IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 9, 2021, accepted April 26, 2021, date of publication April 29, 2021, date of current version May 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3076530

Motion Planning for Mobile Robots–Focusing on Deep Reinforcement Learning: A Systematic Review

HUIHUI SUN[1,2], WEIJIE ZHANG[1], RUNXIANG YU[1,2], AND YUJIE ZHANG[1,2]
[1]College of Mechanical and Electrical Engineering, North China Institute of Science and Technology, Langfang 065201, China
[2]Hebei Key Laboratory of Safety Monitoring of Mining Equipment, Langfang 065201, China

Corresponding author: Yujie Zhang (zhangyujiebetter@163.com)

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 3142019055 and Grant 3142015038, in part funded by the Science and Technology Project of Hebei Education Department under Grant QN2021312, and in part by the Langfang Science and Technology Research Development Program under Grant 2019011058.

ABSTRACT Mobile robots contributed significantly to the intelligent development of human society, and the motion-planning policy is critical for mobile robots. This paper reviews the methods based on motion-planning policy, especially the ones involving Deep Reinforcement Learning (DRL) in the unstructured environment. The conventional methods of DRL are categorized to value-based, policy-based and actor-critic-based algorithms, and the corresponding theories and applications are surveyed. Furthermore, the recently-emerged methods of DRL are also surveyed, especially the ones involving the imitation learning, meta-learning and multi-robot systems. According to the surveys, the potential research directions of motion-planning algorithms serving for mobile robots are enlightened.

INDEX TERMS Mobile robot, deep reinforcement learning, motion planning.

I. INTRODUCTION

Mobile robots are commonly applied in the dangerous and complex environments to complete various tasks [1], including exploring unknown and chaotic environments, searching trapped victims and carrying important materials, etc. In the urgent and dangerous scenes, mobile robots may even be irreplaceable [2].

In recent decades, the research on mobile robot technology may reflect the trend of the advanced technology [3]. Many countries regard mobile robots as a key industry in the robotics field. With the rapid development of Artificial Intelligence (AI), sensors, networks and communication technology, mobile robots become more and more intelligent [4]. Intelligent mobile robot should have the high autonomy, and thus it could quickly and accurately complete tasks without the guidance of human, ignoring any environmental restrictions [5]. The motion planning is to search for a non-collision optimal path from the current position to the destination with the external environment information obtained by the sensors [6]. It is one of the most important capabilities of intelligent robots. Traditional motion planning methods is commonly incapable to complete the tasks in the dynamically-changing scenes lacking of the prior knowledge and maps, such as areas where fire, earthquake, explosion and other emergency cases occur [7].

In this context, motion planning methods with self-learning ability becomes an important research direction [8], [9]. Today, Robot AI technology has been deeply applied to all aspects of society, and Deep Learning (DL) and Reinforcement Learning(RL) technology is considered as the most appropriate methods to solve the motion planning problems in an unknown dynamic environment [10].

Deep learning, as an important part of machine learning, has made remarkable achievements in image processing [11], face recognition [12], video detection [13] and natural language processing [14] field. Through multi-layer network structure and nonlinear transformation, deep learning combines low-level features to form an abstract and easily distinguishable high-level representation, so as to discover distributed feature representation of data and enable mobile robots to have strong environmental awareness.

RL is mainly used to solve sequential decision making problems. Inspired by the trial-and-error method in animal

The associate editor coordinating the review of this manuscript and approving it for publication was Valentina E. Balas.

learning, RL uses the reward value obtained from the interaction between agents and the environment as feedback signals to train agents, which does not need auxiliary means such as artificial markers. Reinforcement signals is provided by the environment, which are used to evaluate the effectiveness of the executive actions, rather than to instruct the agent how to perform the correct actions. Different from the tutor signal in supervised learning, the reinforcement signal in RL is easier to obtain for some decision control problems, and RL has become an effective method to solve the decision problem for mobile robot.

Further, DeepMind team combined DL with RL to propose a deep reinforcement learning(DRL) method. DRL was applied to Alphago and defeated human champions in 2016 [15]. Then, OpenAI team proposed an OpenAI-Five method based on multi-agent DRL (MADRL), which beat human players in the 5v5 mode of Dota2 game [16]. In 2019, DeepMind proposed AlphaStar based on MADRL, reaching the level of human masters in the Starcraft game [17]. In closed, static and deterministic environments, DRL can reach or even surpass human decision-making levels.

The motion planning method based on DRL can enable the robot to have autonomous learning ability and thus answer the question ''how to reach the target position'' [18]. DRL [19] enables robots to gain the perceptual and decision-making abilities. The input data are processed to yield the output in an end-to-end manner [20]. Sensor data (color/depth image, laser scanning, etc.) serve as the input state, and robot motion parameters (coordinates, displacement, direction, etc.) are referred to as outputs. The training depends on the interactions between the robot and the environment [21]. The optimization process does not depend on the environment model and any prior knowledge, which can guide the robot to navigate to the target position without collision in real time [22]. Compared with other algorithms, DRL has the following advantages: First, since it does not need sample labeling, it can solve special cases in the environment more effectively; Secondly, the end-to-end motion planning method can treat the system as a whole, which makes the system more robust. Third, reinforcement learning can better explore and learn some more excellent movements. These advantages play an important role in the optimal decision control of mobile robots. The motion planning method based on DRL has made substantial breakthroughs in many tasks involving high-dimensional raw input data and complex decision control [23]. Many famous research institutions and companies have devoted plentiful resources to study the motion-planning methods based on DRL, and a great deal has been achieved [24].

This paper summarizes the research status and of motion planning for mobile robots [25]. Aiming at the defect of conventional motion planning algorithm in complex environment, the mobile robot motion planning method based on DRL technology is emphatically summarized and compared. The main structure of this paper is as follows:

II. Related research
III. Motion planning based on DRL
IV. Multi-robot Cooperative planning
V. Difficulty for DRL motion planning
VI.. Future Research
VII. Conclusion

II. RELATED RESEARCH

A. CONVENTIONAL ALGORITHM

The optimization goal for mobile robot motion planning [26] is to take the shortest planning path, the minimum system cost and the minimum training time [27]. Conventional motion planning algorithms of mobile robots can be divided into two categories according to the prior knowledge of environmental: the global motion planning algorithm and the local motion planning algorithm, which is shown as FIGURE 1.

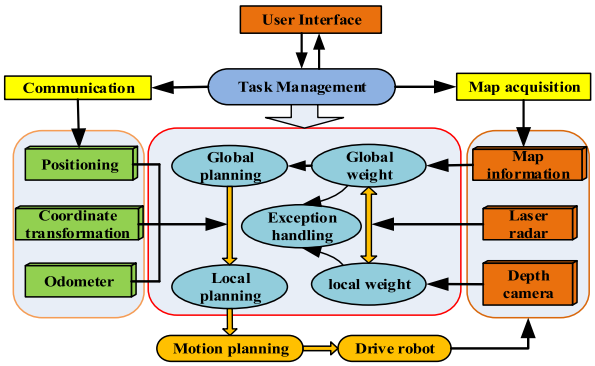


FIGURE 1. Schematic diagram of mobile robot motion planning.

Global motion planning algorithms [28] belongs to static and off-line motion planning, which is generally applied when the obstacle information in the robot operating environment is fully understood. However, the motion planning depends on the accuracy of environment acquisition. Although the planning results are global and superior, they have poor robustness to the error of environment model and noise interference. According to different search methods, global motion planning algorithm include graph search algorithm, random sampling algorithm and intelligent algorithm.

The commonly used graph search algorithms include Dijkstra [29], A* [30], D* algorithm [31] etc. Compared with Dijkstra algorithm, A* algorithm has higher efficiency by increasing heuristic estimation, reducing search volume. But the efficiency of A* algorithm still cannot be guaranteed when the environment is complex and large. Random sampling algorithms mainly include probability graph method (PRM) [32] and rapid Exploration random Tree method (RRT) [33], and these algorithms search for the optimal path in the whole space by randomly selecting scatter points. When the scatters cover the starting point and the end point, the shortest line between the scatters is the calculated path. There are still some problems, such as high cost, poor real-time performance, and the planned path may not be the optimal path.

Intelligent algorithm simulates biological evolution and insect foraging behaviors in nature, mainly including genetic algorithm(GA) [34], ant colony algorithm (ACO) [35], particle swarm algorithm [36], etc. GA has the characteristics of potential parallelism and is suitable for solving and optimizing complex problems. However, there are problems of slow operation speed and immature solutions. Particle swarm algorithm overcomes this disadvantage and has the advantage of fast convergence. The comparison of global motion planning for global motion planning is shown in Table 1.

TABLE 1. Comparison of global motion planning.

Algorithm	Advantages	Disadvantages
Dijkstra	Simple calculation and the optimal path	Too many traversal nodes and the calculation is complicated
A * Algorithm	Highly search efficiency in static environment	Not suitable for dynamic environments
D*Algorithm	Highly search efficiency in dynamic environment	Poor applicability for long distance motion planning
RRT Algorithm	Fast speed, strong search ability, no dependence on the map pretreatment	Low efficiency in the complex environment
Genetic algorithm	Easy to find the global optimal solution Suitable for complex problems	Large calculation scale，slow convergence speed，poor local search ability
Ant colony algorithm	Suitable for solving and optimizing complex problems and has potential parallelism	Slow calculation speed is, precocious solution
Particle swarm optimization	Fast convergence speed, simple and good robustness	Good convergence performance, easy to fall into the local optimal solution

Local motion planning is to enable the robot to autonomously obtain an optimal path without collision in an unknown environment based on the surrounding environment information [37]. Its advantage is that it is more adaptive to the unknown and real-time environment. Local motion planning belongs to dynamic planning and focuses on considering the local environment information of the robot, which enables the robot to have good obstacle avoidance ability [38]. Robots need to be robust enough to environmental errors and noise, and they can provide real-time feedback and correction for planning results through efficient information processing capability [39]. The algorithms used in local motion planning mainly include artificial potential field method [40], simulated annealing method [41], fuzzy logic method [42], neural network method(NN) [43], dynamic window Approach(DWA) [44], etc.

There is no essential difference between global motion planning and local motion planning. Many methods applicable to global motion planning can also be applied to local motion planning after being improved. The comparison of typical local planning for global motion planning is shown in Table 2.

In order to show the relationship between different algorithms better, the algorithm mechanism diagram is shown in

TABLE 2. Comparison of local motion planning.

Algorithm	Advantages	Disadvantages
Artificial potential field	Less computation, high efficiency	Exist local minimal traps
Simulated annealing	Less computation，flexible calculation	Result unstable , too fast convergence
Fuzzy logic	Strong coupling, strong robustness	Requires prior knowledge
Neural network	Strong robustness, memory ability, self-learning ability	Slow convergence rate;Low planning efficiency
Dynamic window	Better self-applicability	Not suitable for complex environments

FIGURE 2. Traditional motion planning algorithms include graph-based, sampling - based and data-based methods. Intelligent bionic algorithm includes genetic algorithm, ant colony algorithm, etc. Learn-based algorithm includes deep learning algorithm, reinforcement learning algorithm, imitation learning algorithm [45] and meta learning [46]. Among them, DRL motion planning algorithm will be reviewed as a key point in this paper.

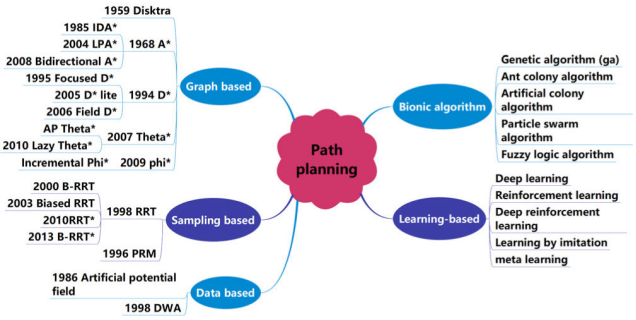


FIGURE 2. Classification diagram of motion planning algorithm.

B. SENSORS

Sensors equipped with mobile robots mainly include three categories: self sensors, Positioning sensors and surrounding sensors [47].

The self sensors uses proprioceptive sensors to measure the current state of the robot using preinstalled measurement units, such as odometer, IMU and CAN bus [48]. Positioning sensors use GPS or inertial measurement unit dead reckoning to determine the robot's global and local positions. Surrounding sensors use external sensors to sense road markings, weather conditions, obstacle status, which includes lidar sensors, vision sensors, infrared sensors and ultrasonic sensors, etc [49].

C. ENVIRONMENTAL PERCEPTION

Environment perception refers to the process of obtaining information from objects around it, road surface and its own state data. It relies on a variety of sensors to help the robot

understand the shapes and the positions of obstacles [50]. The perceptual algorithms includes two categories: Mediated Behavior Perception and Reflex Perception. Mediated Behavior Perception develops a detailed map of the robot's surroundings by analyzing the distance, pedestrians, trees, road markers, etc [51]. Behavior Reflex Perception uses artificial intelligence techniques to apply sensor data directly to the control system. They include vision, point cloud and multi-sensor fusion based algorithms.

D. LOCATION MODE

Most of the general positioning systems of mobile robots are based on GPS. However, they are not suitable for harsh environments for GPS cannot guarantee a signal in enclosed areas. So mobile robots are often equipped with positioning systems that do not rely on GPS [52], which includes visual-based positioning, light reflection positioning, ultrasonic positioning, aircraft calculation, Space beacon positioning, and SLAM(Simultaneous Localization and Mapping) [53].

