2022). To cut down the training time and to reduce the difficulty of the algorithm, this study uses structured data to construct a state space. Due to the overfitting problem for SAC, the intelligent agent finds it difficult to adapt to changing environments. Therefore, to raise the adaptability of the SAC algorithm to the environment, research was conducted with the purpose of making decisions based on the movement status of environmental obstacles and feasible roads. Meanwhile, to improve the generalization ability of the intelligent agent, the maximum number of adjacent vehicles in the state space was set as 6. Moreover, to better express the positional relationship between vehicles, this study converts the state information for relevant vehicles into the Frenet coordinate system, and uses the current position of the vehicle as the origin of coordinates. The longitudinal distance range between the vehicle and the vehicles in front of it and behind it is -30 to 100 meters, the left and right distance range is -5.25 to 5.25 meters, the speed range of participants is 0 to 15 m/s, and the width range of obstacles is 0 to 2 meters. At the same time, to reduce the complexity of the network, the research also converts lane information into obstacle information. The lateral distance between this vehicle and the centerline of the lane ranges from -1.75 to 1.75 meters, and the absolute speed range of this vehicle and the range for the angle between the lane lines are 0 to 15 m/s and -π/3 to π/3, respectively. Besides, the action space has a crucial impact on the search efficiency of the SAC algorithm. For AD tasks, each action includes three dimensions, as well as throttle, steering wheel rotation angle, and braking, with significant differences in their range of variation. Therefore, it is necessary to normalize them (Li, D. et al., 2023). Due to the close relationship between AD decision-making and trajectory planning, convex optimization methods were adopted in the trajectory planning module, and for the decision planning scheme a horizontal and vertical decoupling scheme was adopted (Li, M. et al., 2023). The action space constructed in the study includes 7 action quantities, as well as the recommended values for left and right lane changing, lateral offset of left and right roads, lateral offset of middle lane, lane keeping and speed, with values ranging from 0 to 1, -1.75 to 1.75 m, -1.75 to 1.75 m, 0 to 1, and 0 to 15 m/s, respectively. By using the above-mentioned methods, the state space and action space can be constructed. As it was mentioned earlier, SAC includes a total of 6 networks, where the input and output of the AN are the current state and action, respectively. The input of the Critic state evaluation network is a combination of state and action, while the input of the Critic value evaluation network is a combination of action and value. The AN structure is illustrated in Figure 2.

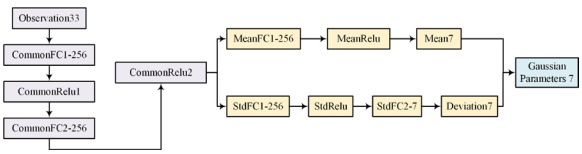


Figure 2. The Structure of the AN

Based on Figure 2, the AN is broken into the input layer, fully connected layer (FCL), activation function layer, and hidden layer. After passing through the activation function layer, the data

is divided into two groups, which are processed through the hidden layer, activation function layer, and FCL, and then integrated through the connection layer function to output action values. The CN structure is shown in Figure 3.

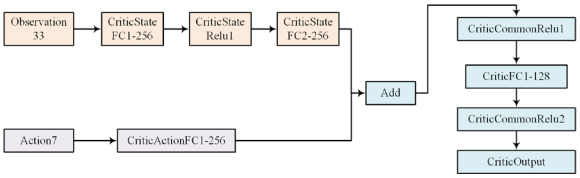


Figure 3. Critic network structure

As it is shown in Figure 3, the state evaluation network concatenates the state data and action data before calculating the state value, and then sequentially passes through the FCL, the activation function layer, and the action input layer. The data processing methods of the value evaluation network and the state evaluation network are consistent. By using the above methods, an AD decision-making intelligent agent based on the SAC algorithm has been constructed. But when faced with complex tasks, the agent finds it difficult to accurately represent environmental semantics. Therefore, to improve this issue, the LSTM model is introduced in this study to enhance the accuracy of AD decisions. As a nonlinear model, LSTM can be used as a complex nonlinear unit to construct larger deep neural networks. At the same time, LSTM can also preserve important features (main features associated with the vehicle status) through various gate functions, which can effectively slow down the gradient vanishing or exploding that may occur in long sequence problems. The main feature of LSTM is its gate structure control, including forget gate, input gate, memory unit, and output gate. Its function is equivalent to adding a "processor" to determine the usefulness of information, enabling it to better process time series tasks. At the same time, it solves the long-term dependency problem of RNN and alleviates the "gradient vanishing" problem caused by backpropagation during RNN training. The formula for calculating the forget gate of LSTM is given in equation (3):

$$f_k = \sigma\left(W_f X_k + U_f h_{k-1} + b_f\right) \tag{3}$$

In equation (3), f_k represents the forget gate, $\sigma(.)$ represents the Sigmoid function, W_f represents the weight of the input data for the forget gate, X_k represents the input at time k; U_f represents the weight of the hidden layer output at the previous moment, h_{k-1} represents the output of the hidden layer at the previous moment, and b_f represents the bias of the forget gate. The calculation formula for the input gate is given in equation (4):

$$i_k = \sigma\left(W_i X_k + U_c h_{k-1} + b_i\right) \tag{4}$$

In equation (4), i_k represents the input gate, W_i represents the input data weight for the input gate, U_c denotes the weight of the hidden layer output at the previous moment, and b_i denotes the bias of the input gate. The calculation formula for the memory unit is shown in equation (5):

$$\bar{C}_k = \tanh\left(W_c X_k + U_c h_{k-1} \_ b_c\right) \tag{5}$$

In equation (5), $\tilde{C}_k$ represents the memory unit at the current time, tanh(.) represents the tanh function, W_c denotes the weight of the input data for the memory unit, U_c represents the weight of the hidden layer output at the previous moment, and b_c denotes the bias of the memory unit. The calculation formula for the hidden layer is shown in equation (6):

$$h_k = o_k \times \tanh\left(C_k\right) \tag{6}$$

In equation (6), C_k represents the memory unit value. The calculation formula for the output gate is shown in equation (7):

$$o_k = \sigma\left(W_o X_t + U_o h_{k-1} + b_o\right) \tag{7}$$

In equation (7), o_k represents the output gate, W_o represents the input data weight for the output gate, U_o represents the weight of the hidden layer output at the previous moment, and b_o represents the bias of the output gate. The improved AN and CN are illustrated in Figure 4.

