Journal of Theoretical and Applied Information Technology
15th July 2023. Vol.101. No 13
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DRIVING MODE SWITCHING MECHANISM BASED ON REINFORCEMENT LEARNING

YILIN ZHU[1], SANG-CHUL KIM[2]
[1,2]School of Computer Science, Kookmin University, Korea
E-mail: [1]z514201255@gmail.com, [2]sckim7@kookmin.ac.kr ([2]Corresponding author)

ABSTRACT

Autonomous driving technology can improve traffic efficiency and safety, making the driving process more convenient and comfortable, thus increasing road safety and reducing environmental pollution. In this study, we explore the mode switching problem between automatic and manual driving and propose a driving mode switching mechanism based on deep reinforcement learning. First, we study the background on driving scenes based on safety degree calculation. Then, a real-time online switching mechanism based on deep reinforcement learning is presented. Finally, the proposed algorithm's effectiveness is verified using a simulation platform. The simulation results show that based on the analysis of lateral position deviation, heading angle deviation, and safety index data, the deep RL method proposed in this project can effectively realize the switch between manual and automatic driving modes. As the results, the mode switching strategy can respond well to the vehicle's lateral position deviation, heading angle deviation, and longitudinal safety degree and thus improve vehicle operation safety.

Keywords: *Autopilot, Intensive Learning, Actor–Critic Framework*

1. INTRODUCTION

In the last several years, with the increasing global trade and the progress of science and technology, the number of vehicles worldwide has increased dramatically. Road networks have become more and more complex and traffic safety, road congestion, and other issues have become increasingly severe. Also, traffic accidents have become more frequent, causing a large number of casualties and property losses. Uncivilized driving behaviors such as lane changing, illegal overtaking, turning around, and not maintaining a safe distance are the main reasons for traffic accidents [1][2]. Autonomous driving technology can improve traffic efficiency and safety, making the driving process more convenient and comfortable, and thus increasing road safety and reducing environmental pollution.

Deep reinforcement learning has developed rapidly in recent years. In some aspects, the agents trained by reinforcement learning can perform close to or even beyond the human level. Compared with rule-based methods, automatic driving decision-making technology based on deep reinforcement learning does not need to establish rules manually. When encountering an unknown environment, the model can be improved through continuous interaction with the environment. The behavioral decision-making method based on deep reinforcement learning can adapt to more conditions and has strong robustness and adaptive ability. In complex traffic scenes, the state space and action space of automatic driving are extremely large. Therefore, it is difficult to establish a decision rule that can be mapped from the state space to the action space manually. Also, if the artificially established rules encounter undefined conditions, serious traffic accidents may happen. Therefore, deep reinforcement learning has promising application value in the study of safe and reliable switching of driving modes during vehicle operation.

2. RELATED WORKS

Currently, there are generally three solutions for the development of unmanned driving technology [3]. The first is the most common solution, which realizes a complex driving control framework using modular construction of the driving system. This solution is called the modular pipeline (MP) [4]. The second scheme is based on human driving data, known as imitation learning (IL) [5]. The third method is learning through the interactive data between vehicles and the environment, also known as reinforcement learning (RL) [6].

To meet the requirements for commercialization and safety regulations, most researchers and commercial companies in the automotive industry choose to adopt the MP scheme [7]. Although the academia is the frontier of scientific and technological innovation, there is still relatively little research on end-to-end unmanned driving using the latter two schemes, such as Drive Lab, an unmanned driving laboratory from NVIDIA, and WAYVE in the UK [8]. Because of the rise of artificial intelligence, many deep learning methods are now being used for end-to-end control of robots, unmanned driving perception, and multi-sensor fusion. However, in the decision-making control of unmanned driving, driving strategies are still designed based on traditional artificial rules [9]. Therefore, considering the deep RL's great potential for unmanned driving and the lack of relevant research at this stage, it is very meaningful to explore the work on decision-making control in unmanned driving using deep RL methods.

There are two main decision-making methods for automatic driving behavior: the rule-based decision-making method and the learning-based decision-making method. The rule-based decision-making scheme is currently the most commonly used. In this method, according to the characteristics of the surrounding road environment, the road surface is divided into several types of driving scenes. Then, the rule base is established based on the experience of human drivers. The decision-making results of the autonomous vehicle in various scenarios are determined according to the preset rule base. Among many rule-based decision-making methods, the finite state machine [10] is the most frequently used because of its strong practicality and clear logic, which is easy to debug and implement.

The focus of this study is on the safe and reliable switching mechanism of the driving mode based on deep RL. Deep RL combines the idea of deep learning based on RL. It uses the powerful fitting capability of deep learning to approximate the model required by RL iteratively. This chapter will introduce the relevant theoretical basis from three aspects: RL, deep RL, and the Actor–Critic framework.

RL, also known as evaluation learning, is an essential branch of machine learning [11][12][13]. It has no prior knowledge, does not need to give correct action, and only needs to provide a return and then adjusts the strategy to maximize the total cumulative return.

Whether it is a value function or a policy gradient method, the fundamental purpose is to optimize the policy and allow the agent to take better actions. However, there is a big difference between the two with regard to how to achieve this purpose.

