Intelligent Robot Cooperative Control Based on Distributed DQN Algorithm

Wei LI[1], Chen CHENG[2]*

[1] Geely University of China, No. 123, Section 2, Chengjian Avenue, Eastern New District, Chengdu, Sichuan Province, China 641423,li.wei.77@163.com

[2] Anju Property Services Co., Ltd., Changjiang Road, Syzhou District, Yibin, Sichuan Province, China 644600,1293494273@qq.com (**Corresponding author)*

Abstract: Intelligent robot-based material handling vehicles (referred to as intelligent robot) for the transportation of small parts of industrial materials at new energy automobile assembly factories cannot work at the same time in the same interval of compute nodes, which affects the efficiency of material handling. This paper presents a cooperative motion control method for intelligent robots based on a regional distributed structure. Firstly, the motion direction for the intelligent robot is taken as the horizontal axis, a rectangular coordinate system is established, and the robot motion strategy is constructed in the framework of the rectangular coordinate system which was divided into four quadrants. Secondly, the observation space, action space and reward function are constructed based on the Markov decision process. Thirdly, a distributed DQN (Deep Q-Learning) reinforcement learning algorithm is employed to enable decision-making for intelligent robot so that they avoid obstacles and choose the optimal paths. In the context of a MATLAB-based simulation test, the proposed model employs 1 to 20 intelligent robots at the same time, and the obtained material handling success rate is higher than 80%. When 20 to 160 intelligent robots are employed at the same time, the material handling success rate is maintained at 78%. In comparison with the other three employed algorithms, namely the TD3 algorithm, the DQN algorithm and the DDPG algorithm, the convergence speed of the proposed model is the highest and its value can be maintained at about 0.6. By creating an electronic map model of the factory environment and testing the cooperative control of four intelligent robots, the experimental process can be completely carried out. In comparison with the existing tracking model, a reduction of 86 seconds for the accumulated idle time for the intelligent robot indicates that the proposed model can be better applied in real-life contexts.

Keywords: Intelligent robot, Cooperative control, Observation space, Action space, Reward function, Convergence speed.

1. Introduction

The rapid advancement of intelligent manufacturing and automation systems has led to an increased focus on research into cooperative control methods for intelligent robots. In complex industrial environments, a single robot often struggles to efficiently and accurately complete tasks, making multi-robot collaborative control essential for improving system performance and flexibility. However, traditional centralised control methods face challenges such as high computational complexity and heavy communication burdens in large-scale and dynamic scenarios. Efficient and cooperative control in multi-agent systems has therefore become a key issue. As a result, the collaborative control method based on intelligent robots based on the distributed DQN (Deep Q-Network) algorithm has emerged as a research hotspot due to its strong learning ability and adaptability. Nevertheless, there are still numerous challenges and shortcomings in the application of the distributed DQN algorithm, collaborative control of multi-agent systems, deep learning application in path planning, and distributed control strategy of energy systems.

This paper introduces the distributed DQN algorithm to address the decision optimisation problem of multiple robots in trackless cooperative transportation, thereby enhancing transportation efficiency and stability. This method takes into account the interaction between robots and environmental obstacles, which makes it suitable for complex environments with scalability and flexibility. It offers new insights and technical support for intelligent robot collaborative control, and is anticipated to play a significant role in the fields of large object handling and logistics.

The remainder of this paper is as follows. Section 2 presents the research status of the multi-agent cooperative control technology. Section 3 sets forth the theoretical basis and research method for the proposed model. Section 4 provides the experimental results of this research and Section 5 concludes this paper and outlines possible future research directions.

2. Literature Review

At present, multi-agent cooperative control technology has been widely adopted. It covers robot cooperative control and management, intelligent manufacturing and automation

https://doi.org/10.24846/v34i1y202505

systems, intelligent distributed computing and network architecture.

The study of Schultz et al. (2023) emphasises the importance of autonomy in multi-robot systems, especially in remote operation, where autonomy can significantly improve the response speed and efficiency of the system. However, the research mainly focuses on the construction of the theoretical framework and lacks the verification of practical applications. Liu et al. (2024) demonstrated how to improve the work efficiency and path smoothness for robots through collaboration in complex environments through the improved path planning algorithm. However, it would face computational complexity and real-time problems in practical applications. The research of Chaudhry & Gautam (2020) focuses on the design of the communication architecture, emphasising the importance of stable and efficient communication for collaborative work in multi-robot systems. However, its universality and extensibility have not been fully verified. Lin (2023) proposed an intelligent anti-jamming scheme for UAV swarm based on DQN. In this study, the optimal channel selection strategy is learned by the DQN model to improve the overall anti-interference performance.

This method is especially suitable for UAV communication in complex environments and can effectively deal with dynamic changing interference sources. However, the study mainly focuses on the communication level and less on the collaborative control of actual physical movements and task execution. Peng et al. (2022) discussed the push-grab collaboration method based on DQN under a dual perspective. By using a RGB-D camera to obtain RGB images of objects and point cloud information, the method solves the problem of missing information and features a high generalisation ability and a good performance when new objects appear. This research has made significant contributions to the flexibility and adaptability of robot operation, but its application scenarios are mainly limited to static environments, and its adaptability to dynamic and changeable environments needs to be further verified.

Wu & Suh (2024) studied the decentralised control application of DQN in multi-robot systems. The study not only demonstrates the potential of DQN in multi-robot control, but it also presents a new perspective for enhancing the overall performance and robustness of the system by integrating local information. This study is of great significance both in theory and practice, but its results are mainly based on the simulation environment, and its effect and stability in practical applications still need further experimental verification.

In (Al-Selwi et al., 2024) an in-depth four-step method based on the PRISMA method is used to improve the performance of RNN-LSTM by weight initialisation and optimisation. Cui et al. (2021) studied the protection technology for distributed power generation networks and proposed to transform this problem into a multi-agent reinforcement learning problem. The author describes in detail the construction methods for the reinforcement learning environment and the deep Q learning network. This study shows how to optimise the protection strategy of distributed networks by the DRL method, but its application scenario is relatively specific, mainly in the field of power systems.

The study of Jankovič et al. (2024) investigated the advantages of different machine learning (ML) methods in production systems, and for a given use case, a framework that enhances production systems with ML facilitates the transition to smarter processes and enables the integration of fast, accurate predictions into decision making and adaptive control, through linear regression (LR), decision tree (DT), support vector machine (SVM), Gaussian process regression (GPR) and neural network (NN) model prediction accuracy performance evaluation indicators RMSE, MAE, MSE and R^2. The rationale illustrates the tradeoffs between model complexity, accuracy, and computational training and prediction rates. Zhu et al. (2022) proposed a distributed cluster regulation strategy based on multi-level deep Q learning for the coordinated control of multi-park integrated energy systems. This strategy takes into account the flexible alternative interval of various energy sources, which can coordinate the adjustment ability of the integrated energy system of each park and improve the stability of the overall system. This study demonstrates the potential of DRL in the management of complex energy systems, but the complexity of its model may limit its scalability in practical applications.

In (Kuo et al., 2024), a hybrid multi-objective meta-heuristic and probabilistic intuitionistic fuzzy C-means (PIFCM) algorithm was proposed for cluster analysis. Three clustering algorithms were proposed, namely PIFCM based on multi-

object GA (MOGA-PIFCM), PIFCM based on multi-object PSO (MOPSO-PIFCM) and PIFCM based on multi-object GE (MOGE-PIFCM). The performance results were compared with those of other clustering algorithms (the intuitive fuzzy C-means (IFCM) algorithm, the probabilistic intuitionistic fuzzy C-means (PIFCM) algorithm, the single-objective GA-PIFCM algorithm, the single- objective PSO-PIFCM algorithm, and the single-objective GE-PIFCM algorithm). The results showed that MOGE-PIFCM achieved a better solution than other clustering algorithms for all performance verification metrics.

Zhang et al. (2020) focused on robot path planning and proposed a method based on deep reinforcement learning. Simulation experiments show that the method can achieve intelligent perception and decision by relying only on partial map information. The advantages of this approach lie in its adaptability and rapid response to environmental changes, which is particularly important for robotic operations in complex and dynamic environments. However, the study does not discuss in detail the performance of the algorithm in practical applications, especially on resource-constrained hardware platforms.