III. MOTION PLANNING BASED ON DRL

In the face of the dynamic unknown task environment, the conventional motion planning algorithm still show many shortcomings for lacking of prior knowledge. DRL combines the decision ability of reinforcement learning (RL) and deep learning (DL) [10]. DL uses the perceptual ability of the neural network to extract the features from the input of the unknown environmental state, and realizes the fitting of environmental state to the action value function. On the other hand, RL completes the decision based on the output of deep neural network and its own exploration, and realizes the mapping from state to action. The DRL algorithm has been applied to solve the problem of motion planning in the high-dimensional environment of robots, and substantial breakthroughs have been made in such aspects as unmanned vehicle driving, mobile robot navigation, and bionic robot skill learning. Mirowski *et al.* [54] used Google street view as as the video input in an interactive navigation environment, and realizes the unmanned vehicle navigation (FIGURE 3(a)) across the city. Everett *et al.* [55] used onboard sensors to judge the state of pedestrians near the robot, so as to select reasonable movements and speeds and realize the obstacle avoidance in the crowded pedestrians (FIGURE3(b)). Through the deep reinforcement learning, Abhik *et al.*. [56] trained the quadruped robot "STOCH" to move in a simulation environment, including all the basic walking modes of the quadruped robot, and gained trot, walk, gallop, and standstill skills in a real environment (FIGURE 3(c)). Chen *et al.*. [57]. propose a Takagi-Sugeno (T-S) fuzzy control system based RL, and use UAVs to pursue UAVs (FIGURE 3(d)).

In additon, DRL algorithms are often combined with conventional motion planning algorithms. Conventional algorithm is preferred for global motion planning in static environment, and then deep reinforcement learning algorithm is used for dynamic obstacle avoidance. This hybrid DRL method can also be suitable for motion planning problems in dynamic unstructured unknown environments.

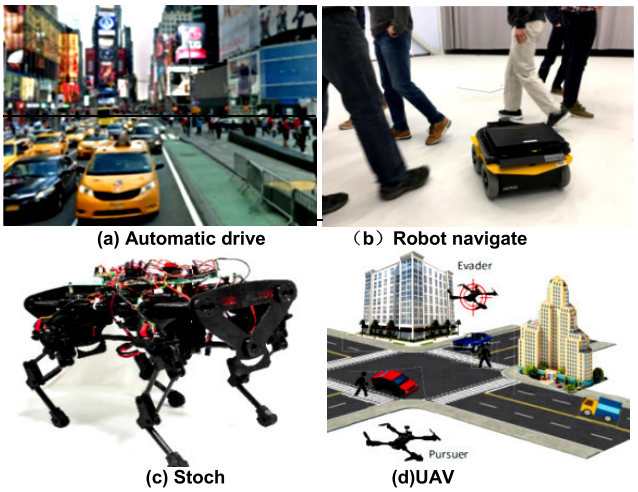


FIGURE 3. Application of DRL in the field of robotics.

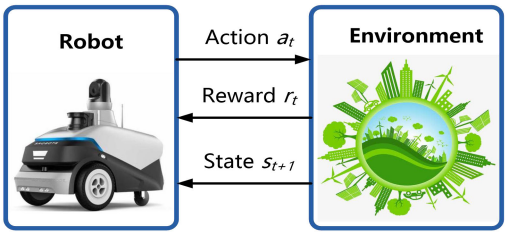


FIGURE 4. MDP for motion planning.

A. MDP

The model of RL is a standard Markov decision process (MDP). Different from supervised learning and unsupervised learning method, RL uses a feedback signal(reward) instead of a large number of labeled samples. Classical motion planning strategies (such as artificial potential field method, A* algorithm, etc.) are based on the environment with certain information, and usually have certain limitations in the application process. MDP provides a unified model description for the decision-making and solving process in the environment with incomplete information, which can solve the motion planning problem in the unstructured environment well.

As shown in FIGURE 4, MDP includes the environment, Agent, Action, Reward and State [58]. The Agent represents the mobile robot; The Environment refers to the map information of the task. The State is the state space of the mobile robot. And the action is a set of actions of a mobile robot. The goal of reinforcement learning is to find a strategy that maximizes cumulative rewards of the robot. Different from the immediate reward, the cumulative reward evaluates the long-term impact of a certain strategy on the performance of agents in the Environment. The commonly used calculation methods include discount cumulative reward and step cumulative reward. When the reward value is not easy to set,

the Inverse Reinforcement Learning can be used to learn the reward value function.

In contrast to conventional motion planning methods, the mobile robot is not told which action to choose, but instead tries to find the one that will get the most reward. Motion planning can be viewed as a process of trial evaluation. Firstly, the Agent will select an action for the environment and changes the state after the environment accepts the action. The action selection of the current state s is determined by the strategy $\pi(a|s)$: $\pi(a|s) = P(A_t = a|S_t = s)$, and it represents the conditional probability distribution of action selection, and the effect is evaluated by the value function $V_\pi(s)$ [59]:

$$V_\pi(s) = \mathbb{E}_\pi(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s) \quad (1)$$

where $\gamma \in [0, 1]$ is the discount factor, between [0,1].

Considering the influence of action A on the value function, the action value function $Q_\pi(s, a)$ is defined as [60]:

$$Q_\pi(s, a) = \mathbb{E}_\pi(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s, A_t = a) \quad (2)$$

According to the definition of the action-value function $Q_\pi(s, a)$ and the state value function $V_\pi(s)$, transformation relationship between them is:

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) q_\pi(s, a) \quad (3)$$

The purpose of robot motion planning is to find an optimal strategy for the mobile robot to consistently get more rewards during movement than any other strategy, and this optimal strategy can be represented by $\pi^*(a|s)$:

$$\pi^*(a|s) = \begin{cases} 1 & if\ a = \text{argmax}_{a \in A} Q_*(s, a) \\ 0 & else \end{cases} \quad (4)$$

where: $Q_*(s, a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \max_{a'} Q_*(s', a')$.

In general, it is difficult to find an optimal strategy for motion planning, but a better strategy can be determined by comparing the advantages of several different strategies.

B. MODEL-FREE RL

RL can be divided into model-based RL and model-free RL according to whether the environment model needs to be learned and understood [61]. The model-free RL motion planning process for the unknown environment model do not need to estimate the MDP model, and the value function or policy function can be evaluated directly by sampling to approximate the solution of reinforcement learning tasks.

The main methods include value-based (Value function approximation), policy-based (strategy gradient method), and Actor-Critic algorithm based on the combination of value and policy, which is shown as FIGURE 5.

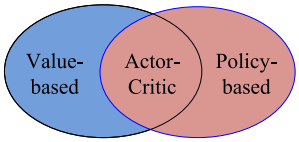


FIGURE 5. Classification of RL algorithm.

1) VALUE-BASED RL

Value-based reinforcement learning optimizes strategy by maximizing value function. The value function of each state of robot can be obtained by estimation. The commonly used estimation methods include Monte Carlo(MC) [62], TD (λ) [63], Q-learning [64], SARSA [65] and so on. Among them, the most widely used algorithm is Q-learning. Q-learning algorithm continuously optimizes the strategy through the information perceived by the robot in the environment, so as to achieve the optimal performance. It is one of the most effective reinforcement learning algorithms, which uses greedy strategy to select actions and can guarantee the convergence of the algorithm without knowing the model.

The state-action value function Q (s, a) is used as the evaluation function, and the ε-greedy strategy is used to select actions. The convergence of the algorithm can be guaranteed without knowing the model. It is one of the most effective reinforcement learning algorithms at present. Value function Q (s, a) is the estimated value rather than the actual value. The algorithm selects the maximum Q value function from the estimated values of different actions to update. The update method of evaluation function is as follows:

$$Q^*(s_t, a_t) = Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a)) \quad (5)$$

where, Q^* is the optimal value function, r_t is the current reward, α is the step factor, and γ is the discount factor.

The motion planning algorithm of mobile robot based on Q-Learning generally uses lidar data or depth camera data as the state input, stores the discrete state-action value function Q (s, a) in the Q table, and updates the Q value in the Q table through iteration. In the complex dynamic environment, improving the learning efficiency of Q-learning algorithm is an important problem to be solved in robot motion planning. Jaradat M [66] established a new state space to limit the number of state inputs, so as to greatly reduce the size of the Q table, thus improving the speed of the planning algorithm; Song [67] based on the known environmental information and Q table, a mapping is created between the initial values of the value table. The dynamic wave expansion neural network is used to specify the initial Q value properly. The prior knowledge in the environment is integrated into the learning system to speed up the learning efficiency and obtain better strategies

In recent years, scholars have been trying to improve Q-learning algorithm for robot motion planning. Chakraborty *et al.* [68] used RL method to drive the robot to

navigation in an unknown map and perform the forecasting task. At the same time, the fuzzy logic control method and Q-learning are combined to carry out self reinforcement learning of robot motion strategy. Based on the known topological map, Romero [69] and others use Q-learning method to define state, action and reward, and provide navigation map related to environment map to realize online navigation algorithm. However, this method only uses odometer measurement to navigate, and can only obtain less and inaccurate sensor information.

As shown in FIGURE 6(a), in order to enhance the fitting ability of Q-learning, Yi *et al.* [70] proposed a type-2 fuzzy neural network to realize the robot path planning in a complex environment. Reinforcement learning based on value function is also applied to the research of UAV flight control system. Behzad *et al.*. [71] studied the problem of aircraft path planning in a single environment by using a double-Q learning reinforcement learning method. As shown in FIGURE 6(b), the problem of overestimation of the Q value of Q-learning has been successfully solved.

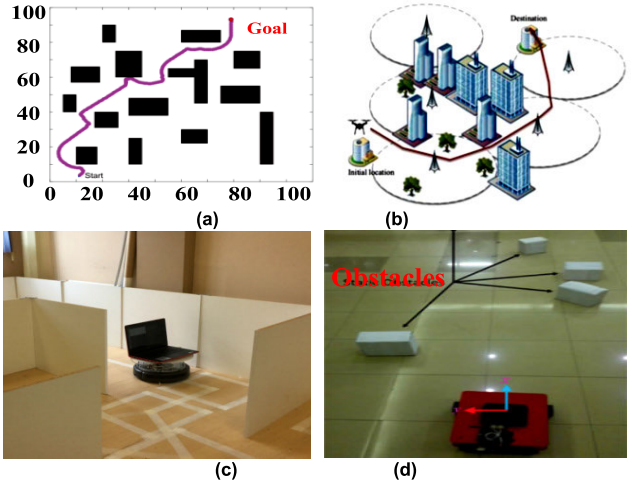


FIGURE 6. Robot motion planning based on Q-learning.

As shown in FIGURE 6(c), Song [72] combined Q-learning with Boltzmann strategy to improve the efficiency of multi robot system and reduce the number of explorations. Yuan *et al.* [73] used a Q-learning approach for obstacle avoidance by imitating human behavior, which is shown in FIGURE 6(d).

DQN: When dealing with the problems of motion control, navigation and obstacle avoidance, the robot needs to make correct and efficient decisions for the next action according to the information it feels, such as visual information, LIDAR point cloud and so on. Although the basic reinforcement learning method has strong decision-making ability, its perception of the environment is weak. Especially facing high-dimensional problems with large state space, there is still the danger of dimensional disaster. In recent years, deep Q network (DQN) algorithm and its related improved algorithms are widely used in robot motion planning tasks.

In order to deal with the task of robot motion planning based on visual perception, Mnih [74] first combined CNN with traditional reinforcement learning Q-learning algorithm and proposed a DQN model. FIGURE 7 has shown the network diagram of mobile robot based on vision navigation.

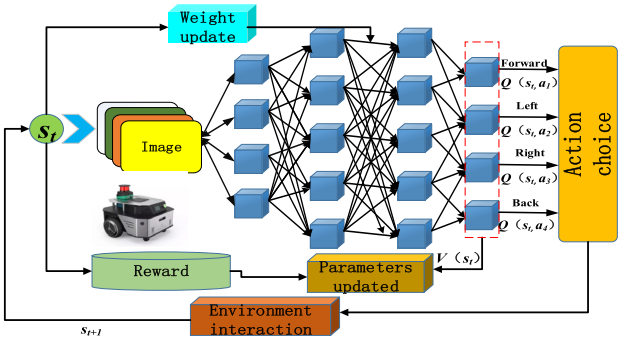


FIGURE 7. Architecture diagram of mobile robot motion planning.

DQN uses image or video as input information, constructs value network with memory ability through convolution neural network and long-term and short-term memory model (LSTM), and estimates value function. The number of neurons in the output layer of neural network is the number of actions that the agent can perform, such as forward, backward, left turn, right turn and so on.

There are still two problems in the process of robot motion planning based on DQN reinforcement learning algorithm, one is that the Q value is easy to be highly estimated, the other is that the sample correlation is too high. To solve these two problems, Van *et al.* [75] proposed a double DQN algorithm based on the double neural network. One network selects the optimal action, and the other network estimates the value function, which can better solve the problem. In order to break the correlation between the collected samples.

FIGURE 8 shows the network structure of the most widely used dqn algorithm, including two neural networks and a memory playback unit.

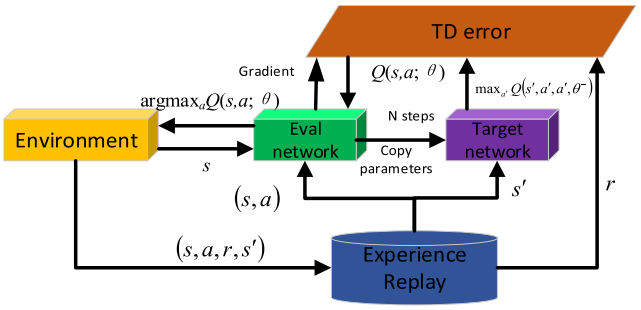


FIGURE 8. DQN algorithm network architecture.

Schaul *et al.* [76] proposed prioritized replay DQN algorithm by maximizing the absolute value of *TD* error on the Q-network, which makes the neural network update more efficient. With the increase of the training data scale, the

neural network structure needs to be further optimized to obtain more effective information. Dueling DQN [77] algorithm decomposes the Q value into State-value and Advantage function, which improves the learning efficiency of the network.

Furthermore, Noisy DQN [78] improves the exploration ability of the algorithm, Distributed DQN enhances the stability of the algorithm [79], and Asynchronous DQN makes the estimation of target value more accurate [80]. ''Deepmind'' finally integrated the advantages of the improved DQN algorithm, and proposed a comprehensive algorithm, named ''Rainbow'' [81]. Tai *et al.* [82] use the depth image as the input of the Q-network, and the action and speed of the mobile robot are used as the output to train the robot turnlebot to move (forward, right and left) in the indoor corridor while avoiding the wall. The navigation effect is verified by using a variety of different 3D simulation environments. The motion planning of the mobile robot in the case of a variety of static obstacles is realized,which has been shown in FIGURE 9 (a).

