TYPE Original Research
PUBLISHED 15 January 2025
DOI 10.3389/fnbot.2024.1428358

EDITED BY
Petia D. Koprinkova-Hristova,
Institute of Information and Communication Technologies (BAS), Bulgaria

REVIEWED BY
Ahmad Ataka,
Gadjah Mada University, Indonesia
Raul-Cristian Roman,
Politehnica University of Timișoara, Romania

*CORRESPONDENCE
Dou Yinke
✉ douyinke@tyut.edu.cn

RECEIVED 06 May 2024
ACCEPTED 26 December 2024
PUBLISHED 15 January 2025

CITATION
Khan S, Niaz A, Yinke D, Shoukat MU and Nawaz SA (2025) Deep reinforcement learning and robust SLAM based robotic control algorithm for self-driving path optimization.
Front. Neurorobot. 18:1428358.
doi: 10.3389/fnbot.2024.1428358

Deep reinforcement learning and robust SLAM based robotic control algorithm for self-driving path optimization

Samiullah Khan[1], Ashfaq Niaz[1], Dou Yinke[1]*, Muhammad Usman Shoukat[2] and Saqib Ali Nawaz[3]

[1]College of Electrical and Power Engineering, Taiyuan University of Technology, Taiyuan, China, [2]Hubei Key Laboratory of Advanced Technology for Automotive Components, School of Automotive Engineering, Wuhan University of Technology, Wuhan, China, [3]School of Information and Communication Engineering, Hainan University, Haikou, China

A reward shaping deep deterministic policy gradient (RS-DDPG) and simultaneous localization and mapping (SLAM) path tracking algorithm is proposed to address the issues of low accuracy and poor robustness of target path tracking for robotic control during maneuver. RS-DDPG algorithm is based on deep reinforcement learning (DRL) and designs a reward function to optimize the parameters of DDPG to achieve the required tracking accuracy and stability. A visual SLAM algorithm based on semantic segmentation and geometric information is proposed to address the issues of poor robustness and susceptibility to interference from dynamic objects in dynamic scenes for SLAM based on visual sensors. Using the Apollo autonomous driving simulation platform, simulation experiments were conducted on the actual DDPG algorithm and the improved RS-DDPG path-tracking control algorithm. The research results indicate that the proposed RS-DDPG algorithm outperforms the DDPG algorithm in terms of path tracking accuracy and robustness. The results showed that it effectively improved the performance of visual SLAM systems in dynamic scenarios.

KEYWORDS

autonomous navigation, robotic control, path tracking, deep reinforcement learning, SLAM, RS-DDPG algorithm

1 Introduction

At present, self-driving technology is one of the research hotspots in the field of artificial intelligence, and reinforcement learning (RL) has drawn significant attention in recent years. In self-driving systems, the most fundamental challenge is the control of path tracking (Yuan et al., 2021). The goal of path tracking control is to enabling vehicles to travel along a predetermined path and approach the fixed trajectory as closely as possible. SLAM is one of the rapidly developing robot perception technologies in recent years, which has been applied in fields such as autonomous navigation (Zheng et al., 2023), augmented reality (Shoukat M. U. et al., 2023), and medical equipment (Shoukat K. et al., 2023). Path tracking control methods are mainly divided into two categories: model based and non-model based. Among them, model-based path tracking control methods mainly rely on the kinematic and dynamic models of the robot, and control the robot's motion by outputting control signals from the controller. Common model-based control methods include proportional integral differential (PID) control (Arrieta et al., 2023; Gopi Krishna Rao et al., 2014; Huba et al., 2023), fuzzy control (Benbouhenni et al., 2023; Zhang et al., 2023), model predictive control (MPC; Fiedler et al., 2023; Wei and Calautit, 2023; Bayat and Allison, 2023), etc. Non-model based path tracking control methods do not require an accurate robot model, but control is achieved through perception and decision-making modules. Common non-model based control methods include neural network control methods (Legaard et al., 2023; Sun et al., 2023).

In the past, researchers have sought solutions from model-based algorithms to address issues such as low tracking accuracy and poor robustness of AVs towards target paths during operation. Mendoza and Yu (2023) proposed a fuzzy adaptive PID control method based on vehicle kinematics and dynamics is used to plan the next driving path based on preview theory. They first used the position relationship between the vehicle center of mass and the desired path preview point to calculate the lateral deviation and heading deviation, and then used the fuzzy adaptive PID controller to adjust the front wheel angle by adjusting the error. Although this method is simple and feasible, its adaptability and control accuracy are limited in high demand control situations. Chen et al. (2023) proposed a horizontal and vertical fuzzy control method based on dynamic dual point preview strategy, which dynamically controls the dual point preview distance through fuzzy control, thereby controlling the vehicle to track the corresponding trajectory. However, the effectiveness of fuzzy control is greatly affected by changes in the preview distance. Wu et al. (2023) proposes a composite fuzzy control method based on lateral error and heading error. This method adjusts the output of two fuzzy controllers by specifying corresponding weight variables, and uses integral compensation to solve the problem of low steady-state accuracy in traditional fuzzy control. However, this method has poor path tracking accuracy in complex roads.

In order to ensure the stability and overall performance of the control-loop system, Roman et al. (2024) suggested a data-driven approach that combines an algorithm with continuous-time active disturbance rejection control. Nguyen et al. (2023) proposed a control algorithm for vehicle trajectory tracking using linear time-varying MPC. Compared to nonlinear control, this method has a global optimal solution and smaller computational complexity. Nevertheless, this method requires high modeling requirements for vehicles, linear approximation for nonlinear systems, and the construction of a quadratic cost function. At the same time, this method requires high hardware storage space and computing power, and needs to consider the limitations of computing resources appropriately. Xue et al. (2024) suggested a novel control technique, which utilizes a neural network (NN) and policy iteration (PI) algorithm to achieve H∞ control in a nonlinear system. Jing (2024) introduced a NN modelling method that utilizes evolutionary computation (EC). The method includes techniques such as NN model compression, distributed NN model, and knowledge distillation. Although the control algorithm can stably track the path, but the system is too complex and has poor stability performance. Chi et al. (2022) presented a method for improving the P-type controller using set point learning called indirect adaptive iterative learning control to improve both linear and nonlinear systems. Zhou et al. (2023) adopts a combination of neural network and fuzzy control method to control the driving direction of the vehicle by controlling the steering wheel angle. The control effect of this method is relatively stable, but there are problems such as untimely steering control and large tracking errors. Model based control algorithms in path tracking rely on robot models, and robot modeling is a complex process that not only needs to consider the influence of various factors such as mechanical structure, dynamic characteristics, control strategies, etc., but also needs to consider the influence of various uncertain factors, making modeling difficult. There are primarily three problems in raising and ensuring the proper functioning of the intelligent vehicular path optimization system.

- Model based control algorithms have a high degree of dependence on the model in the path tracking process, but robot modeling is difficult, which can lead to poor tracking accuracy.
- Non model based control algorithms require a large amount of robot and environmental data for neural network learning, and the completeness of environmental data collection is difficult to meet, which leads to poor tracking performance.
- Current technical conditions are difficult to ensure the integrity of environmental data collection. Lack of complete environmental data can lead to inaccurate information learned by neural networks, resulting in poor tracking performance.

An integrated method combining DRL's adaptability for learning optimal pathways and SLAM's robustness for reliable localization and mapping is necessary to address these problems. To fill in data gaps, the suggested system fuses inputs from several sensors (e.g., LiDAR, radar, and cameras) and using SLAM to produce a continuous, real-time environment map. Table 1 shows the advantages, disadvantages, and main application scenarios of several common visual sensors.

From Table 1, it can be seen that the visual SLAM using cameras as sensors has gradually become one of the main research directions in the field of SLAM. Yuan et al. (2023) calculated the transformation matrix between two frames based on the results of feature point matching and then used this matrix to extract line features and evaluate their static weights. Finally, the remaining static features were used for camera pose estimation to complete the tracking task. Montemerlo and Thrun (2003) proposed FastSLAM by combining EKF and RBPF algorithms. This algorithm estimates the robot pose using the RBPF algorithm and then updates the map using EKF, achieving accurate localization of the robot in unknown environments. Shoukat et al. (2024) introduced a graph SLAM system that utilized YOLOv5 and Wi-Fi fingerprint sequence matching. This algorithm aims to improve the accuracy and resilience of closed-loop detection for robot navigation. Zhang et al. (2018) improved the traditional SLAM system and proposed a dynamic object detection algorithm based on geometric constraints. Dai et al. (2020) used the Delaunay triangulation method to establish a structure similar to the graph for map points, in order to determine their adjacency relationship.