In the study of Pan, W (2024), multi-layer perceptron artificial neural networks (MLP ANN) are used to predict thinning results from input parameters, and genetic algorithms (GA) are used to optimise these parameters. The results demonstrate the effectiveness of the proposed method in minimising thinning. Savran et al. (2024) and Negoiță & Borangiu (2023) propose a genetic algorithm (GA) and a Whale Optimisation algorithm (WOA), respectively to optimise motor power, battery capacity, and propulsion ratio for two different driving modes. GA optimisation is found to create vehicle architectures suitable for long distance and high performance driving, and can provide shorter optimisation times. Long et al. (2023) proposes a self-optimising control system for an unmanned line marking machine (ULMM) based on visual navigation.

A new algorithm based on HAAR-like features is used to detect the guide line (GL) of ULMM in order to reduce the influence of complex road surfaces and light. To solve the problem of the inaccurate ULMM model and local navigation information, an online self-optimising control algorithm is proposed, which can detect the GL accurately. In contrast, Ibrahim et al. (2023) focused on the load balancing problem in SDN and proposed a sequential deep Q learning network (tDQN). The model optimises the switch and controller mapping through a reward-punishment mechanism, aiming to reduce network latency.

The experimental results show that tDQN significantly improves decision quality during iterative learning. Although the method performs well in simulated environments, its effectiveness in dealing with complexity and uncertainty in real networks needs to be further validated. Huang et al. (2022) propose an automatic modulation and resource allocation (AMC and RA) algorithm based on dual deep Q networks (DDQN), which considers users as agents and improves throughput at a single link and overall system level by learning from past experience and implementing the distributed policies. Li et al. (2020) proposed a security management algorithm applied to power networks.

By constructing a low-delay real-time communication network and learning individual characteristic information, the reliability of the communication network is trained. Experimental results showed that the proposed algorithm greatly enhanced the reliability of the communication network. Guo et al. (2024) proposed a power transformer intelligent cooperative control system based on a deep Q network (DQN).

By establishing the state space and action of the transformer control system, deep reinforcement learning was used to optimise the control strategy. A comprehensive evaluation of these studies shows that they provide important technological advances in the field of intelligent manufacturing and automation systems. However, there are some drawbacks to these studies. First, their adaptability and expansibility in complex environments are not discussed in detail. Second, their applicability in the actual industrial environment is not verified. Third, although the collaborative collision avoidance method performs well in simulations, its real-time performance and reliability in practical applications still need to be further verified. Agarwal & Sharma (2023) trained a reinforcement learning agent to tell the user which route network had less congestion, and to constantly interact with the surrounding environment to help the user find the best route. In the experiment carried out, a DQN (Deep Q-Network) based on the Q value was used, and the DQN agent obtained 345 rewards

in 500 iterations. Guo et al. (2024) proposed a coordinated sequence optimisation method for road network signals based on heterogeneous multi-agent self-attention networks, so as to improve the performance of multi-intersection signal control strategies within the road network. Li et al. (2025) proposes an improved deep reinforcement learning algorithm for robot path planning. First, the Dueling DQN network architecture is adopted, combined with the preferential experience replay strategy, to learn and utilise the experience data more effectively. Second, it expands the mobile space of the robot and enhances the diversity and flexibility of the action space.

In the process of action selection, the Artificial Potential Field (APF) algorithm is introduced to intervene in action selection with a certain probability, thus speeding up the convergence process of the network. At the same time, the greedy strategy is used to balance exploration and development in favour of better exploring the environment and utilising the existing knowledge. On this basis, a composite reward function is designed to integrate various reward mechanisms, which improves the convergence performance and path planning ability of the algorithm.

Cui, X. Y. (2024) proposed an anomaly detection scheme based on a Graph Attention Network (GAT) and Informer. GAT can effectively learn sequence features, and Informer has an excellent performance in long-time series prediction. Multivariate time series anomalies are detected by using the long-term forecast loss and the short-term forecast loss. The short-term forecast is used for predicting the next time value, and the long-term forecast is used to assist the short-term forecast. The experiments carried out show that the proposed method can accurately locate anomalies and achieve interpretability. Devi et al. (2024) proposes an elastic distributed formation controller based on an attack signal compensator. Wang, S. et al. (2023) proposes a predictive control scheme based on a dynamic event drive.

By introducing networked predictive control methods, the negative effects of time-varying delay and aperiodic DoS attacks on system performance are effectively reduced. Homod et al. (2023) and Li et al. (2024b) propose an obstacle avoidance control method for heterogeneous agent formation based on deep reinforcement learning, which enables agents to gradually optimise comprehensive strategies to cope with complex interactive information. Wang, X. et al. (2023) propose a reactive power-voltage control strategy for distribution networks based on multi-agent deep reinforcement learning, and adopts multi-agent double-delay deep deterministic strategy gradient algorithm to solve the real-time optimal control strategy.

In (Hu et al., 2023) the multi-agent double-delay deep deterministic strategy gradient algorithm is used to study the obstacle avoidance and target reaching problems for multi-UAV. Guo et al. (2023) propose a distributed multi-agent reinforcement learning decision algorithm based on trajectory prediction, which can solve the problem of pursuing intelligent escape targets under incomplete information. Oroojlooy & Hajinezhad (2023) propose the design of a multi-intelligent car body segment based on RT-Thread, which improves the execution efficiency and stability of the system.

Yin et al. (2024) proposed an improved DDQN algorithm based on the average Q value estimation and reward redistribution. First, in order to improve the accuracy of the target Q value, the average of multiple previously learned Q values in the target Q-network is used to replace the single Q value in the current target Q-network. Second, the reward redistribution mechanism is designed so as to adjust the final reward of each action by using the round reward in the trajectory information to overcome the sparse reward problem. In addition, a reward-first experience selection method is introduced to rank the experience samples according to reward values to ensure the frequent utilisation of high-quality data. Finally, the effectiveness of the proposed algorithm in fixed-location scenarios and random environments is verified by simulation experiments. Li et al. (2024a) propose a two-stage voltage control strategy for an active distribution network in the power grid, which adopts the method of offline training and online operation. In literature, the multi-agent deep deterministic strategy gradient (MADDPG) algorithm was applied to construct a multi-conveyor intelligent agent cooperative control system.

In (Luo et al, 2023), aiming at the pursuit and escape game problem of UAVs in complex combat environments, the Markov model was established, and the training process of multi-agent deep deterministic reinforcement learning

algorithm with centralised training and distributed execution was constructed by adopting the zero-sum game idea, and the Nash equilibrium solution of the pursuit and escape game was obtained. Luo et al. (2023) propose a multi-AGVS path planning algorithm based on the multi-agent deep deterministic strategy gradient.

In (Mani et al., 2023), the multi-agent reinforcement learning MA-SARSA algorithm is proposed, which lays the foundation for the research of more complex and changeable multi-agent cooperation and competition scenarios.

In (Fayti et al., 2023), a multi-agent deep reinforcement learning scheme was proposed to complete the cooperative rounding task in the game confrontation of unmanned boat groups. Zeng et al. (2023) propose an improved sample-based elastic dynamic event-triggering mechanism that includes dynamic auxiliary variables. Yao, Li & Gao (2024) propose a Unified model for Multi-agent Reinforcement Learning and AI Planning (UniMP), which improves the flexibility of reinforcement learning algorithms. Gu et al. (2023) proposes a Q-learning algorithm based on action sampling, in which each agent selects actions independently, effectively reducing the amount of computation in the learning phase.

In (Sivaranjani & Vinod, 2023), a consistency control protocol was proposed to solve the time-varying control problem of queue following for an autonomous vehicle with external interference and unknown input by using a proportional integral observer, but the problem of uncertain parameter changes during operation was not considered. Li et al. (2024b) propose an error model based on vehicle dynamics, and deduces a steady-state controller to track and control the vehicle path to achieve a relatively high-precision control effect. Chai, G. et al. (2023) propose a cooperative formation control scheme to cope with the normal operation of the convoy under sensor fault. Tilki, U., Ölgün (2023) propose a distributed model predictive controller (DMPC) based on the leader-follower approach to deal with multi-robot cooperative formation tasks. Zhang et al. (2023) propose a region division positioning algorithm of the truncated octahedral (TO) model controlled by underwater vehicles, which features a small error, a high positioning coverage and a strong robustness.

A comprehensive evaluation of the above literature shows that remarkable progress has been made in the fields of robot cooperative control and management, intelligent manufacturing and automation systems, and intelligent distributed computing and network architecture. These studies provide important technical advances and theoretical foundations for task assignment, collaborative path planning, communication architecture design, intelligent operation algorithm improvement, and applications in intelligent manufacturing of multi-robot systems. However, these studies also have some common limitations and shortcomings, especially with regard to the verification of their practical applicability, computational complexity, real-time performance and reliability, system adaptability and scalability.