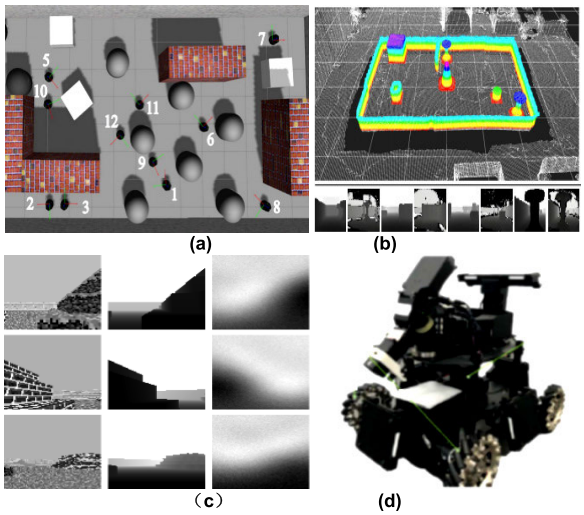


FIGURE 9. Motion planning using DQN algorithm.

DQN algorithm also uses other types of images as input. Based on the depth image information, Zhang *et al.* [83] migrated the learned navigation strategy to the unknown environment by means of inheriting features, and realized the migration of robot navigation strategy from simulation to reality by DQN, as shown in FIGURE 9(b). In the 3D simulation environment, Barron T *et al.* [84] used RGB image as the input of DQN and used deeper neural network to train the robot in complex tasks, which achieved better performance than the depth image and video. As shown in FIGURE 9(c), their theory was verified in the virtual natural environment.

As shown in FIGURE 9(d), Haora *et al.* [85] trained special robots to make automatic navigation exploration in unknown environments, and proposed a decision algorithm named AFCQN. The algorithm used deep neural network to learn exploration strategies from local maps and constructed a full convolutional Q network (FCQN). Compared with the basic DQN method, AFCQN can effectively reduce the probability of navigation failure. Similarly, Hussein *et al.* [86] proposed an improved DQN method combining demonstrations learning and experiential learning, which can shape a potential reward function by training a supervised convolutional neural network, and the algorithm is proved to be efficient in the navigation task of learning from raw pixels.

Kang *et al.* [87] applied DQN method to control four-rotor UAV. This method uses real-world data to learn system dynamics, and uses simulated data to learn a scalable sensing system, so that the robot can avoid collision with only one monocular camera, which is shown in FIGURE10.

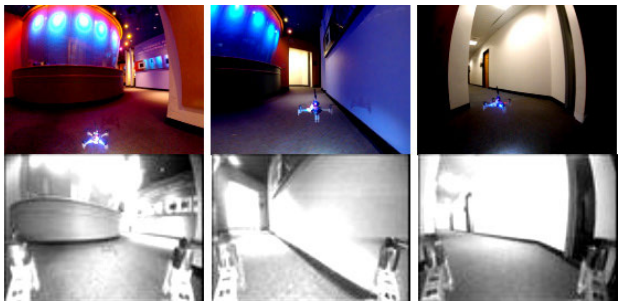


FIGURE 10. Motion planning for four rotor UAV.

The Value-based reinforcement learning motion planning method has the advantages of simplicity and efficiency, but it also has some disadvantages:

(a) *Deterministic Strategy:* Value-based RL is an indirect method to get the optimal strategy. According to value function, the action with the best value is selected by greedy. Facing the same state every time, the selected action is the same.
(b) *Policy Degradation:* The method based on value function is to use approximators to fit the real value function. There must be some errors, which may lead to poor strategy.
(c) *Discrete Control:* DQN is not suitable for scenes with continuous control or large action space, because the output action value of DQN is discrete and finite
(d) *Slow Convergence:* Value-based RL optimizes iteratively through value function and strategy, which has the disadvantage of low efficiency

2) POLICY GRADIENT ALGORITHM

The output motion parameters of value-based RL is not suitable for the continuous action space system of mobile robot. Policy-based reinforcement learning makes it possible to solve the above problems based on policy gradient [88], [89]. Rather than learning a value function, the policy gradient directly attempts to optimize the policy function π. The strategy is expressed as a function π_θ with parameter θ, and the optimization of the strategy is indirectly transformed into the optimization of parameter θ. The given policy evaluation

function is:

$$\eta(\theta) = \mathrm{E}\left[\sum_{t=0}^{H} r\left(s_t, a_t\right)|\pi_\theta\right] \tag{6}$$

According to the policy gradient method, by solving the derivative of the policy evaluation function with respect to parameter θ, the search direction of the strategy parameter θ is $\nabla_\theta \eta(\theta)$:

$$\nabla_\theta \eta(\theta) = \sum_{\tau} p(\tau;\theta)\nabla_\theta \log p(\tau;\theta) R(\tau) \tag{7}$$

where, $p\ (\tau;\theta)$ represents the probability distribution of trajectory τ obtained by executing strategy π_θ.

Then, the updated strategy parameter θ_{i+1} is:

$$\theta_{i+1} = \theta_i + \alpha \nabla_\theta \eta(\theta) \tag{8}$$

where, α is the updating step size.

In the aspect of robot motion planning [90], as shown in FIGURE 11 (a), Andrew [91] used the appropriate "prompt" to shape the optimal return function, and proposed a PEGASUS search strategy to automatically design a stable UAV controller, which got the flight test though the remote control helicopter.

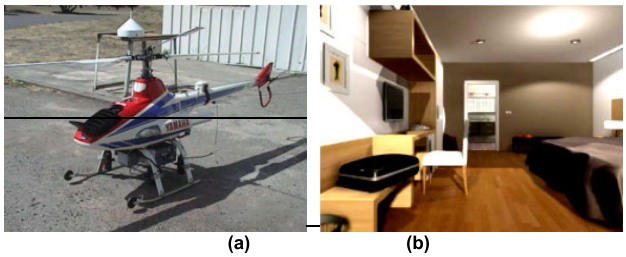


FIGURE 11. Policy-based RL Motion Planning.

In order to reduce the training time of mobile robot motion planning based on visual navigation, as shown in FIGURE 11 (b), Kulhanek *et al.* [92] added auxiliary tasks of visual data processing to pre train part of the network based on the extended strategy search algorithm, thus significantly reducing the training time and increasing the learning efficiency. Li *et al.* [93] based on the learning termination function interruption mechanism, added human experience into the algorithm framework, derived a hybrid hierarchical reinforcement learning structure, and realized the motion planning and target exploration tasks of mobile robot in indoor environment in simulation environment.

The policy-based RL uses the rewards as the estimation of state value. Although it is unbiased, the noise is relatively large. The policy gradient needs complete sequence samples to iterate, and the variability is always too high. In addition, the direction of updating the policy parameters is probably not the optimal direction of the policy gradient. Therefore, the policy-based RL needs to be further improved. So the most widely used improvement method is actor-critic RL algorithm.

3) ACTOR-CRITIC

The motion planning method based on Actor-Critic (AC) combines policy gradient and value function, which includes two parts: Actor and Critic [93]. Among them, Actor is the policy function equivalent to the policy gradient, which is responsible for generating actions and interacting with the environment. Critic uses the value function similar to DQN instead of Monte Carlo method to calculate the value of each step and get the value function. Then, the value function is used to evaluate the performance of the Actor and guide the action of the Actor in the next stage. In the Actor-Critic algorithm, two groups of approximations are needed

$$\pi_\theta(s, a) = P(a|s, \theta) \approx \pi(a|s) \tag{9}$$

The second group is the approximation of value function

$$\hat{v}(s, w) \approx v_\pi(s); \quad \hat{q}(s, a, w) \approx q_\pi(s, a) \tag{10}$$

The parameter update formula of the strategy is as follows:

$$\theta = \theta + \alpha \nabla_\theta log \pi_\theta(s_t, a_t) v_t \tag{11}$$

where, v_t is the optimal state value calculated by Critic through Q-network.

Actor uses v_t to update the parameter θ of policy function, and then select actions to get feedback and new state. Critical uses feedback and new state to update Q-network parameter w. Finally, Critic uses new network parameter w to help Actor calculate the optimal value of state v_t. The mobile robot motion planning with continuous action space can obtain better performance based on Actor-Critic.

Jaderberg M *et al.* [95] added auxiliary control and reward prediction into Actor-Critic algorithm, and Let the robot learn to navigate in the 3D Labyrinth environment. The method improved the data efficiency and parameter robustness successfully; Brunner G *et al.* [96] based on Actor-Critic and relocation unit, used 2DMaze global map and first-person perspective image as input to train the robot. In the DeepMind Lab virtual environment, the robot can quickly locate itself and estimate its orientation angle.

Compared with the traditional strategy iteration method, the Actor-Critic method can be updated in every step. But its disadvantage is that the network convergence depends on the evaluation of the Critic. However, the Critic itself is difficult to converge.

a: DDPG

One of the main reasons for Actor-Critic motion planning algorithm difficult to converge is that the data correlation is too strong. Based on the experience replay technology of DQN, Lillicrap *et al.* [97] from the DeepMind team fused deep neural networks with Actor-Critic, and proposed a model-free off-line Actor-Critic algorithm: Depth Deterministic Policy Gradient algorithm (DDPG). DDPG is also divided into Actor part and Critical part. Actor is responsible for updating the network, and Critic part is responsible for updating the action-value function.

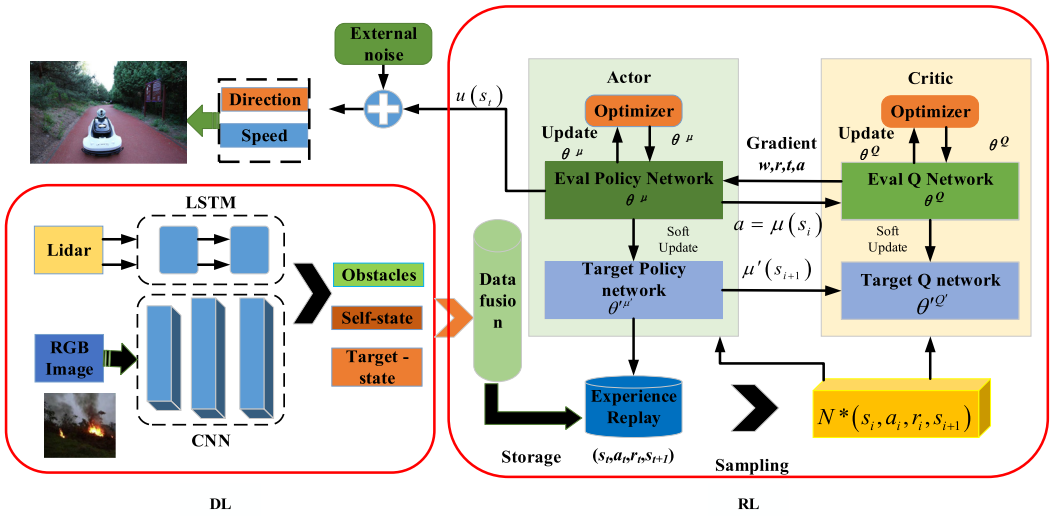


FIGURE 12. Motion Planning principle of DDPG.

DDPG uses deterministic strategy μ to select actions $a_t = \mu(s_t|\theta^\mu)$, θ^μ are the parameters of the policy network that generates deterministic actions. The strategy network μ is the actor, and the value network is the critic part for $Q(s, a)$ function. In the process of motion planning, the objective function of DDPG is as follows:

$$J(\theta^\mu) = E_{\theta^\mu}[r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots] \tag{12}$$

The goal of DDPG is to maximize the objective function and minimize the loss function of value-network Q. The optimal deterministic behavior policy μ^* could be found through the objective function, which is equivalent to maximizing the gradient of the objective function on the parameters θ^μ of the policy network,

Similar to the structure of using DQN, DDPG uses Q networks to fit Q functions. In other words, policy μ is used to select robot actions in state s, and the expected return could be obtained:

$$Q^\mu(s_t, a_t) = E[r(s_t, a_t) + \gamma Q^\mu(s_{t+1}, \mu(s_{t+1}))] \tag{13}$$

The gradient of Actor network is:

$$\nabla_\theta J_\beta(\mu_\theta) = E_{s\sim\rho^\beta}[\nabla_\theta \mu_\theta(s) Q^\mu(s, a)|_{a=\mu_\theta}] \tag{14}$$

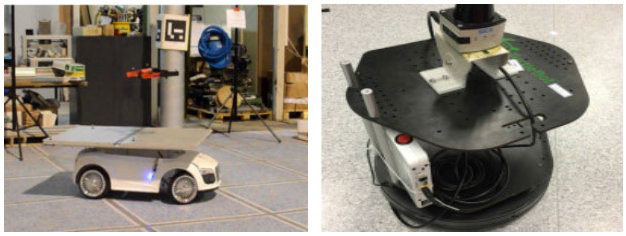
On the other hand, the value gradient of Critic network is

$$\frac{\partial L(\theta^Q)}{\partial \theta^Q} = E_{s,a,r,s'\sim D}[(TargetQ - Q(s, a|\theta^Q)\frac{\partial Q(s, a|\theta^Q)}{\partial \theta^Q}] \tag{15}$$

where, $TargetQ = r + \gamma Q\prime(s', \pi(s'|\theta^{\mu\prime})|\theta^{Q\prime})$

Using the gradient formulas of *Eq.*(14,15), the network can be updated with gradient descent algorithm. The motion planning method of mobile robot based on DDPG is shown in FIGURE 12.

Heess *et al.* [98] further extended the DDPG method to POMDP by using two recursive neural networks to approximate Actor and Critic, and developed an algorithm called



(a) (b)

FIGURE 13. Motion Planning Method Based on DDPG.

repeated deterministic policy gradient (RDPG). In terms of robot motion planning, Alejandro *et al.* [99] applied the new DDPG algorithm to design a general reinforcement learning framework, which solved the landing operation of UAV on a mobile platform. Better results were achieved in both simulation and actual flight experiments, which is shown in FIGURE 13 (a). Tai L *et al.* [100] proposed a learning-based Mapless motion planner: asynchronous DDPG (ADDPG) using the improved DDPG.