This balanced strategy improves tracking precision, data integrity, and system resilience under real-world uncertainty. This system could improve self-driving technology by improving navigation accuracy, reliability, and adaptability in changing surroundings. The algorithms for RL include SARSA (state action reward state action; Naderi et al., 2023), Q-learning (Puente-Castro et al., 2024), DQN (deep Q-network; Yang and Han, 2023), DDPG (Na et al., 2023), etc. SARSA first creates a Q table and updates its status through interaction with the environment, then takes actions based on the values in the Q table. However, SARSA can only target some simple games. Q-learning is similar to SARSA. The difference in Q-learning is that different strategies are chosen when updating the Q-table, but it is essentially in the form of a table. Q-learning selects the optimal strategy through the Q-table. Moreover, Tampuu et al. (2017), utilizes DQN to train individual agents in a two-player Pong game. However, considering other agents as part of the environment causes instability because

TABLE 1 Sensor performance.

Sensor type	Advantages	Disadvantages	Application scenarios
LiDAR (Li et al., 2024)	Good robustness and high stability; Low computational complexity and lower CPU requirements than camera sensors	Unable to obtain semantic information; Not suitable for harsh environments, such as rainy and foggy weather; Unable to obtain depth information of the perspective body	Indoor low-speed small-scale scenes
Monocular camera (Yu et al., 2023)	Simple structure; Low cost; The calibration and identification process are easy	Unable to determine the depth information of individual images and the true size of objects	Indoor and outdoor scenes
Binocular camera (Zhang et al., 2024)	Can determine the true scale of an object	Large computational load; The calibration process is complex; GPU or FPGA acceleration is required, which consumes a huge amount of computing power	Indoor and outdoor small-scale scenes
RGB-D camera (Song et al., 2023)	Strong dynamism and low computational complexity	Narrow field of view, small measurable range, easily affected by light, unable to recognize transmissive objects	Indoor small-scale scenes
Event camera (Messikommer et al., 2022)	Low latency; Low computational power consumption and low computational power requirements; High dynamic range	Strong data sparsity; More redundant information, less effective information	High speed and high dynamic scenes

agents may adjust their strategies independently. DQN is based on Q-learning and introduces neural networks instead of Q-tables to save software space, but it is not suitable for continuous spaces.

DDPG is a strategy that facilitates depth function approximation, which can be applied in high-dimensional and continuous spaces, while the first three algorithms are only applicable to low dimensional and discrete behavioral spaces. However, in high-dimensional, continuous action autonomous driving, the reward and penalty mechanism of DDPG cannot be well set. Based on the above analysis, it can be concluded that both models based and non-model based path tracking control algorithms have some shortcomings in the path tracking process. Within this particular context, the primary contributions of this paper can be summarized as follows:

- In response to the drawbacks of the above path tracking control algorithms, we developed a reward-shaping deep deterministic policy gradient (RS-DDPG) algorithm for path tracking control maneuvers. This algorithm does not rely on precise data models of the system or requires a large amount of environmental data.
- This article proposed a visual SLAM method for dynamic scenes by combining semantic segmentation networks and multi view geometry methods.
- RS-DDPG for continuous-action tasks in DRL framework to address optimization and robustness concerns. This approach promotes agent collaboration.
- In the proposed algorithm, the robot's path tracking control is achieved by designing reward functions and adaptive weight coefficients based on factors such as the yaw deviation between the robot and its expected trajectory, the lateral angular velocity of the robot, and other relevant parameters.

2 Preliminary

2.1 Markov decision process

The essence of DRL is the interaction process between intelligent agents and the environment, which can be regarded as a Markov decision process (MDP). MDP is a time-dependent sequential decision-making process, where the state at the next moment depends only on the current state and action. MDP defines a five tuple (S,A,R,P,γ), where, $s=\{s_1,s_2,s_3,\ldots\}$ represents the state of the robot; $A=\{a_1,a_2,a_3,\ldots\}$ signifies the actions output by the intelligent agent in the current state; $R=\{r_1,r_2,r_3,\ldots\}$ denotes the reward for the output action in the current state, with lag effect; $P=p\left[s_{t+1},r_t\,|s_t,a_t\right]$ represents the probability function of s_t output action at a_t transferring to the next state s_{t+1} and receiving reward r_t in the current state; and γ is the discount factor, and $\gamma\in[0,1]$.

This study specifies the state-action rate role for any policy π in a very large state-action space. Because getting an exact estimate of $Q_\pi(s,a)$ is not practicable, function approximations such linear functions and neural networks are used (Yang et al., 2022; Yang et al., 2022). Neural networks' strong function approximation abilities have led to their extensive practical application across many domains.

2.2 Reinforcement learning process

In the process of RL, the agent gives action A based on the current state parameter S at each time point, and then enters the next environmental state, providing feedback reward R (schematic diagram of RL process is shown in Figure 1). Then a series of data $(s_1,a_1,r_1,s_2.a_2,r_2,\ldots,s_t,a_t,r_t)$ will be recorded in the memory pool, and the cumulative return G_t will be calculated using the following formula:

$$G_t=R_{t+1}+\gamma R_{t+2}+\ldots=\sum_{k=0}^{+\infty}\gamma^k R_{t+k+1} \tag{1}$$

Use π to represent the policy of the intelligent agent $\pi(a,s)=p(A_t=a|S_t=s)$, and select the probability of outputting action a based on the current state s. Use the value function Q to represent the value of action a taken by s in the current state as $Q(s,a)=E_\pi(G_t|A_t=a,S_t=s)$. Where, $E(x)$ is the expected function. The value function obtained by recursion using the Bellman equation is as in Equation 2:

$$Q(s,a)=E_\pi\left(R_{t+1}+\gamma Q(S_{t+1},A_{t+1})|A_t=a,S_t=s\right) \tag{2}$$

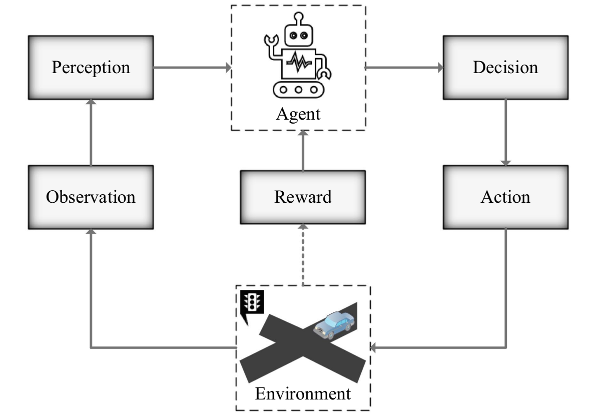

FIGURE 1
Schematic diagram of reinforcement learning process.

The core task of RL is to continuously adjust strategies to maximize the value of the reward function. In the process of reinforcement learning, the agent updates the strategy by maximizing the value of the reward function. The strategy then gives the next action and receives the reward, which loops through to ultimately achieve the system control goal.

In the camera mode of SLAM, combined with semantic segmentation, a semantic segmentation module and a thread for constructing a semantic octree map are added on top of the original front-end odometry, local mapping, and loop detection threads. The overall framework is shown in Figure 2. Firstly, the RGB image obtained by the RGB-D camera is fed into the tracking thread. In the tracking thread, the GCNv2 network is used to extract the key points and descriptors of the current frame. Afterwards, pixel level semantic segmentation is performed on the RGB image through a semantic segmentation network to segment specific objects, including dynamic and static target objects, and preliminary removal of dynamic feature points, such as walking people, is performed. And combined with multi view geometric methods for detection (Cui and Ma, 2019), further removing dynamic objects, and using the remaining static features for pose estimation. Finally, in the semantic map construction thread, the semantic information extracted through semantic segmentation is used to generate a point cloud map and convert it into an octree map.

2.3 Feature extraction

The GCNV2 is a network trained for 3D projection geometry that can be used to extract feature points and descriptors. In contrast to the conventional approach of training with a single image, GCNv2 trains on the TUM and SUN-3D datasets using image pairs. In order to obtain feature points and their corresponding descriptors that are uniformly distributed, GCNv2 takes the input single-channel image and scales it to 320 × 240. The network then takes this adjusted image, extracts its features, and processes them using homogenization and non-maximum suppression. The GCNv2 method's feature extraction procedure is shown in Figure 3 (Shao et al., 2022). It can clearly see from the graph that the extracted features are evenly distributed throughout the entire image.

