World Journal of Engineering Research
Print ISSN: 2959-9865
Online ISSN: 2959-9873
DOI:https://doi.org/10.61784/wjer3035

APPLICATION OF REINFORCEMENT LEARNING IN AUTONOMOUS DRIVING SCENARIOS: PATH PLANNING USING POLICY GRADIENT METHODS

JunRan Wu
School of Mathematical, Chengdu University of Technology, Yibin 644000, Sichuan, China.
Corresponding Email: Wjr13031955011@foxmail.com

Abstract: This study explores reinforcement learning-based autonomous path planning in China's mixed traffic flow, employing a policy gradient approach to optimize decision-making through environmental interaction. China's traffic environment, characterized by a diverse mix of vehicles, pedestrians, and non-motorized traffic, presents unique challenges. To address these, a state space is meticulously constructed, incorporating real-time traffic light status, the precise positions and orientations of surrounding vehicles, providing a comprehensive representation of the traffic scenario. The action space, covering operations such as going straight, turning left, turning right, and stopping, enables the autonomous system to make practical driving decisions. A "time-saving and violation-avoidance" reward function is designed, effectively transforming the path planning task into a sequential decision optimization problem. The policy network, parameterized by a three-layer neural network, learns from the environment through repeated trials. Initial reward fluctuations gradually stabilize, achieving a 100% success rate with an average of 18.2 steps, successfully realizing shortest-path selection. However, the negative average reward (-1.22) indicates that the current reward function may have an excessive penalty bias. While validating the effectiveness of the policy gradient method for intelligent transportation, the results underscore the necessity of refining the reward function to strike a balance between negative penalties and positive incentives. This framework provides valuable methodological guidance for autonomous driving in complex scenarios, yet further optimization of the reward mechanisms remains essential for practical implementation.
Keywords: Autonomous driving; Policy gradient methods; Reward function; Policy network

1 INTRODUCTION

We have now entered an era surrounded by intelligent agents. Intelligent agents can be found in various fields, such as the AI field, the autonomous driving field, the geographical simulation field, and the economic calculation field. The arrival of intelligent agents has brought many new experiences to our lives. Piero even once said, "Machines are the key to the happiness of our future lives,... Machine intelligence is likely to be indispensable for solving the most serious problems of our time[1]. " So, how can we use machines as a carrier to simulate a certain aspect of human intelligence? In Wu Fei's work, it was introduced that "To simulate a certain aspect of human intelligence using machines as a carrier, it can be achieved through methods such as logical reasoning centered on symbolism, search exploration centered on problem-solving, and decision-making intelligence centered on data-driven approaches." Inspired by the laws of biological learning, reinforcement learning interacts with the environment through a trial-and-error mechanism. It learns and optimizes by maximizing the cumulative rewards, and ultimately achieves the optimal strategy[2]. Artificial intelligence aims to enable machines to learn, think, and understand just like humans, that is, to simulate human intelligence using computers. Since the external environment provides very little information directly, the decision-making agent in reinforcement learning must interact with the environment through trial and error, and continuously optimize the control strategy from the action-evaluation process to improve the control performance of the system. Therefore, essentially, reinforcement learning approximates the mapping relationship of "state-action" through parameterized functions to find the optimal strategy for solving decision-making problems[3]. We hope that machines can learn autonomously just like humans. What we need to do is to enable the intelligent agent to maximize the cumulative rewards obtained from its responses during the interaction with the environment. This reward is abstract. It can be numerical addition or subtraction, time addition or subtraction, or other forms. Since the intelligent agent may face multiple states, we can simplify this problem to the issue of an autonomous vehicle passing through a traffic light intersection. Then, the final cumulative return can be regarded as the time required for the autonomous vehicle to complete this section of the journey.
Typical reinforcement learning has the following three characteristics[4]:
1) Different actions generate different rewards.
2) Rewards are delayed in time.
3) The return of a certain action is related to the specific environment in which it is performed.
Wahba proposed two methods related to machine learning models: supervised learning and unsupervised learning. Unsupervised learning does not require a decision-making process, and its learning objective is merely to obtain the distribution of data within the same category. Supervised learning provides data information with labels and can make