As shown in FIGURE13(b), with sparse 10-dimensional Lidar information and target position coordinates as inputs and continuous steering instructions as outputs, the mapless motion planner can be trained end-to-end without any prior human intervention. The trained policy can be applied directly to both virtual and real environments. Wang *et al.* [101] used the deterministic policy gradient to control the flight of the quadrotor helicopter. DDPG directly maps the system state to the control command, and introduces the integral compensator into the criticism structure, so that the tracking accuracy and robustness of the quadrotor helicopter are greatly improved.

DDPG deals with the problem of continuous control of mobile robots well, and solves the problem of Actor-Critic

convergence through dual network structure and priority experience replay mechanism, which is widely used in the field of robot motion planning. However, in the process of updating policy parameters, parameter replication needs to choose a appropriate step, which will directly affect the training effect of the algorithm, and even lead to the collapse of the motion planning system.

b: TRPO

In order to find an appropriate step in the policy gradient and guarantee the return function monotonically increased, Schulmam *et al.* [102] from Berkeley proposed a Trust region policy optimizatio(TRPO) method. TRPO decompose the return function of the new policy into the old policy and other items. If the other terms of the new strategy are greater than or equal to zero, the new policy could ensure that the return function remains monotonous. The objective function of TRPO method is:

$$\begin{aligned} &\underset{\theta}{\text{maximize}}\, \hat{\mathrm{E}}_t \left[\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{old}}\left(a_t \mid s_t\right)} \widehat{A}_t \right] \\ &\text{subject to}\, \hat{\mathrm{E}}_t \left[KL\left[\pi_{\theta_{old}}\left(\bullet \mid s_t\right), \pi_\theta\left(\bullet \mid s_t\right)\right]\right] \leq \delta \end{aligned} \quad (16)$$

where, $\hat{A}_t$ is the estimated value of the Advantage function; π_θ and $\pi_{\theta old}$ respectively represent the new and old policy on the same batch of training data. δ is a small value, which is used to limit the differences in KL divergence between the old and new policy. The motion planning principle of mobile robot based on TRPO is shown in FIGURE 14.

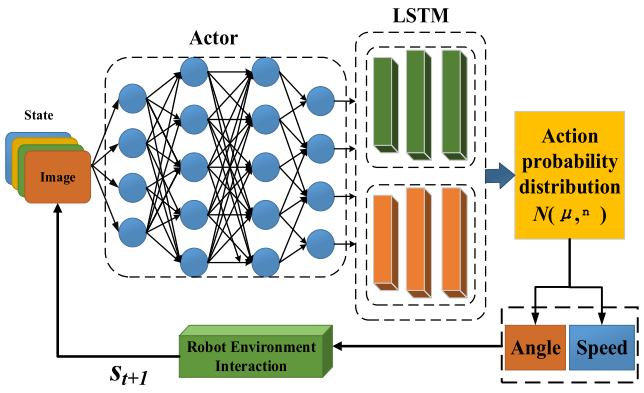


FIGURE 14. Motion Planning principle of TRPO.

Finally, the data samples are obtained by sampling to solve the optimization problem. TRPO algorithm has been successfully applied to robot operation skill learning in virtual and real scene. Li *et al.* [103] adopted TRPO for end-to-end optimization to solve the problem of social navigation in living environment. William *et al.* [104] used the TRPO algorithm to realize the autonomous flight of the quadrotor UAV with a stable posture in the high-fidelity simulation environment.

TRPO motion planning algorithm has solved the problem of choosing appropriate update step. However, Thre too many approximations used in the solution process, which not only complicate the process, but also bring large errors. At the same time, it is a difficult process to solve a constrained optimization problem. Because of these problems, TRPO has not been widely used in the field of robot motion planning.

c: PPO

In order to simplify the solution process of TRPO, OpenAI [105] proposed an algorithm that is simpler than TRPO theory and simpler in specific operation: Proximal Policy Optimization (PPO). PPO not only has a good performance in continuous action space of mobile robot, but also is easier to implement compared to TRPO. The objective function of PPO method is:

$$\begin{aligned} L_t^{CLIP+VF+S}(\theta) = \hat{\mathrm{E}}_t \Big[& L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) \\ & + c_2 S\left[\pi_\theta\right]\left(s_t\right) \Big] \end{aligned} \quad (17)$$

where, $L_t^{CLIP}(\theta)$ is the main objective function of strategy optimization;The loss between network output value and TD error target value is required to be minimum by $L_t^{VF}(\theta)$; $S\left[\pi_\theta\right]\left(s_t\right)$ is the information entropy of measuring the latest strategy gradient function.

Under the condition of satisfying the sampling to be the maximum likelihood probability, the policy is random and has the least factors assumes. The motion planning principle of mobile robot based on PPO is shown in FIGURE 15.

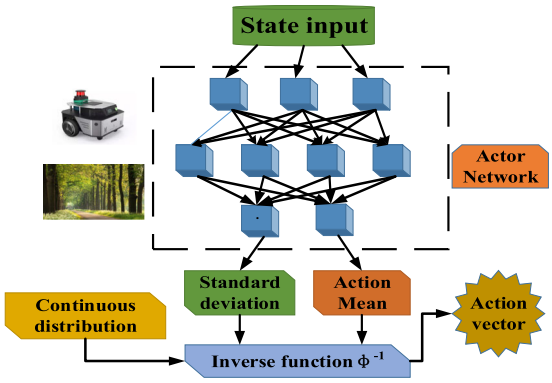


FIGURE 15. Motion Planning principle of PPO.

PPO policy can realize some very complex motion planning task, and the policy can even help mobile robot to make rotation, roll and other random behaviors when arriving at its destination. Chen *et al.* [106] trained the motor skills of wheel-leg robots based on an improved PPO algorithm. This algorithm directly maps high-dimensional images to motor commands and decomposes complex navigation tasks into manageable navigation behaviors.

In order to solve the problem of indoor maze navigation, Marchesini *et al.* [107] proposed a PPO algorithm based on double deep Q-network, which greatly reduces the training time though multi-batch priority to experience replay, which is shown in FIGURE16. In addition, a domain randomization technique is proposed in PPO motion planning algorithm, which enables the robot to navigate to the target position

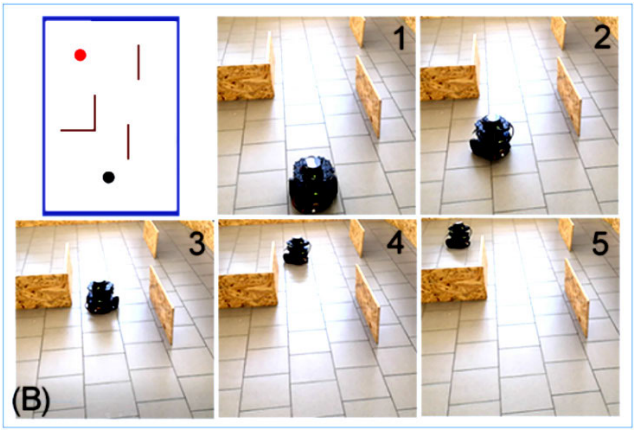


FIGURE 16. Navigate through the PPO algorithm.

while avoiding obstacles. It overcomes the problems of low data efficiency and sparse reward, and improves the efficiency of robot motion planning.

d: A3C

An important problem in robot motion planning is the fast convergence. To achieve this goal, the correlations between data must be broken. Both DQN and DDPG make use of the principle of empirical replay. However, experience replay is not the only method. Another method is the principle of asynchronous training. The asynchronous approach is that the data are not generated simultaneously. Based on this idea, DeepMind team Mnih *et al.* [108] developed Asynchronous Advantage Actor-critic(A3C) for reinforcement learning. The basic framework of A3C is also the Actor-Critic framework. Instead of using a single thread, it uses multiple threads to train simultaneously. Each thread trains a robot in a random exploration environment, multiple robots explore and compute policy gradients in parallel. These policy gradients are used together to update the weights of the shared model θ.

For the loss function part of Actor and Critic network, the entropy term of policy π is added, and the coefficient is c. That is, the gradient of the strategy parameter is updated as:

$$\theta = \theta + \alpha \nabla_\theta log \pi_\theta(s_t, a_t) A(S, t) + c \nabla_\theta H(\pi(S_t, \theta)) \quad (18)$$

Using asynchronous advantage reinforcement learning algorithm, Zhang *et al.* [109] trained four-wheel robot navigation in USAR environment on rugged terrain. As shown in FIGURE 17 (a). The training process provides the target position for the task, and the robot autonomously learns how to move forward, backward and turn. The whole navigation task consists of the robot performing a series of these basic actions to reach a predefined target position. This method has been successfully validated in different 3D simulated USAR environments.

Zhu *et al.* [110] input both the first-angle image and the target object image into the A3C model and approximate the target function based on the general value function. After 100 million frames of training, the strategy obtained by A3C algorithm can not only be applied to the simulation

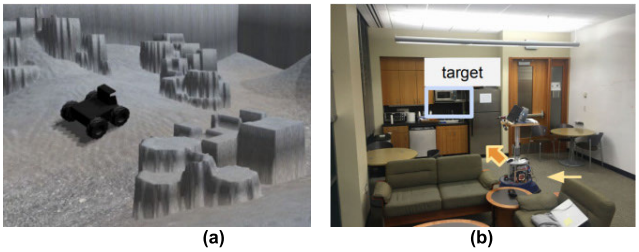


FIGURE 17. Navigating in complex environments based on A3C.

environment, but also be able to navigate the robot in the real environment, which is shown in FIGURE 17(b).

Niroui *et al.* [111] applied the deep reinforcement learning algorithm to the urban search and rescue robot. The method combines A3C network with frontier exploration to enable the robot to explore the unknown messy environment autonomously. This exploration method can effectively navigate to the appropriate boundary position in the unknown clutter environment of different sizes and layouts. It is also robust to different environment layouts and layouts.

Zeng *et al.* [112] proposed an Actor-Critic method (MK-A3C) based on asynchronous advantages of memory and knowledge for the navigation problem of nonholonomic robots with continuous control. MK-A3C constructs a memory neural network based on GRU to enhance the temporal reasoning ability of the robot. At the same time, the robot is given a certain memory ability. Through the estimation of the environment model, the local minimum trap is avoided and the non-convergence strategy problem caused by reward sparse is solved. Mk-A3C can effectively navigate in unknown dynamic environments, and the robot can successfully navigate in unknown and challenging environments through the output of continuous acceleration instructions.

e: SAC

In order to improve the exploratory and robust of mobile robots motion planning, Tuomas *et al.* [113] from the University of California, Berkeley proposed an off-policy algorithm Soft Actor-Critic (SAC) in 2018. SAC has achieved good performance in robot motion control and can be directly applied to robots in real environment. In contrast to DDPG, SAC uses a random strategy. At the same time, the problems of high sensitivity to over parameter and weak convergence are improved.

Compared with TRPO, A3C and PPO, SAC can reuse the past experience and improve the utilization efficiency of training samples. It is based on maximum entropy reinforcement learning framework, in which the entropy increase objective function is:

$$J(\pi) = \mathbb{E}_\pi \left[\sum_t r\left(\mathbf{s}_t, \mathbf{a}_t\right) - \alpha \log\left(\pi\left(\mathbf{a}_t | \mathbf{s}_t\right)\right) \right] \quad (19)$$

where, s and a represent states and actions respectively, and the E_π expectation contains dynamic parameters from the real system.

The goal of strategy optimization not only needs to maximize the expectation, but also needs to maximize the entropy of expectation, and the parameter α balances the influence of these two parts on the result.

When α is 0, the Eq(19) degenerates to the traditional objective function. The objective function can be regarded as the maximum expected return of entropy constraint, and the α parameter is automatically learned instead of the hyperparameter. The robot motion planning principle based on SAC algorithm is shown in FIGURE 18.

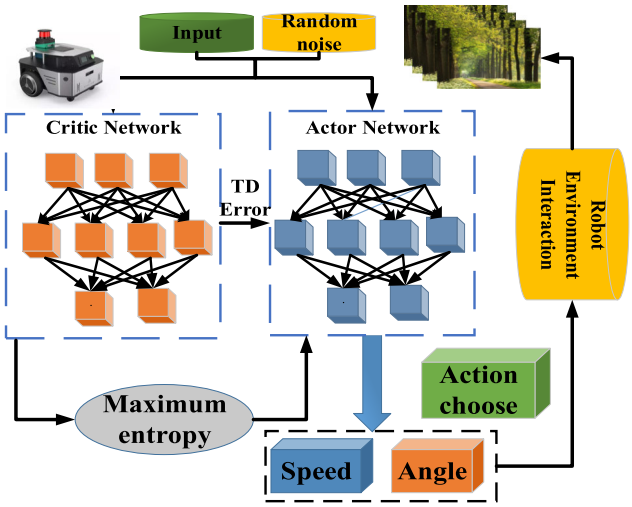


FIGURE 18. Robot motion planning principle based on SAC.

In order to verify the ability of SAC in the real world, Haarnoja *et al.* [114] made the robot learn from zero without relying on simulation or manual teaching, and realized the autonomous planning in the complex walking system of the foot robot. The algorithm directly mapped the sensor input to low-level actions. The learning process requires only a small task adjustment and a moderate number of experiments to learn better neural network strategies. As shown in FIGURE19, the robot learns a stable gait in the real world within two hours without relying on any model or simulation. The strategy obtained is robust enough to moderate changes in the environment.

FIGURE 19. Quadruped robot in real and simulation environment.

The gait control training data of robot in simulated environment were transferred to reality, and the stable walking of a medium-sized dog-sized quadruped robot ANYmal was realized.

AnyMal can run faster than ever before, and can even self-recover from falls in complex environments with a high level speed commands, which is shown in FIGURE20.