2.4 Semantic segmentation network

This paper uses the segmentation network DeepLabv3+ (Hu et al., 2022) to complete the semantic segmentation task of image frames. In recent years, many scholars have continuously proposed new semantic segmentation networks, such as PSPNet with a pixel accuracy of 0.9293 and BiSeNet with a pixel accuracy of 0.9337. DeepLabv3+ adds a decoder module to the framework of DeepLabv3, and integrates multi-scale information in the atrous spatial pyramid pooling layer (ASPP) module based on dilated convolution. In the decoder architecture, more accurate object boundaries are obtained through spatial information recovery, optimizing segmentation results, and achieving a pixel accuracy of 0.9431, so, this article chooses DeepLabv3+. Figure 4 shows the process of semantic segmentation algorithm, where pixels represent people and blue pixels represent display screens. After inputting the image into the network, two output values are obtained through DCNN feature extraction: feature map 1 containing high-level semantic information and feature map 2 containing low-level features. Map1 first passes through the ASPP module, and then utilizes 1 × 1 to adjusting the number of channels for convolution yields map1'. Map2 utilizes 1 × 1 to adjust the number of channels for convolution to obtain map2'. Perform 4 up-sampling operations on map1' and concatenate it with map2'. Finally, use 3 × 3 for channel adjustment with 3 convolutions, the final segmentation result is obtained through quadruple up-sampling.

2.5 Semantic map construction

In the semantic map construction thread, PCL library is used to generate point clouds by combining keyframes and depth maps. Then, the pose of the current frame and its point cloud are used for point cloud stitching and filtering processing to generate a point cloud map, and semantic information is annotated in the point cloud map. However, although point cloud maps give people a very intuitive feeling, they have disadvantages such as occupying a large amount of storage space, redundant location information, and cannot be directly used for navigation. Compared to this, octree maps (Ju et al., 2020) also have the intuitiveness of point cloud maps, but their storage space is much smaller, making them suitable for various navigation purposes. Therefore, this article further processes point cloud maps by converting them into octree maps and constructing semantic octree maps based on semantic information. However, during the mapping process, due to camera noise and errors caused by dynamic objects, the same node may have different states at different time points. So, we use probability to explain whether a node is occupied or not. However, this method may result in a probability greater than 1, which can interfere with data processing. Therefore, the probability logarithm is used to describe whether a node is occupied. Let $y \in R$ (real number set) represent the probability logarithm, and the range of occupied probability p is [0,1]. The logit transformation formula is $y = \log(p) = \log\left(\frac{p}{1-p}\right)$. The reversible transformation for logit transformation is as in Equation 3:

$$p = \log it^{-1}(y) = \left(\frac{1}{1+\exp(-y)}\right) \tag{3}$$

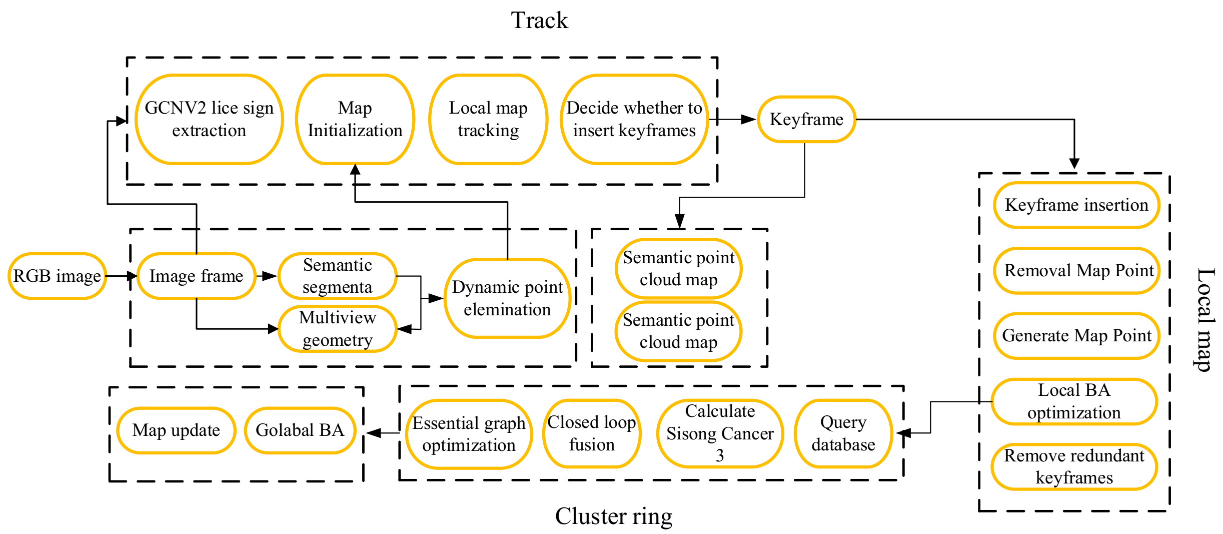

FIGURE 2
Overall framework of visual SLAM algorithm.

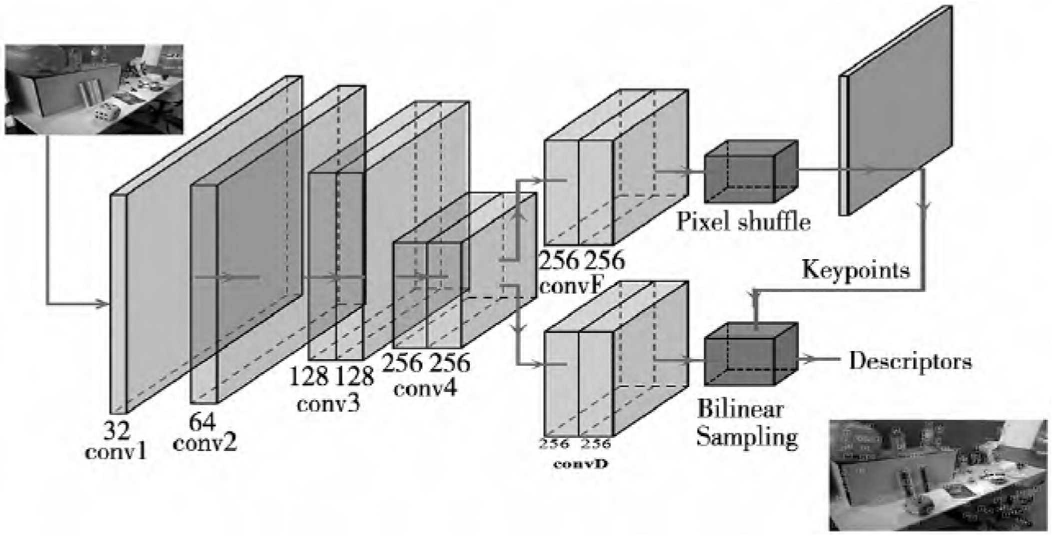

FIGURE 3
GCNv2 network structure.

Assuming that the observation probability of a node n at time T is $P(n|Z_t)$, where Z represents the observation data. The probability of its occupation $P(n|Z_{1:T})$ is represented as in Equation 4:

$$P(n|Z_{1:T}) = \left[1 + \frac{1-P(n|Z_T)}{P(n|Z_T)}\frac{1-P(n|Z_{1:T-1})}{P(n|Z_{1:T-1})}\right] - \frac{P(n)}{1-P(n)} \quad (4)$$

where, $P(n)$ represents the prior probability of node n being occupied, and $P(n|Z_{1:T-1})$ represents the estimated probability of node n from the beginning to the $T-1$ moment. In this article, we set the prior probability $P(n)$ to 0.5, and the above equation is transformed into a probability pair in the form of $L(n|Z_{1:T})$, which represents the logarithmic value of the probability of node n from the beginning to time T. Therefore, at time $T+1$, it is as in Equation 5:

$$L(n|Z_{1:T+1}) = L(n|Z_{1:T-1}) + L(n|Z_T) \quad (5)$$

here, $L(n|Z_{1:T+1})$ and $L(n|Z_T)$ represent the logarithmic values of the probability of node n being occupied before and at time T. According to Equation 3, when a node is repeatedly observed and occupied, its probability logarithm increases, otherwise it decreases. Based on the obtained information, the occupancy probability of this node can be dynamically adjusted to continuously update the octree map.