2.1 Value Function

The value function method is an evaluation of the policy π. The value functions include the state-value functions and the state-action value function. The state-value functions refer to the total expected return $V^{\pi}(s)$ obtained by executing policy π in state s, and the state-action value functions refer to the total expected return obtained by taking action a in state s and performing policy π. The state-action value function is often referred to as the Q function, denoted as $Q^{\pi}(s,a)$. In general RL problems, Q-functions are usually used due to the complexity of the problem. It is known that the value function is the evaluation of policy π. When the space of policy π is a finite state-action space, argmax can be determined for all Q-functions to obtain the optimal policy. However, this method is often challenging to use because even though the number of states and actions is limited, their strategy space is usually a geometric multiple. Currently, the most common operation is to carry out policy iteration, which can be divided into deterministic and random strategies. The deterministic strategy is

$a = \mu(s)$.

The expression form of the random strategy is

$a \sim \pi(\theta|s)$.

The advantage of the deterministic strategy is that it requires less sampling data and the algorithm is efficient. On the other hand, the advantage of the random strategy lies in its ability to explore.

2.2 On-policy and Off-policy

Deterministic policy algorithms are efficient but cannot explore the environment. To solve this problem, the concepts of off-policy and on-policy are introduced. When iterating a policy, the method is to follow an established or new policy. If it follows the established policy, it is an on-policy. Otherwise, it is an off-policy.

Deep learning is one of the branches of machine learning, which has achieved a lot in image recognition and natural language processing [14][15][16]. A simple explanation of deep learning is that when you know the input and mapping, you can get its output. However, in actuality, mapping is complex and cannot be expressed by expression, or it is simply an unknown state. Currently, deep learning constructs a deep neural network as the bridge between sample input and feature output. It continuously inputs samples for training so the neural network can fit and approximate a mapping corresponding to the samples as much as possible. Finally, the network completed by training can be approximately equivalent to the required mapping.

Q-learning is one of the most common and widely used RL algorithms. However, when encountering more complex situations, the method of updating tables can no longer meet the requirements. Therefore, we have to look for other methods to replace tables to describe the status and actions. Since the number of input and output neurons of the neural network can be determined at will, introducing deep learning into the RL framework and replacing the original Q table with the neural network can significantly improve the operation of the RL algorithm and can handle higher dimensional and more complex situations. In other words, the neural network provides the perception ability lacking in RL. By combining the fitting and approximation ability of deep learning with RL's ability to generate strategies, and transforming the neural network from table to network form, deep RL can be achieved.

The Actor–Critic algorithm is a progressive iterative algorithm that combines a policy and a value function, known as the Actor–Critic framework [17][18][19]. With the Actor–Critic framework as the core, many excellent deep RL algorithms have emerged over the years, such as DDPG, A3C, and PPO.

Let us analyze several essential branches of machine learning. The algorithm development process discusses the sequence from RL to deep RL to the Actor–Critic framework. First, we analyze the theoretical principles of RL and expound on their advantages, disadvantages, and characteristics from the following aspects: RL based on value function or strategy function and intensive learning of policy or off-policy. Then, because RL has difficulty in dealing with high-dimensional state problems, we introduce the principle of deep RL. Deep RL skillfully combines the fitting approximation ability of deep learning with the generation strategy ability of RL, realizing the transformation of RL from table to neural network and significantly improving the computing ability and application range. Finally, this chapter introduces the Actor–Critic framework, a popular deep RL framework in recent years. This framework uses Actor and Critic networks to update parameters alternately. Currently, many deep RL algorithms are based on the Actor–Critic framework.

3. PROPOSED ALGORITHM

We introduce the method of achieving smooth mode switching between automatic driving and manual driving, including the evaluation of driving safety, the design of the mode switching module, and the algorithm design based on deep RL.

3.1 Safety Algorithm

This project takes a typical three-lane scenario as an example to determine and calculate the lane safety degree in automatic driving, which requires vehicles to meet the minimum safe distance. As shown in Fig. 1, the minimum safe distance includes (1) the driving distance of the driver's reaction time, (2) the braking deceleration distance, and (3) the stopping distance of traffic vehicles at the maximum deceleration speed. Based on the setting of the minimum safe distance, the project gives the calculation formula for this distance, as shown in Eq. (1).

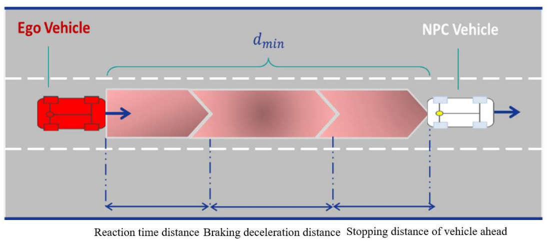
Ego Vehicle
d_{min}
NPC Vehicle
Reaction time distance
Braking deceleration distance
Stopping distance of vehicle ahead

Figure 1. Schematic Diagram of Minimum Safe Distance for Intelligent Driving Vehicles

$$d_{min} = v_{\text{ego}} * t + \frac{1}{2} a_{\text{ego}} * t^2 + \frac{(v_{\text{ego}} + a_{\text{ego}} * t)^2}{2a_{\text{ego}}} + \frac{(v_{\text{npc}})^2}{2a_{\text{npc}}}, \quad (1)$$

where, v_{ego} , a_{ego} represent the speed and acceleration of the vehicle, v_{npc}, a_{npc} are the speed and acceleration of the traffic vehicle, t is the driver's reaction time, and d_{min} denotes the calculated minimum safe distance. Using Eq. (1), this project can calculate the security threshold for situational assessment, as shown in Eq. (2):

$$T_{\text{therod}} = \frac{d_{\min}}{v_{\text{ego}} - v_{\text{npc}}}. \quad (2)$$

It can be seen from Eq. (2) that the calculation result of the safety threshold T_{therod} is closely related to the speed v_{ego} of the vehicle and the speed v_{npc} of the traffic vehicle.