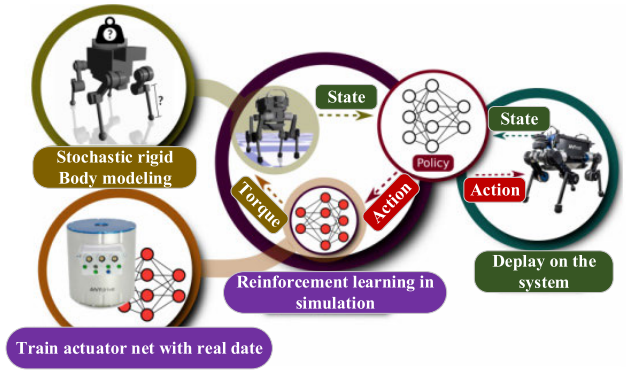


FIGURE 20. Train the quadruped robot ANYmal to walk.

DRL algorithms have been widely used in motion planning of mobile robots. Value-based DQN algorithm is the most widely used and simplest method in reinforcement learning field, and has a good performance in most discrete action Spaces. The algorithms based on Actor-Critic framework, such as DDPG, TRPO, PPO, A3C and SAC, have great advantages compared with DQN in continuous action space. The comparison of these deep reinforcement learning algorithms is shown in Table 3.

TABLE 3. Comparison of DRL Algorithms.

Algorithm	Advantages	Disadvantages
DQN	Offline update, simple and reliable	Not suitable for continuous motion space
DDPG	less sampling data is and faster convergence rate	Not suitable for random environment problems, poor exploration ability
TRPO	Low variance, more stable convergence	Complicated calculation, large error
PPO	Simple and efficient, multi-threaded training	Online update, low sample efficiency
A3C	Asynchronous parallel training, high efficiency, suitable for multi-robot system	Difficult model migration,more training data
SAC	Random strategy, high sample efficiency, better robustness	Large model size

C. MODEL BASED RL

Motion-planning method based on Model-Free Reinforcement Learning requires a lot of sample training, which limits its wide application in real environment. For example, we can train a control algorithm for a UAV in a simulated environment by constantly letting the UAV explore and learn. But in the real world, it is very difficult. We have to consider the risk of collision, the cost of repairs, the cost of exporting data to the GPU for training. All these problems indicate that the cost of data acquisition in the actual physical scene is far greater than that in the simulation environment.

In real physics scenarios, especially robotics projects, the more data-efficient approach is quite popular.

Model-based reinforcement learning is one of them. It obtains an environment model through supervised training, and then expands to learn this value function implicitly in order to expect maximum return. Compared with model-free RL (MFRL), model-based RL (MBRL) has higher sample efficiency, faster convergence speed and lower data requirement.

Xi *et al.* [116] combined MBRL with MFRL and proposed a Gaussian process (GP) model with two independent levels. The algorithm overcomes the problem of over-fitting in the complexity model, and reduceds the amount of training data. The biped robot can stabilize itself on the rotating platform, and can adapt to the platform with different angular velocity (FIGURE 21).

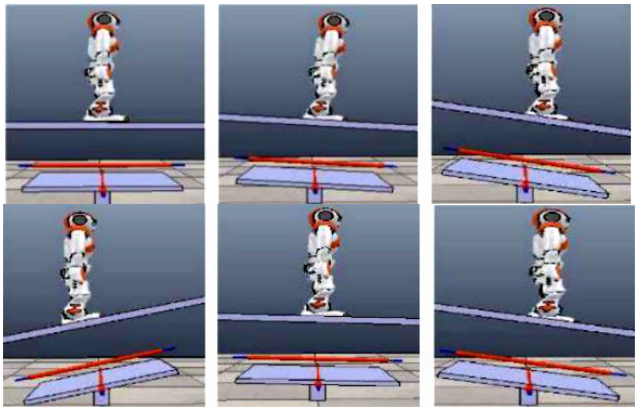


FIGURE 21. Biped Robot on a Rotating Platform.

Munk *et al.* [117] proposed an model learning Deep Deterministic Policy Gradient(ML-DDPG) for continuous control policies of robot based on model-state representation learning. ML-DDPG includes three deep neural networks: Model network, Critic network and Actor network. As shown in FIGURE 22, this method can learn a series of different challenging continuous control strategies, which improves the robustness and efficiency of the algorithm. Nursultan *et al.* [118] applied model-based algorithm to solve the flight control problem of UAV in the execution of practical tasks.

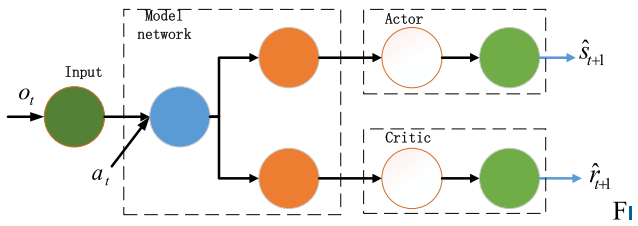


FIGURE 22. ML-DDPG Algorithm architecture.

Model-based RL algorithm can realize rapid and targeted exploration of uncertain effective strategies and fast learning with few samples.

D. IMITATION LEARNING

Imitation learning refers to an mobile robot's ability to mimic a desired behavior by learning from observations, which aims to learn reward functions from a expert-controlled agent. The mobile robot will be trained to perform a task from demonstrations by learning a mapping between observations and actions [119]. The mothin planning methods based on Imitation learning will use prior knowledge and recordedvexpert experiences to generate reward functions and movement trajectories, which can rapidly improve the performance of the initially learned policy and significantly reduce the quantity demand of samples.

Many researchers apply imitation learning to 3D motion planning for mobile robots. Based on DL and Imitation learning, as shown in FIGURE 23, Hussein *et al.* [120] proposed a deep imitative learning method to perform navigation tasks in 3D simulation environment. Combining demonstrations and experience, the algorithm employs deep CNN and raw visual input, which significantly improves the training efficiency and generalization ability.

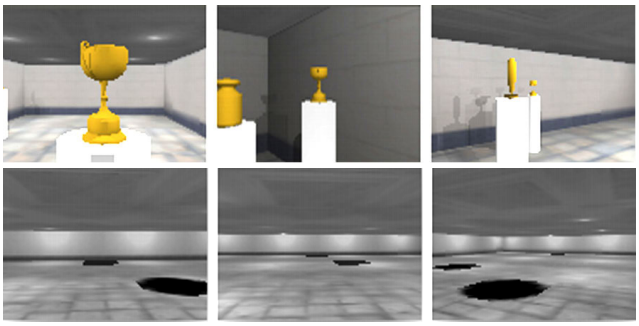


FIGURE 23. Imitation learning in a 3D simulation environment.

On the other hand, Wu *et al.* [121] present a motion planning method for UAV, which is based on generative imitation learning, and realizes mapless navigation in indoor scene. As shown in FIGURE 24(a), this method can automatically learn a navigation strategy, imitate data by experts, and apply to multi-robot system. As shown in FIGURE 24(b), Zhang *et al.* [122] proposed an imitation learning model (IADRL) to drive UAV to cooperate with each other to complete more complex tasks. The model provides a strategy for multi-UAV system, which makes multi-UAV complete the task with the minimum system cost.

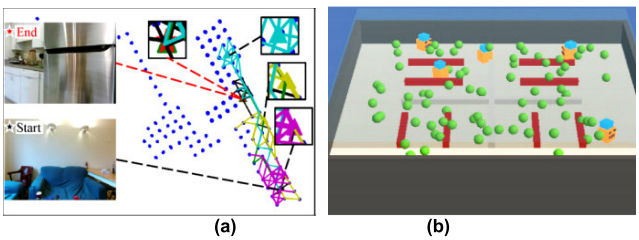


FIGURE 24. Navigation in a complex environment.

In order to solve the obstacle avoidance problem of UAV visual navigation, Park *et al.* [123] proposed an imitation learning strategy based on human experts sharing flight data. As shown in FIGURE 25, the strategy takes UAV visual image as input, uses neural network to extract features, and trains UAV obstacle avoidance strategy.

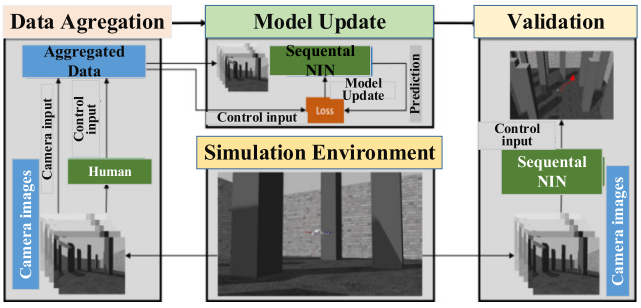


FIGURE 25. Human data-based imitation learning framework using sequential neural networks.

E. META-LEARNING

A certain amount of training data is required whether for reinforcement learning or imitation learning. Using a small amount of training data to learn new operational skills has become more important for the application of robots.

Meta learning is an improved reinforcement learning method based on a small amount of training data, which trains policy through a large number of related tasks. Each task contains a small amount of tagged data in the task set, which can automatically learn the common knowledge of the training task set. Many scholars have applied this method to the fast motion planning learning of mobile robots. Gaudet *et al.* [124] proposed an adaptive guidance model based on meta learning, which combines loop strategy and value function approximator. As shown in FIGURE 26, the model solves the problem of landing on Mars and asteroids, and realizes the combination of guidance and motion planning.

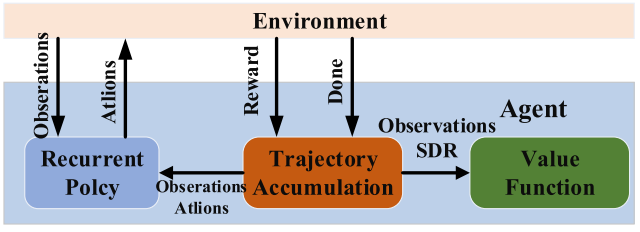


FIGURE 26. Adaptive guidance system.

In order to improve the adaptability and robustness of robot motion planning, Wortsman *et al.* [125] proposed an adaptive visual motion planning method (SAVN) based on meta reinforcement learning. As shown in FIGURE 27 (a), this method does not need any explicit supervision, but optimizes its own navigation strategy by learning a self supervised interaction

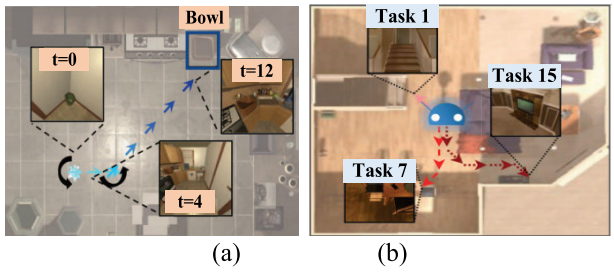


FIGURE 27. Indoor navigation based on meta-learning.

loss. In the low resource training environment with only a few labeled objects, Liu *et al.* [126] proposed a new unsupervised transferable meta skill based on visual navigation meta reinforcement learning. As shown in FIGURE 27 (b), as long as the reward information is provided to the robot, the robot can use these meta skills to quickly adapt to the environment with few resources.

IV. MULTI-ROBOT COOPERATIVE PLANNING

As the working environment becomes more and more complex, the working mode of a single robot shows disadvantages, such as limited perception ability, low task reliability and low execution efficiency. In the past 10 years, it has become more advantageous to carry out multi-robot cooperation to perform tasks together [127], and it has gradually become a research consensus and hotspot in the field of robotics. The working mode of multi-robot cooperation brings more challenges to the inter-individual motion planning in the group. How to carry out the cooperative motion planning effectively becomes the unique feature of this field, which is different from the single robot motion planning. The architecture of reinforcement learning motion planning system for multi-mobile robots can be mainly divided into two categories: centralized [128] and distributed [129].

Centralized reinforcement learning takes the common task of multiple robots as the training goal, and there is a centralized computing unit that can obtain the state and sensor information of all robots, and the centralized computing unit is responsible for the centralized strategy training and distribution. The advantage of centralized reinforcement learning is that real-time position, speed and target information of any robot can be obtained directly. Moreover, it can direct all individual robots to complete tasks such as task objective assignment, cooperation mechanism and path collaboration under optimal conditions. While ensuring the completion of the task, energy, time and other personal losses can be reduced as much as possible [130].

Distributed reinforcement learning is different from centralized reinforcement learning. Firstly, the whole task needs to be segmented in real time, and then the segmented sub task is sent to a single robot. After receiving the task, the robot, as a separate individual, participates in the training of the sub task independently, just like the movement plan of single robot reinforcement learning. Taking the maximum reward as

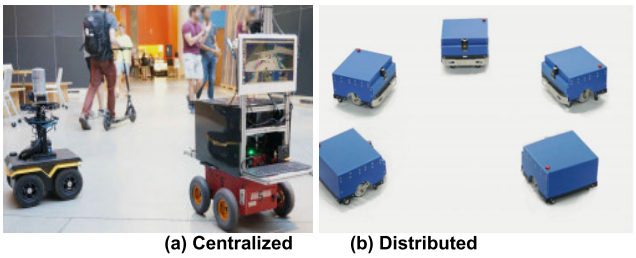


FIGURE 28. Motion planning for RL of multi-robots.

the goal and considering the overall total return, the overall task and planning can be realized through cooperation. The advantage of distributed reinforcement learning is that it can fully mobilize the resources of each robot and improve the utilization of hardware. Each robot is an independent individual with strong fault tolerance and redundancy.

Although the centralized reinforcement learning motion planning algorithm can get the global information, but the computational consumption will become huge when facing the large-scale mobile robot system, and the accuracy and real-time performance cannot be guaranteed. In order to solve this problem, Chen *et al.* [131] estimated the reach-time to the target by predicting value function, and degraded the centralized online calculation into a distributed offline calculation process. By learning a value function of ''implicitly encoding cooperative behavior'', a more real-time and efficient multi-robot motion planning system is obtained. As shown in FIGURE 28 (a), the cooperative obstacle avoidance in a crowded environment for multi-robot system is realized. To solve the efficient logistics transportation problem in dynamic production environment, Malus *et al.* [132] used reinforcement learning method to train multiple mobile robots for logistics scheduling. The modified algorithm learns to bid on orders based on their observations and their own positions and current plans to maximize the benefits of the system. Compared with the common transportation scheduling rules, the cooperative planning based on reinforcement learning is more efficient.