This article treats a moving person as a dynamic object, and uses a KinectV2 camera mounted on a mobile robot to move uniformly in a

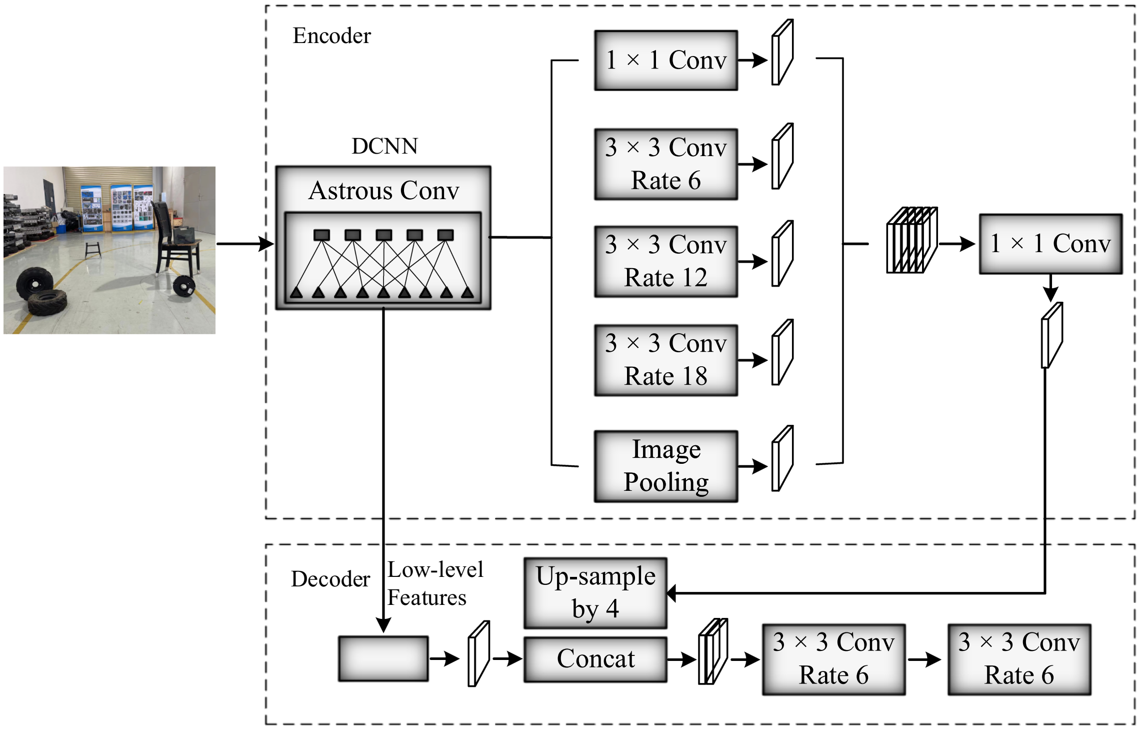

FIGURE 4
DeepLabv3+ semantic segmentation.

dynamic environment according to a previously designed rectangular path, perceive the surrounding environment, and collect information in the scene. Subsequently, using ROS tools, the obtained real scene data was split into frame images, and the TUM dataset was used as the standard to produce the obtained real scene data in the format of the TUM dataset. The algorithm proposed in this paper and the ORB - SLAM2 algorithm were tested to verify their effectiveness and feasibility. Figures 5A,B respectively represent the three-dimensional motion trajectories generated by our algorithm and ORB - SLAM2 algorithm. From Figure 5, it can be clearly seen that due to the presence of moving objects in the experimental scene, the trajectories generated by ORB - SLAM2 algorithm show significant fluctuations compared to the actual motion trajectories. However, the trajectories generated by our algorithm are basically consistent with the actual motion trajectories, and the fluctuation amplitude is relatively small.

This study uses GCNv2 for feature extraction, compared with traditional SLAM system feature extraction methods, the extracted feature points are more evenly distributed. The semantic segmentation network DeepLabv3+ is used to assign semantic information to the image frames in the visual SLAM system, detect moving targets in the objects, and then combine geometric information to detect dynamic feature points.

3 Reward shaping DDPG algorithm

3.1 DDPG algorithm

Among the many actor-critic algorithms that use neural network approximations, DDPG is among the most well-known. DDPG is a model shaping algorithm based on deterministic policy gradients, which is based on the actor-critic framework and can be applied to continuous behavior spaces. The actor network denoted as $\mu\left(s|\theta^{\mu}\right)$, maps a state s to an action a using parameters θ^{μ}. The critic network $Q\left(s,|a,|\theta^{Q}\right)$, evaluating actions with a learning rate α_Q with parameters θ^{Q}. Training parameters like the total number of episodes and steps per episode establish the overall training duration. The function of the actor network is to output action A based on the state S feedback from the environment; The function of the critic network is to output the Q value based on the state S feedback from the environment and the corresponding action A of the actor. The function of actor target network and critic target network is to improve the stability of the network. The network first fixes its own parameters for a period of time, and then updates its own parameters by copying the parameters of the actor network and the critic network, as shown in Figure 6.

On the basis of state observation, the actor network outputs corresponding decision behaviors and parameterizes these behaviors into an n-dimensional vector θ with policy π is $\pi\left(a|s,\theta\right)=p\left(A_t=a|S_t=s\right)$. The projected long-term return is estimated by a critic parameterized by ω in DDPG, while an actor parameterized by θ generates a deterministic policy π_θ.

The actor network is updated based on the policy gradient method, and the policy is improved through the policy gradient. The policy gradient $\nabla_\theta J\left(\theta\right)$ expression is represented in Equation 6:

$$\nabla_\theta J\left(\theta\right)=\left[\frac{\partial J\left(\theta\right)}{\partial\theta_1}\frac{\partial J\left(\theta\right)}{\partial\theta_2}\cdots\frac{\partial J\left(\theta\right)}{\partial\theta_n}\right]^T \tag{6}$$

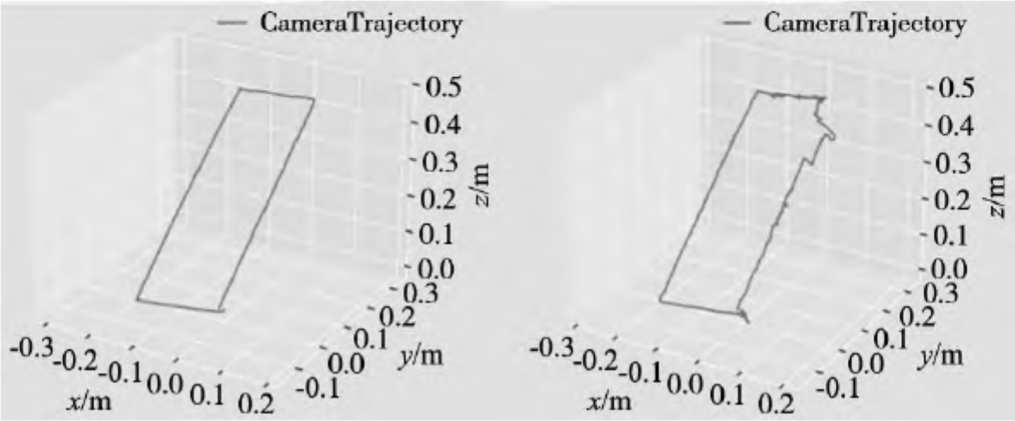

FIGURE 5
Comparison of motion trajectories: **(A)** Algorithm **(B)** ORB-SLAM2 algorithm in this article.

where, $J(\pi_\theta)$ is the policy objective function. The policy gradient $\nabla_\theta J(\theta)$ expression for stochastic policies is $\nabla_\theta J(\pi_\theta) = E_{\pi\theta}\left(\left[\nabla_\theta \log \pi_\theta(s,a)\right] Q^\pi(s,a)\right) \cdot \nabla_\theta \log \pi_\theta(s,a)$ is a fractional function, which can be expressed in Equation 7:

$$\nabla_\theta \log \pi_\theta(s,a) = \left(\nabla_\theta \pi_\theta(s,a)\right) / \left(\pi_\theta(s,a)\right) \tag{7}$$

In deterministic strategy, $a = \mu_\theta(s)$, the gradient of deterministic strategy is a special form of stochastic policy where the gradient variance approaching 0, and its gradient expression is as in Equation 8:

$$\nabla_\theta J(\mu_\theta) = E_{\mu\theta}\left(\nabla_\theta \mu_\theta(s) Q^\mu(s,a) \middle| \mu_\theta(S)\right) \tag{8}$$

In the learning process of intelligent robot trajectory tracking control, the input of the actor neural network is the observed environmental state variables, such as position, angle, speed, etc. Its output is decisions made based on strategies, such as steering wheel angle and throttle braking. At the same time, critic's approach is based on the behavioral value function, where the input variables are state and behavior, and the output variables are return values. During the learning process, critic uses the estimated value function as the benchmark for updating the actor function, while evaluating the actor's strategy. The advantage of the actor-critic method is that critic provides a more accurate evaluation through the value function, thereby improving the actor strategy and making it more optimized. In addition, the actor-critic method can not only use critic to update actor policies, but also update the value function of critic, which can better evaluate behavioral value.