single-step decisions based on the label information. However, these two learning methods completely fail to meet the data processing and decision-making needs of intelligent agents in practice. In the actual environment, the state is changing. Since supervised learning requires the construction of a large number of labeled samples, it is difficult to be applied in practice. Currently, most studies use the deep reinforcement learning method to train the model . Reinforcement learning does not require a supervision signal to directly guide the learning. It only depends on a feedback reward signal to evaluate its "trial-and-error" process, and indirectly guides the intelligent agent to learn in the direction of maximizing the feedback reward value, thus reducing the dependence on an accurate system model[5].
Reinforcement learning refers to the learning of the mapping from environmental states to actions, with the aim of maximizing the cumulative reward value obtained by the actions from the environment. The basic components of reinforcement learning include a set of states S that represent the environment, a set of actions A that represent the actions of the intelligent agent, and the reward r for the intelligent agent. In the actual process, the interaction between these two roles includes the state, action, and reward (the reward can be either positive or negative), as shown in Figure 1 Environmen-Agent . The large-scale application of reinforcement learning requires the use of generalization function approximation[6], such as neural networks, decision trees, or instance-based methods. In the past decade, the dominant method has been the value function method. The goal of reinforcement learning is to enable the intelligent agent to learn the optimal control strategy in an unknown environment, thereby maximizing the expected future returns[7]. The core idea of reinforcement learning is to continuously interact between the intelligent agent and the environment, and select reasonable actions with the goal of maximizing the cumulative return. This coincides with the process of acquiring experiential knowledge and making decisions in human intelligence.

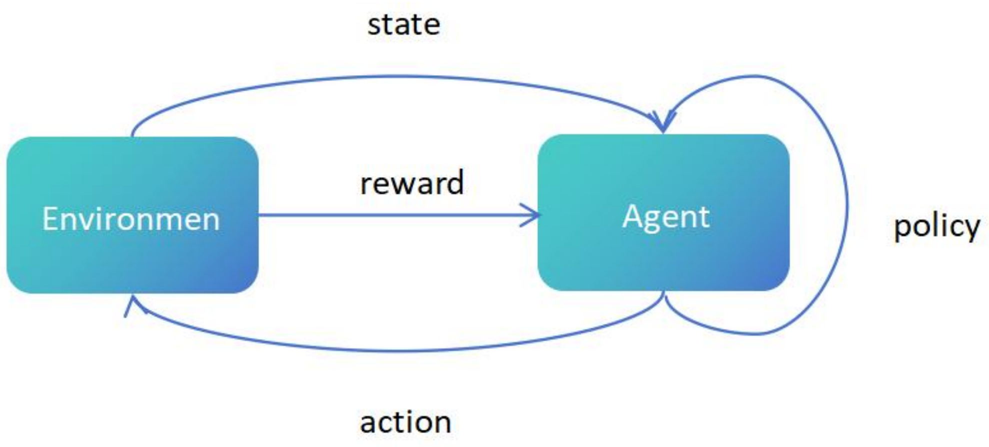

Figure 1 Environmen-Agent

The behavior policy function is the code of conduct for an intelligent agent. In essence, it is a mapping, which maps the set of environmental states to the probability distribution function or probability density function of the set of behaviors, guiding the intelligent agent to select the best behavior. The reward function is the evaluation criterion for the intelligent agent. Usually, positive rewards are given for good behaviors, and negative rewards are given for the opposite. The value function is used to describe the quality of the current state, including the state value function and the action value function [8]. Generally, the interaction process between the intelligent agent and the environment in reinforcement learning can be modeled as a Markov Decision Process (MDP)[9]. The core idea of model-based reinforcement learning originates from the theory of dynamic programming. Its core algorithm realizes the optimization process of policy iteration and value iteration by applying the Bellman equation and the Bellman optimality equation. Model-free reinforcement learning, on the other hand, adopts a learning method of directly interacting with the environment for sampling. The current research mainly focuses on three technical directions: optimization algorithms centered around the value function, optimization methods based on the search in the policy space, and optimization methods combined with the construction of the environment.