On the other hand, Long *et al.* [133] proposed a distributed RL model based on multi-sensor in order to solve the motion planning problem of multiple robots. As shown in FIGURE 28 (b), the algorithm maps the original sensor data containing speed information to the steering command, and uses the multi-scene and multi-stage framework for training to obtain the optimal strategy.

In order to ensure that each driverless vehicle learns to cooperate with each other, Shi *et al.* [134] uses interactive feature attention mechanism to capture interactive information between autopilot cars, and promotes information sharing strategy of multiple autonomous driving vehicles. As shown in FIGURE 29, the cooperation performance of the whole transportation system is improved by adjusting the human driven traffic flow.

The number of mobile robots directly affects the motion planning effect of multi robot system. In order to solve the

FIGURE 29. A highly collaborative autonomous driving.

problem that the DRL multi robot motion planning system deviates from the real goal with the increase of the number of robots, Michael [55] used an A3C framework to simulate the complex interaction and cooperation between robots. As shown in FIGURE 30, with LSTM recurrent neural network, it breaks the limitation that the conventional collision avoidance network needs to input fixed length in the convolution layer and pooling layer feed-forward network, and realizes the collision avoidance algorithm only through learning without preset behavior norms of other dynamic robots.

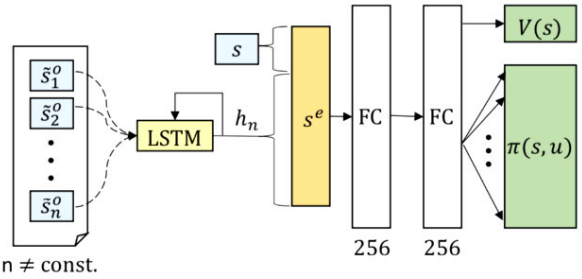


FIGURE 30. [122] RL network framework for multiple mobile robots.

In addition, Samaneh *et al.* [135] designed a hybrid reinforcement learning motion planning method based on the mechanical motion planning method. A new mixed reward function reduces the chance of collisions in crowded environments. As shown in FIGURE 31, the algorithm can switch automatically according to the complexity of different environments, and achieves good results in both 3D virtual environment and real environment.

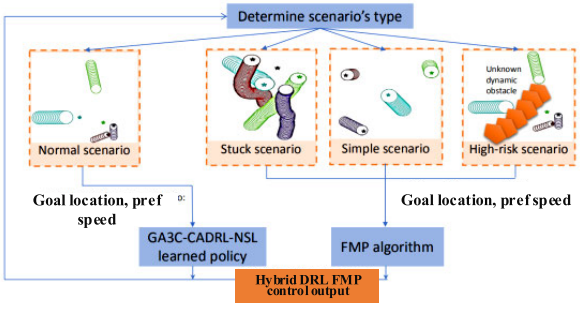


FIGURE 31. Hybrid control framework for DRL.

In conclusion, the centralized reinforcement learning motion planning system has strong coordination ability, and does not need to consider the cooperation between robots.

However, the system needs huge amount of computation and has poor scalability. The distributed reinforcement learning motion planning system is more flexible and convenient, and the processing ability is stronger. However, the cooperation between individual robots and collision avoidance are important problems to be studied in motion planning of multi-robot in complex dynamic environment.

V. DIFFICULTY FOR DRL MOTION PLANNING

A. DIFFICULTY FOR DESIGNING REWARD FUNCTION

In the motion planning task of DRL, mobile robots learn optimal policies by estimating cumulative rewards, and it can be performed well in a conventional task environment. However, this training type based on cumulative rewards has a huge search space and needs to make multi-step decisions in the whloe process. when facing a complex working environment, the robots will not get rewards unless they reach the predetermined goals, and less effective information will be obtained from the intermediate proces. Such ''sparse rewards'' will make it difficult for robots to learn effective strategies and fail in motion planning. How to design an efficient reward function to improve search efficiency is one of the problems in DRL movement planning field.

B. INSUFFICIENT DATA SAMPLE

DRL motion planning requires a large number of data samples to train the strategy, and the data samples are mainly acquired in the process of ''trial and error''. However, the collection process in the real environment is difficult due to mechanical reliability and time cost,, which will lead to the problem of insufficient training data sample. So the advantages of data-driven DRL method in feature extraction and parameter optimization will not be fully exerted. How to overcome the problem of insufficient data sample in the real environment is one of the problems to be solved urgently at present.

C. POOR UNDERSTANDING ABILITY

DRL can solve problems of modeling with conventional algorithms, but it still relies on a large amount of data training to fit the motion model. The intelligence level of DRL is low and still in the stage of computational intelligence for lacking of knowledge and understanding of environmental models. The time consumption and sample data requirements are enormous even in the face of simple models. How to enhance the cognitive and understanding ability of the DRL and learn from integrate the experience of human experts, is one of the important problems

D. INSUFFICIENT GENERALIZATION ABILITY

At present, DRL motion planning is mainly used in static, closed and deterministic environments, such as mazes, factories and indoor scenarios. However, many task environments of mobile robots are dynamic, opening and uncertain. The performance of the training model has certain deviation when the simulation is transferred to the real environment, and it is difficult to apply the strategy of the same task to different physical robots. Therefore, how to improve the generalization ability of DRL motion planning algorithm in practical application scenarios, so that mobile robots can obtain better adaptability, will be a huge challenge.

VI. FUTURE RESEARCH

A. REPRESENTATION OF COMPLEX ENVIRONMENT

Environment perception of mobile robots is one of the key abilities of motion planning in DRL. At present, the main way to obtain environmental state is to rely on optical sensors, such as ordinary cameras, depth cameras, three-dimensional Lidar, etc., but these sensors generally have common problems such as insufficient sampling efficiency, delay in real-time feedback, lack of environmental information, and insufficient accuracy [3].

In addition, the real environment of the robot has the changes of light, season and weather, which further restricts the effect of mobile robot on environment observation and recognition, and then poses a huge challenge to the robot motion planning. How to ensure DRL accurately perceive the characteristics of time-varying real environment in complex environment, and obtain the ability for semantic description and abstract reasoning representation, will be an important research direction in the field of motion planning for deep reinforcement learning.

B. COLLABORATIVE PLANNING FOR HETEROGENEOUS SYSTEMS

With the demand of three-dimensional task, the working environment becomes collaborative working mode. At present, the DRL motion planning is mostly used to solve the problem of navigation and collision avoidance in the low dimensional plane ground environment, while the research on the comprehensive planning in the three-dimensional space is less, such as the land marine amphibious robot and land air cooperative bionic robot system [136]. Compared with two-dimensional land space, high-dimensional motion planning will face more complex dynamic constraints and more uncertainties. This will put forward more stringent requirements for data collection and perception, deep network model construction, policy training and migration. In the future, large-scale deep neural networks can be combined to construct deeper Actor-Critic networks. At the same time, data preprocessing should be enhanced to further improve the perception and extraction ability for input features, and different data priorities should be used to cope with the surging data information in heterogeneous.

C. MULTI-SOURCE DOMAIN MIGRATION

The training process of motion planning is time-consuming. A model with good generality and mobility will greatly improve the efficiency of robot motion planning. However, in the practical application, especially in the face of

multi-robot motion planning system, model migration in the actual environment is faced with huge problems due to the difference of original data and target domain and the gap between robot systems. At present, most of the multi-source domain transfer learning algorithms for reinforcement learning are still in their infancy, and the effect of task transfer depends on the correlation between the source domain and the target domain. The source and target domains need to be in the same data space. In the future, we will study how to measure the difference of characteristics of multiple source domains and target domains adaptively according to environmental samples of different source domain tasks of mobile robots. How to extract common features or parameters and automatically align them to a specific state space. How to express features in a unified standard form, obtain invariant feature parameters between multi-source domains, and realize task, parameter and feature migration of multi-source domains in DRL. All these are important guarantees for mobile robots to perform tasks in a more diversified environment in the future.

D. DATA-MODEL HYBRID RL

The model-free DRL motion planning algorithm is based on data and does not depend on the system model, which requires less prior knowledge of the environment. However, it faces the problems of very low sample efficiency and difficult design of reward function, so that the algorithm is easy to fall into local optimality and can not find the optimal programming model. Model-based motion planning algorithm is based on robot ontology and environment model, and even shows more efficient performance than model-free DRL in static environment, especially when the map is known. However, in the face of complex tasks, the modeling is too complex, and the dynamic characteristics cannot be represented.

One of the solutions is to combine the model-based RL motion planning algorithm with the model-free RL motion planning algorithm. The hybrid algorithm can learn the environment model from the data, then optimize the policy based on the learned model, and reverse update and improve the model. It can make full use of environmental samples to approach the model, greatly improve the use efficiency of training sample data and shorten the learning process of the robot. In addition, when the mobile robot encounters a new environment, it can combine with multi-source domains, quickly adapt to the new model and greatly improve its own generalization ability based on the models. This Data-model hybrid reinforcement learning algorithm will be an important direction of motion planning in the future.

VII. CONCLUSION

The robot-motion-planning method based on DRL promotes the policy improvement of the robot through its interactions with the environment. Robots based on the method may obtain the robust ability of automatic learning and decision-making. This ability is critical for the unstructured environment, e.g., partial mapped and dynamically-changing environments. The method based on DRL lowers the programming complexity, and removes the dependency of the prior knowledge about environments. The method not only enables the robot to analyze the environment, but also boosts its ability of planning motion, avoiding obstacles in real time, searching object, and making decisions. Therefore, the methods based on DRL promote the development of intelligent robots.

However, the robot-motion-planning method based on DRL is rarely found in practice mainly due to its limited theoretical development and the low interpretability. Accordingly, the positioning accuracy, motion dexterity and stability required by the real-time application cannot be guaranteed. Additionally, laboratory environment differs drastically from the real world, e.g., ground disorganization, data delay and the mechanical structure unreliability, etc. These facts challenge the trial-and-error way of training the robots based on DRL. Thus, the solution remains to be developed in the further.

REFERENCES

[1] H.-W. Chae, J.-H. Choi, and J.-B. Song, "Robust and autonomous stereo visual-inertial navigation for non-holonomic mobile robots," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9613–9623, Sep. 2020.

[2] A. Muhammad, M. A. H. Ali, and I. H. Shanono, "Path planning methods for mobile robots: A systematic and bibliometric review," *ELEKTRIKA-J. Elect. Eng.*, vol. 19, no. 3, pp. 14–34, 2020.

[3] Y. D. V. Yasuda, L. E. G. Martins, and F. A. M. Cappabianco, "Autonomous visual navigation for mobile robots: A systematic literature review," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–34, 2020.

[4] A. K. Varma, J. Karjee, H. K. Rath, and A. Pal, "Dynamic path selection for cloud-based multi-hop multi-robot wireless networks," *IETE Tech. Rev.*, vol. 37, no. 1, pp. 98–107, 2020.

[5] E. A. Oyekanlu, A. C. Smith, W. P. Thomas, G. Mulroy, D. Hitesh, M. Ramsey, D. J. Kuhn, J. D. Mcghinnis, S. C. Buonavita, N. A. Looper, M. Ng, A. Ng'oma, W. Liu, P. G. Mcbride, M. G. Shultz, C. Cerasi, and D. Sun, "A review of recent advances in automated guided vehicle technologies: Integration challenges and research areas for 5G-based smart manufacturing applications," *IEEE Access*, vol. 8, pp. 202312–202353, 2020.

[6] K. I. Ghathwan, A. J. Mohammed, and Y. Yusof, "Optimal robot path planning using enhanced particle swarm optimization algorithm," *Iraqi J. Sci.*, vol. 61, no. 1, pp. 178–184, 2020.

[7] J. Lian, W. Yu, K. Xiao, and W. Liu, "Cubic spline interpolation-based robot path planning using a chaotic adaptive particle swarm optimization algorithm," *Math. Problems Eng.*, vol. 2020, pp. 1–20, Feb. 2020.

[8] S. Tian, Y. Li, Y. Kang, and J. Xia, "Multi-robot path planning in wireless sensor networks based on jump mechanism PSO and safety gap obstacle avoidance," *Future Gener. Comput. Syst.*, vol. 118, pp. 37–47, May 2021.

[9] S. K. Pattnaik, D. Mishra, and S. Panda, "A comparative study of meta-heuristics for local path planning of a mobile robot," *Eng. Optim.*, pp. 1–19, Jan. 2021, doi: 10.1080/0305215X.2020.1858074.

[10] K. Manchella, A. K. Umrawal, and V. Aggarwal, "FlexPool: A distributed model-free deep reinforcement learning algorithm for joint passengers and goods transportation," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2035–2047, Apr. 2021.

[11] T. Tao, K. Liu, L. Wang, and H. Wu, "Image recognition and analysis of intrauterine residues based on deep learning and semi-supervised learning," *IEEE Access*, vol. 8, pp. 162785–162799, 2020.

[12] J. Sun, Y. Lv, C. Tang, H. Sima, and X. Wu, "Face recognition based on local gradient number pattern and fuzzy convex-concave partition," *IEEE Access*, vol. 8, pp. 35777–35791, 2020.

[13] T. H. Trojahn and R. Goularte, "Temporal video scene segmentation using deep-learning," *Multimedia Tools Appl.*, pp. 1–27, Feb. 2021, doi: 10.1007/s11042-020-10450-2.

[14] V Osadchiy, T. Jiang, J. N. Mills, and S. V. Eleswarapu, "Low testosterone on social media: Application of natural language processing to understand patients' perceptions of hypogonadism and its treatment," *J. Med. Internet Research.*, vol. 22, no. 10, 2020, Art. no. e21383.

[15] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[16] C. Berner *et al.*, "Dota 2 with large scale deep reinforcement learning," 2019, *arXiv:1912.06680*. [Online]. Available: https://arxiv.org/abs/1912.06680

[17] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, and J. Oh, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, Nov. 2019.

[18] Q. Zhang, J. Lin, Q. Sha, B. He, and G. Li, "Deep interactive reinforcement learning for path following of autonomous underwater vehicle," *IEEE Access*, vol. 8, pp. 24258–24268, 2020.