In practice, the value function of critic is updated using the Bellman equation $Q'(s,a) = Q(s,a) + \alpha\left[R(s,a) + \gamma \max Q(s,a) - Q(s,a)\right]$, where, α is for learning rate and Q' is a new value function. The actor network updates the parameter θ using chain differentiation (Equation 9).

$$\nabla_\theta J(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left[\nabla Q(s_i,a_i)\nabla \pi_\theta(a_i|s_i)\right] \tag{9}$$

The critical network updates the parameter *w*, by taking the mean square error (MSE) between the expected and actual values, i.e, as represented in Equation 10.

$$J(w) = \frac{1}{n}\sum_{t=1}^{n}\left(Q(s',a',w') - Q(s_i,a_i,w)\right)^2 \tag{10}$$

here, $Q(s',a',w')$ is the target value calculated by the critic target network.

3.2 Reward function design

The quality of the reward function is a key factor affecting the results of the model. Intelligent agents for a single task have clear reward goals, so it is necessary to maximize the reward value. However, in dealing with complex autonomous driving tasks, it is difficult to have a single clear reward objective. Therefore, this paper intends to design a reward function through a combination approach, known as:

1) Path tracking capability. The lateral distance between the robot's center of mass position y_i and the expected trajectory y_j was designed to describe the tracking accuracy of the robot as in Equation 11:

$$R_1 = \Delta_y = \left|y_i - y_j\right| \tag{11}$$

The ratio of tracking accuracy error to allowable error is Δ_1, and its mentioned in Equation 12.

$$\Delta_1 = \Delta_y / 0.3 \tag{12}$$

2) Speed. $R_2 = V_x \cos(\theta)$, where, $V_x \cos(\theta)$ is the speed of the vehicle along the expected path direction, and it is expected to complete the driving task quickly in limited time and safety.

3) Robot stability. The stability of a robot is mainly reflected by its yaw rate and center of mass lateral deviation angle. The yaw rate is $R_3 = \Delta_\omega = \left|\omega_p - \omega_t\right|$ and described as the difference between the actual yaw rate ω_p and the expected yaw rate ω_t, where, $\omega_t = \mathrm{mim}\left(\left|\omega_{\mathrm{des}}\right|, \omega_d\right)\mathrm{sgn}(\delta)$ and ω_d is the upper limit of lateral angular velocity. ω_{des} is the yaw rate under steady-state steering, and $\omega_{des} = G_{\omega zss} \times \delta$. Here, $G_{\omega zss}$ is known as steady-state gain of the yaw rate and δ is the angle of the steering wheel.

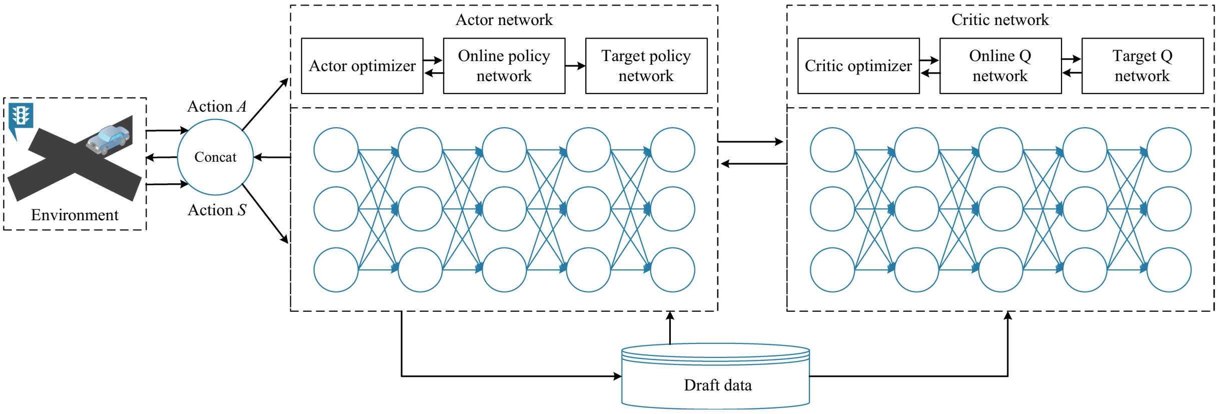

FIGURE 6
DDPG algorithm with actor and critic networks.

The ratio of lateral angular velocity error to expected angular velocity is $\Delta 2 = \Delta_\omega / \omega_t$. Similarly, the center of mass deviation angle is described by the difference between the actual center of mass deviation angle β_p and the expected center of mass deviation angle β_t as represented in Equations 13, 14.

$$R_4 = \Delta_\beta - \left|\beta_p - \beta_t\right| \tag{13}$$

$$\beta_t = \min\left(\left|\beta_{\text{des}}\right|, \beta_d\right) \operatorname{sgn}(\delta) \tag{14}$$

where, β_d is the upper limit of the lateral deviation angle of the center of mass. β_{des} is the lateral deviation angle of the center of mass under steady-state turning, and $\beta_{des} = G_{\beta zss} \times \delta$. $G_{\beta zss}$ is the steady-state gain of the center of mass sideslip angle.

The ratio of the deviation angle of the center of mass to the expected deviation angle of the center of mass is $\Delta 3 = \Delta_\beta / \beta_t$.

4) Steering stability. The smoothness of steering represents the degree of steering wheel oscillation, and a coefficient of variation R_5 is $R_5 = C_v = \sigma / \tilde{\theta}$. Where, σ is the standard deviation of the steering wheel angle and $\tilde{\theta}$ is the average value of the steering wheel angle. The aforementioned RS-DDPG approach involves a collective reward shaping that is distributed across all agents in collaborative circumstances. Nevertheless, this factor is frequently ignored in several real-world scenarios.

3.3 Adaptive weight design

The accuracy of path tracking and the stability performance of robots have a significant impact on the control of autonomous driving path tracking. When both cannot be met simultaneously, it is necessary to determine which indicator with a large gap should be dealt with first. This study designed adaptive weight coefficients. When the percentage of tracking accuracy error is greater than the percentage of stability error, the weight of the reward function for tracking accuracy will increase, and vice versa. The weight and stability weight coefficients for tracking accuracy are in Equations 15, 16

$$C_1 = 0.5 + e^{\Delta_1} / \left(e^{\Delta_1} + e^{\Delta_2 + \Delta_3}\right) \tag{15}$$

$$C_2 = 0.5 + e^{e^{\Delta_2 + \Delta_3}} / \left(e^{\Delta_1} + e^{\Delta_2 + \Delta_3}\right) \tag{16}$$

The tracking accuracy weight coefficient and stability weight coefficient satisfy $C_1 = C_2 = 2$ expressions. During the training process, AVs may encounter two situations: normal driving and exceeding the lane. The reward function for normal driving has been designed, and the situation of exceeding the lane is uniformly set to 0 here. The expression for the reward function is represented in Equation 17:

$$R = \begin{cases} R_2 - C_1 R_1 - C_2\left(R_3 + R_4\right) - R_5, \text{Normal} \\ 0, \qquad \text{Beyond the lane} \end{cases} \tag{17}$$

4 Simulation testing and analysis

In order to evaluate the advantages and disadvantages of the proposed autonomous driving robot control method in this study, a model will be built on the Apollo simulation platform, and the trajectory tracking process of intelligent robots will be simulated and analyzed using proposed algorithm and actual DDPG algorithm. The RS-DDPG algorithm in this study is based on the actor-critic network structure, where the actor network is updated using the policy gradient method, and the policy is optimized in a better direction based on the policy gradient. The input of the actor network is observation (position, angle, speed, etc.), and the output is control signals, such as steering wheel angle and accelerator brake.

The critic network develops using the behavioral value function, which takes state and behavior as input factors and outputs return values as output variables. This network is utilized to assess the efficiency of different techniques. The paper's RS-DDPG approach is more generalizable and robust than the traditional DDPG method since it uses a novel reward function. The evaluation methods of the DDPG algorithm and RS-DDPG algorithm are shown in Figure 7.