To ensure the safety of vehicle operation, it is essential to develop a real-time online human–computer fusion autonomous mode switching mechanism. This project defines the mode switching process (module) of the autonomous vehicle as an optimal control problem based on RL and develops a man–machine integration mode switching controller with autonomous capability to cope with the driving safety caused by the external environment in the process of vehicle movement.

RL is an online optimization control method based on environment interaction, which depends on the description of the controlled object and the current state of the operating environment. State information is used as the input to the mode switching strategy, which defines the environmental observations received at each time step. Here, the lateral position offset difference, *e*, of the vehicle, the heading angle deviation, $\Delta\varphi$, the three-lane safety degree, and a total of five parameters are selected to form the state space as shown in Eq. (3).

$$S_{\text{t}} = \{e, \Delta\varphi, \text{left}_{\text{safe}}, \text{current}_{\text{safe}}, \text{right}_{\text{safe}}\}. \quad (3)$$

Specifically, the smart vehicle needs to travel along the centerline of the lane. In the ideal operating state, under a given speed condition, the lateral displacement deviation, e, and the heading angle deviation, $\Delta\varphi$, between the centerline of the vehicle body and the lane centerline is close to zero. Figure 2 shows the lateral position offset difference, *e*, and the heading angle deviation, $\Delta\varphi$, of the vehicle, respectively.

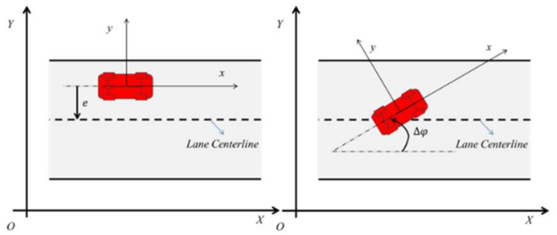

Figure 2. Vehicle Lateral Position Offset Difference e And Heading Angle Deviation $\Delta\varphi$

Also, it is necessary to ensure the vehicle's safety during operation. Therefore, the evaluation index of safety degree under the current driving situation should also be considered as a state space parameter of the mode switching strategy. This study takes a typical three-lane scenario as an example. The three lanes have different safety assessments, which are respectively expressed as left lane safety (left_safe), middle lane safety (current_safe), and right lane safety (right_safe), as shown in Fig. 3.

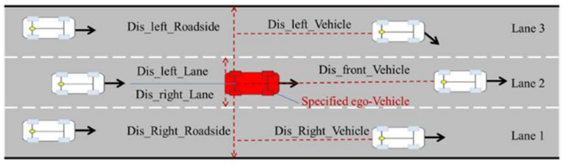

Figure 3. Safety Degree Of Three-Lane Scene

3.2 Motion Space Variable

Based on the current environmental state, the action space of the autonomous vehicle includes lateral behavior and longitudinal behavior. Lateral behavior refers to left lane change, right lane change, and lane keeping mode. Longitudinal behavior refers to acceleration, deceleration, and speed-keeping patterns. The human–machine integration switching mechanism based on RL is for evaluating the risk cost of intelligent control system behavior, and the driver's expected driving strategy under the constraint of the human driver's experience, redistributing the weight of automatic driving and manual driving, and output mode switching flag bit as shown in Eq. (4).

$$A_t = \left\{ \begin{array}{c} \text{left}_{\text{change}}, \text{right}_{\text{change}}, \text{lane}_{\text{keep}}, \\ \text{acceleration}, \text{deceleration}, \text{speed}_{\text{keep}} \end{array} \right\}. \quad (4)$$

3.3 Design of Cost Function for Risk Assessment

As a key element of the RL framework, the objective evaluation optimization function maximizes the objective reward function and minimizes driving risk by adjusting the control strategy and optimizing the action sequence through a reward and punishment mechanism. In the vehicle driving task, the purpose of the design of the cost function is to find the optimal mode switching strategy to reduce the motion error of the vehicle as much as possible and improve driving safety. The design criterion is shown in Fig. 4.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

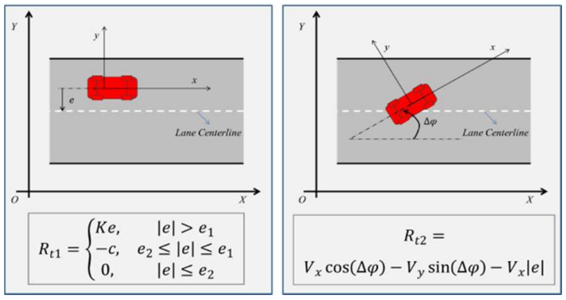
Y
y
x
e
Lane Centerline
O
X
$R_{t1} = \begin{cases} Ke, & |e| > e_1 \\ -c, & e_2 \leq |e| \leq e_1 \\ 0, & |e| \leq e_2 \end{cases}$
Y
y
x
Δφ
Lane Centerline
O
X
$R_{t2} =$
$V_x \cos(\Delta\varphi) - V_y \sin(\Delta\varphi) - V_x|e|$

Figure 4. Design Criteria Of Reward Function For Mode Switching

3.4 Algorithm Design

The overall design is shown in Fig. 5. The proposed human–machine fusion mode switch based on the RL framework is mainly composed of three parts: the Actor network, the Critic network, and the mode switch state machine based on the risk degree evaluation, for weight redistribution, and driving mode flag bit output.