[19] B. Rubí, B. Morcego, and R. Pérez, "Deep reinforcement learning for quadrotor path following with adaptive velocity," *Auto. Robots*, vol. 45, no. 1, pp. 119–134, Jan. 2021.

[20] P. Cai, Y. Zhang, and C. Pan, "Coordination graph-based deep reinforcement learning for cooperative spectrum sensing under correlated fading," *IEEE Wireless Commun. Lett.*, vol. 9, no. 10, pp. 1778–1781, Oct. 2020.

[21] X. Cao, H. Sun, and L. Guo, "Potential field hierarchical reinforcement learning approach for target search by multi-AUV in 3-D underwater environments," *Int. J. Control*, vol. 93, no. 7, pp. 1677–1683, Jul. 2020.

[22] S. Yeo, S. Lee, B. Choi, and S. Oh, "Integrate multi-agent simulation environment and multi-agent reinforcement learning (MARL) for real-world scenario," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*. Jeju Island, Republic of Korea: IEEE Computer Society, Oct. 2020, pp. 523–525.

[23] T. Fernando, S. Denman, S. Sridharan, and C. Fookes, "Deep inverse reinforcement learning for behavior prediction in autonomous driving: Accurate forecasts of vehicle motion," *IEEE Signal Process. Mag.*, vol. 38, no. 1, pp. 87–96, Jan. 2021.

[24] A. Jing, Z. Tang, J. Gao, and G. Pan, "An improved DDPG reinforcement learning control of underwater gliders for energy optimization," in *Proc. 3rd Int. Conf. Unmanned Syst. (ICUS)*. Harbin, China: Institute Electrical Electronics Engineers, Nov. 2020, pp. 621–626.

[25] L. Xie, Y. Miao, S. Wang, P. Blunsom, Z. Wang, C. Chen, A. Markham, and N. Trigoni, "Learning with stochastic guidance for robot navigation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 166–176, Jan. 2021.

[26] Y. Lu, W. Wang, and L. Xue, "A hybrid CNN-LSTM architecture for path planning of mobile robots in unknow environments," in *Proc. Chin. Control Decis. Conf. (CCDC)*. Hefei, China: Institute Electrical Electronics Engineers, Aug. 2020, pp. 4775–4779.

[27] F. Okumu and A. F. Kocamaz, "Comparing path planning algorithms for multiple mobile robots," in *Proc. Int. Conf. Artif. Intell. Data Process. (IDAP)*. Malatya, Turkey: Institute Electrical Electronics Engineers, 2018, pp. 1–4.

[28] M. Indri, M. Possieri, F. Sibona, P. D. C. Cheng, and V. D. Hoang, "Supervised global path planning for mobile robots with obstacle avoidance," in *Proc. 24th IEEE Int. Conf. Emerg. Technol. Factory Automat. (ETFA)* Zaragoza, Spain: Institute Electrical Electronics Engineers, Sep. 2019, pp. 601–608.

[29] Z. Cheng, H. Zhang, and Q. Zhao, "The method based on Dijkstra of multi-directional ship's path planning," in *Proc. Chin. Control Decis. Conf. (CCDC)*. Hefei, China: Institute Electrical Electronics Engineers, Aug. 2020, pp. 5142–5146.

[30] R. Pi, D. Youakim, P. Cieslak, and P. Ridao, "Multi-representation multi-heuristic A* motion planning for a dual-arm underwater vehicle manipulation system," *IFAC-PapersOnLine*, vol. 52, no. 21, pp. 205–210, 2019.

[31] M. M. Costa and M. F. Silva, "A survey on path planning algorithms for mobile robots," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*. Porto, Portugal: Institute Electrical Electronics Engineers, Apr. 2019, pp. 1–7.

[32] A. Francis, A. Faust, H.-T.-L. Chiang, J. Hsu, J. C. Kew, M. Fiser, and T.-W.-E. Lee, "Long-range indoor navigation with PRM-RL," *IEEE Trans. Robot.*, vol. 36, no. 4, pp. 1115–1134, Aug. 2020.

[33] Y. Li, W. Wei, Y. Gao, D. Wang, and Z. Fan, "PQ-RRT*: An improved path planning algorithm for mobile robots," *Expert Syst. Appl.*, vol. 152, Aug. 2020, Art. no. 113425.

[34] K. Hao, J. Zhao, K. Yu, C. Li, and C. Wang, "Path planning of mobile robots based on a multi-population migration genetic algorithm," *Sensors*, vol. 20, no. 20, p. 5873, Oct. 2020.

[35] H. Nobahari and S. Nasrollahi, "A terminal guidance algorithm based on ant colony optimization," *Comput. Electr. Eng.*, vol. 77, pp. 128–146, Jul. 2019.

[36] Y. Ling, N. Yang, H. Yu, and Y. Zhu, "Novel Bayesian network incremental learning method based on particle swarm optimization algorithm," in *Emerging Trends in Intelligent and Interactive Systems and Applications*. Shanghai, China: Springer, 2021.

[37] Z. Liu, Y. Li, and L. Zheng, "Local path planning for autonomous vehicles based on sparse representation of point cloud in unstructured environments," *Jixie Gongcheng Xuebao/J. Mech. Eng.*, vol. 56, no. 2, pp. 163–173, 2020.

[38] P. Lin, W. Y. Choi, and C. C. Chung, "Local path planning using artificial potential field for waypoint tracking with collision avoidance," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*. Rhodes, Greece: Institute Electrical Electronics Engineers, Sep. 2020, pp. 1–7.

[39] K. Wyrbkiewicz, T. Tarczewski, and L. Niewiara, "Local path planning for autonomous mobile robot based on APF-BUG algorithm with ground quality indicator," in *Advanced, Contemporary Control*. Lodz, Poland: Springer, 2020.

[40] W. Di, L. Caihong, G. Na, S. Yong, G. Tengteng, and L. Guoming, "Local path planning of mobile robot based on artificial potential field," in *Proc. 39th Chin. Control Conf. (CCC)*. Shenyang, China: IEEE Computer Society, Jul. 2020, pp. 3677–3682.

[41] P. Grabusts, J. Musatovs, and V. Golenkov, "The application of simulated annealing method for optimal route detection between objects," in *Proc. ICTE Transp. Logistics (ICTE)*, vol. 149, Jan. 2019, pp. 95–101.

[42] F. L. Silva, S. F. D. Silva, F. M. Santiciolli, J. J. Eckert, L. C. Silva, and F. G. Dedini, "Multi-objective optimization of the steering system and fuzzy logic control applied to a car-like robot," in *Multibody Mechatronic Systems*. Cham, Switzerland: Springer, 2021, p. 94 and 195–202.

[43] H. Li, Y. Mao, W. You, B. Ye, and X. Zhou, "A neural network approach to indoor mobile robot localization," in *Proc. 19th Int. Symp. Distrib. Comput. Appl. Bus. Eng. Sci. (DCABES)*. Xuzhou, Jiangsu, China: Institute Electrical Electronics Engineers, Oct. 2020, pp. 66–69.

[44] D. Feng, L. Deng, T. Sun, H. Liu, H. Zhang, and Y. Zhao, "Local path planning based on an improved dynamic window approach in ROS," in *The 10th International Conference on Computer Engineering and Networks*. Xi'an, China: Springer, 2021.

[45] S. Yu, X. Chen, L. Yang, D. Wu, M. Bennis, and J. Zhang, "Intelligent edge: Leveraging deep imitation learning for mobile edge computation offloading," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 92–99, Feb. 2020.

[46] A. Dantas and A. Pozo, "On the use of fitness landscape features in meta-learning based algorithm selection for the quadratic assignment problem," *Theor. Comput. Sci.*, vol. 805, pp. 62–75, Jan. 2020.

[47] C. Zong, Z. Ji, Y. Yu, and H. Shi, "Research on obstacle avoidance method for mobile robot based on multisensor information fusion," *Sensors Mater.*, vol. 32, no. 4, pp. 1159–1170, 2020.

[48] A. F. M. Paijens, L. Huang, and A. M. Al-Jumaily, "Implementation and calibration of an odometry system for mobile robots, based on optical computer mouse sensors," *Sens. Actuators A, Phys.*, vol. 301, Jan. 2020, Art. no. 111731.

[49] M. Varposhti, V. Hakami, and M. Dehghan, "Distributed coverage in mobile sensor networks without location information," *Auto. Robots*, vol. 44, nos. 3–4, pp. 627–645, Mar. 2020.

[50] J. N. Chen, S. L. Cai, J. Wang, X. F. Wu, and C. N. Yu, "Method for detecting obstacles in reservoir culverts based on circular structured light," *Lasers Eng.*, vol. 46, no. 4, pp. 203–224, 2020.

[51] D. Wang, W. Li, X. Liu, N. Li, and C. Zhang, "UAV environmental perception and autonomous obstacle avoidance: A deep learning and depth camera combined solution," *Comput. Electron. Agricult.*, vol. 175, Aug. 2020, Art. no. 105523.

[52] W. Guan, S. Chen, S. Wen, Z. Tan, H. Song, and W. Hou, "High-accuracy robot indoor localization scheme based on robot operating system using visible light positioning," *IEEE Photon. J.*, vol. 12, no. 2, pp. 1–16, Apr. 2020.

[53] P. H. Truong, S. You, and S. Ji, "Object detection-based semantic map building for a semantic visual SLAM system," in *Proc. 20th Int. Conf. Control, Autom. Syst. (ICCAS)*. Busan, Republic of Korea: IEEE Computer Society, Oct. 2020, pp. 1198–1201.

[54] P. Mirowski, M. K. Grimes, M. Malinowski, K. M. Hermann, K. Anderson, D. Teplyashin, K. Simonyan, K. Kavukcuoglu, A. Zisserman, and R. Hadsell, "Learning to navigate in cities without a map," in *Proc. 32nd Conf. Neural Inf. Process. Syst. Found. (NeurIPS)*, Montreal, QC, Canada, Dec. 2018, pp. 2491–2430.

[55] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. Madrid, Spain: Institute Electrical Electronics Engineers, Oct. 2018, pp. 3052–3059.

[56] A. Singla, S. Bhattacharya, D. Dholakiya, S. Bhatnagar, A. Ghosal, B. Amrutur, and S. Kolathaya, "Realizing learned quadruped locomotion behaviors through kinematic motion primitives," in *Proc. Int. Conf. Robot. Autom. (ICRA)*. Montreal, QC, Canada: Institute Electrical Electronics Engineers, May 2019, pp. 7434–7440.

[57] D. Chen, Y. Wei, L. Wang, C. S. Hong, L.-C. Wang, and Z. Han, "Deep reinforcement learning based strategy for quadrotor UAV pursuer and evader problem," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*. Dublin, Ireland: Institute Electrical Electronics Engineers, Jun. 2020, pp. 1–6.

[58] H. Ali, D. Gong, M. Wang, and X. Dai, "Path planning of mobile robot with improved ant colony algorithm and MDP to produce smooth trajectory in grid-based environment," *Frontiers Neurorobotics*, vol. 14, pp. 1–13, Jul. 2020, doi: 10.3389/fnbot.2020.00044.

[59] B. Cunha, A. M. Madureira, B. Fonseca, and D. Coelho, "Deep reinforcement learning as a job shop scheduling solver: A literature review," in *Hybrid Intelligent Systems*. Porto, Portugal: Springer-Verlag, 2020.

[60] H. Nguyen and H. La, "Review of deep reinforcement learning for robot manipulation," in *Proc. 3rd IEEE Int. Conf. Robotic Comput. (IRC)*. Naples, Italy: Institute Electrical Electronics Engineers, Feb. 2019, pp. 590–595.

[61] Q. Huang, "Model-based or model-free, a review of approaches in reinforcement learning," in *Proc. Int. Conf. Comput. Data Sci. (CDS)*. Stanford, CA, USA: Institute Electrical Electronics Engineers, Aug. 2020, pp. 219–221.

[62] M. C. Fu, "Simulation-based algorithms for Markov decision processes: Monte Carlo tree search from AlphaGo to AlphaZero," *Asia–Pacific J. Oper. Res.*, vol. 36, no. 6, 2019, Art. no. 1940009.

[63] H. van Hasselt, A. R. Mahmood, and R. S. Sutton, "Off-policy TD(λ) with a true online equivalence," in *Proc. 13th Conf. Uncertainty Artif. Intell.*, AUAI Press, Jul. 2014, pp. 330–339.

[64] P. K. Das, S. C. Mandhata, H. S. Behera, and S. N. Patro, "'An improved Q-learning algorithm for path-planning of a mobile robot," *Int. J. Comput. Appl.*, vol. 59, no. 9, pp. 40–46, 2012.

[65] L. Harwin and P. Supriya, "Comparison of SARSA algorithm and temporal difference learning algorithm for robotic path planning for static obstacles," in *Proc. 3rd Int. Conf. Inventive Syst. Control (ICISC)*. Coimbatore, India: Institute Electrical Electronics Engineers, Jan. 2019, pp. 472–476.

[66] M. A. Kareem Jaradat, M. Al-Rousan, and L. Quadan, "Reinforcement based mobile robot navigation in dynamic environment," *Robot. Comput.-Integr. Manuf.*, vol. 27, no. 1, pp. 135–149, Feb. 2011.

[67] Y. Song, Y.-B. Li, C.-H. Li, and G.-F. Zhang, "An efficient initialization approach of Q-learning for mobile robots," *Int. J. Control, Autom. Syst.*, vol. 10, no. 1, pp. 166–172, Feb. 2012.

[68] C. I. Goswami, P. K. Das, A. Konar, and R. Janarthanan, "Extended Q-learning algorithm for path-planning of a mobile robot," in *Simulated Evolution and Learning*. Kanpur, India: Springer Verlag, 2010.

[69] D. P. Romero-Marti, J. I. Nunez-Varela, C. Soubervielle-Montalvo, and A. Orozco-de-la-Paz, "Navigation and path planning using reinforcement learning for a Roomba robot," in *Proc. 18th Congreso Mexicano De Robotica*. Mazatlan, Mexico: Institute Electrical Electronics Engineers, Nov. 2016, pp. 1–5.