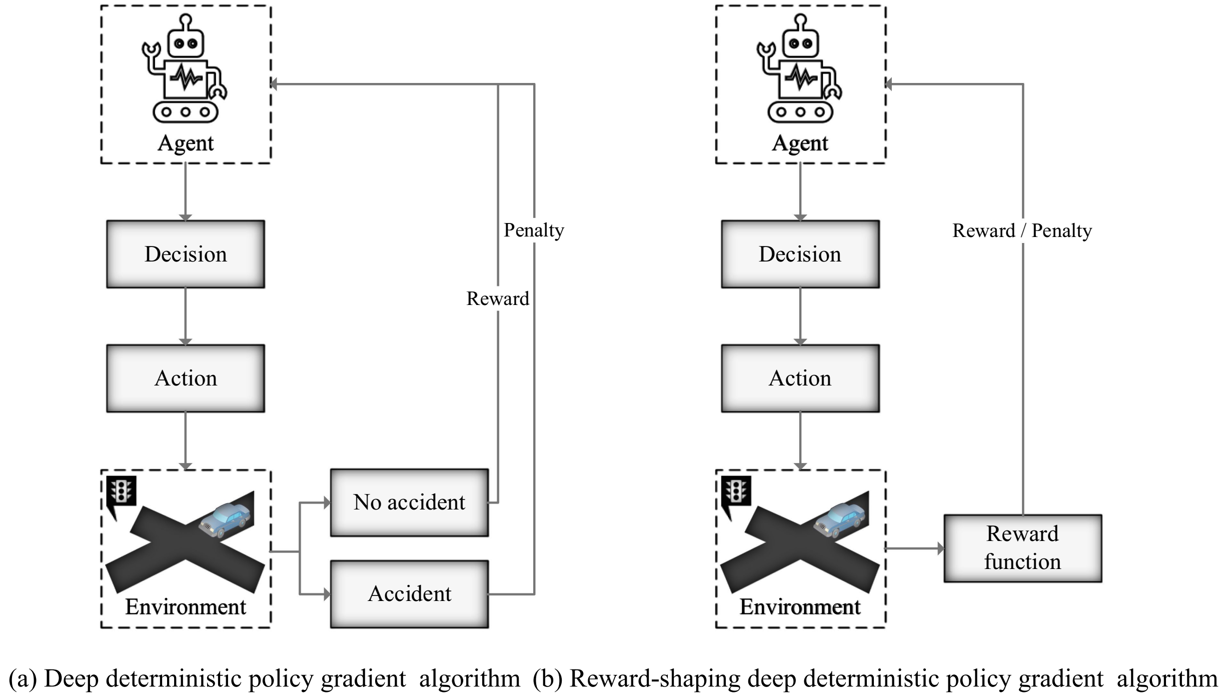

FIGURE 7
Evaluation methods of DDPG and RS-DDPG algorithms.

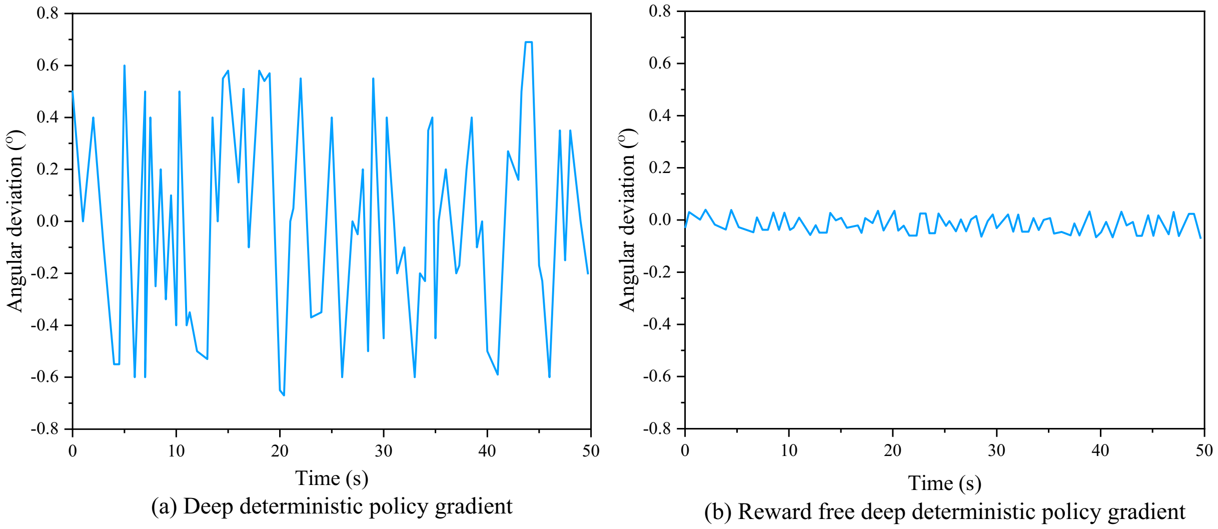

FIGURE 8
Heading angle deviation curves of autonomous robots before and after algorithm improvement.

Figure 7A shows the actual evaluation algorithm. It is noticeable that the actual algorithm differentiates between evaluation techniques for intelligent robots that are based on accidents and non-accidents. However, the trained results failed to achieve the required accuracy standards for intelligent robots path tracking. Figure 7B shows the enhanced assessment achieved using a combined method in designing a reward function, resulting in a more logical assessment and better accuracy of the control effect after training. The heading angle deviation curve, yaw rate deviation curve, and center of mass lateral deviation curve of AVs before and after algorithm improvement are shown in Figures 8–10, respectively.

From Figures 8–10, it can be seen that the stability performance of AVs using RS-DDPG control algorithm during the experimental process is significantly higher than that of robots using DDPG algorithm, and the control process is more reasonable. This study

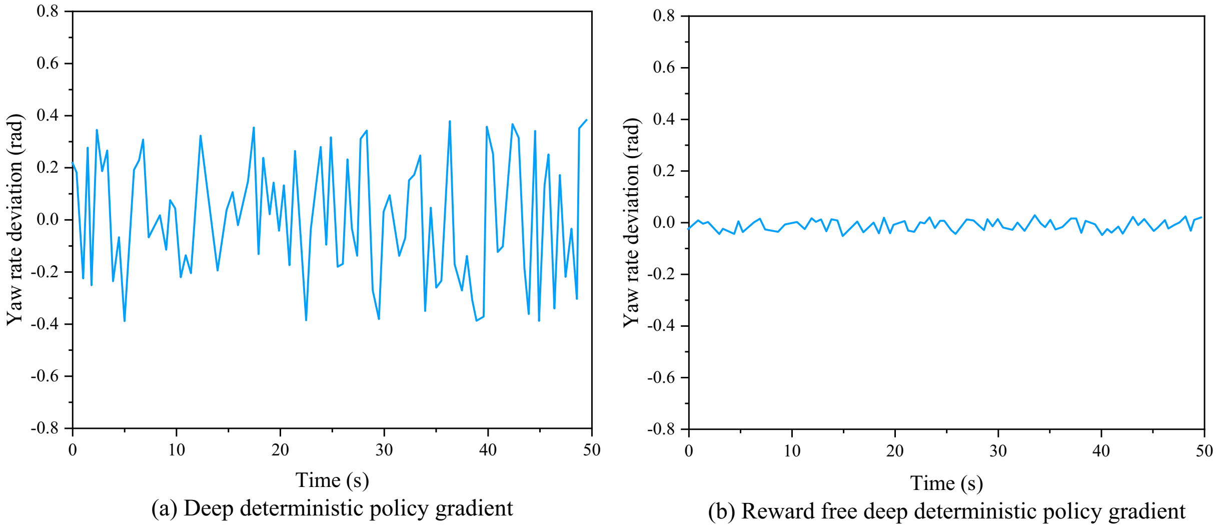

FIGURE 9
Yaw rate deviation curve of autonomous robots before and after algorithm modification.

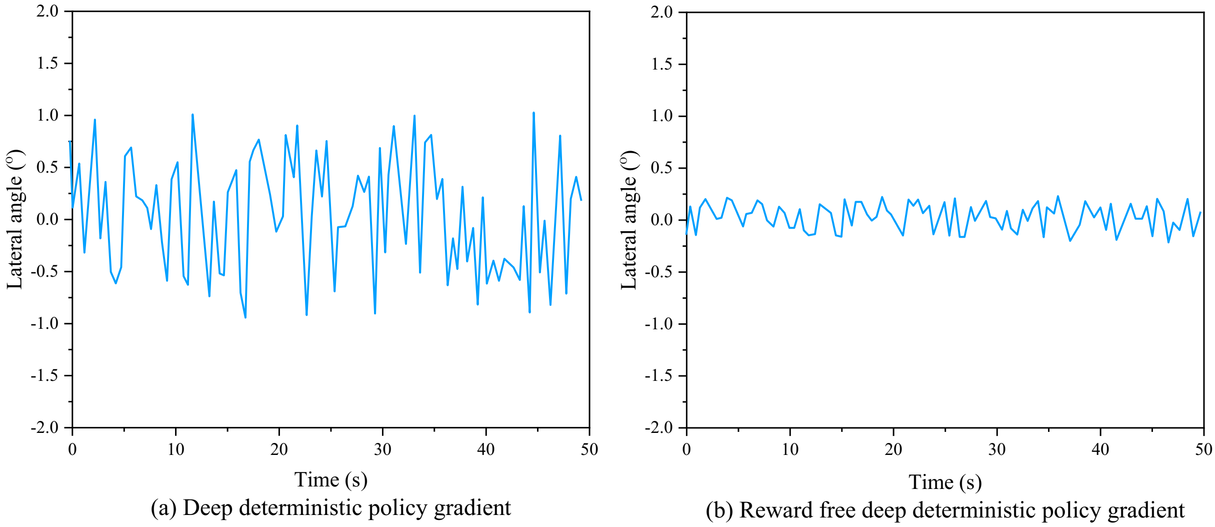

FIGURE 10
Curve of center of mass lateral deviation angle of autonomous driving robots before and after algorithm enhancement.

not only confirms the efficacy of the algorithm strategy output, but also demonstrates the strong application of the improvement approach in the simulated environment. Figure 11 shows a comparison of lateral errors between AVs using RS-DDPG and DDPG algorithms.