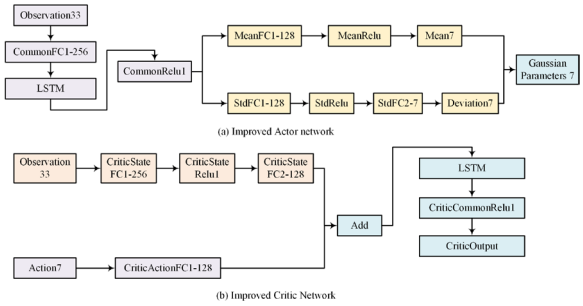


Figure 4. Improved Actor and Critic networks

In Figure 4, in the AN, LSTM was used to replace the first hidden layer, while in the CN, LSTM was used to replace the FCL after the feature layer, and the original input layer was replaced by a sequence input layer. Due to the significant influence of the amount of hidden units in LSTM on its effectiveness, after comprehensive consideration, the number of LSTM hidden units selected for this study was 64.

3.2 Design of Autonomous Driving Decision Execution Strategy Based on SAC

The actions obtained by the decision-making intelligent agent need to be processed before they can be effectively utilized by the planning module to ensure the accurate execution of the strategy. Therefore, for an accurate execution of AD strategies, research was conducted on decision execution strategies. When using SAC to generate paths, considering the continuity between decision frames, the study selected the lane with the highest recommended value as the driving lane. The determination of the start and end points of the decision path is shown in Figure 5.

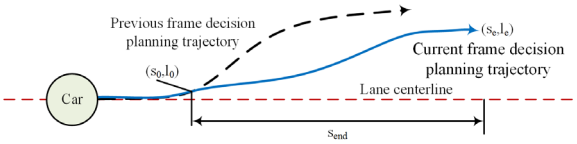


Figure 5. Schematic diagram of the start and end points of the decision path

In Figure 5, the Frenet coordinate system is established based on the centerline of the lane. However, since the decision result for the current frame needs to wait until the next frame is tracked, the projection of the position of the previous frame was selected in the coordinate system as the coordinate origin, and the length of the endpoint corresponding to the lateral offset was determined based on the size of the endpoint of the previous frame. The path fitting formula is shown in equation (8):

$$l = c_5 s^5 + c_4 s^4 + c_3 s^3 + c_2 s^2 + c_1 s + c_0 \quad (8)$$

In equation (8), l represents the path, c_o to c_5 are coefficients and s represents the horizontal axis of the endpoint. By pacing the generated path, it is possible to determine whether there will be a collision between vehicles. For the convenience of calculation, it was assumed that the vehicle is moving at a constant speed, and the speed of the vehicle on the generated path is calculated for an interval of 4 s. The current position of the vehicle is calculated every 0.2 s to obtain its horizontal axis (Sun, 2023). To determine if there is a risk of collision with the chosen vehicle, other vehicles in the surrounding area are driving at a constant speed in the same coordinate system as this vehicle, and are recursive at intervals of 0.2 s. Due to the fact that vehicles only need to consider the rear vehicles from adjacent lanes and the front vehicles from the current lane and adjacent lanes when driving, a maximum number of 5 obstacles can be calculated. When calculating this number, the vehicle is considered as a particle. If the lateral and longitudinal distances between two vehicles are less than 1.5 meters, then a collision has occurred between the two vehicles. At this point, the algorithm proposed in this paper will abandon the decision result for the given path and calculate whether the remaining paths are feasible. If there is a collision risk for all paths, the first recommended road will be used as the path decision. For paths with collision risks, lateral position constraints will be determined based on the current vehicle position and the collision position. The lateral position constraints are shown in Figure 6.

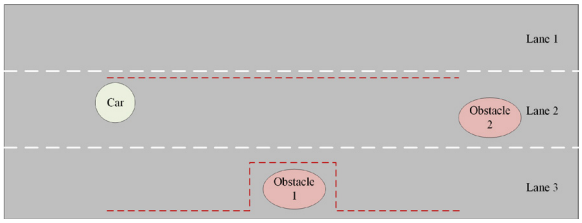


Figure 6. Lateral position constraints for path planning

From Figure 6, when the vehicle chooses a lane change decision, due to the collision risk associated with that decision, the algorithm will delineate lateral position constraints at the collision location. To obtain the optimal path decision, it is necessary to score and evaluate each strategy (Banerjee et al., 2022). Therefore, whether the reward function is reasonable will determine whether the agent can obtain the optimal strategy. In AD, safety is the top priority, followed by efficiency and comfort. Therefore, when constructing the function, safety rewards should be taken into consideration first. The calculation method for safety rewards is expressed in equation (9):

$$r_{safe-long} = \max\left(\left\{\begin{array}{ll} 2*\log\left(\dfrac{s_{ego-obs}}{s_{safe}}\right), & s \le s_{safe} \\ 0, & s > s_{safe} \end{array}\right\}, -2\right) \quad (9)$$

In equation (9), $r_{safe\text{-}long}$ represents the safety reward. $s_{ego\text{-}obs}$ represents the following distance between the current vehicle and the preceding vehicle. s_{safe} represents the safe following distance. In the horizontal direction, if there are obstacles that are too close to the horizontal axis of the vehicle, a penalty will be imposed, and its calculation is given in equation (10):

$$r_{safe-lat1} = \max\left(\left\{\begin{array}{ll} 2*\log\left(\dfrac{D_{ego-obs}}{D_{safe}}\right), & D \le D_{safe} \\ 0, & D > D_{safe} \end{array}\right\}, -2\right) * \frac{|\Delta v|}{15} \quad (10)$$

In equation (10), $r_{safe\text{-}lat1}$ represents the horizontal penalty. $D_{ego\text{-}obs}$ represents the lateral distance when the vehicle collides with an obstacle, D_{safe} represents the lateral safety distance. Δv represents the longitudinal speed difference between the vehicle and the obstacle vehicle. Due to excessive deviation of vehicles from the centerline of the lane, it can also cause safety issues. Therefore, the calculation of the lateral deviation penalty is given in equation (11):

$$r_{safe-lat2} = 0.2 * |l - 0.5| \tag{11}$$

In equation (11), $r_{safe\text{-}lat2}$ represents the lateral offset penalty and l represents the horizontal decision quantity. The formula for the collision penalty is given in equation (12):

$$r_{safe-col} = \begin{cases} -50, & if\ collision \\ 0, & other \end{cases} \tag{12}$$