2 BACKGROUND

Under the traffic conditions of mixed traffic flow in China, the traditional solution of presetting situations in advance is obviously no longer suitable for today's autonomous driving. Cars need to make autonomous decisions in situations that have not been preset by humans in advance. The autonomous decision-making technology dynamically adjusts the path through reinforcement learning to find the optimal path. In recent years, many scholars have made progress in the field of path planning by using reinforcement learning methods[10-12]. As a key technology of autonomous driving [13], path planning serves as a bridge connecting environmental perception and motion control. The quality of the planned path directly affects the driving trajectory of the vehicle. It is the foundation for achieving autonomous driving and has broad scientific research value and commercial application prospects[14].

3 PROBLEM FORMULATION

Suppose an autonomous vehicle needs to reach its destination through a rectangular grid route, where each grid intersection is equipped with a traffic light displaying the remaining time for each signal phase. When the traffic light is green, the vehicle is allowed to go straight or turn left; when it is red, the vehicle can only stop or turn right. Based on literature review, in real-world scenarios, traffic light cycles at different intersections vary, and even within the same intersection, the durations of straight-going and left-turning green phases may differ. For simplicity, we assume that all traffic lights along the route share identical settings, with red and green lights in different directions switching simultaneously. The research objective is to determine how the autonomous vehicle can reach its destination in the shortest possible time under these conditions.
The problem exhibits the characteristics of reinforcement learning algorithms: the agent conducts trial and exploration in the environment, and optimal policy selection is made based on rewards from the environment. We model this problem as follows:

3.1 Problem Modeling

Environment Setup:
An *N*N* grid of intersections, with the starting point at coordinates (1, 1) and the destination at *(N, N).* The vehicle's initial orientation is north. The path matrix is denoted as W, the average speed of the vehicle is v, and the total traffic light cycle duration is 160 seconds—80 seconds for the green phase and 80 seconds for the red phase. The north-south direction starts with a green light followed by a red light, while the east-west direction starts with a red light followed by a green light.
State Space Definition:
$S = (x_{1i}, x_2, x_3)$,where $x_{1i} = (x_{11}, x_{12}, x_{13})$ respectively represent the distances traveled by the vehicle when going straight, turning left, and turning right to the next intersection (unit: meters).
x_2 represents the remaining time for the transition from red - light to green - light (if it is green - light, then it is directly 0). $x_3 \in \{N, E, S, W\}$ represents the current driving direction of the vehicle.
Action Space Definition:
The action vector $a_t \in R^4$ is one of the four unit vectors:

$$a_t \in \{e_1, e_2, e_3, e_4\} \tag{1}$$

where:
$e_1 = (1,0,0,0)$: go straight ; $e_2 = (0,1,0,0)$: turn left ;
$e_3 = (0,0,1,0)$: turn right ; $e_4 = (0,0,0,1)$: stop.
Above are the state space and action space of this problem. Next, the state transition function (deterministic) is defined as follows: Let the current time be t, the accumulated passage time be T_t, the state be $S = (x_{1i}, x_2, x_3)$ and the action be $a_t = e_i$.
Then the next state is $S = (T_{1i,k+1}, t_{2,k+1}, x_{3,k+1})$, and the calculation logic is as follows:
(1) $x_{1i,k+1}$ Path Update
According to the current position and vehicle orientation $x_{3,t}$, using the path matrix W and action $a_t = e_i$, calculate the distances in each direction after the next position reaches the new node.
The following Table 1 Vehicle Orientation Changes is used to illustrate the changes in the direction of the car.

Table 1 Vehicle Orientation Changes

Current Orientation	Go Straight	Turn Left	Turn Right
N	*N*	*W*	*E*
E	*E*	*N*	*S*
W	*W*	*S*	*N*
S	*S*	*E*	*W*

(2) Calculation of the remaining time for red - light to turn green for $x_{2,k+1}$:Calculate the current cumulative passage

$$\begin{cases} x_{3,t} \in \{N, S\} \to green\ first, then\ red \\ x_{3,t} \in \{\mathrm{E}, W\} \to red\ first, then\ green \end{cases} \tag{2}$$

Calculate the current cumulative time: $T_{k+1} = T_t + \dfrac{vdistance\ of\ action - taken\ direction}{v} + x_{2,t}$

Determine the state of the traffic light: $p = T_{t+1} \bmod 160$

When $x_{3,t} \in \{N,S\}$, $x_2 = \begin{cases} 0 & , p < 80 \quad green \\ 160-p, & p > 80 \quad red \end{cases}$;When $x_{3,t} \in \{E,W\}$, $x_2 = \begin{cases} 0 & , p < 80 \quad r \\ 160-p, & p > 80 \quad red \end{cases}$

Design objective of the reward function: save time and avoid violations.