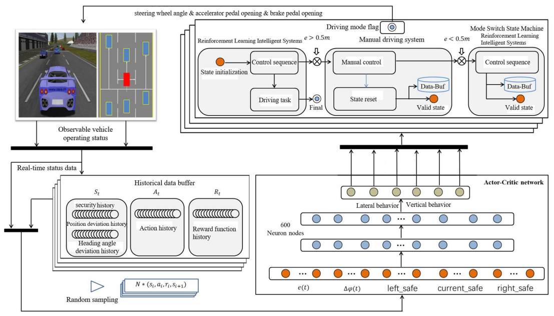

Figure 5. Overall Framework Of Project Design

The mode switching for human–computer integration has the following features:

(1) The adaptive feature can constantly update the behavior strategy when there is a change in the environment, which is completed by the Actor feedforward control network;
(2) Self-evaluation characteristics, according to the objective cost function, evaluates the current driving strategy risk, which is completed by the Critical feedback evaluation network;
(3) The driving weight is dynamically allocated. Based on the risk evaluation function and the experience constraints of manual drivers, the weight is redistributed.

The deep RL algorithm used in this project is the DDPG algorithm. The algorithm is a deep RL algorithm for continuous action space, so it is suitable as an autonomous driving decision algorithm based on deep RL. The DDPG algorithm combines the core ideas of the value function method and the policy gradient method, uses a neural network to fit the value function, adopts experience playback similar to the DQN algorithm, and adopts a dual network structure in both the Actor and the Critic networks. In the DDPG algorithm, the strategy network is used to update the strategy, corresponding to the Actor-network in the Actor–Critic algorithm framework, and the parameter θ_μ is used to represent the deterministic strategy $a = \mu(s|\theta_\mu)$. The value network is used to fit the action value function and give the gradient information to the Actor-network, corresponding to the critic network in the Actor–Critic algorithm framework, and its parameter is θ_Q.

The objective function of the Actor-network in the DDPG algorithm is the expectation of cumulative reward value. It has been proved that the gradient of the objective function concerning the policy network θ_μ is equal to the expected gradient of the action value function $Q(s, a|\theta_Q)$ with respect to θ_μ by Silver et al. [20]. The gradient ascent formula of the policy network $\mu(s|\theta_\mu)$ is shown in Eq. (5).

$$\nabla_{\theta_\mu} J \approx E_{s_t}\left[\nabla_{\theta_\mu} Q(s, a|\theta_Q)|_{s=s_t, a=\mu(s_t|\theta_\mu)}\right] = E_{s_t}\left[\nabla_a Q(s, a|\theta_Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta_\mu} \mu(s_t|\theta_\mu)|_{s=s_t}\right]. \quad (5)$$

The value network $Q(s, a|\theta_Q)$ updates the parameter θ_Q of the value network through the DQN algorithm, and the gradient of the value network is (6)

$$\nabla_{\theta_Q} = E_{s,a,r,s',\ldots,R}\left[\left(\left(r + \nabla_{\theta_Q'} \gamma Q'\left(s, \mu'(s|\theta_\mu')\right)\right) - Q(s, a|\theta_Q')\right) \nabla_{\theta_Q} Q(s, a|\theta_Q)\right]. \quad (6)$$

The Actor-network is a policy function that needs reward and punishment information to adjust the strategy, select different actions according to different states, and is responsible for generating actions and interacting with the environment. The Critic-network is an evaluation function responsible for evaluating the performance of actors and guiding the output of the Actor-network in the next phase. Theoretically, a neural network with only one hidden layer is enough to achieve the global approximation and description of any nonlinear function. In the process of model training, this study designs a fully connected network structure. By optimizing the objective function and using the algorithm of DDPG to find the optimal control strategy, the mapping from the state space to the action space can be realized. The architecture of the Actor–Critic network is shown in Fig. 6. They are all composed of three layers: input, output, and hidden layers with 600 neurons.

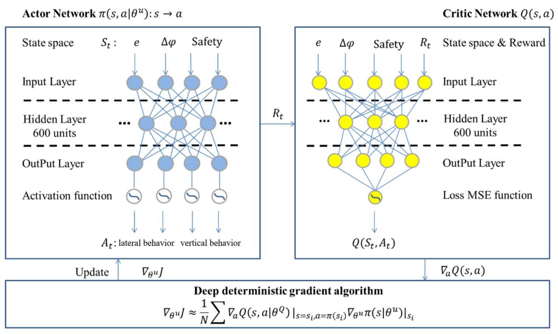
Actor Network $\pi(s,a|\theta^u): s \rightarrow a$
Critic Network $Q(s,a)$
State space S_t: e $\Delta\varphi$ Safety
Input Layer
Hidden Layer 600 units
OutPut Layer
Activation function
A_t: lateral behavior vertical behavior
R_t
e $\Delta\varphi$ Safety R_t State space & Reward
Input Layer
Hidden Layer 600 units
OutPut Layer
Loss MSE function
$Q(S_t, A_t)$
Update
$\nabla_{\theta^u} J$
$\nabla_a Q(s,a)$
Deep deterministic gradient algorithm
$\nabla_{\theta^u} J \approx \frac{1}{N}\sum \nabla_a Q(s,a|\theta^Q)|_{s=s_i,a=\pi(s_i)} \nabla_{\theta^u}\pi(s|\theta^u)|_{s_i}$

Figure 6. Architecture Of Actor–Critical Network

4. EXPERIMENT AND DISCUSSION

4.1 Simulation Platform

The scene is an essential part in the simulation test of automatic driving. Scenes generally include two parts: the scene in the vehicle's process and the driving site, reflecting the real environment and driving behavior. The real environment has static traffic facilities and dynamic change objectives. Driving behavior includes cruise control, car following, overtaking, and other driving tasks. Our study uses TORCS (The Open Racing Car Simulator) as the simulation environment. TORCS can simulate a car's transmission, clutch, engine, and various road characteristics. The environment is widely used in game competitions, simulation, and other fields. The graphical interface of TORCS is shown in Fig. 7.