In equation (12), $r_{safe\text{-}col}$ represents the collision penalty. At this point, the total safety reward is calculated as the sum of equations (9) to (12). With regard to efficiency rewards, if the vehicle reaches the finish line, a one-time reward of 100 points will be given. During the driving process, the path is decomposed, and a reward of 30 points is obtained for each reward point achieved. In addition, to improve driving efficiency, overtaking rewards have been included in the study, and their calculation method is shown in equation (13):

$$\begin{cases} r_{eff-spd} = \dfrac{v}{15} \\ r_{eff-over} = \begin{cases} 2, & over \\ 0, & other \end{cases} \end{cases} \tag{13}$$

In equation (13), $r_{eff\text{-}spd}$ represents acceleration, while v represents the speed of the vehicle. $r_{eff\text{-}over}$ represents the reward for completing an overtake. The total efficiency reward is the sum of the endpoint reward, driving reward, and overtaking reward. As comfort rewards are concerned, the calculation formulas for longitudinal and lateral comfort rewards are given in equation (14):

$$\begin{cases} r_{com-long} = 0.5 * (1 - |\Delta a|) - 0.3 * |jerk| \\ r_{com-lat} = \max\left(-0.05 * \sum_{i=0}^{n} \left(l_i''^2 + l_i'''^2\right), -2\right) \end{cases} \tag{14}$$

In equation (14), $r_{com\text{-}long}$ represents the longitudinal comfort reward. Δa represents the difference between actual acceleration and recommended acceleration and *jerk* represents the rate of change in longitudinal acceleration. $r_{com\text{-}lat}$ represents the lateral comfort reward, while l_i'' and l_i''' represent the second and third derivative of the vehicle's position, respectively. In addition, as changing lanes reduces comfort, penalties will be imposed when changing lanes, and their calculation is given in equation (15):

$$r_{com-lc} = \begin{cases} -2, & lane\ change \\ 0, & other \end{cases} \tag{15}$$

In equation (15), $r_{com\text{-}lc}$ represents the lane change penalty. The planning gap reward is expressed in equation (16):

$$r_{qp} = -0.2 * \left(|\Delta l_e| + |\Delta v_e|\right) \tag{16}$$

In equation (16), r_{qp} represents the planning gap reward, Δl_e represents the lateral position difference at the endpoint and Δv_e represents the speed difference at the endpoint. The total reward can be obtained by summing up the comfort reward and the planning gap reward. By using the above method, the AD decision model with the SAC algorithm has been constructed, and then it can be trained. The training parameters for AN and CN are included in Table 1.

Table 1. Training parameters for Actor and Critic networks

Network	Parameter	Value
Actor network	Target entropy	-log7
	Sampling time	0.1 s
	Initial entropy weight	1
	Learning rate of entropy weight	10^{-4}
	Discount	0.998
	Experience buffer length	10^{6}
	Target smooth factor	10^{-3}
	Minimum batch size	64
Critic network	Learning rate	0.001
	Gradient threshold	1
	Optimizer	Adam
	L2 Regularization factor	0.0002

According to Table 1, the minimum batch size, experience buffer length, and sampling time of the AN are 64, 10^6, and 0.1 s, respectively. The initial entropy weight and target entropy are 1 and - log7, respectively. The learning rate of entropy weight, the gradient threshold, the optimizer, and L2 regularization factor of the Critical network are 0.001, 1, Adam, and 0.0002, respectively. Since the total reward is the sum of safety reward, driving efficiency reward, comfort reward and planning gap reward, and the goal of SAC algorithm is to maximize state value and strategy entropy, its

objective function involves reward and entropy, which can be converted into formula (17):

$$\pi^* = arg\max_{\pi} E_{\tau\sim\pi}[\sum_{t=0}^{\infty}\gamma^t(R_t(s_t,a_t,s_{t+1})+\alpha H(\pi(\cdot|s_t)))] \quad (17)$$

From equation (17), Discount can regulate the effects of short-term and long-term rewards, and the closer the Discount is to 1, the more long-term goals can be considered. The Discount needs to be considered comprehensively with a view to training difficulty and long-term reward. If the reward for the endpoint can be designed at the beginning of a round, a rough calculation can be made according to the number of steps necessary for completing the task.

4. Results and Discussion

To determine the performance of the proposed AD decision strategy based on the SAC-LSTM algorithm, this study validated it using traditional kinematic models as benchmark models. At the same time, the improved algorithm was compared with the baseline model and SAC model in specific deteriorating mixed traffic scenarios.

4.1 Specific Scenario Validation and Analysis

Due to the complexity of the actual driving scenarios, this study analyzed the AD decision-making strategies in lane change overtaking, front vehicle insertion, and front vehicle cutting-out scenarios. During the lane change overtaking test, the vehicle followed the target vehicle at a speed of 15m/s, with a distance of 100m between them. After self-stabilization, the target vehicle underwent a uniform deceleration of 3m/s2. The lane changing overtaking planning results of SAC-LSTM are shown in Figure 7.

As it is shown in Figure 7(a), at 5.9 s, the vehicle output a lane change command and began to shift to the left. At the 6-th second, the distance from the centerline of the lane reached its maximum of -2.4 m. At 9.0 s, it completed the lane change and returned to its position at the centerline of the lane. With regard to Figure 7(b), the vehicle followed the preceding vehicle steadily within the first 3 s, with a driving speed of 15.0 m/s. After 3.0 s, the speed slightly decreased. After issuing the lane change command, the speed was basically maintained at around 14.7 m/s. The above results indicated that the AD decision strategy based on the SAC-LSTM algorithm proposed in this study has a good performance in lane changing scenarios, and can maintain constant speed during lane changing. In the case of the front vehicle insertion scenario, the vehicle's driving speed was still 15 m/s. When the distance between vehicles was 28 m, the front vehicle was inserted ahead of the other vehicle. The decision results for four different algorithms in the scenario of front vehicle insertion are shown in Figure 8.

As it is shown in Figure 8(a), when VT1 crossed the lane line, the GSDLCM model made a lane change decision at 7.9 s and completed the lane change at 10.5 s. The SAC and SAC-LSTM models made lane changing decisions at 7.6 s and 7.2 s, respectively, and completed lane changing at 9.3 s and 9.1 s, respectively. In Figure 8(b), when the front vehicle was inserted, the GSDLCM model and SAC model started to decelerate at 5.9 s and 5.2 s, respectively, with minimum speeds of 6.8 m/s and 9.7 m/s. The SAC-LSTM model started decelerating at 4.2 s, with a minimum speed of 9.6 m/s. The above results indicate that when the front vehicle was inserted, the SAC-LSTM model's deceleration and lane change decision-making timing was faster than for other models. The safety of each algorithm in the scenario of front vehicle insertion is shown in Figure 9.