First of all, most studies focus on the construction of theoretical frameworks or experimental verification in specific scenarios, and lack an extensive verification of their practical applicability in complex and dynamic environments. The computational complexity and real-time problems related to the path planning method in practical applications have not been fully solved. Second, although the design of the communication architecture performs well in specific scenarios, its universality and scalability have not been fully verified. In addition, although the algorithms in intelligent manufacturing and automation systems perform well in experiments, the validation of their applicability in real industrial environments is still insufficient.

In this paper, an intelligent robot cooperative control framework based on a regional distributed structure is proposed, which divides the whole task into multiple sub-regions, and the robots in each sub-region are responsible for completing specific transportation tasks. The distributed DQN algorithm is applied to the cooperative control of intelligent robots for the first time, which achieves the cooperative decision and motion control of multiple robots in complex environment. This structure not only helps to reduce the complexity of the system, but also improves the collaborative efficiency of the robots and the accuracy of the task completion. This method can not only improve the decision-making efficiency of the robots, but also effectively reduce the consumption of computing resources, making the system more expandable and flexible.

3. Research Method

3.1 Model Building

The running track of the material handling robot in the new energy vehicle assembly factory is simplified as shown in Figure 1 below. It is assumed that the target is K, the obstacle is Z, and the target area reached by the transport is M. The task requires n intelligent handling robots to effectively reach the K region and then smoothly move the target object to the M region. Therefore, the goal of each robot is to control its direction and speed of operation and avoid collisions between individual robots and between robots and obstacles.

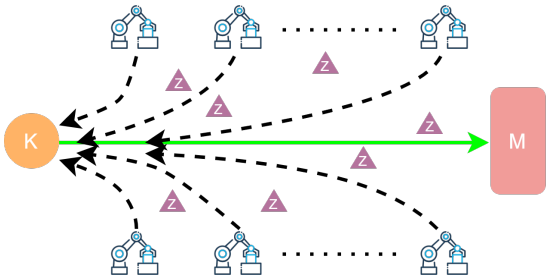

Figure 1. Operation diagram for the intelligent handling robot

Therefore, the operation process for the robot consists in constantly making collision decisions regarding other robots or obstacles. The behavior of the decision process is described as a reinforcement learning problem pertaining to the Markov decision process, which encompasses the observation space, action space, reward function and algorithm framework.

In order to better study the operational problem for the intelligent robot, n intelligent robots control problems shall be treated as multi-agent cooperative control problems. Taking the position coordinates of the i-th intelligent robot as the origin, a rectangular coordinate system is established with its forward direction as the positive direction of the horizontal axis (x). The intelligent robot operation plan is divided into 4 quadrants, as shown in Figure 2.

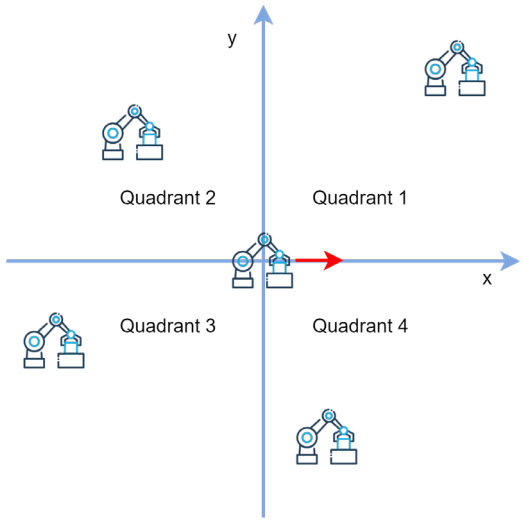

Figure 2. Operation area division for the intelligent handling robot

The collision avoidance strategies between the i-1 intelligent robot and the i intelligent robot or obstacles are shown in Table 1.

The optimal path planning for n intelligent robots can be described as:

$$\begin{cases} |d_i - d_j| > d_s, i, j = 1,2,3,...,n; \\ |d_i - d_k| > d_t, i, k = 1,2,3,...,n; \\ |d_k - d_m| > d_p, k, m = 1,2,3,...,n; \\ \min \sum_{i,j=1}^{n} D_i, V, \delta \end{cases} \quad (1)$$

where d_i and d_j are the positions of any two robots,d_k is the target starting point, d_m is the target end point, d_s is the minimum safe distance between any two robots, d_t is the minimum distance between the robot and the moving target, and d_p is the minimum distance between the intelligent robot and the obstacle. D_i is the distance covered

Table 1. Collision avoidance strategies for intelligent robots

i position	i-1 position			
	quadrant 1	quadrant 2	quadrant 3	quadrant 4
quadrant 1	quadrant 4	Quadrant 3 turns to Quadrant 4	quadrant 4	quadrant 4
quadrant 2	quadrant 1	quadrant 1	quadrant 4	quadrant 1
quadrant 3	quadrant 1	quadrant 1	quadrant 4	Quadrant 3 turns to Quadrant 4
quadrant 4	quadrant 1	quadrant 1	quadrant 1	quadrant 1

by the robot, V is the speed of the robot, and δ is the direction of the robot.

The i-th dynamic model of the intelligent robot is expressed as follows:

$$\begin{cases} v_{i,x} = v_{0,x} + a_{i,x} \times t_i, \\ x_i = x_0 + v_{i,x} \times t_i, \\ v_{i,y} = v_{0,y} + a_{i,y} \times t_i, \\ y_i = y_0 + v_{i,y} \times t_i, \\ D_i = \sqrt{x_i^2 + y_i^2}, \end{cases} \tag{2}$$

where $v_{0,x}$ is the initial velocity of the robot in the x direction, and $a_{i,x}$ is the acceleration of the robot in the x direction, $v_{0,y}$ is the initial velocity of the robot in the y direction, and $a_{i,y}$ is the acceleration of the robot in the y direction, x_i and y_i represent the distance the robot moves in the x direction and the y direction, respectively, and t_i is the time step.

In the model expressed in equation (1), the reward function is mainly divided into the collision reward function involving the i-th intelligent robot and the j-th intelligent robot, the reward function of the i-th intelligent robot approaching the target point, the reward function of the i-th intelligent robot colliding with the obstacle and the speed reward function.

The greater the distance between the i-th intelligent robot and the j-th intelligent robot, the greater the distance reward. The distance reward function is expressed as follows:

$$R_{i-j} = \begin{cases} -1, d_{i-j} < 1; \\ 2d_{i-j} - 3, 1 \le d_{i-j} \le 2; \\ 1, d_{i-j} > 2; \end{cases} \tag{3}$$

otherwise, $d_{i-j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ 。

The smaller the distance between the i-th intelligent robot and the target object, the greater the distance reward. The distance reward function is expressed as follows:

$$R_{i-t} = \begin{cases} 1, d_{i\text{-}t} < 0.5; \\ 2d_{i-t} - 2, 0.5 \le d_i \le 1; \\ -1, d_{i-t} > 1; \end{cases} \tag{4}$$

otherwise, $d_{i-t} = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2}$ 。

The greater the distance between the i-th intelligent robot and the obstacle, the greater the distance reward. The distance reward function is expressed as follows:

$$R_{i-k} = \begin{cases} -1, d_{i-k} < 1; \\ 2d_{i-k} - 2, 1 \le d_{i-k} \le 1.5; \\ 1, d_{i-k} > 1.5; \end{cases} \tag{5}$$

otherwise, $d_{i-k} = \sqrt{(x_i - x_k)^2 + (y_i - y_k)^2}$ 。

The speed of the i-th intelligent robot is rewarded, and the speed is set to the upper and lower critical values, 0.2 and 1, respectively. When the speed varies between critical values, the faster the speed, the greater the reward. The speed reward function is expressed as follows:

$$R_{v_i} = \begin{cases} -1, otherwise; \\ 5v_i - 4, 0.2 \le v_i \le 1 \end{cases} \tag{6}$$

otherwise, $v_i = \sqrt{v_{i,x}^2 + v_{i,y}^2}$ 。

With regard to the acceleration reward for the i-th intelligent robot, the greater the acceleration, the greater the reward. The acceleration reward function is expressed as follows:

$$R_{a_i} = \begin{cases} -1, otherwise; \\ \dfrac{a_i}{0.2}, 0.2 \le a_i; \end{cases} \tag{7}$$

otherwise, $a_i = \sqrt{a_{i,x}^2 + a_{i,y}^2}$ 。

Therefore, the total reward function for the i-th intelligent robot is:

$$R_{all} = R_{i-j} + R_{i-t} + R_{i-k} + R_{v_i} + R_{a_i} \tag{8}$$

The reinforcement learning framework is shown in Figure 3 below.