[70] Z. Yi, G. Li, S. Chen, W. Xie, and B. Xu, "A navigation method for mobile robots using interval type-2 fuzzy neural network fitting Q-learning in unknown environments," *J. Intell. Fuzzy Syst.*, vol. 37, no. 1, pp. 1113–1121, Jul. 2019.

[71] B. Khamidehi and E. S. A. Sousa, "A double Q-learning approach for navigation of aerial vehicles with connectivity constraint," in *Proc. IEEE Int. Conf. Commun. (ICC)*. Dublin, Ireland: Institute Electrical Electronics Engineers, Jun. 2020, pp. 1–6.

[72] Q. L. Song and J. B. Zhao, "Multi-robot path planning based on learningbclassifier system with policy gradient reinforcement learning and support vector machine (PGRL-SVM)," *Appl. Mech. Mater.*, vol. 347, pp. 3208–3211, Aug. 2013.

[73] R. Yuan, F. Zhang, Y. Wang, Y. Fu, and S. Wang, "A Q-learning approach based on human reasoning for navigation in a dynamic environment," *Robotica*, vol. 37, no. 3, pp. 445–468, Mar. 2019.

[74] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and S. Petersen, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[75] H. H. Van, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.* Phoenix, AZ, USA: AAAI press, 2016, pp. 1–7.

[76] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2015, *arXiv:1511.05952*. [Online]. Available: http://arxiv.org/abs/1511.05952

[77] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.* New York City, NY, USA: International Machine Learning Society (IMLS), 2016, pp. 1995–2003.

[78] M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin, and C. Blundell, "Noisy networks for exploration," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2018, pp. 1–21.

[79] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *Proc. Int. Conf. Mach. Learn.* Sydney, NSW, Australia: International Machine Learning Society (IMLS), 2017, pp. 449–458.

[80] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.* New York City, NY, USA: International Machine Learning Society (IMLS), 2016, pp. 1928–1937.

[81] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.* New Orleans, LA, USA: AAAI Press, 2018, pp. 1–8.

[82] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. Vancouver, BC, Canada: Institute Electrical Electronics Engineers, Sep. 2017, pp. 31–36.

[83] Y. Wang, J. Sun, H. He, and C. Sun, "Deterministic policy gradient with integral compensator for robust quadrotor control," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 10, pp. 3713–3725, Oct. 2020.

[84] T. Barron, M. Whitehead, and A. Yeung, "Deep reinforcement learning in a 3-D blockworld environment," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, IJCAI/AAAI Press, Jul. 2016, pp. 1–6.

[85] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2064–2076, Jun. 2020.

[86] A. Hussein, E. Elyan, M. M. Gaber, and C. Jayne, "Deep reward shaping from demonstrations," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 510–517, doi: 10.1109/IJCNN.2017.7965896.

[87] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine, "Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight," in *Proc. Int. Conf. Robot. Autom. (ICRA)*. Montreal, QC, Canada: Institute Electrical Electronics Engineers, May 2019, pp. 6008–6014.

[88] J. Xu, Z. Wei, L. Xia, Y. Lan, D. Yin, X. Cheng, and J.-R. Wen, "Reinforcement learning to rank with pairwise policy gradient," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.* Virtual, China: Association Computing Machinery, Jul. 2020, pp. 509–518.

[89] L. Jia, C. Chen, S. Xu, and J. Shen, "Fabric defect inspection based on lattice segmentation and template statistics," *Inf. Sci.*, vol. 512, pp. 964–984, Feb. 2020.

[90] H.-T.-L. Chiang, J. Hsu, M. Fiser, L. Tapia, and A. Faust, "RL-RRT: Kinodynamic motion planning via learning reachability estimators from RL policies," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4298–4305, Oct. 2019.

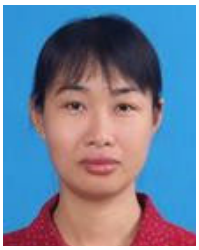
[91] A. Y. Ng, "Shaping and policy search in reinforcement learning," Univ. California, Berkeley, CA, USA, Tech. Rep. AAI3105322, 2003.

[92] J. Kulhanek, E. Derner, T. D. Bruin, and R. Babuska, "Vision-based navigation using deep reinforcement learning," in *Proc. Eur. Conf. Mobile Robots (ECMR)*, Sep. 2019, pp. 1–8.

[93] T. Li, J. Pan, D. Zhu, and M. Q.-H. Meng, "Learning to interrupt: A hierarchical deep reinforcement learning framework for efficient exploration," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*. Kuala Lumpur, Malaysia: Institute Electrical Electronics Engineers, Dec. 2018, pp. 648–653.
[94] W. Song, S. Liu, Y. Li, Y. Yang, and C. Xiang, "Smooth actor-critic algorithm for end-to-end autonomous driving," in *Proc. Amer. Control Conf. (ACC)*. Denver, CO, USA: Institute Electrical Electronics Engineers, Jul. 2020, pp. 3242–3248.
[95] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu, "Reinforcement learning with unsupervised auxiliary tasks," 2016, *arXiv:1611.05397*. [Online]. Available: http://arxiv.org/abs/1611.05397
[96] G. Brunner, O. Richter, Y. Wang, and R. Wattenhofer, "Teaching a machine to read maps with deep reinforcement learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, Nov. 2018, pp. 1–10.
[97] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *Comput. Sci.*, vol. 8, no. 6, p. A187, 2015.
[98] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," *Comput. Sci.*, pp. 1–11, Dec. 2015.
[99] A. Rodriguez-Ramos, C. Sampedro, and H. Bavle, "A deep reinforcement learning strategy for UAV autonomous landing on a moving platform," *J. Intell. Robot. Syst.*, vol. 93, nos. 1–2, pp. 351–366, 2019.
[100] L. Tai and M. Liu, "Towards cognitive exploration through deep reinforcement learning for mobile robots," 2016, *arXiv:1610.01733*. [Online]. Available: http://arxiv.org/abs/1610.01733
[101] Y. Wang, J. Sun, H. He, and C. Sun, "Deterministic policy gradient with integral compensator for robust quadrotor control," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 10, pp. 3713–3725, Oct. 2020.
[102] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.* Lile, France: International Machine Learning Society (IMLS), 2015, pp. 1889–1897.
[103] M. Li, R. Jiang, S. S. Ge, and T. H. Lee, "Role playing learning for socially concomitant mobile robot navigation," *CAAI Trans. Intell. Technol.*, vol. 3, no. 1, pp. 49–58, Mar. 2018.
[104] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for UAV attitude control," *ACM Trans. Cyber-Phys. Syst.*, vol. 3, no. 2, pp. 1–21, Mar. 2019.
[105] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithm," *Mach. Learn.*, pp. 1–12, Aug. 2017.
[106] X. Chen, A. Ghadirzadeh, J. Folkesson, M. Bjorkman, and P. Jensfelt, "Deep reinforcement learning to acquire navigation skills for wheel-legged robots in complex environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. Madrid, Spain: Institute Electrical Electronics Engineers, Oct. 2018, pp. 3110–3116.
[107] E. Marchesini and A. Farinelli, "Discrete deep reinforcement learning for mapless navigation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. Paris, France: Institute Electrical Electronics Engineers, May 2020, pp. 10688–10694.
[108] V. Mnih, A. P. Badia, M. Mirza, and A. Graves, "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, vol. 48, Jun. 2016, pp. 1928–1937. [Online]. Available: https://jmlr.org/
[109] K. Zhang, F. Niroui, M. Ficocelli, and G. Nejat, "Robot navigation of environments with unknown rough terrain using deep reinforcement learning," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot. (SSRR)*. Philadelphia, PA, USA: Institute Electrical Electronics Engineers, Aug. 2018, pp. 1–7.
[110] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. Singapore: Institute Electrical Electronics Engineers, May 2017, pp. 3357–3364.
[111] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 610–617, Apr. 2019.
[112] J. Zeng, R. Ju, L. Qin, Y. Hu, Q. Yin, and C. Hu, "Navigation in unknown dynamic environments based on deep reinforcement learning," *Sensors*, vol. 19, no. 18, p. 3837, Sep. 2019.
[113] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.* Stockholm, Sweden: International Machine Learning Society (IMLS), 2018, pp. 1861–1870.
[114] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," in *Proc. Robot., Sci. Syst.*, 2018, pp. 1–8.
[115] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Sci. Robot.*, vol. 4, no. 26, pp. 1–13, 2019.
[116] A. Xi, T. W. Mudiyanselage, D. Tao, and C. Chen, "Balance control of a biped robot on a rotating platform based on efficient reinforcement learning," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 4, pp. 938–951, Jul. 2019.
[117] J. Munk, J. Kober, and R. Babuska, "Learning state representation for deep actor-critic control," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*. Las Vegas, NV, USA: IEEE, Dec. 2016, pp. 4667–4673.
[118] N. Imanberdiyev, C. Fu, E. Kayacan, and I.-M. Chen, "Autonomous navigation of UAV by using real-time model-based reinforcement learning," in *Proc. 14th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Nov. 2016, pp. 1–6.
[119] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1–30, 2017.
[120] A. Hussein, E. Elyan, M. M. Gaber, and C. Jayne, "Deep imitation learning for 3D navigation tasks," *Neural Comput. Appl.*, vol. 29, no. 7, pp. 389–404, Apr. 2018.
[121] Q. Wu, X. Gong, K. Xu, D. Manocha, J. Dong, and J. Wang, "Towards target-driven visual navigation in indoor scenes via generative imitation learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 175–182, Jan. 2021.
[122] J. Zhang, Z. Yu, S. Mao, S. C. G. Periaswamy, J. Patton, and X. Xia, "IADRL: Imitation augmented deep reinforcement learning enabled UGV-UAV coalition for tasking in complex environments," *IEEE Access*, vol. 8, pp. 102335–102347, 2020.
[123] B. Park and H. Oh, "Vision-based obstacle avoidance for UAVs via imitation learning with sequential neural networks," *Int. J. Aeronaut. Space Sci.*, vol. 21, no. 3, pp. 768–779, Sep. 2020.
[124] B. Gaudet, R. Linares, and R. Furfaro, "Adaptive guidance and integrated navigation with reinforcement meta-learning," *Acta Astronautica*, vol. 169, pp. 180–190, Apr. 2020.
[125] M. Wortsman, K. Ehsani, M. Rastegari, A. Farhadi, and R. Mottaghi, "Learning to learn how to learn: Self-adaptive visual navigation using meta-learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA: IEEE Computer Society, Jun. 2019, pp. 6743–6752.
[126] J. Li, X. Wang, S. Tang, H. Shi, F. Wu, Y. Zhuang, and W. Y. Wang, "Unsupervised reinforcement learning of transferable meta-skills for embodied navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Long Beach, CA, USA: IEEE Computer Society, Jun. 2020, pp. 12120–12129.
[127] P. Yu and D. V. Dimarogonas, "A fully distributed motion coordination strategy for multi-robot systems with local information," in *Proc. Amer. Control Conf. (ACC)*. Denver, CO, USA: Institute Electrical Electronics Engineers, Jul. 2020, pp. 1423–1428.
[128] X. Zhou, P. Wu, H. Zhang, W. Guo, and Y. Liu, "Learn to navigate: Cooperative path planning for unmanned surface vehicles using deep reinforcement learning," *IEEE Access*, vol. 7, pp. 165262–165278, 2019.
[129] D. Wang, T. Fan, T. Han, and J. Pan, "A two-stage reinforcement learning approach for multi-UAV collision avoidance under imperfect sensing," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3098–3105, Apr. 2020.
[130] S. Tang, J. Thomas, and V. Kumar, "Hold or take optimal plan (HOOP): A quadratic programming approach to multi-robot trajectory generation," *Int. J. Robot. Res.*, vol. 37, no. 9, pp. 1062–1084, Aug. 2018.
[131] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. Singapore: Institute Electrical Electronics Engineers, May 2017, pp. 285–292.
[132] A. Malus, D. Kozjek, and R. Vrabi, "Real-time order dispatching for a fleet of autonomous mobile robots using multi-agent reinforcement learning," *CIRP Ann.*, vol. 69, no. 1, pp. 397–400, 2020.
[133] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. Brisbane, QLD, Australia: Institute Electrical Electronics Engineers, May 2018, pp. 6252–6259.
[134] T. Shi, J. Wang, Y. Wu, and L. Sun, "Towards efficient connected and automated driving system via multi-agent graph reinforcement learning," *Mach. Learn.*, pp. 1–13, Jul. 2020.

[135] S. H. Semnani, H. Liu, M. Everett, A. de Ruiter, and J. P. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3221–3226, Apr. 2020.
[136] S. Tarbouriech and W. Suleiman, "Bi-objective motion planning approach for safe motions: Application to a collaborative robot," *J. Intell. Robotic Syst.*, vol. 99, no. 1, pp. 45–63, Jul. 2020.

HUIHUI SUN was born in Xuzhou, Jiangsu, China, in 1989. He received the B.S. and M.S. degrees from the China University of Mining and Technology. He is currently a Lecturer with the North China Institute of Science and Technology. Since 2011, he has been the author of two books, six articles, and three inventions. His research interests include mechanism optimization and automatic motion control system of mobile robots.



WEIJIE ZHANG was born in Handan, Hebei, China, in 1972. She received the Ph.D. degree from the China University of Mining and Technology, Beijing, China. She is currently an Associate Professor with the North China Institute of Science and Technology. Since 1997, she has been the author of eight books, 30 articles, and six patents. Her research interests include coal mine electromechanical and smart robots.

RUNXIANG YU was born in Jining, Shandong, China, in 1986. He received the Ph.D. degree from the Beijing Institute of Technology. He is currently a Lecturer with the North China Institute of Science and Technology. His research interests include bionic robot development and motor control.

YUJIE ZHANG was born in Yingcheng, Hubei, China, in 1987. She received the B.S. and M.S. degrees from the Huazhong University of Science and Technology. She is currently a Lecturer with the North China Institute of Science and Technology. Since 2013, she has been the author of two books, five articles, and three patents. Her research interests include coal mine rescue robots and gait planning of quadruped robots.