(1) Time Cost

$$r_1 = -\frac{d(a_t)}{v}, d(a_t) = \begin{cases} x_{11}, & if\ a_t = e_1 \\ x_{11}, & if\ a_t = e_1 \\ x_{13}, & if\ a_t = e_3 \\ 0\ , & if\ a_t = e_4 \end{cases} ; \tag{3}$$

(2) Violation Penalty : $r_2 = \sigma_\delta(red(x_3,\phi))I_{a_t=e_1 \vee a_t=e_2} + \sigma_\delta(red(x_3,\phi))I_{a_t=e_4}$;

(3) Right - turn Reward when x_2 is Large $r_3 = \sigma(\frac{x_2}{\tau})\sigma_\delta(red(x_3,\phi))I_{a_t=e_3}$;

Finally, the total reward is set as $r(s_t,a_t) = w_1 r_1 - w_2 r_2 + w_3 r_3$

Table 2 Symbol Explanation is for explaining the symbols.

Table 2 Symbol Explanation

Symnol	Explanation
a_k	Current Action
$I_{a_t=e_i}$	Indicator function, which equals 1 when $a_t = e_i$
$red(x_3,\phi)$	Judge whether the current is a red light
$\sigma(z)$	Used for smoothing processing
x_2	The time left for the red light
$h_j(s_t;\theta)$	The j -th action approximated and output by the neural network.
θ	Parameter set of the neural network
γ	Discount rate
w	Weight

In the reinforcement learning framework, the learning objective is to maximize the expected cumulative reward. This objective guides the agent to make optimal decisions in successive actions, aiming to achieve the highest long - term reward and optimize its environmental behavior.

$$J(\theta) = \mathrm{E}\left[\sum_{t=0}^{T} \gamma^t \cdot r(s_t,a_t)\right] \tag{4}$$

Decision function $\pi(s,a)$:

$$\pi(a_t \mid s_t;\theta) = \frac{\exp(h_i(s_t;\theta))}{\sum_{j=1}^{4}\exp(h_j(s_t;\theta))}, a_t = e_i \tag{5}$$

With the specific form of the decision function, we can further analyze the impact of policy parameters on performance. Let θ represent the vector of policy parameters, μ represent the performance of the corresponding policy. In the policy gradient method, the policy parameters are updated approximately proportionally to the policy gradient: $\Delta\theta \approx \alpha\frac{\partial\mu}{\partial\theta}$.

Where α represents the learning rate. Through the above formula, it can be ensured that θ converges to a locally optimal strategy.

Policy Gradient Theorem:

$$\frac{\partial \mu}{\partial \theta}=\sum_{S} d^{\pi}(S)\sum_{a}\frac{\partial \pi(a \mid S;\theta)}{\partial \theta}Q^{\pi}(S,a) \tag{6}$$

The policy gradient formula provides a method for the gradient of the parameter θ . Substituting the decision function into this theorem enables further exploration of the optimized policy function. Assume $g_w : S \times a \to \mathbb{R}$ approximates $Q^{\pi}(S,a)$, substitute into the formula to find the extreme point:

$$\sum_{S} d^{\pi}(S)\sum_{a}\frac{\partial \pi(S,a)}{\partial \theta}\left[Q^{\pi}(S,a)-g_w(S,a)\right]=0 \tag{7}$$

3.2 Strategy Optimization Implementation

In practical engineering, parameterize the decision function using a neural network, use the $PolicyNet$ function, and adopt a three - layer fully - connected network. Input the state vector, and the output is transformed into a decision function (action probability distribution) $\pi(a_t \mid S;\theta)$ through the $Soft\max$ function. As shown in Figure 2 Parameter Update Flowchart.