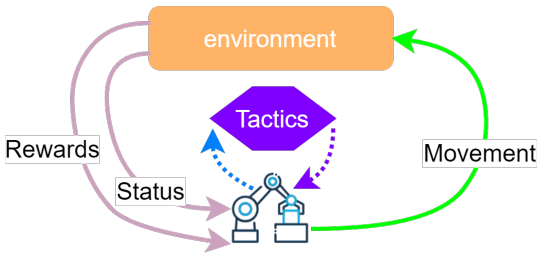

Figure 3. Reinforcement learning framework for the Markov decision process

In the framework of reinforcement learning, the intelligent robot makes a decision based on the current state of the environment (S, state) and decides what action to take next (A, action). After each action, the environment will give a

corresponding reward (R, reward) based on the action of the intelligent robot, and the state S will be transferred to the state S^. This is repeated, and the robot generates a large amount of S, A, and R data in the process of continuous interaction with the environment. Based on this data, the reinforcement learning algorithm allows the agent to make more correct decisions, that is, to learn the strategy. Therefore, it can be understood that intelligent robots interact with the environment through actions, states, and rewards, and constantly learn strategies for maximising those rewards.

The behavior of n intelligent robot control problems is described as a Markov decision process This process includes three states: cooperation, competition and independence. The framework of n robot reinforcement learning algorithms is shown in Figure 4 below.

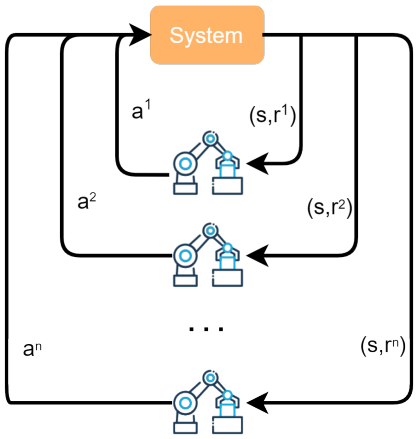

Figure 4. Markov decision process for n robots

The Markov decision process is defined as follows:

$$(S, A, \pi, R, \gamma) \tag{9}$$

where S is the set of the current environmental states of the intelligent robot, A is the set of actions taken by n intelligent robot, R is the reward set, π represents the learning strategy and γ is the calculated return factor. The learning strategy π is calculated based on the following formula:

$$\pi(s \mid a) = p\left[A_t = a \mid S_t = s\right] \tag{10}$$

To maximise the employed strategy, one should know the cumulative return, which is expressed as follows:

$$G_1 = \sum_{m=0}^{\infty} \gamma^m R_{t+m+1} \tag{11}$$

For the cumulative return, the value function is used to represent the cumulative return expectation. Therefore, the expectation of the i-th intelligent robot is expressed by the following formula:

$$q_\pi(s,a) = E_\pi[\sum_{m=0}^{\infty} \gamma^m R^i_{t+m+1} \mid S_t = s, A_t = a] \tag{12}$$

In equation (6), if the action (a-1) taken by another intelligent robot (i-1) makes the return of the i-th intelligent robot worse, the i-th intelligent robot can get the maximum expected return. This return can be expressed as:

$$V_i(s,a) = \max_\pi \min_a - \sum_a Q_i(s,a,a-)\pi(s,a) \tag{13}$$

Then the optimal strategy is found by maximising the action value function, according to the following formula:

$$\pi(a \mid s) = \begin{cases} 1; a = \arg\max_{a \in A} q(s,a) \\ 0; otherwise \end{cases} \tag{14}$$

3.2 The Distributed DQN Algorithm

The DQN algorithm is composed of a Q-Learning algorithm and a neural network. It is one of the most representative algorithms in the family of reinforcement learning algorithms (Lu & Liu, 2023; Kim & Han, 2023; Niu et al., 2024). As its characteristics are concerned, it does not need to understand the environment, it directly gives the value of each action in the current environment, and updates the neural network once that action is taken. During algorithm execution, one can learn while executing. The DQN algorithm uses the experience pool as shared data, adopts the structure of centralised training and features a decentralised execution. The DQN algorithm framework is shown in Figure 5. The DQN algorithm uses two neural networks, namely the Q-value network and the target Q network, the two networks having the same structure.

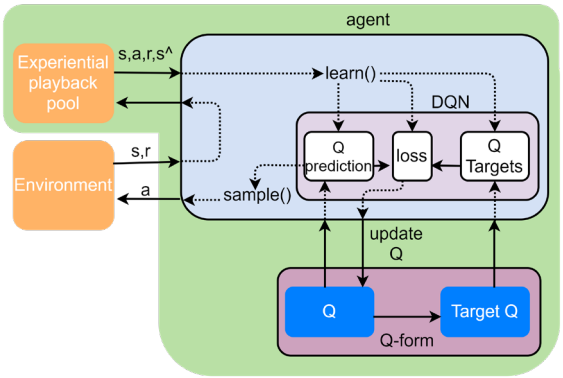

Figure 5. The DQN algorithm structure

The intelligent robot constantly interacts with the environment. At the moment of state s, it generates action a according to its own strategy, and action a makes the environment produce a new state s^ and at the same time it gets a reward r from the environment. The interactive data (s,a,r,s^) is introduced into the experience playback pool. When there is enough data (s,a,r,s^) in the experience pool, a unit of data is randomly selected from the experience pool, the predicted value of Q is calculated using the current network, the target value of Q is calculated using the target Q network, then the loss function between the two values is calculated, and the current network parameters are updated using the gradient descent minimisation loss function. After the above process is repeated several times, the parameters of the current network should be copied to the target Q network. The gradient descent minimisation loss function is used to continuously debug the network weights. The calculation formula is as follows:

$$L_i(\theta_i) = E_{(s,a,r,\hat{s})} \sim U(D)[R + \gamma \max_{\hat{a}} Q(\hat{s}, a^i; \hat{\theta}_i) - Q(s,a;\theta_i)]^2 \quad (15)$$

where θ_i is the Q network parameter and $\hat{\theta}_i$ is the target network parameter of the i-th iteration. The gradient descent update network weight θ_i is calculated as follows:

$$\frac{\partial L_i(\theta_i)}{\partial \theta_i} = E_{(s,a,r,\hat{s})} \sim U(D)[(R + \gamma \max_{\hat{a}} Q(\hat{s}, a^i; \hat{\theta}_i) - Q(s,a;\theta_i))\nabla_{\theta_i} Q(s,a;\theta_i)] \quad (16)$$

The distributed DQN algorithm proposed in this paper includes 4 quadrants according to the intelligent robot operation area as shown in Figure 2. The distributed DQN algorithm framework is shown in Figure 6.

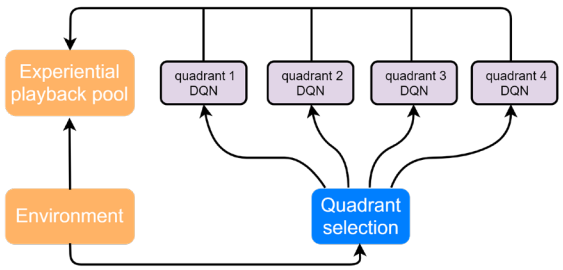

Figure 6. The distributed DQN algorithm structure

Quadrant 1: The focus is on the parallelisation of algorithms and the improvement of distributed computing capabilities, and the aim is to achieve a more efficient model training by optimising data parallelisation and parameter server mechanisms. The training data is spread across multiple compute nodes, each of which independently processes a portion of the data and calculates gradients, which are then aggregated to a parameter server for updating model parameters. The parameter server is responsible for storing and updating global model parameters and ensuring that all compute nodes have access to the latest parameters.

Quadrant 2: The focus is on distributed storage and optimising the access to the experience replay pool to ensure that each compute node can efficiently use the experience pool data for training. The experience replay pool is split into multiple parts and stored on different compute nodes. This way, each node can access and update its part of responsibility independently, without waiting for other nodes. Data access strategies should be optimised using caching, prefetching, and parallel reading to further improve the efficiency of the access to the experience replay pool. In addition, data compression and coding techniques are used to reduce the overhead of data transmission and storage.