As shown in Figure 11, the comparison of lateral errors between AVs using RS-DDPG and DDPG approaches can also be seen intuitively that the RS-DDPG control algorithm has higher tracking accuracy performance than the DDPG algorithm, and the control process is more reasonable. Table 2 compares the results of different tracking control values using DDPG and RS-DDPG control algorithms. From the data in Table 2, it can be concluded that the tracking control values of RS-DDPG algorithm are better than the corresponding values of DDPG algorithm.

5 Conclusion

This article takes intelligent robots as the research object and uses reinforcement learning based methods to study the optimal control problem of robots in tracking trajectories. A DRL based RS-DDPG and visual SLAM path tracking algorithms are proposed, aiming to optimize the tracking accuracy and operational stability of robots. Enhanced the robustness of the visual SLAM system in dynamic environments, and utilized semantic information to generate a static

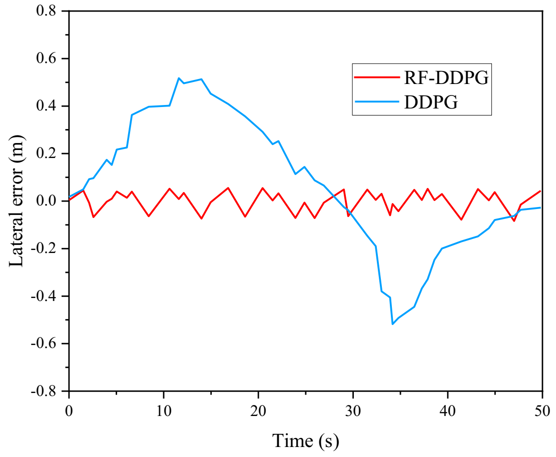

FIGURE 11
Comparison of lateral errors of autonomous robots before and after algorithm improvement.

TABLE 2 Comparison of tracking control values before and after algorithm improvement.

Parameters	Tracking control values for different algorithms	
	DDPG	RS-DDPG
Maximum absolute value of lateral error (*m*)	0.52	0.07
Average absolute value of lateral error (*m*)	0.22	0.03
Maximum absolute value of angular deviation (°)	0.60	0.05
Average absolute value of angular deviation (°)	0.29	0.03
Maximum absolute value of angular velocity deviation (rad)	0.40	0.03
Average absolute deviation of angular velocity (rad)	0.22	0.01

semantic octree map, saving a lot of storage space. At the same time, the generated map can be directly used for robot path planning. On the basis of DRL, these algorithm designs a reward function and adaptive weight coefficients for intelligent robots in trajectory tracking, thereby optimizing the parameters of RS-DDPG. The controller takes the current position, speed, tracking path information, and heading angle of the robot as inputs, and outputs the steering wheel angle and throttle brake. Intelligent robot trajectory tracking performance using the algorithm proposed in this paper and the actual DDPG algorithm was tested on a simulation platform. The simulation results prove that the RS-DDPG based RL method, proposed in this paper, has substantial enhancements in tracking accuracy and control effectiveness compared to the actual DDPG method. Furthermore, it guarantees the safety and stability of the robot's driving process. To explore the problem of intelligent robot trajectory tracking further, the next research will continue to conduct trajectory planning, apply the following control strategies to the planned trajectory and conduct simulation verification of the trajectory tracking strategy. On this basis, the RS-DDPG algorithm can be further improved to enhance its control accuracy and robustness. This study is of great significance for intelligent robots' autonomous driving and intelligent transportation systems' development. It is expected to provide effective technical support for achieving safe driving of robots and smooth traffic.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

AN: Data curation, Formal analysis, Methodology, Writing – original draft, Writing – review & editing. MS: Conceptualization, Data curation, Formal analysis, Methodology, Writing – original draft, Writing – review & editing. SN: Resources, Visualization, Writing – original draft, Writing – review & editing. SK: Methodology, Software, Writing – original draft. DY: Supervision, Funding acquisition, Project administration, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. Research on the radar echo inversion mechanism of shallow ice and snow material balance parameters in the Antarctic ice sheet under the funding of National Natural Science Foundation of China General Project (4216231).

Acknowledgments

The author(s) wish to acknowledge Dou Yinke (Taiyuan University of Technology, Taiyuan City (030024), Shanxi, China) for providing the facilities and funding needed for this research work.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Arrieta, O., Campos, D., Rico-Azagra, J., Gil-Martínez, M., Rojas, J. D., and Vilanova, R. (2023). Model-based optimization approach for PID control of pitch–roll UAV orientation. *Mathematics* 11:3390. doi: 10.3390/math11153390

Bayat, S., and Allison, J. T. (2023). SS-MPC: a user-friendly software based on single shooting optimization to solve model predictive control problems. *Softw. Impacts* 17:100566. doi: 10.1016/j.simpa.2023.100566

Benbouhenni, H., Bizon, N., Mosaad, M. I., Colak, I., Djilali, A., and Gasmi, H. (2023). Enhancement of the power quality of DFIG-based dual-rotor wind turbine systems using fractional order fuzzy controller. *Expert Syst. Appl.* 238:121695. doi: 10.1016/j.eswa.2023.121695

Chen, L., Qin, Z., Hu, M., Gao, H., Bian, Y., Xu, B., et al. (2023). Trajectory tracking control of autonomous heavy-duty mining dump trucks with uncertain dynamic characteristics. *SCIENCE CHINA Inf. Sci.* 66:202203. doi: 10.1007/s11432-022-3713-8

Chi, R., Li, H., Shen, D., Hou, Z., and Huang, B. (2022). Enhanced P-type control: indirect adaptive learning from set-point updates. *IEEE Trans. Autom. Control* 68, 1600–1613. doi: 10.1109/TAC.2022.3154347

Cui, L., and Ma, C. (2019). SOF-SLAM: a semantic visual SLAM for dynamic environments. *IEEE Access* 7, 166528–166539. doi: 10.1109/ACCESS.2019.2952161

Dai, W., Zhang, Y., Li, P., Fang, Z., and Scherer, S. (2020). Rgb-d slam in dynamic environments using point correlations. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 373–389. doi: 10.1109/TPAMI.2020.3010942

Fiedler, F., Karg, B., Lüken, L., Brandner, D., Heinlein, M., Brabender, F., et al. (2023). Do-mpc: towards FAIR nonlinear and robust model predictive control. *Control. Eng. Pract.* 140:105676. doi: 10.1016/j.conengprac.2023.105676

Gopi Krishna Rao, P. V., Subramanyam, M. V., and Satyaprasad, K. (2014). Design of internal model control-proportional integral derivative controller with improved filter for disturbance rejection. *Syst. Sci. Cont. Eng.* 2, 583–592. doi: 10.1080/21642583.2014.956372

Hu, Z., Zhao, J., Luo, Y., and Ou, J. (2022). Semantic SLAM based on improved DeepLabv3$^+$ in dynamic scenarios. *IEEE Access* 10, 21160–21168. doi: 10.1109/ACCESS.2022.3154086

Huba, M., Vrancic, D., and Bistak, P. (2023). Series PID control with higher-order derivatives for processes approximated by IPDT models. *IEEE Trans. Autom. Sci. Eng.* 21:4406–4418. doi: 10.1109/TASE.2023.3296201

Jing, N. I. N. G. (2024). Neural network-based pattern recognition in the framework of edge computing. *Sci. Technol.* 27, 106–119. doi: 10.59277/ROMJIST.2024.1.08

Ju, C., Luo, Q., and Yan, X. (2020). "Path planning using an improved a-star algorithm." in *2020 11th international conference on prognostics and system health management (PHM-2020 Jinan)*. pp. 23–26. IEEE.