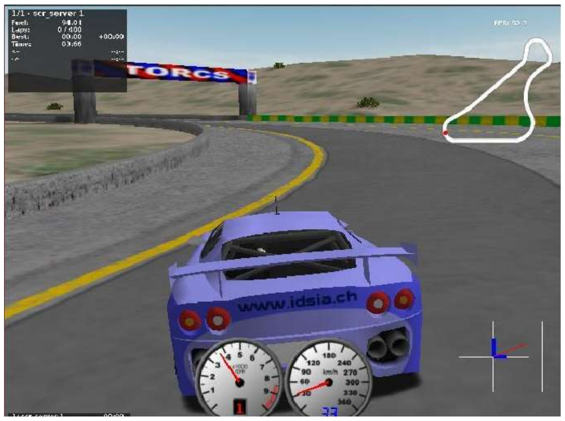

Figure 7. TORCS Simulation Environment

TORCS is an open-source auto-driving simulator for racing cars, which consists of two parts: client and server. The client is responsible for receiving the sensor information sent by the server and returning the control quantity to the server after making decisions based on the obtained information to realize the vehicle motion control. The server is responsible for rendering the image quality of the simulation environment and controlling the vehicle based on the client's decision results. The TORCS autopilot simulation platform can provide a variety of state information so that this project can decide according to the task requirements. The platform provides 19 kinds of sensors, as shown in Table 1. Moreover, the physical isolation between the game engine and the driver is established so that users can obtain the status information of all vehicles in the driving environment and the surrounding environment without knowing the internal data structure, which significantly improves the development efficiency of the automatic driving decision-making algorithm. Many machine learning researchers choose the TORCS platform for algorithm simulation verification [21], [22].

Table 1. Sensor Types Of TORCS Autonomous Driving Simulation Platform

Name	Range (unit)	Description
angle	$[-\pi, +\pi]$ (rad)	The angle between the longitudinal axis of the car and the axis of the road
track	$(0, 200)$ (m)	Distance between a vehicle within 200 meters and the edge of the road
trackPos	$(-\infty, +\infty)$	Distance between car and road axis
speedX	$(-\infty, +\infty)$ (km/h)	Velocity along the longitudinal axis of the vehicle
speedY	$(-\infty, +\infty)$ (km/h)	Speed along the transverse axis of the vehicle
speedZ	$(-\infty, +\infty)$ (km/h)	Speed along the Z-axis of the car
damage	$[0, +\infty)$ (point)	Current damage to the car
wheelSpinVel	$[0, +\infty)$ (m)	wheel rotation speed
rpm	$(0, +\infty)$ (rpm)	Engine speed
curLapTime	$[0, +\infty)$ (s)	The time spent in the current loop
distFromStart	$[0, +\infty)$ (m)	The distance from the car to the starting line
distRaced	$[0, +\infty)$ (m)	Distance from the start line to the finish
fuel	$[0, +\infty)$ (l)	Current remaining fuel level
gear	$\{-1, 0, 1 \ldots 6\}$	Current gear
lastLapTime	$[0, +\infty)$ (s)	Start time of the last lap

opponents	[0,200] (m)	Distance to the nearest vehicle within 200m
racePos	$\{1,2,3\ldots N\}$	Current ranking
z	$(-\infty,+\infty)$ (m)	Distance from the center of mass of the car to the Z-axis track surface

In this study, we use the TORCS autopilot simulation platform for the experiment as follows. First, the TORCS autopilot simulation platform stores the state feature sequence in the experience playback pool (also known as the sample buffer, Replay Buffer). The DDPG algorithm then receives the small batch of state sample information features from the Replay Buffer. Finally, it outputs the direction, speed, acceleration, braking, and other information of the control vehicle to evaluate the current identifier to determine whether manual driving or automatic driving should be performed.

4.2 Network Parameter Configuration

This project is based on the DDPG algorithm in deep RL to achieve smooth driving mode switching. The algorithm principle, calculation method, framework design, and pseudo-code implementation of the DDPG algorithm have already been discussed in Section 3. Here, we give the hyperparameter settings of the DDPG algorithm in the simulation experiment, as shown in Table 2

Table 2. Hyperparameters of the DDPG Algorithm

Hyperparameter	Preset
Actor-network learning rate	0.001
The critical network learning rate	0.01
State space dimension	5
Action space dimension	6
Discount factor	0.95
Network iteration step size	200,000

Driving mode switch refers to the switching process between the driver's manual driving mode and the vehicle's autonomous driving mode in a vehicle equipped with the automated driving system (ADS).

As defined by the SAE J3016 standard [23], driving automation systems from levels L0 to L3 require a human driver to support dynamic driving tasks and take over driving control when needed. Even at L4 autopilot, the system cannot cover all driving scenarios. Therefore, a human driver needs to take over the controls if the vehicle goes beyond the intended operating zone ODD. The essence of take-over is the conversion of vehicle driving control between "human" and "machine." According to the initiator and executor of driving right conversion, take-over can be divided into request take-over initiated by the ADS and active intervention initiated by users. The direct conversion between intelligent driving and manual driving is shown in Fig. 8.