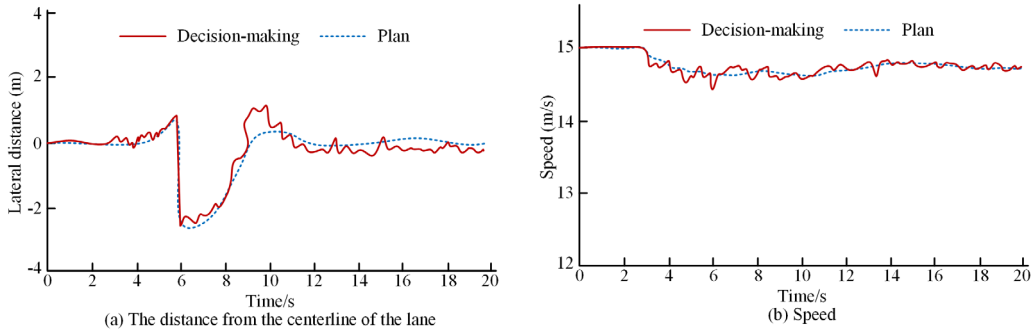
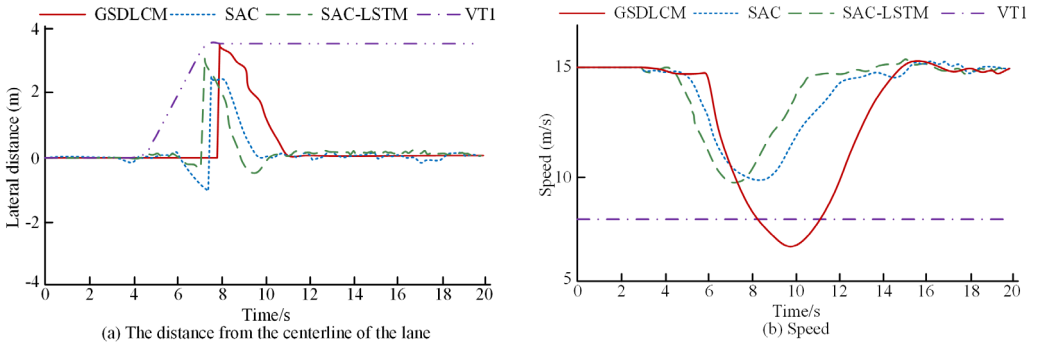


Figure 7. Lane change overtaking planning results for SAC-LSTM

According to Figure 9(a), the IDM model and SAC model obtained the minimum collision time with regard to the preceding vehicle at 6.0 s and 5.2 s, respectively, and at 2.1 s and 3.1 s, respectively. The SAC-LSTM model also obtained the smallest collision time with regard to the preceding vehicle at 5.2 s, but the collision time between the two vehicles was 3.6 s. As it is shown in Figure 9(b), for the IDM model and SAC model the distances from the preceding vehicle when making lane change decisions were 8.3 m and 10.2 m, respectively. When making lane changing decisions, SAC-LSTM was 10.5 m away from the previous vehicle. The above results indicate that in comparison with the IDM model and SAC model, the SAC-LSTM model features a higher decision safety when the front vehicle is inserted. When verifying the decision performance for the front vehicle in the front vehicle cut-out scenario, the driving speeds of the analysed vehicle and the front vehicle VT2 were 15 m/s and 10 m/s respectively, and there was another vehicle following VT2 at a speed of 15 m/s 10 m behind the left side of this vehicle, and there was a bicycle in front of it, to the right. When the distance between VT2 and the preceding vehicle was 40 m, VT2 began to change lanes. The decision results for each algorithm in the front vehicle cut-out scenario are shown in Figure 10.

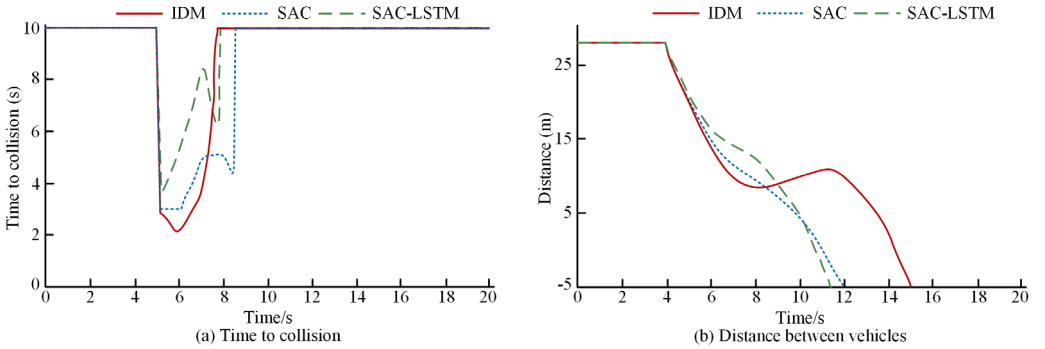
As it can be seen in Figure 10(a), in the scenario where the front vehicle is cut out, the IDM model, SAC model, and SAC-LSTM model made three different decisions. The IDM model



Note: VT1 represents the vehicle inserted ahead

Note: GSDLCM represents the Gipps Safety Distance Lane Changing Model

Figure 8. Decision results for four different algorithms in the front car insertion scenario



Note: The IDM represents the Intelligent Driver Model

Figure 9. Safety of each algorithm in the front car insertion scenario

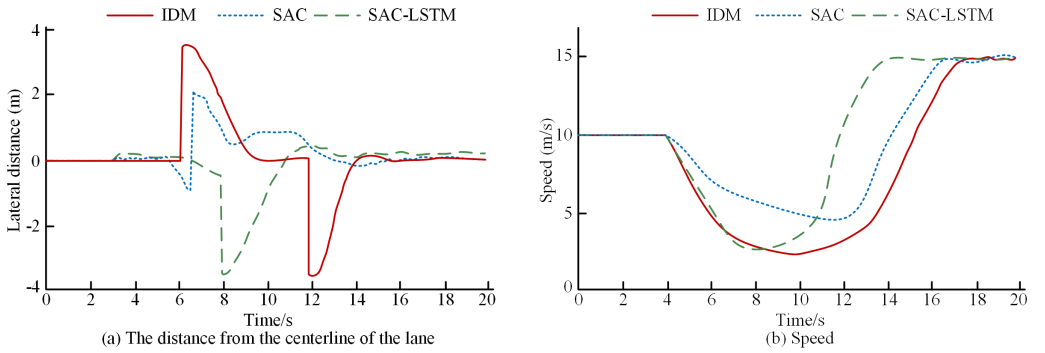


Figure 10. Decision results for each algorithm in the front car cutting-out scenario

made a lane change decision at 6.1 s, driving towards the right lane, and then changed lanes again at 11.9 s, driving towards the left lane. The SAC model made a lane change decision at 6.3 s and drove towards the right lane. The SAC-LSTM model made a decision to change lanes to the left lane at 6.2 s. According to Figure 10(b), all three models started to decelerate at the same time, while SAC-LSTM reached the minimum speed earlier than the other models. The safety of each algorithm in the front car cutting-out scenario is shown in Figure 11.