Legaard, C., Schranz, T., Schweiger, G., Drgoňa, J., Falay, B., Gomes, C., et al. (2023). Constructing neural network based models for simulating dynamical systems. *ACM Comput. Surv.* 55, 1–34. doi: 10.1145/3567591

Li, H., Zou, Y., Chen, N., Lin, J., Liu, X., Xu, W., et al. (2024). MARS-LVIG dataset: a multi-sensor aerial robots SLAM dataset for LiDAR-visual-inertial-GNSS fusion. *Int. J. Robot. Res.* 43, 1114–1127. doi: 10.1177/02783649241227968

Mendoza, A. M. E. R., and Yu, W. (2023). Fuzzy adaptive control law for trajectory tracking based on a fuzzy adaptive neural PID controller of a multi-rotor unmanned aerial vehicle. *Int. J. Control. Autom. Syst.* 21, 658–670. doi: 10.1007/s12555-021-0299-2

Messikommer, N., Georgoulis, S., Gehrig, D., Tulyakov, S., Erbach, J., Bochicchio, A., et al. (2022). "Multi-bracket high dynamic range imaging with event cameras." in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 547–557.

Montemerlo, M., and Thrun, S. (2003). "Simultaneous localization and mapping with unknown data association using FastSLAM." in *2003 IEEE international conference on robotics and automation (Cat. No. 03CH37422)*. Vol. 2, pp. 1985–1991. IEEE.

Na, Y., Li, Y., Chen, D., Yao, Y., Li, T., Liu, H., et al. (2023). Optimal energy consumption path planning for unmanned aerial vehicles based on improved particle swarm optimization. *Sustain. For.* 15:12101. doi: 10.3390/su151612101

Naderi, M., Mahdaee, K., and Rahmani, P. (2023). Hierarchical traffic light-aware routing via fuzzy reinforcement learning in software-defined vehicular networks. *Peer Peer Netw. Appl.* 16, 1174–1198. doi: 10.1007/s12083-022-01424-2

Nguyen, H. D., Kim, D., Son, Y. S., and Han, K. (2023). Linear time-varying MPC-based autonomous emergency steering control for collision avoidance. *IEEE Trans. Veh. Technol.* 72, 12713–12727. doi: 10.1109/TVT.2023.3269787

Puente-Castro, A., Rivero, D., Pedrosa, E., Pereira, A., Lau, N., and Fernandez-Blanco, E. (2024). Q-learning based system for path planning with unmanned aerial vehicles swarms in obstacle environments. *Expert Syst. Appl.* 235:121240. doi: 10.1016/j.eswa.2023.121240

Roman, R. C., Precup, R. E., Petriu, E. M., and Borlea, A. I. (2024). Hybrid data-driven active disturbance rejection sliding mode control (SMC) with tower crane systems validation. *Sci. Technol.* 27, 3–17.

Shao, X., Pan, W., Liu, G., Tan, C., and Zhong, Q. (2022). "A RGB-D visual SLAM algorithm based on GCNv2 and GMS for dynamic scenes." in *Proceedings of the 2022 5th International Conference on Signal Processing and Machine Learning*. pp. 262–267.

Shoukat, K., Jian, M., Umar, M., Kalsoom, H., Sijjad, W., Atta, S. H., et al. (2023). Use of digital transformation and artificial intelligence strategies for pharmaceutical industry in Pakistan: applications and challenges. *Artif. Intell. Health* 1:1486. doi: 10.36922/aih.1486

Shoukat, M. U., Yan, L., Deng, D., Imtiaz, M., Safdar, M., and Nawaz, S. A. (2024). Cognitive robotics: deep learning approaches for trajectory and motion control in complex environment. *Adv. Eng. Inform.* 60:102370. doi: 10.1016/j.aei.2024.102370

Shoukat, M. U., Yan, L., Zhang, J., Cheng, Y., Raza, M. U., and Niaz, A. (2023). Smart home for enhanced healthcare: exploring human machine interface oriented digital twin model. *Multimed. Tools Appl.* 83, 31297–31315. doi: 10.1007/s11042-023-16875-9

Song, P., Li, Z., Yang, M., Shao, Y., Pu, Z., Yang, W., et al. (2023). Dynamic detection of three-dimensional crop phenotypes based on a consumer-grade RGB-D camera. *Front. Plant Sci.* 14:1097725. doi: 10.3389/fpls.2023.1097725

Sun, X., Zhang, L., and Gu, J. (2023). Neural-network based adaptive sliding mode control for Takagi-Sugeno fuzzy systems. *Inf. Sci.* 628, 240–253. doi: 10.1016/j.ins.2022.12.118

Tampuu, A., Matiisen, T., Kodelja, D., Kuzovkin, I., Korjus, K., Aru, J., et al. (2017). Multiagent cooperation and competition with deep reinforcement learning. *PLoS One* 12:e0172395. doi: 10.1371/journal.pone.0172395

Wei, Z., and Calautit, J. (2023). "Evaluation of model predictive control (MPC) of solar thermal heating system with thermal energy storage for buildings with highly variable occupancy levels" in Building simulation, vol. *16* (Beijing: Tsinghua University Press), 1915–1931.

Wu, G., Wang, G., Bi, Q., Wang, Y., Fang, Y., Guo, G., et al. (2023). Research on unmanned electric shovel autonomous driving path tracking control based on improved pure tracking and fuzzy control. *J. Field Robot.* 40, 1739–1753. doi: 10.1002/rob.22208

Xue, Z., Cheng, S., Li, L., and Zhong, Z. (2024). Nonlinear H∞ path following control for autonomous ground vehicles via neural network and policy iteration algorithm. *Proc. Inst. Mech. Eng. Part D J. Automobile Eng.* 238, 1670–1683. doi: 10.1177/09544070221145468

Yang, X., and Han, Q. (2023). Improved DQN for dynamic obstacle avoidance and ship path planning. *Algorithms* 16:220. doi: 10.3390/a16050220

Yang, Y., Pan, Y., Xu, C. Z., and Wunsch, D. C. (2022). Hamiltonian-driven adaptive dynamic programming with efficient experience replay. IEEE Transactions on Neural Networks and Learning Systems.

Yang, Y., Zhu, H., Zhang, Q., Zhao, B., Li, Z., and Wunsch, D. C. (2022). Sparse online kernelized actor-critic learning in reproducing kernel Hilbert space. *Artif. Intell. Rev.* 55, 23–58. doi: 10.1007/s10462-021-10045-9

Yu, Y., Fan, S., Li, L., Wang, T., and Li, L. (2023). Automatic Targetless monocular camera and LiDAR external parameter calibration method for Mobile robots. *Remote Sens.* 15:5560. doi: 10.3390/rs15235560

Yuan, X., Liu, H., and Qi, R. (2021). Research on key technologies of autonomous driving platform. *J. Phys. Conf. Ser.* 1754:012127. doi: 10.1088/1742-6596/1754/1/012127

Yuan, C., Xu, Y., and Zhou, Q. (2023). PLDS-SLAM: point and line features SLAM in dynamic environment. *Remote Sens.* 15:1893. doi: 10.3390/rs15071893

Zhang, X., Lv, T., Dan, W., and Minghao, Z. (2024). High-precision binocular camera calibration method based on a 3D calibration object. *Appl. Opt.* 63, 2667–2682. doi: 10.1364/AO.517411

Zhang, H., Wang, L., and Shi, W. (2023). Seismic control of adaptive variable stiffness intelligent structures using fuzzy control strategy combined with LSTM. *J. Build. Eng.* 78:107549. doi: 10.1016/j.jobe.2023.107549

Zhang, L., Wei, L., Shen, P., Wei, W., Zhu, G., and Song, J. (2018). Semantic SLAM based on object detection and improved octomap. *IEEE Access* 6, 75545–75559. doi: 10.1109/ACCESS.2018.2873617

Zheng, S., Wang, J., Rizos, C., Ding, W., and El-Mowafy, A. (2023). Simultaneous localization and mapping (SLAM) for autonomous driving: concept and analysis. *Remote Sens.* 15:1156. doi: 10.3390/rs15041156

Zhou, Z., Ding, X., Shi, Z., Yin, X., and Meng, X. (2023). In-wheel motor electric vehicle based on fuzzy neural network yaw stability optimization control. *Eng. Lett.* 31:1717–1723. Available at: https://www.engineeringletters.com/issues_v31/issue_4/EL_31_4_42.pdf