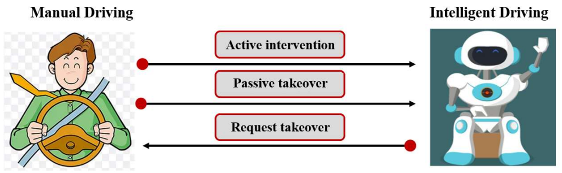

Figure 8. Direct Conversion Between Intelligent Driving And Manual Driving

4.2.1 Passive Takeover

When the system related to the dynamic driving task fails, or the vehicle exceeds the specified operating area (generally determined by parameters such as lateral distance deviation, heading angle deviation, and safe driving area), the ADS sends a takeover request to the user, and the user responds by controlling the transverse and longitudinal motion control. Passive takeover emphasizes that it is initiated by the ADS and executed passively by the user. The driving mode switching strategy based on deep RL proposed in this project is that the ADS outputs the driving mode flag and requests the human driver to take over when the vehicle exceeds the expected specified driving area or does not meet the safe driving state by observing the real-time environmental operation state.

4.2.2 Active Intervention

When the autopilot system is still running, the user actively provides input to the transverse and longitudinal motion control actuator. The system will choose to exit the autopilot function or continue to perform dynamic driving tasks according to whether the user's input reaches the threshold. Active intervention emphasizes the initiative of users. The ADS can allow various active intervention methods. For example, the user can control the brake pedal, accelerator pedal, parking brake switch, turn signal switch, hazard warning lamp switch, and other ways to implement the active intervention. The ADS proposed in this project based on deep RL will automatically detect the running state of the vehicle. When the car is restored to its initial state through manual intervention, the automatic driving sign will be sent, and the ADS will take over passively.

4.3 Driving Mode Switching Sequence Diagram

The sequence diagram of the manual driving mode switching to the automatic driving mode is shown in Fig. 9.

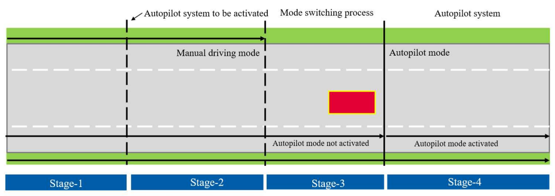

Figure 9. Time Sequence Diagram Of Mode Switching (From Manual Driving To Automatic Driving)

Figure 9 shows that the sequence diagram is divided into four stages:

- Stage 1: Human drivers drive to keep the driving state of vehicles stable.
- Stage 2: The automatic driving mode is in the state to be activated, and the system will prompt that the automatic driving function can be started.
- Stage 3: The driver starts the automatic driving mode and enters the manual automatic mode switching process.
- Stage 4: The vehicle enters the automatic driving mode.

Figure 10 shows the sequence diagram of automatic driving mode switching to manual driving mode.

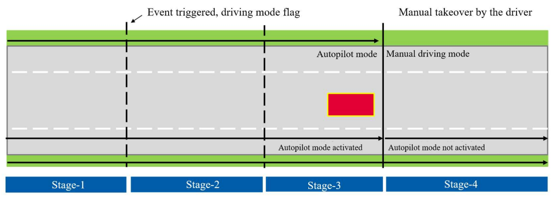
Event triggered, driving mode flag
Manual takeover by the driver
Autopilot mode
Manual driving mode
Autopilot mode activated
Autopilot mode not activated
Stage-1
Stage-2
Stage-3
Stage-4

Figure 10. Time Sequence Diagram Of Mode Switching (From Automatic Driving To Manual Driving)

It can be seen from Fig. 10 that the sequence diagram is divided into four stages:

- Stage 1: The ADS controls the vehicle, and the driving state remains stable.
- Stage 2: Event triggering, output mode switching driving flag bit based on mode switching strategy of RL by observing vehicle operation status, such as lateral position deviation, heading angle deviation, and three-lane driving safety.
- Stage 3: The vehicle requests the driver to take over and enter the automatic manual mode switching process.
- Stage 4: The driver takes over and controls the vehicle to keep it running stably.

4.5 Simulation Results

Two different scenarios are set up to select the conservative and aggressive driving strategies, respectively. The simulation experiment of driving mode switching is conducted to verify the generalization performance of the proposed deep RL model. The generalization performance described in this project refers to whether the deep RL model can adapt to different scenarios to achieve smooth switching of driving modes in different scenarios.

- Scenario Case 1: Set the initial vehicle to drive on the left lane, with the vehicle's driving speed in front of the user at 38 km/h and the vehicle's driving speed in front of the user at 41 km/h. Select a conservative driving strategy. The simulation results obtained in this scenario are shown in Fig. 11.

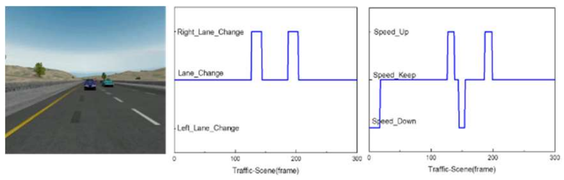

Figure 11. The Scenario Case 1's Simulation Results

It can be seen from the simulation results of Case 1 (Fig. 11) that in the current scenario, the automatic driving lateral behavior decision first executes lane keeping. With the appearance of distant obstacles in front, the longitudinal behavior decision executes deceleration and enters the speed following mode. At the same time, determine whether the right lane meets the conditions for a safe lane change. After meeting the conditions, the horizontal decision-making behavior executes the right lane change, the longitudinal behavior decision-making executes the vehicle acceleration mode, and finally completes the lane change and overtaking, ensuring driving efficiency

- Scenario Case 2: Set the initial vehicle driving in the middle lane, with the driving speed of the front car at 43 km/h, the driving speed of the left front car at 43 km/h, and the driving speed of the right front car at 47 km/h. The aggressive driving strategy is selected. Figure 12 shows the simulation results obtained in this scenario.