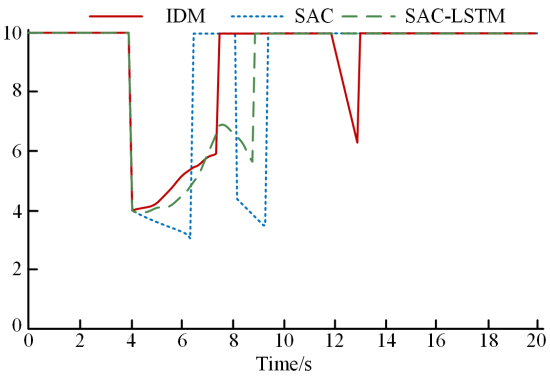


Figure 11. Safety of each algorithm in the front car cutting-out scenario

As it is shown in Figure 11, the minimum collision times for the IDM model, SAC-LSTM and SAC model were 4.0 s, 3.2 s, and 3.9 s, respectively. The minimum collision time for the SAC model occurred when eluding the preceding vehicle VT1, and the collision time for the SAC model was significantly lower in comparison with the other model when eluding bicycles in the right lane, namely between 8.1 s and 9.2 s. Overall, in the front car cutting-out scenario, the safety of the IDM model was not greatly different from the safety of the SAC-LSTM model, while the safety of the SAC model was greatly inferior to the safety of the other two models.

4.2 Validation Analysis for the Pedestrian Crossing and Mixed Traffic Scenarios

In urban transportation, pedestrians are also very important participants, but when facing vehicles, pedestrians are in a disadvantaged position. Therefore, to verify the impact of AD decision models on pedestrian safety, an analysis was carried out to verify the decision results and safety for pedestrian crossing scenarios. In the case of the pedestrian crossing scenario, the vehicle was traveling at 10 m/s. When they were 35 m away from the vehicle, pedestrians began to cross the road at a speed of 1.5 m/s. Meanwhile, bicycles with a speed of 3 m/s were starting to come forward in the right lane. Since there is no baseline model for pedestrian forward interpolation, only the SAC and SAC-LSTM models were compared. The decision results for each algorithm in the pedestrian crossing scenario are shown in Figure 12.

According to Figure 12, when pedestrians crossed into the lane where the vehicle is located, the SAC model first decelerated to avoid pedestrians, and when facing a bicycle that was suddenly inserted in front of the vehicle, it braked sharply and changed lanes. The SAC-LSTM model decelerated and changed lanes to the left when pedestrians crossed into the lane where the vehicle was located, so there was no need to face a bicycle that was suddenly inserted in front of the vehicle. As it can be seen in Figure 12(a), the SAC and SAC-LSTM models made lane change decisions at 6.8 s and 7.5 s, respectively. Further on, as it is shown in Figure 12(b), the SAC model and SAC-LSTM model began to decelerate at 6.0 s and 3.7 s, respectively, and reached their minimum speeds at 10.0 s and 8.3 s, respectively. The average vehicle speeds under the decision-making of the SAC model and SAC-LSTM model were 10.28 m/s and 10.29 m/s, respectively, which are basically

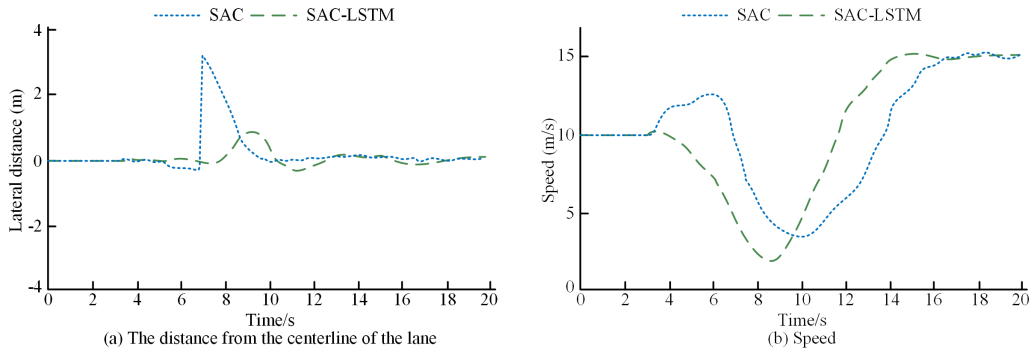


Figure 12. Decision results for each algorithm in the pedestrian crossing scenario

the same. The decision safety for the SAC and SAC-LSTM models is shown in Figure 13.

As it is shown in Figure 13(a), the collision times for the SAC model and the SAC-LSTM model were the smallest at 8.1 s and 8.3 s, respectively. The minimum collision times for the two models were 0.3 s and 1.2 s, respectively. That is, the collision time was higher for the SAC-LSTM model than for the SAC model. As it can be seen in Figure 13(b), at 8.1 s, the SAC model was changing lanes and the analysed vehicle almost collided with the bicycle. At 12.7 s, although the distance traveled by the bicycle and the vehicle in the SAC-LSTM model was the same, there was no risk of collision at this time because the SAC-LSTM model had already completed lane changing. The above results indicate that in comparison with the SAC model, the SAC-LSTM model features a higher safety when facing pedestrian crossings. Due to the fact that vehicles often face different scenarios during actual driving, to verify the generalization ability of AD decision models, an analysis was carried out to verify the decision safety and comfort of the employed models in mixed traffic scenarios. When conducting mixed traffic scenario validation, the traffic density was 20, the number of pedestrians and non-motorized vehicles was 8, and the number of stationary obstacles was 6. The safety, and the efficiency and comfort for each algorithm in mixed traffic scenarios are depicted in Figure 14.