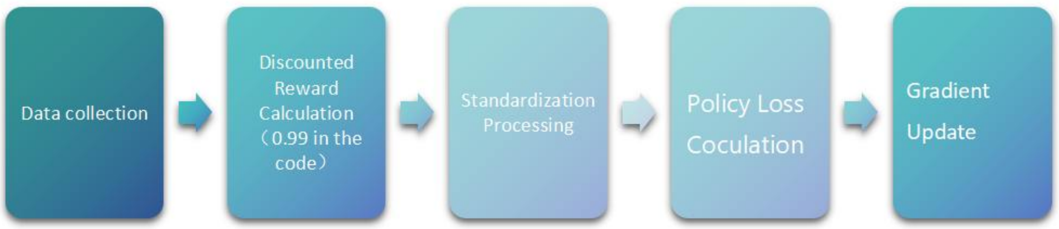
Data collection
Discounted Reward Calculation（0.99 in the code）
Standardization Processing
Policy Loss Coculation
Gradient Update

Figure 2 Parameter Update Flowchart

During the training process, the change in the number of cumulative - reward episodes of the agent is shown in the figure:

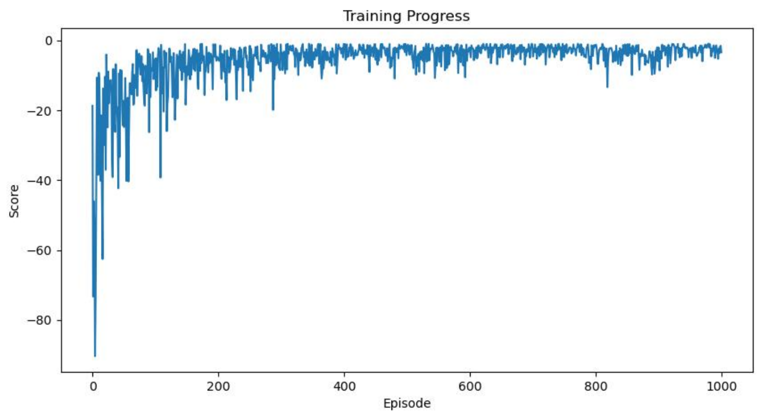

Figure 3 Reward Variation

As can be seen from the Figure 3 Reward Variation, in the early stage, the agent explores the environment and the reward fluctuates greatly. In the later stage, with the progress of training, the reward gradually stabilizes at a relatively high level, indicating that the policy network has effectively learned and chosen the shortest path.