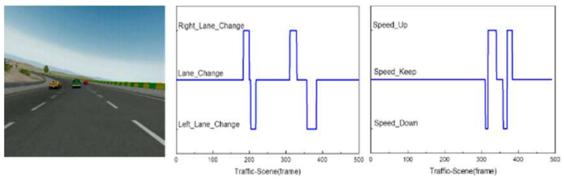

Figure 12. Case 2's Simulation Results

It can be seen from the simulation results of Case 2 (Fig. 12) that in the current scenario, the horizontal behavior decision first implements lane keeping. With the approaching obstacles in front, the vertical behavior decision implements deceleration but does not enter the speed following mode. The horizontal behavior decision directly executes right lane changing. In the first overtaking behavior, the horizontal behavior decision again executes left lane changing. It can be seen that when the aggressive driving style is a constraint, the lateral decision-making mode will frequently perform lane changing.

The simulation results of Case 1 and Case 2 show that the driving mode switching algorithm based on the deep RL proposed in this project has good generalization performance and can adapt to different scenarios.

4.6 Effectiveness

It is confirmed that the deep RL model proposed in this study has good generalization performance. We verify the effectiveness of this algorithm for driving mode switching tasks. The proposed model collects vehicle operation status data, including lateral position deviation, heading angle deviation, and safety index. The model also records the current driving mode, verifies the effectiveness of the driving mode switching strategy based on deep RL, and records the distance that can be safely driven. The experimental simulation results are shown in Fig. 13.

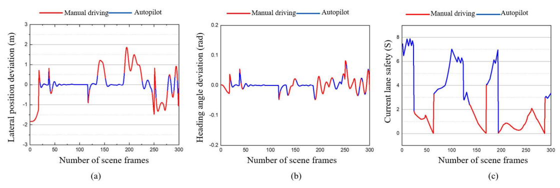

Figure 13. Effectiveness Verification Results Of Driving Mode Switching Strategy

It can be seen from the experimental simulation results (Fig. 13) that the driving mode switching strategy based on deep RL proposed in this project can be successfully switched to manual driving when the lateral position deviation and heading angle deviation are significant by observing the vehicle operational state data.

The mathematical results show that the proportion of manual driving is 55.73%, and the proportion of automatic driving is 44.26%. At the same time, the driving mode switching strategy can respond to the lane safety degree. When it is lower than the safety degree, the proposed strategy switches to the manual driving mode, and when it is higher than the safety degree threshold, it enters the automatic driving mode. The results show that the proportion of manual driving is 48.57%, and the proportion of automated driving is 51.43% at this point. The simulation results show that based on the analysis of lateral position deviation, heading angle deviation, and safety index data, the deep RL method proposed in this project can effectively realize the switch between manual and automatic driving modes.

5. CONCLUSION

This paper illustrates the components in a modern autonomous driving systems based on the real-time online switching mechanism with deep reinforcement learning. The first achievement of the paper is the understandings of distance and safety degree of moving obstacles implemented by the motion space variable, design of cost function for risk assessment, and safety algorithm. The second achievement of the paper is the decision and planning with Actor–Critic framework followed by switching strategy.

Taking a typical three-lane traffic scene as an example, to calculate and judge the safety degree of moving obstacles in the lane, a real-time online switching mechanism based on RL is adopted. Through real-time observation of the safety degree index of the current driving situation and the running data of the vehicle, the state space parameters of the vehicle are formed, and the cost function based on risk evaluation is designed. Through the constructed Actor–Critic network, the output mode switches the flag bit. Finally, the effectiveness of the proposed algorithm is verified by the simulation platform. Through the case setting in the simulation scenario, the success rate of mode switching is verified. The results show that the mode switching strategy can respond well to the vehicle lateral position deviation,

heading, angle deviation, and longitudinal safety degree and improve the vehicle's operational safety. This paper verifies the effectiveness of the driving strategy designed in this project in the two scenarios presented, but in general, there is still much room for research and improvement. In the future, the work that can be improved in this project includes the following:
(1) Research on automatic driving strategies transfer learning in different scenarios [24]. The experimental scenarios designed in this project are few compared with the actual scenarios. In future research, we can study the application of transfer learning in RL. Not only can we explore how to transfer the knowledge or skills learned in a single scene to other environments, but we can also study how to transfer the knowledge learned in the virtual environment to the actual scene, which will undoubtedly greatly promote the application of driving strategies based on RL in reality.
(2) Hot start research using expert strategies. This project uses random strategy initialization in the training process, so the exploration efficiency is relatively low, and the training process is lengthy. In succeeding research, we can use IL to obtain an initialization strategy from expert data, which can significantly accelerate the training process.
(3) Research on the design of reward function. The reward function is the critical factor that determines whether the agent can learn effective strategies. If the designed reward function needs to adjust the coefficients of different reward items constantly, it is highly subjective based on manual observation and continuous experiments. Therefore, in future studies, a mechanism similar to "internal curiosity" can be added to the agent exploration strategy to guide the agent to accelerate learning. In addition, we can also consider using the inverse RL method to avoid the process of designing reward functions and automatically find more effective reward functions.