According to Figure 14(a), the average collision frequency, average lane change frequency, average following distance, and average distance from the lane centerline for the baseline model were 1.4 m, 8.5 m, 43 m, and 0.12 m, respectively. The values of these parameters for the SAC model were 0.9 m, 11.2 m, 21 m, and 0.27 m, respectively, while those achieved by the SAC-LSTM model were 0.9 m, 9.8 m, 19 m, and 0.36 m, respectively. According to Figure 14(b), the average arrival times for the baseline model, the SAC model, and the SAC-LSTM model were 167 s, 132 s, and 127 s, respectively. The values for the root mean square (RMS) of acceleration were 0.78, 1.1, and 1.1, respectively. The values for the RMS of the acceleration change rate were 0.85, 1.4, and 1.3, respectively. According to the above results, the safety of the SAC-LSTM model is higher than that of the baseline model and of the SAC model, and the SAC-LSTM model provides a higher comfort and flexibility. In order to further understand the driving decision-making performance of the proposed SAC-LSTM model, it was compared with the state-of-the-art Deep Q-Network Long Short-Term Memory Self Attention (DQN-LSTM-SAT) model and with the Bayesian Network RoboSim (BNRoboSim) model.

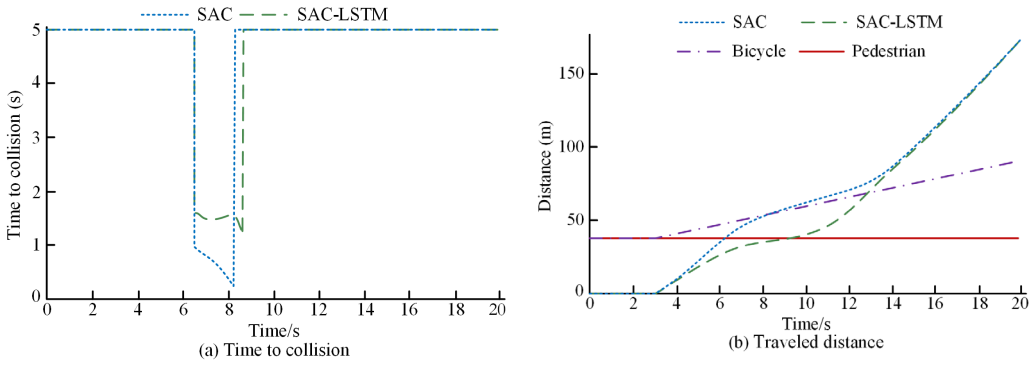
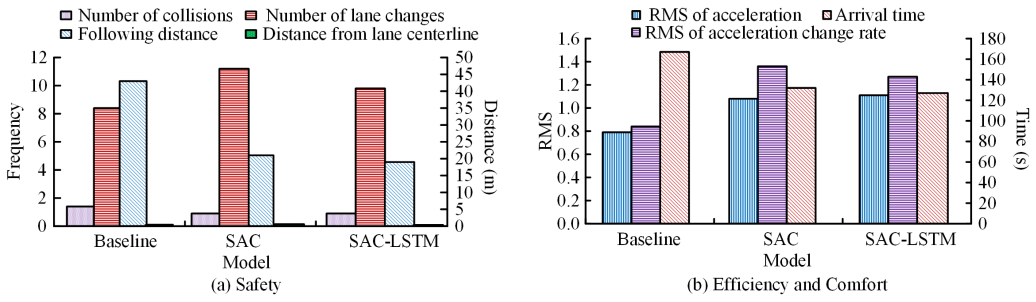


Figure 13. Decision safety for the SAC and SAC-LSTM models



Note: The baseline models are the IDM and the GSDLCM.

Figure 14. Safety and comfort of each algorithm in mixed traffic scenarios

The results for the decision-making performance of the three models are shown in Figure 15.

As it can be seen in Figure 15(a), the DQN-LSTM-SAT and BNRoboSim models chose to slow down in front of the people who were crossing the street on the pedestrian crossing, while in the face of sudden forward crossing, the SAC-LSTM model simultaneously slowed down and changed lanes to the left, in order to achieve the purpose of avoiding pedestrians and bicycles simultaneously. According to Figure 15(b), both the DQN-LSTM-SAT and BNRoboSim models basically started to slow down at around 5.0s, reaching the minimum speed at 8.9s and 9.7s, respectively. However, the SAC-LSTM model started to decelerate at 3.7s and reached a minimum speed at 8.3s. The decision safety for the three algorithms is illustrated in Figure 16.

According to Figure 16(a), the collision times for both the DQN-LSTM-SA and the BNRoboSim models are the smallest at 8.1 seconds, with minimum collision times of 0.9 seconds and 0.7 seconds, respectively. The collision time for the SAC-LSTM model is the smallest at 8.3 seconds, with a minimum collision time of 1.2 seconds. The collision time for the SAC-LSTM model is higher than that achieved by the two other models. In Figure 16(b), it can be seen that all three models managed to avoid a collision with bicycles and pedestrians, but the SAC-LSTM model features a larger distance between pedestrians and bicycles. It can be seen that the SAC-LSTM model features a higher safety. In order to verify the comfort provided by the decision-making algorithms, the acceleration values for the previous front vehicle insertion scenario were compared as an example. The acceleration for each algorithm is shown in Figure 17.

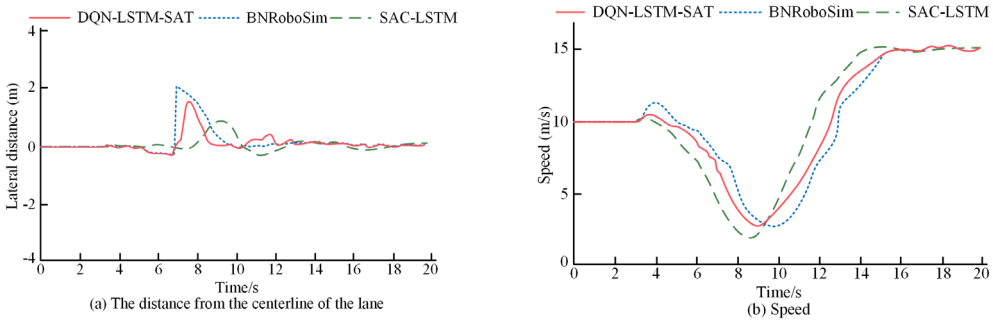


Figure 15. The results for the decision-making performance of the three algorithms