Quadrant 3: The focus is on improving the stability of the algorithm, and reducing computational bias and sample fluctuation by improving the experience playback mechanism and target network update strategy. Using priority experience playback and batch experience playback, the sample fluctuation and sample bias are reduced, and the convergence speed and stability of the algorithm are improved. The target network is an important component of the DQN algorithm, which is used to calculate the target Q value. Soft update and progressive update are adopted to reduce the calculation deviation or overestimation caused by the calculation of max Q value, and further improve the stability of the algorithm.

Quadrant 4: The ability of the algorithm to handle complex environments and large-scale data sets should be explored, and the robustness and generalisation ability of the algorithm should be improved by optimising the model structure and training strategies. A deep convolutional neural network, a recurrent neural network and regularisation techniques are introduced to

improve the expressiveness and generalisation ability of the model. RMSprop and cross-validation are used to improve the performance and stability of the algorithm.

Based on the above design ideas, TensorFlow is used to implement the distributed DQN algorithm. The execution process for the distributed DQN algorithm is as follows.

3.2.1 Initialisation Phase

(1) Environment initialisation. Set up reinforcement learning environment, including state space, action space, reward function, etc.

(2) Agent initialisation. In a distributed environment, multiple agents are initialised, and each agent has its own neural network model (i.e. Q network) and state and action space.

(3) Experience replay pool initialisation. Initialises a distributed experiential replay pool that stores experiential data (state, action, reward, next state) generated by the agent's interaction with the environment.

(4) Parameter server initialisation. Set up a parameter server to store and update global model parameters to ensure parameter consistency across compute nodes.

3.2.2 Training Phase

(1) Agents interact with the environment. Each agent selects an action based on its current state and performs that action.

(2) The environment returns the next state and reward according to the agent's action.

(3) The agent stores the current state, action, reward, and next state into a distributed experiential replay pool.

(4) Experience replay and model update. Sample a batch of empirical data randomly from a distributed empirical replay pool.

(5) The sampled data is sent to the Q network for calculation and the predicted Q value is obtained.

(6) Use the target network to calculate the target Q value and calculate the loss function.

(7) Update the Q network parameters through backpropagation.

(8) Parameter synchronisation and update. The parameter server periodically collects model parameters from each compute node and synchronises and updates them. The updated parameters are distributed to each compute node to ensure parameter consistency.

(9) Performance evaluation and adjustment. Periodically evaluate the performance of the agent, such as cumulative rewards, accuracy, etc. Adjust the training strategy and model structure according to the performance evaluation results.

In order to reduce the overestimation of the model, two estimation functions and Double Q-learning are also used in the framework of the algorithm. The two estimation functions are Q^A and Q^B, and each Q function updates the next state with the value of the other Q function. The algorithm`s steps are as follows:

Algorithm

1. Randomly initialise Q(s,a), where s represents the environment and a represents the action;

2. Initialise s;

3. Select action a from the neural network according to the current s and constraint condition ();

4. Perform a, observe s^ (the new environment), and r (the incentive).

The implementation process is as follows:

$$Q^A(s,a) \leftarrow Q^A(s,a) + \beta(s,a)[r + \gamma Q^B(s^{\wedge}, a^*) - Q^A(s,a)] \quad (17)$$

$$Q^B(s,a) \leftarrow Q^B(s,a) + \beta(s,a)[r + \gamma Q^A(s^{\wedge}, b^*) - Q^B(s,a)] \quad (18)$$

$s \leftarrow s^{\wedge}$

Repeat Steps 3 and 4 until s reaches the end state.

otherwise， $a^* = \arg\max_a Q^A(s^{\wedge}, a)$ 。

At this point, the algorithm is built.

4. Results and Discussion

In order to improve the clarity of result presentation, this paper uses charts and visualisation methods to illustrate the performance of each employed algorithm under the cooperative

working condition for multiple intelligent robots. In particular, the success rate curves of each algorithm in different training rounds and the specific behavior trajectories of multi-robot cooperative tasks are drawn. These charts and visualisations not only help to understand the performance of each algorithm more intuitively, but also provide a strong basis for a subsequent optimisation and improvement.

4.1 Parameter Selection

The algorithm parameters set in this paper mainly include the number of intelligent robots, the number of obstacles, the number of model training times, the number of samples extracted from the experience pool, the reward factor, the Q network parameters and the network learning rate, and the parameter values are set as shown in Table 2 below. The simulation platform is MATLAB R2022a.

Table 2. Model parameters

Parameters	Values
Number of intelligent robots (i)	1-160
Number of obstacles	5
Training times (N)	200
Q Network parameters (θ_i)	0.99
Reward factor (γ)	0.99
Number of samples extracted from the experience pool	120
Network learning rate	0.001

4.2 Experiment and Discussion

First, the percentage of the robots successfully completing a handling process with regard to the total handling times is defined as the handling success rate.

Second, model training is grouped. The model was divided into four groups to carry out model training under the conditions of only one intelligent robot working, only 2 intelligent robots working, only 4 intelligent robots working and 8 intelligent robots working. The first group was trained 10 times, the second group was trained 100 times, the third group was trained 200 times, and the fourth group was trained 280 times.

Third, the handling success rate was tested, as shown in Figure 7. As it can be seen, with the increase in the number of model training times, the material handling success rate was slightly improved. With the increase in the number of robots at the same time, the material handling success rate decreased slightly. The reason for this analysis is the increase in the number of training times and the overfitting of the employed model. However, the overall handling success rate of the model is higher than 80%.

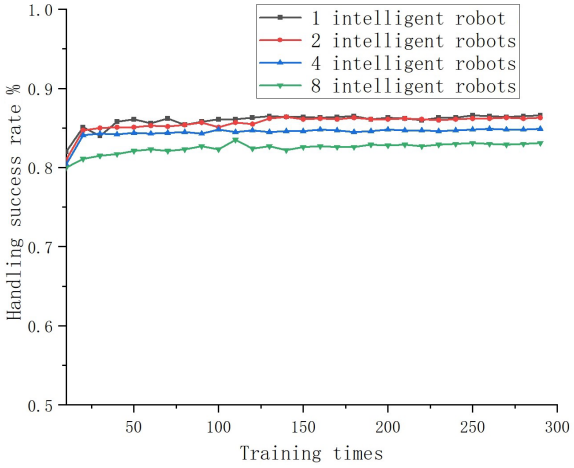

Figure 7. Material handling success rates for intelligent robots

The in-depth analysis shows that with the increase in the number of model training times, although the handling success rate features a weak upward trend, this increase may be accompanied by the risk of overfitting the model. Overfitting means that the model performs very well on the training data but it is difficult to generalise it for previously unseen scenarios, which explains why the handling success rate drops slightly in more complex scenarios where the number of robots increases at the same time. This shows that while the model is robust for a single task or a small number of robot tasks, its generalisation ability needs to be strengthened when dealing with the more complex and dynamic environment of multi-robot collaboration.

Then the performance of the proposed algorithm is compared with that of traditional intelligent algorithms.

First, the algorithms were chosen. In this study, the Twin Delayed Deep Deterministic Policy Gradient Algorithm (TD3), the DQN algorithm, and the DDPG (Deep Deterministic Policy Gradient) algorithm were used and compared

with the proposed algorithm, and the number of experimental intelligent robots was set at 8.

Second, the handling success rate was tested. As shown in Figure 8 below, 10 numbers are drawn at different intervals (between 10 and 280).

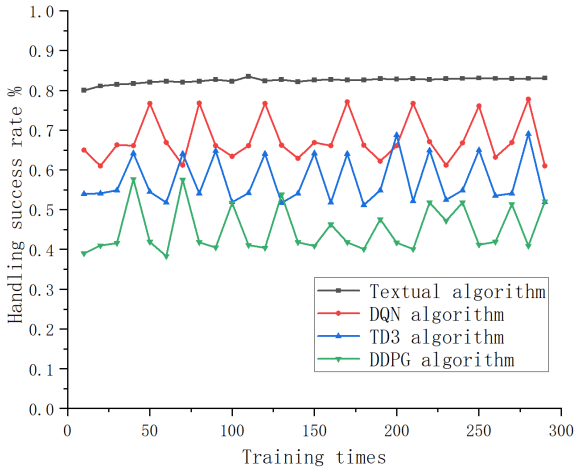

Figure 8. Comparison of the success rates for intelligent robot material handling under the four employed algorithms

As it can be seen from Figure 8, the proposed algorithm obtained the highest success rate. The overall volatility of the proposed algorithm is very small, and the robustness of the model is good.