ACKNOWLEDGMENT:

"This research was supported by the MIST(Ministry of Science, ICT), Korea, under the National Program for Excellence in SW), supervised by the IITP(Institute of Information & communications Technology Planning & Evaluation) in 2022"(2022-0-00964). In addition, This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ICAN(ICT Challenge and Advanced Network of HRD) program (IITP-2023- 2020-0-018260201001) supervised by the IITP(Institute of Information & Communications Technology Planning & Evaluation. This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2023-2018-0-01396) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation. "

REFERENCES:

[1] Hamilton A, Waterson B, Cherrett T, et al. The evolution of urban traffic control: changing policy and technology, *Transportation Planning and Technology*, Vol. 36, 2013, pp. 24-43.
[2] Mannering F, Bhat C R, Shankar V, et al. Big data, traditional data and the tradeoffs between prediction and causality in highway-safety analysis, *Analytic Methods in Accident Research*, Vol. 25, 2020, pp. 100113.
[3] Dosovitskiy A, Ros G, Codevilla F, et al. CARLA: An open urban driving simulator, *Conference on Robot Learning. PMLR*, 2017, pp. 1-16.
[4] Konwar K M, Hanson N W, Pagé A P, et al. MetaPathways: a modular pipeline for constructing pathway/genome databases from environmental sequence information, *BMC Bioinformatics*, Vol. 14, 2013, pp. 1-10.
[5] Le Mero L, Yi D, Dianati M, et al. A survey on imitation learning techniques for end-to-end autonomous vehicles, *IEEE Transactions on Intelligent Transportation Systems*, 2022.
[6] Qi Q, Wang J, Ma Z, et al. Knowledge-driven service offloading decision for vehicular edge computing: A deep reinforcement learning approach, *IEEE Transactions on Vehicular Technology*, Vol. 68, 2019, pp. 4192-4203.
[7] Paden B, Čáp M, Yong S Z, et al. A survey of motion planning and control techniques for self-driving urban vehicles, *IEEE Transactions on Intelligent Vehicles*, Vol. 1, 2016, pp. 33-55.
[8] Kendall A, Hawke J, Janz D, et al. Learning to drive in a day, 2019 *International Conference on Robotics and Automation (ICRA). IEEE*, 2019, pp. 8248-8254.
[9] Muller U, Ben J, Cosatto E, et al. Off-road obstacle avoidance through end-to-end learning, *Advances in Neural Information Processing Systems*, 2005, pp. 18.
[10] Talebpour A, Mahmassani H S, Hamdar S H. Modeling lane-changing behavior in a connected environment: A game theory approach, *Transportation Research Procedia*, Vol. 7, 2015, pp. 420-440.

[11] Coronato A, Naeem M, De Pietro G, et al. Reinforcement learning for intelligent healthcare applications: A survey, *Artificial Intelligence in Medicine*, Vol. 109, 2020, pp. 101964.

[12] Mousavi S S, Schukat M, Howley E. Deep reinforcement learning: an overview, Proceedings of SAI Intelligent Systems Conference. *Springer, Cham*, 2016, pp. 426-440.

[13] Moerland T M, Broekens J, Jonker C M. Emotion in reinforcement learning agents and robots: A survey, *Machine Learning*, Vol. 107, 2018, pp. 443-480.

[14] Dargan S, Kumar M, Ayyagari M R, et al. A survey of deep learning and its applications: a new paradigm to machine learning, *Archives of Computational Methods in Engineering*, Vol. 27, 2020, pp. 1071-1092.

[15] Otter D W, Medina J R, Kalita J K. A survey of the usages of deep learning for natural language processing, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 32, 2020, pp. 604-624.

[16] Wang D L, Chen J. Supervised speech separation based on deep learning: An overview, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 26, 2018, pp. 1702-1726.

[17] Bhatnagar S, Ghavamzadeh M, Lee M, et al. Incremental natural actor-critic algorithms, *Advances in Neural Information Processing Systems*, 2007, pp. 20.

[18] Vamvoudakis K G, Lewis F L. Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem, *Automatica*, Vol. 46, 2010, pp. 878-888.

[19] Fujimoto S, Hoof H, Meger D. Addressing function approximation error in actor-critic methods, *International Conference on Machine Learning, PMLR*, 2018, pp. 1587-1596.

[20] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, Martin Riedmiller, Deterministic policy gradient algorithms, ICML'14: *Proceedings of the 31st International Conference on International Conference on Machine Learning*, Vol. 32, 2014, pp. I-387–I-395.

[21] Sabir Hossain, Deok-Jin Lee, Autonomous-driving vehicle learning environments using unity real-time engine and end-to-end CNN approach, *Journal of Korea Robotics Society*, Vol. 14, 2019, pp. 122-130, https://doi.org/10.7746/jkros.2019.14.2.122

[22] Zhang T, Mo H. Reinforcement learning for robot research: A comprehensive review and open issues, *International Journal of Advanced Robotic Systems*, Vol. 18, 2021, doi:10.1177/172988142110073054.

[23] Society of Automotive Engineers (SAE), “J3016,” SAE international taxonomy and definitions for terms related to on-road motor vehicle automated driving systems, levels of driving automation, 2014.

[24] B. R. Kiran et al., "Deep Reinforcement Learning for Autonomous Driving: A Survey," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 6, pp. 4909-4926, June 2022, doi: 10.1109/TITS.2021.3054625.