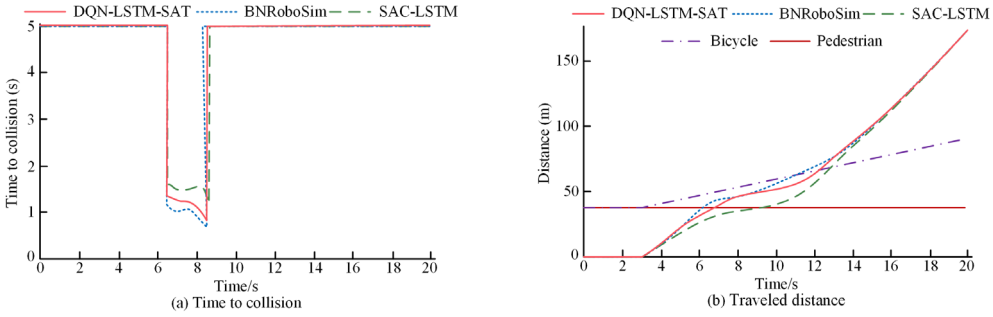


Figure 16. The decision safety for the three algorithms

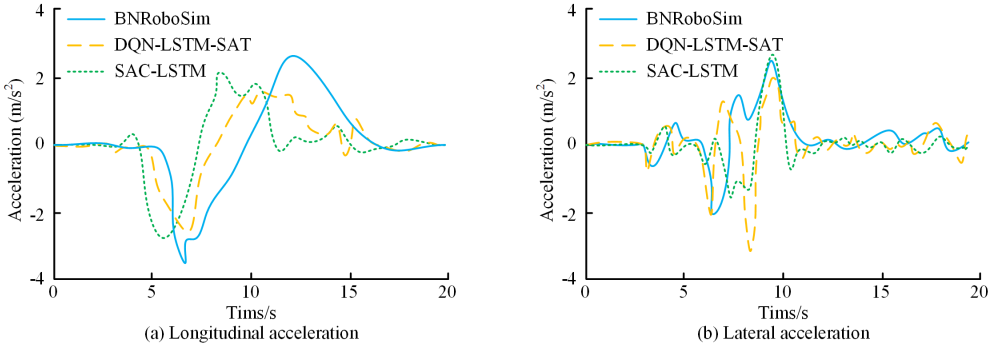


Figure 17. Acceleration for the three algorithms

In Figure 17, it can be seen that for the front vehicle insertion scenario, the target vehicle of BNRoboSim experiences a sudden brake when entering this lane, with a deceleration of -5m/s2 and a high longitudinal deceleration rate, resulting in a significant impact. Although the DQN-LSTM-SA and the SAC-LSTM models feature smaller impacts, the SAC-LSTM model is more stable in the subsequent deceleration stage. The root mean square values for the longitudinal acceleration of the three algorithms are 1.22, 1.09, and 1.10, respectively, and the root mean square values for the lateral acceleration of the three algorithms are 1.17, 1.09, and 0.98, respectively. The above results indicate that in comparison with the other two algorithms, the SAC-LSTM model provides a higher comfort.

5. Conclusion

In the autonomous driving system, the decision-making system, as an important component, has a great influence on the effectiveness of the entire driving system. A good decision-making system can effectively ensure the safety of drivers and pedestrians, and ensure driving efficiency. Therefore, to enhance the safety and driving efficiency in the context of AD decision-making, a decision model for AD based on the SAC algorithm and a LSTM network is proposed and tested. The experimental findings showed that in the front vehicle insertion scenario, the GSDLCM model made a lane change decision at 7.9 s and completed the lane change at 10.5 s. The SAC and SAC-LSTM models made lane changing decisions at 7.6 s and 7.2 s, respectively, and completed lane changing at 9.3 s and 9.1 s, respectively. At this point, the minimum collision times for the IDM model, the SAC model, and the SAC-LSTM model were 2.1 s, 3.1 s, and 3.6 s, respectively. In the pedestrian crossing scenario, the SAC model first decelerated to avoid pedestrians, and when facing a bicycle that was suddenly inserted in front of the vehicle, it braked sharply and changes lanes. The SAC-LSTM model decelerated and changed lanes to the left when pedestrians crossed into the lane where the vehicle was located, so there was no need to face a bicycle that was suddenly inserted in front of the vehicle. At this point, the minimum collision times for the two models were 0.3 s and 1.2 s, respectively. For a mixed traffic scenario, the average number of collisions for the baseline model, the SAC model, and the SAC-LSTM model was 1.4, 0.9, and 0.9, respectively. The outcomes illustrated that the safety of the SAC-LSTM model was superior to the other employed models for both single and mixed traffic scenarios. However, due to the fact that this study only considered the nearest obstacle in relation to the analysed vehicle when the proposed model was developed, the model lacks foresight. Therefore, further research on decision models will be conducted in the future, taking into account more obstacles.

REFERENCES

Banerjee, C., Chen, Z., Noman, N. & Zamani, M. (2022) Optimal actor-critic policy with optimized training datasets. *IEEE Transactions on Emerging Topics in Computational Intelligence*. 6(6), 1324-1334. doi: 10.1109/TETCI.2022.3140375.

Belattar, K., Zemali, E. A., Baouni, S. & Dehni, S. (2022) Parallel multiple DNA sequence alignment using genetic algorithm and asynchronous advantage actor critic model. *International Journal of Bioinformatics Research and Applications*. 18(5), 460-478. doi: 10.1504/IJBRA.2022.128236.

Chen, P., Liu, S., Zhang, D. & Yu, L. (2022) A deep asynchronous actor-critic learning-based event-triggered decentralized load frequency control of power systems with communication delays. *International Journal of Robust and Nonlinear Control*. 32(5), 3039-3061. doi: 10.1002/rnc.5516.

Chen, L., Dong, C., He, S., & Dai, S.-L. (2023) Adaptive optimal formation control for unmanned surface vehicles with guaranteed performance using actor-critic learning architecture. *International Journal of Robust and Nonlinear Control*. 33(8), 4504-4522. doi: 10.1002/rnc.6623.

Dey, S. & Xu, H. (2023) Hierarchical game theoretical distributed adaptive control for large scale multi-group multi-agent system. *IET Control Theory & Applications*. 17(17), 2332-2352. doi: 10.1049/cth2.12506.

Du, L., Sun, B., Huang, X., Wang, X. & Li, P. (2023) A Learning-Based Nonlinear Model Predictive Control Approach for Autonomous Driving. *IFAC – PapersOnLine*. 56(2), 2792-2797. doi: 10.1016/j.ifacol.2023.10.1388.