The in-depth analysis shows that the TD3 algorithm effectively alleviates the overestimation problem of the traditional deep reinforcement learning algorithm by introducing the double-delay deep deterministic strategy gradient method, and improves the stability of this strategy. Under the condition of 8 intelligent robots working together, the TD3 algorithm shows a good generalisation ability and robustness, and the handling success rate is maintained at a high level. However, with the increase in the number of robots, the complexity of the environment also increases, and the TD3 algorithm still faces certain challenges in handling the interaction and coordination of multiple robots, resulting in a slight decline in the handling success rate. As a classic algorithm in the field of deep reinforcement learning, the DQN algorithm achieves an effective decision making in complex environments by using a convolutional neural network to approximate the Q-value function. However, the DQN algorithm faces limitations in dealing with a continuous action space and is easily affected by the overestimation problem. In the scenario of eight intelligent robots working together, the handling success rate of the DQN algorithm is relatively low, especially when the number of robots is high, the decline in its performance is more obvious. The DDPG algorithm is an extension of the DQN algorithm on a continuous action space. By introducing the deterministic strategy gradient method, it achieves an effective decision making in a continuous action space. Under the condition of 8 intelligent robots working together, the DDPG algorithm shows a good adaptability and stability, and its handling success rate is higher than that of the DQN algorithm. However, compared with the TD3 algorithm, the DDPG algorithm still has some shortcomings in dealing with the interaction and coordination of multiple machines, resulting in its performance being slightly inferior to that of the TD3 algorithm in a complex environment. Based on the advantages of the TD3 algorithm and the DDPG algorithm, the proposed algorithm has been improved and optimised. By introducing a more efficient exploration strategy, improving the state representation method and optimising network structure, the algorithm in this paper shows a higher success rate and a stronger generalisation ability under the condition of 8 intelligent robots working together. Compared with the TD3 algorithm, the DQN algorithm and the DDPG algorithm, the proposed algorithm achieved a better performance in a complex environment.

Third, the average reward obtained by the four algorithms is illustrated in Figure 9.

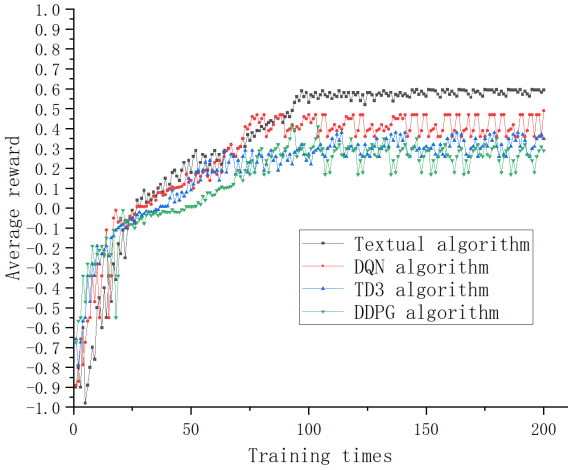

Figure 9. Average reward

As it can be seen from Figure 9, compared with other algorithms, the proposed algorithm has

the highest convergence speed and can obtain the highest reward after convergence, and the obtained reward is more stable after convergence, maintaining itself at about 0.6.

The fitting coefficient (R^2) was used for model evaluation (Gao et al., 2023). In the context of the evaluation, the higher the R^2 value, the better the model. The definition of R^2 is as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y_i})}{\sum_{i=1}^{n}(Y_i - \bar{Y}_i)} \tag{19}$$

where n is the number of training times, Y_i is the real measured value, $\hat{Y_i}$ is the value measured by employing this model, and $\bar{Y}_i$ is the average of the true measured values.

The robots were trained in different numbers and the results for the calculated R^2 values are shown in Figure 10.

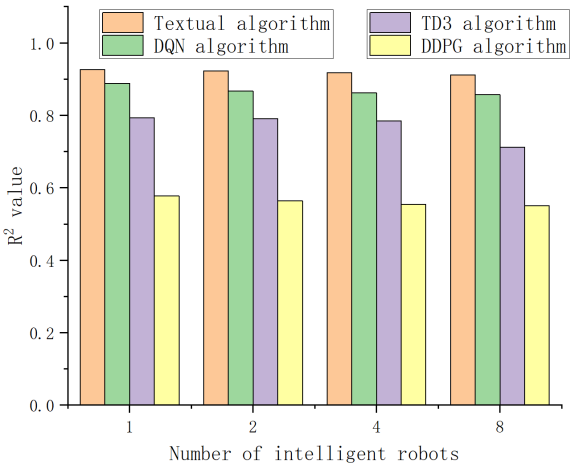

Figure 10. R^2 values under the four employed algorithms

As it can be seen from Figure 10, among the four employed algorithms, the proposed algorithm has the best fitting ability. For different numbers of robots, the values of the fitting coefficient R^2 are all greater than 0.9, indicating that the model has a good fitting ability.

In order to improve the survivability of the algorithm in practical applications, the number of robots and the complexity of the environment are increased in the experiment carried out to verify the capacity expansion performance of the proposed algorithm. The experimental situation is as follows.

First, the number of intelligent robots employed in the model was increased to 20, 40, 80, and 160. Figure 11 shows the model capacity in relation to the handling success rate.

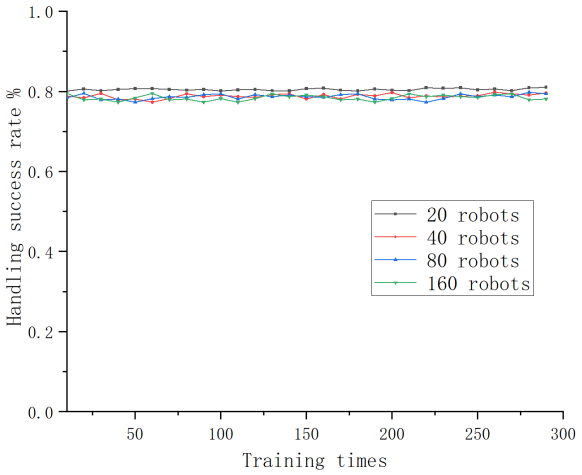

Figure 11. Model capacity test

The experimental results show that when the number of intelligent robots is 20, the handling success rate can be stably maintained above 0.8, which fully validates the model's efficient coordination ability and excellent task execution ability in a relatively complex environment.

However, with the further increase in the number of intelligent robots, the success rate of material handling has shown a slight downward trend. Specifically, when the number of robots climbed to 40, 80 or 160, the handling success rate dropped slightly to 0.78, but it still remained at a relatively high level. This result not only reveals that the model has a good scaling ability in the face of large-scale robot-based collaborative tasks, but it also points out the challenges in maintaining an efficient collaboration and task execution in high-density and highly interactive multi-robot environments.

The in-depth analysis shows that with the increase in the number of robots, the complexity of the environment is significantly increased, and the interaction and coordination needs of robots are sharply increased, which brings about higher requirements with regard to the model`s decision-making ability, environmental perception ability and real-time response ability. Although the model proposed in this paper shows a good performance in dealing with these challenges, in some extreme cases, such as resource competition, path conflict

and communication delay caused by an excessive number of robots, the success rate of the model may be affected to some extent.

Second, the complexity of the environment should be improved. To verify the generalisation ability of the model, the number of intelligent robots should be increased and the random placement of obstacles should be adjusted. The results for the model generalisation ability are shown in Figure 12.