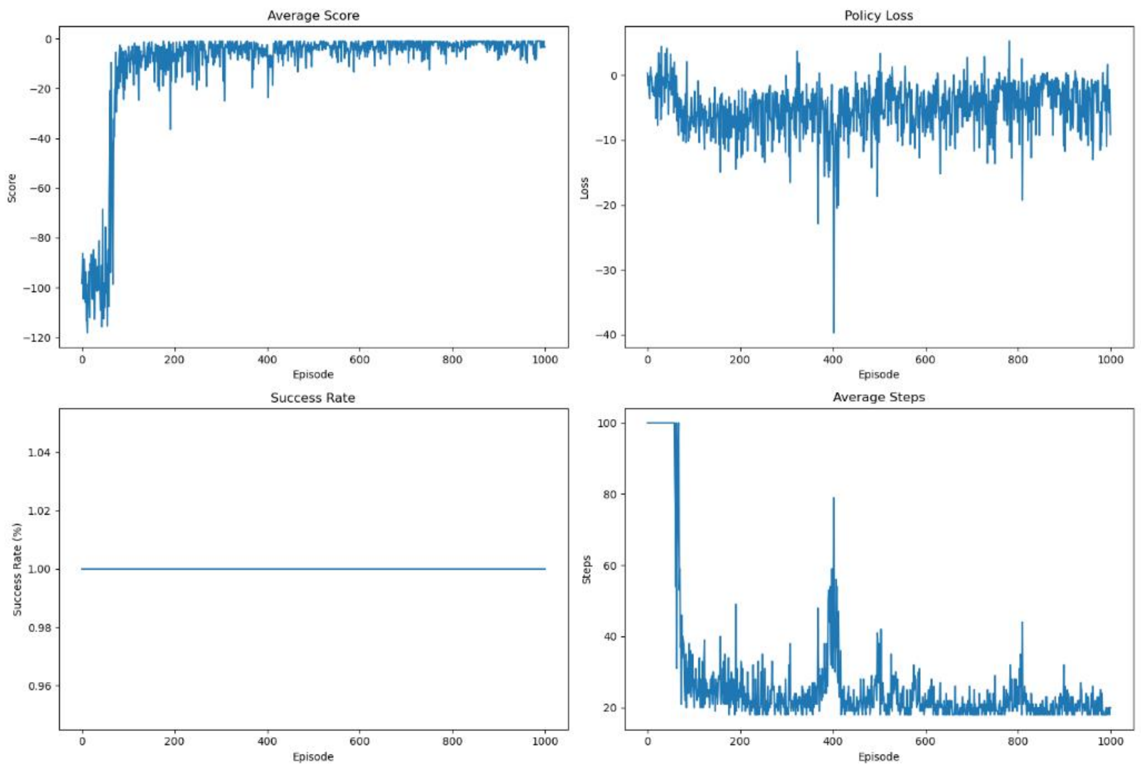

Figure 4 Data Experiment Visualization Results

As can be seen from Figure 4 Data Experiment Visualization Results, the average score fluctuates greatly and has a low value in the early stage of training. This is because the agent hardly obtains positive rewards in the early exploration stage. In the later stage, the average score rises and tends to stabilize, indicating that the agent has learned to implement effective strategies, which shows that the policy gradient algorithm successfully enables the driverless car to learn high - reward strategies. The policy loss fluctuates during training but generally tends to decrease, reflecting the effectiveness of the algorithm. The success rate is nearly 100% with almost no fluctuation, indicating that the model strategy is correct. The agent has good adaptability, can stably achieve the goal, and has a high degree of reliability and generalization ability. The average number of steps fluctuates greatly with peak values in the early stage, and the number of steps to complete the task is unstable. However, it decreases and stabilizes at a lower level in the later stage, indicating that the agent has found a better action sequence and improves efficiency.
Here are the key metrics tracked from Table 3 Data Presentation: success rate, average steps per episode, average reward, and individual episode rewards/steps:

Table 3 Data Presentation

Success Rate	Average Steps:	Average Reward	Rewards per epi -sode	Steps per episode
100.0%	18.2	-1.22	[-0.92,-0.89,-0.94,-1.05,-2.30']	[18, 18, 18, 18, 19]

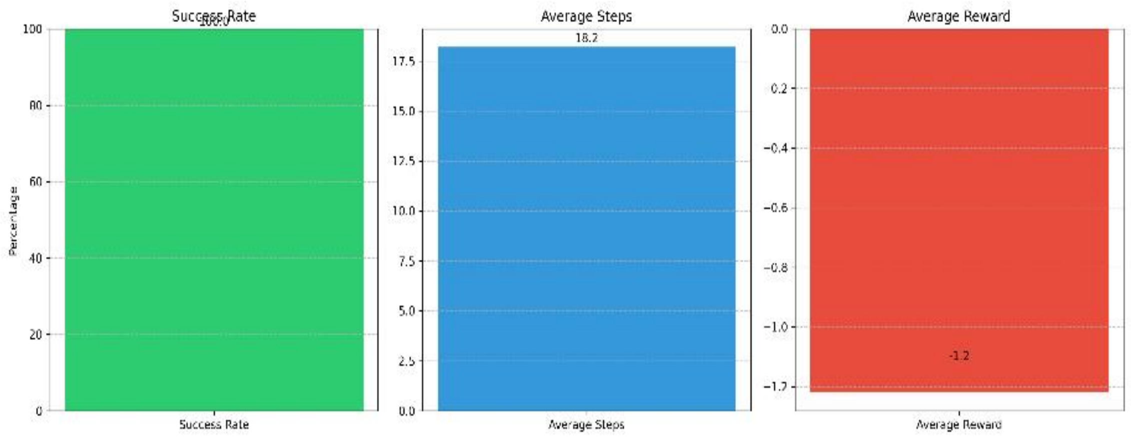

Figure 5 Data Presentation

We can see from Figure 5 Data Presentation table that Average episode steps stabilized at 18.2, demonstrating consistent task - completion patterns and reliable behavioral repeatability.
Notably, the average reward remained negative at -1.22. Despite high success rates and stable episode lengths, this negative reward profile indicates the agent's decision - making under the current reward schema fails to generate net positive returns. Key contributors may include:
Reward Function Formulation: Over - penalization of minor deviations or insufficient goal - achievement incentives.
Policy Optimization Bias: Frequent suboptimal choices prioritizing task completion (high success) over reward maximization, likely due to exploration - exploitation imbalances.