Huang, C., Lv, C., Hang, P. & Xing, Y. (2021) Toward safe and personalized autonomous driving: Decision-making and motion control with DPF and CDT techniques. *IEEE/ASME Transactions on Mechatronics*. 26(2), 611-620. doi: 10.1109/TMECH.2021.3053248.

Huang, T., Song, S., Liu, Q., He, W., Zhu, Q. & Hu, H. (2023) A novel multi-exposure fusion approach

for enhancing visual semantic segmentation of autonomous driving. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*. 237(7), 1652-1667. doi: 10.1177/09544070221097851.

Khan, A. W., Al-Obeidat, F., Khalid, A., Amin, A. & Moreira, F. (2023) Sentence embedding approach using LSTM auto-encoder for discussion threads summarization. *Computer Science and Information Systems*. 20(4), 1367-1387. doi: 10.2298/CSIS221210055K.

Li, X., Xiao, J., Cheng, Y. & Liu, H. (2023) An actor-critic learning framework based on Lyapunov stability for automatic assembly. *Applied Intelligence*. 53(4), 4801-4812. doi: 10.1007/s40684-023-00547-y.

Li, D., Yang, R. & Su, C. (2023) Generative adversarial network based on LSTM and convolutional block attention module for industrial smoke image recognition. *Computer Science and Information Systems*. 20(4), 1707-1728. doi: 10.2298/CSIS221125027L.

Li, M., Zhang, Z., Lu, M., Jia, X., Liu, R., Zhou, X. & Zhang, Y. (2023) Internet Financial Credit Risk Assessment with Sliding Window and Attention Mechanism LSTM Model. *Tehnički vjesnik [Technical Gazette]*. 30(1), 1-7. doi: 10.17559/TV-20221110173532.

Liao, L., Xiao, H., Xing, P., Gan, Z., He, Y. & Wang, J. (2024) Perception Enhanced Deep Deterministic Policy Gradient for Autonomous Driving in Complex Scenarios. *Computer Modeling in Engineering & Sciences*. 140(1), 557-576. doi: 10.32604/cmes.2024.047452.

Madan, K. & Bhatia, R. (2021) Crawling the Deep Web Using Asynchronous Advantage Actor Critic Technique. *Journal of Web Engineering*. 20(3), 879-902. doi: 10.13052/jwe1540-9589.20314.

Nan, J., Deng, W. & Zheng, B. (2022) Intention prediction and mixed strategy Nash equilibrium-based decision-making framework for autonomous driving in uncontrolled intersection. *IEEE Transactions on Vehicular Technology*. 71(10), 10316-10326. doi: 10.1109/TVT.2022.3186976.

Sun, H. (2023) Optimizing manufacturing scheduling with genetic algorithm and LSTM neural networks. *International Journal of Simulation Modelling*. 22(3), 508-519. doi: 10.2507/IJSIMM22-3-CO13.

Sun, Q. & Si, Y.-W. (2023) Supervised actor-critic reinforcement learning with action feedback for algorithmic trading. *Applied Intelligence*. 53(13), 16875-16892. doi: 10.1007/s10489-022-04322-5.

Wang, H., Wang, W., Yuan, S. & Li, X. (2022) Uncovering interpretable internal states of merging tasks at highway on-ramps for autonomous driving decision-making. *IEEE Transactions on Automation Science and Engineering*. 19(4), 2825-2836. doi: 10.1109/TASE.2021.3103179.

Wang, W., Qie, T., Yang, C., Liu, W., Xiang, C. & Huang, K. (2022) An intelligent lane-changing behavior prediction and decision-making strategy for an autonomous vehicle. *IEEE Transactions on Industrial Electronics*. 69(3), 2927-2937. doi: 10.1109/TIE.2021.3066943.

Wu, R., Yao, Z., Si, J. & Huang, H. H. (2022) Robotic knee tracking control to mimic the intact human knee profile based on actor-critic reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*. 9(1), 19-30. doi: 10.1109/JAS.2021.1004272.

Wu, G., Wang, G., Bi, Q. Wang, Y., Fang, Y., Guo, G. & Qu, W. (2023) Research on unmanned electric shovel autonomous driving path tracking control based on improved pure tracking and fuzzy control. *Journal of Field Robotics*. 40(7), 1739-1753. doi: 10.1002/rob.22208.

Xu, C., Zhao, W., Liu, J., Wang, C. & Lv, C. (2022) An integrated decision-making framework for highway autonomous driving using combined learning and rule-based algorithm. *IEEE Transactions on Vehicular Technology*. 71(4), 3621-3632. doi: 10.1109/TVT.2022.3150343.

Yuan, M., Shan, J. & Mi, K. (2022) Deep reinforcement learning based game-theoretic decision-making for autonomous vehicles. *IEEE Robotics and Automation Letters*. *7*(2), 818-825. doi: 10.1109/LRA.2021.3134249.

Zha, Z., Wang, B. & Tang, X. (2023) Evaluate, explain, and explore the state more exactly: an improved Actor-Critic algorithm for complex environment. *Neural Computing and Applications*. 35(17), 12271-12282. doi: 10.1007/s00521-020-05663-3.

Zhang, J. & Xue Q. (2022) Actor-critic-based decision-making method for the artificial intelligence commander in tactical wargames. *Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*. 19(3), 467-480. doi: 10.1177/1548512920954542.

Zhang, T., Zhan, J., Shi, J., Xin, J. & Zheng, N. (2023) Human-Like Decision-Making of Autonomous Vehicles in Dynamic Traffic Scenarios. *IEEE/CAA Journal of Automatica Sinica*. 10(10), 1905-1917. doi: 10.1109/JAS.2023.123696.

Zhang, Y., Song, P., Jing, Q. & Li, Q. (2023) A Novel Point Cloud Compression Algorithm for Vehicle Recognition Using Boundary Extraction. *Tehnički vjesnik [Technical Gazette]*. 30(6), 1899-1910. doi: 10.17559/TV-20230507000612.

Zhang, D., Cui, Y., Xiao, Y., Fu, S., Cha, S. W., Kim, N., Mao, H. & Zheng, C. (2024) An Improved Soft Actor-Critic-Based Energy Management Strategy of Fuel Cell Hybrid Vehicles with a Nonlinear Fuel Cell Degradation Model. *International Journal of Precision Engineering and Manufacturing - Green Technology*. 11(1), 183-202. doi: 10.1007/s40684-023-00547-y.