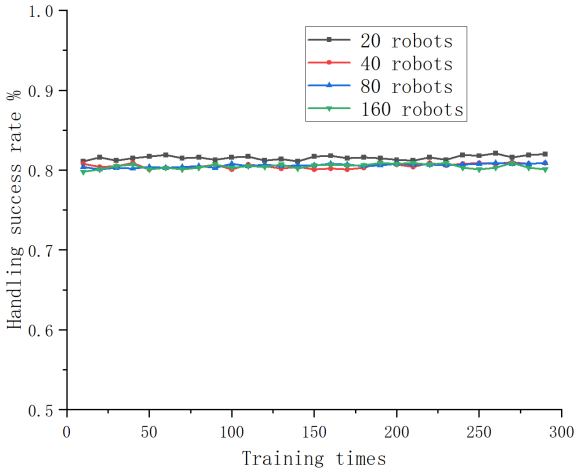

Figure 12. Model generalisation ability test

As it can be seen from Figure 12, when the number of robots is 20, the handling success rate remains at 0.8, however, when their number increases to 40, 80 and 160, the success rate decreases slightly, but it remains at 0.78. The experimental results show that when the number of intelligent robots is 20, the model shows an excellent performance, and the success rate of material handling remains at the high level of 0.8. When the number of intelligent robots is further increased to 40, 80 or 160, although the handling success rate drops slightly to 0.78, the model still maintains a high success rate on the whole, which indicates that this model shows a good capacity expansion and generalisation ability when faced with large-scale and high-density robot collaborative work and dynamically changing obstacle layout. The model can effectively deal with the problems of path planning, obstacle avoidance strategy and task assignment among robots in a complex interactive environment, which ensures the efficient execution of tasks.

Finally, the algorithm was applied to the small area environment of the actual production shop for testing purposes. The experimental process is as follows.

First, spatial modeling is carried out. The operation scenario for the handling robot in the automobile intelligent assembly factory is modeled by means of a grid, arranged according to an equal spacing criterion, and structured into 11 rows and 11 columns with 121 nodes. It is assumed that the target is K, the obstacle is Z, and the target area reached by the transport is M. The task requires n intelligent handling robots to effectively reach the K region and then smoothly move the target object to the M region, and the four intelligent handling robots are represented by the letters A, B, C and D, as shown in Figure 13 below. The aim of the material transportation process is to find the optimal path.

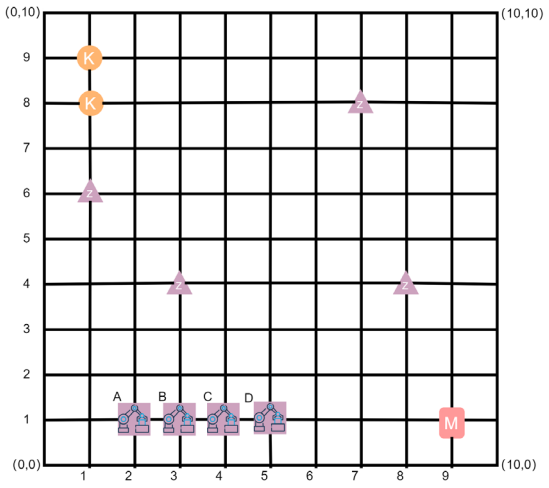

Figure 13. Electronic map of the automobile intelligent assembly factory

Second, model testing is carried out. The intelligent handling robot first runs from the current location to the specified pick-up point, loads the goods from the specified pick-up point, and then moves the goods to the target point. For each task, multiple intelligent robots move at the same time, and the proposed algorithm coordinates the task allocation among the intelligent handling robots to carry out the path planning for each intelligent handling robot for transporting goods. Intelligent handling robot A and D should be taken as an example, as shown in Figure 1. The optimal running path of robot A is represented by blue and red trajectories. The optimal running path of robot D is represented by purple and green trajectories.

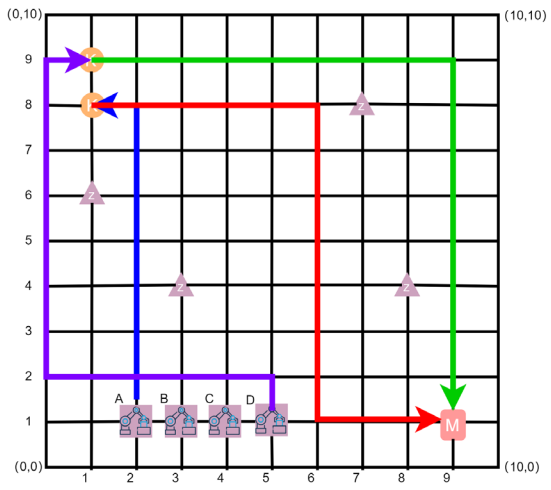

Figure 14. Optimal running path of intelligent robot A and D

Third, practical testing is carried out. By applying the electronic map model shown in Figure 13 to the intelligent workshop for regional testing, the operation results shown in Figure 14 can be obtained. The results show that the proposed method is feasible.

Taking the implementation of four intelligent robots as an example, between two adjacent nodes separated by a distance of 28 meters, the four robots completed one handling operation for testing purposes. The task completion efficiency before and after the application of the proposed model is shown in Table 3 below. The accumulated idle time refers to the stationary waiting time for the intelligent robot.

Table 3. Application efficiency of the proposed model

The employed method	Completion time	Accumulated idle time
Traditional model	178s	98s
Textual model	56s	12s

It can be seen from Table 3 that under the condition of handling the same material, the application of the model proposed in this paper reduces the task completion time by 122 s and the waiting time for the intelligent robot by 86 s in comparison with the traditional model.

5. Conclusion

In this paper, an intelligent robot motion coordination control method based on a regional distributed structure is proposed to fulfill the simultaneous transportation task for multiple robots. The method mainly relies on the deep Q-network (DQN) algorithm, and by means of reinforcement learning, the intelligent robot learns the optimal control strategy in the process of interacting with the industrial environment.

However, although the proposed method shows a great potential in theory and practice, there are still some limitations to it. First, the DQN algorithm needs to discretise the continuous action space when dealing with continuous control problems, which leads to a loss of accuracy and a dimensional disaster, affecting the performance of robot in complex environments. Secondly, the training process for the DQN algorithm is slow and requires a lot of training data and computing resources, which will bring about certain challenges with regard to its practical application. In addition, the DQN algorithm will face oscillation problems, resulting in an unstable training process, which would require the further optimisation and stabilisation of this algorithm.

Future work could further study and optimise the cooperative control strategy of distributed robot systems in order to improve the cooperative efficiency and stability of multiple robots in complex environments. This could include research into more refined task allocation algorithms, more efficient communication and perception mechanisms, etc.

REFERENCES

Agarwal, V. & Sharma, S. (2023) DQN Algorithm for network resource management in vehicular communication network. *International Journal of Information Technology*. 15(6), 3371-3379. https://doi.org/10.1007/s41870-023-01399-0.

Al-Selwi, S. M., Hassan, M. F., Abdulkadir, S. J., Muneer, A., Sumiea, E. H., Alqushaibi, A. & Ragab, M. G. (2024). RNN-LSTM: From applications to modeling techniques and beyond - Systematic review. *Journal of King Saud University-Computer and Information Sciences*. 36(5), 102068. https://doi.org/10.1016/j.jksuci.2024.102068.

Chai G. F., Xia Y. Z. (2023) Multi-Robot Path Optimization and Simulation for Multi-Route Inspection in Manufacturing. *Int. Journal of Simulation Modelling*, 22(1), 121-132. https://doi.org/10.2507/IJSIMM22-1-CO1.

Chaudhry, S. & Gautam, N. (2020) A Communication Architecture Based on ROS2 for the Control in Collaborative and Intelligent Automation Systems. *Psychology and Education Journal*. 57(9), 710-713.

Cui X. Y., Li Z. Z., Yin B. L., Wang W. Q., Liu Y. X., Bai Z. H. (2023) Modelling Analysis of Coupling Deformation between Strip Steel and Roller System. *Int. Journal of Simulation Modelling*, 22(2), 267-278. https://doi.org/10.2507/IJSIMM22-2-644.

Cui, S., Zeng, P., Wang, Z. & Song, C. (2021) Research on Intelligent Protection Technology for Distribution Network with Distributed Generation. In: *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), 12-14 March 2021, Chongqing, China*. Piscataway, NJ, USA, IEEE. pp. 1549-1554.

Devi, K. V. R., Smitha, B. S., Lakhanpal, S., Kalra, R., Sethi, V. & Thajil, S. K. (2024) A Review: Swarm Robotics: Cooperative Control in Multi-Agent Systems. In: *E3S Web of Conferences* (Vol. 505, 03013, *3rd International Conference on Applied Research and Engineering (ICARAE2023)*). Les Ulis, France, EDP Sciences. https://doi.org/10.1051/e3sconf/202450503013.

Fayti, M., Mjahed, M., Ayad, H., Kari, A. E. (2023) Recent Metaheuristic-Based optimization for system modeling and PID controllers tuning. *Studies in Informatics and Control*, 32(1), 57–67. https://doi.org/10.24846/v32i1y202306.