4 CONCLUSION

This paper applies policy gradient methods to address the path planning problem in autonomous driving under mixed traffic flow scenarios. By constructing a state-action space and designing a reward function focused on time efficiency and safety, a three-layer neural network is used to parameterize the decision function, realizing iterative optimization of the policy network. Experimental results show that the agent can stably achieve the shortest path with a 100% success rate, verifying the effectiveness of the policy gradient approach. However, the negative average reward indicates the need for refining the reward function to balance penalty and incentive mechanisms. This study provides a valuable framework for integrating reinforcement learning into intelligent transportation systems, highlighting both the potential of policy gradient methods and the importance of reward function design for practical autonomous driving applications. In the future, we plan to expand the application scope of this approach by incorporating more complex traffic scenarios, such as adverse weather conditions and emergency situations. Additionally, we aim to improve the generalization ability of the model through the integration of transfer learning techniques, enabling it to adapt more effectively to different road networks and traffic regulations.

COMPETING INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

REFERENCES

[1] Piero Scaruffi. The Nature of Intelligence: 64 Big Questions in the Fields of Artificial Intelligence and Robotics. Beijing: People's Posts and Telecommunications Press, 2018.
[2] Sun Changyin, Mu Chaoxu. Several Key Scientific Issues of Multi-Agent Deep Reinforcement Learning. Acta Automatica Sinica, 2020, 46(7): 1301-1312.
[3] Dong Hao, Yang Jing, Li Shaobo, et al. Research Progress of Robot Motion Control Based on Deep Reinforcement Learning. Control and Decision Making, 2022, 37(2): 278-292.
[4] Li Kaiwen, Zhang Tao, Wang Rui, et al. Research Progress of Combinatorial Optimization Based on Deep Reinforcement Learning. Acta Automatica Sinica, 2021, 47(11): 2521-2537.
[5] Xiong Luolin, Mao Shuai, Tang Yang, et al. A Survey of Integrated Energy System Management Based on Reinforcement Learning. Acta Automatica Sinica, 2021, 47(10): 2321-2340.
[6] Yang Ting, Zhao Liyuan, Liu Yachuang, et al. Dynamic Economic Dispatch of Integrated Energy System Based on Deep Reinforcement Learning. Automation of Electric Power Systems, 2021, 45(5): 39-47.
[7] Sutton R S, McAllester D, Singh S, et al. Policy gradient methods for reinforcement learning with function approximation. Advances in neural information processing systems, 1999, 12.
[8] Chen Jiapan, Zheng Minhua. A Survey of Robot Manipulation Behavior Research Based on Deep Reinforcement Learning. Robot, 2022, 44(2): 236-256.
[9] Wang Han, Yu Yang, Jiang Yuan. A Survey of Advances in Multi-Agent Reinforcement Learning Based on Communication. Scientia Sinica (Informationis), 2022, 52(5): 742-764.
[10] Liu Hongqing, Wang Shimin. Research on Vehicle Routing Problem Based on Reinforcement Learning. Computer Applications and Software, 2021, 38(8): 303-308.
[11] Zhang Rongxia, Wu Changxu, Sun Tongchao, et al. Research Progress of Deep Reinforcement Learning and Its Application in Path Planning. Journal of Computer Engineering & Applications, 2021, 57(19).
[12] Huang Dongjin, Jiang Chenfeng, Han Kaili. Three-Dimensional Path Planning Algorithm Based on Deep Reinforcement Learning. Journal of Computer Engineering & Applications, 2020, 56(15).
[13] Xu Hongxin, Wu Zhizhou, Liang Yunyi. A Review of Research on Path Planning Methods for Autonomous Driving Vehicles Based on Reinforcement Learning. Application Research of Computers/Jisuanji Yingyong Yanjiu, 2023, 40(11).
[14] Zhu Maofei, Hu Fangya, Li Nake, Zhu Shouli, Wu Qiong. A Survey of Path Planning Algorithms for Driverless Vehicles. Agricultural Equipment & Vehicle Engineering, 2023, 61(11): 18-22.