Gao, X., Deng, F., Wu, G., Pan, Q., Zheng, C., Wang, W., Cai, T. & Jiang, L. (2023) Divide and Conquer Q-Learning (DCQL) algorithm based Photovoltaic (PV) array reconfiguration scheme for alleviating the partial shading influence. *Solar Energy*. 249, 21-39. https://doi.org/10.1016/j.solener.2022.09.005.

Gu, Y., Zhu, Z., Lv, J., Shi, L., Hou, Z. & Xu, S. (2023) DM-DQN: Dueling Munchausen deep Q network for robot path planning. *Complex & Intelligent Systems*. 9(4), 4287-4300. https://doi.org/10.1007/s40747-022-00948-7.

Guo, J., Cheng, L. & Wang, S. (2023) CoTV: Cooperative Control for Traffic Light Signals and Connected Autonomous Vehicles Using Deep Reinforcement Learning. *IEEE Transactions on Intelligent Transportation Systems*. 24(10), 10501-10512. https://doi.org/10.1109/TITS.2023.3276416.

Guo, J., Du, W., Yang, G., Zhang, W. & Tian, M. (2024) An Intelligent Coordinated Control System for Power Transformers Using Deep Q-Network. *IEEE Access*. 12, 108797-108809. https://doi.org/10.1109/ACCESS.2024.3439015

Homod, R. Z., Mohammed, H. I., Abderrahmane, A., Alawi, O. A., Khalaf, O. I., Mahdi, J. M., Guedri, K., Dhaidan, N. S., Albahri, A. S., Sadeq, A. M. & Yaseen, Z. M. (2023) Deep clustering of Lagrangian trajectory for multi-task learning to energy saving in intelligent buildings using cooperative multi-agent. *Applied Energy*. 351, 121843. https://doi.org/10.1016/j.apenergy.2023.121843.

Hu, M., Wang, X., Bian, Y., Cao, D. & Wang, H. (2023) Disturbance Observer-Based Cooperative Control of Vehicle Platoons Subject to Mismatched Disturbance. *IEEE Transactions on Intelligent Vehicles*. 8(4), 2748-2758. https://doi.org/10.1109/TIV.2023.3237703.

Huang, Y.-H., Zhang, Z., Wang, J., Huang, C. & Zhong, C. (2022) Joint AMC and Resource Allocation for Mobile Wireless Networks Based on Distributed MARL. In: *2022 IEEE International Conference on Communications Workshops (ICC Workshops), 16-20 May 2022, Seoul, Republic of Korea*. Piscataway, NJ, USA, IEEE. pp. 397-402.

Ibrahim, M. M., Ma, L., Zhao, Y., Liu, H. (2023) Robust direct current control of Single-Phase PWM rectifiers based on a mixed H2/H∞ controller. *Studies in Informatics and Control*, 32(1), 81–90. https://doi.org/10.24846/v32i1y202308.

Jankovič, D., Šimic, M. & Herakovič, N. (2024) A comparative study of machine learning regression models for production systems condition monitoring. *Advances in Production Engineering & Management*. 19(1), 78-92. https://doi.org/10.14743/apem2024.1.494.

Kim, Y. T. & Han, S. Y. (2023) Cooling Channel Designs of a Prismatic Battery Pack for Electric Vehicle Using the Deep Q-Network Algorithm. *Applied Thermal Engineering*. 219(C), 119610. https://doi.org/10.1016/j.applthermaleng.2022.119610.

Kuo, R. J., Hsu, C. C., Nguyen, T. P. Q. & Tsai, C. Y. (2024) Hybrid multi-objective metaheuristic and possibilistic intuitionistic fuzzy c-means algorithms for cluster analysis. *Soft Computing*. 28(2), 991-1008. https://doi.org/10.1007/s00500-023-09367-3.

Li W., Wang J., Sun K., Li Y., Yin Y. & Li B. (2020) Realization of intelligent management device for high voltage line electroscope and ground line based on "Internet +". *Science, Technology and Engineering Journal*. 34, 14085-14094.

Li, C., Yue, X., Liu, Z., Ma, G., Zhang, H., Zhou, Y. & Zhu, J. (2025) A modified dueling DQN algorithm for robot path planning incorporating priority experience replay and artificial potential fields. *Applied Intelligence*. 55, art. 366. https://doi.org/10.1007/s10489-024-06149-8.

Li, W., Meng, X.-Q. & Ma, C.-S. (2024a) Optimal Differential Control for Accurate Positioning of Medical Electronic Wristband. *Tehnički vjesnik* [*Technical Gazette*]. 31(3), 792-799. https://doi.org/10.17559/TV-20230809000866.

Li, W., Wang, Q., Lan, Y.-S. & Ma, C.-S. (2024b) Multi-Environment Adaptive Fast Constant False Alarm Detection Algorithm Optimization Strategy. *Tehnički vjesnik* [*Technical Gazette*]. 31(3), 936-944. https://doi.org/10.17559/TV-20230703000781.

Li, Y. (2022) Distributed Cluster Regulation Strategy of Multipark Integrated Energy System Using Multilayer Deep Q Learning. *Computational Intelligence and Neuroscience*. 2022, Article ID 5151369. https://doi.org/10.1155/2022/5151369.

Lin, Z. (2023) Intelligent anti-jamming scheme of UAV swarm based on DQN. In: Li, Y., Yao, H. and Liu, X.

(eds.) *Second International Conference on Informatics, Networking, and Computing (ICINC) 2023), 27-29 October 2023, Wuhan, China.* Cergy-Pontoise, France, SPIE. https://doi.org/10.1117/12.3024761.

Liu, Q., Yan, J. & Huang, H. (2024) Substation Inspection Method Based on Air-Ground Collaboration. In: *2024 IEEE 2nd International Conference on Control, Electronics and Computer Technology (ICCECT)*, *26-28 April 2024, Jilin, China.* Piscataway, NJ, USA, IEEE. pp. 695-703.

Long, G. L., Shi, L., Xin, G., Gao, S., Zhang, W. & Xu, J. (2023) Machine-vision-based online Self-optimizing Control System for Line Marking Machines. *Studies in Informatics and Control*. 32(2), 93–104. https://doi.org/10.24846/v32i2y202309.

Lu, T. & Liu, B. (2023) Automatic Driving Operation Strategy of Urban Rail Train Based on Improved DQN Algorithm. *Journal on Artificial Intelligence.* 5, 113-129. https://doi.org/10.32604/jai.2023.043970.

Luo, L., Zhao, N., Zhu, Y. & Sun, Y. (2023) A* guiding DQN algorithm for automated guided vehicle pathfinding problem of robotic mobile fulfillment systems. *Computers & Industrial Engineering*. 178, 109112. https://doi.org/10.1016/j.cie.2023.109112.

Mani, V., Yarlagadda, S. R., Ravipati, S., Swarnamma, S. C. (2023) ANN optimized Hybrid Energy Management Control System for electric vehicles. *Studies in Informatics and Control*, 32(1), 101-110. https://doi.org/10.24846/v32i1y202310.

Negoiţă, R.F. & Borangiu, T. (2023) Robotic Process Automation of Inventory Demand with Intelligent Reservation. *Studies in Informatics and Control*. 32(2), 5-14. https://doi.org/10.24846/v32i2y202301.

Niu, Y., Yang, Y., Zhang, K., Mu, Y., Wang, Q. & Wang, Y. (2024) Path Planning Method for Unmanned Surface Vessel in On-call Submarine Search Based on Improved DQN Algorithm. *Acta Armamentarii*. 45(9), 3204-3215. https://doi.org/10.12382/bgxb.2023.0909.

Oroojlooy, A. & Hajinezhad, D. (2023) A Review of Cooperative Multi-Agent Deep Reinforcement Learning. *Applied Intelligence*. 53(11), 13677-13722. https://doi.org/10.1007/s10489-022-04105-y.

Pan, W., Sun, Z., Sang, H., Wang, Z. (2023) Encrypted data learning and prediction using a BFV-based cryptographic convolutional neural network. *Studies in Informatics and Control*, 32(1), 37–48. https://doi.org/10.24846/v32i1y202304.

Peng, G., Liao, J., Guan, S., Yang, J. & Li, X. (2022) A pushing-grasping collaborative method based on deep Q-network algorithm in dual viewpoints. *Scientific Reports*. 12, 3927. https://doi.org/10.1038/s41598-022-07900-2.

Savran E., Karpat E., Karpat F. (2024) GA and WOA-Based Optimization for Electric Powertrain Efficiency. *International Journal of Simulation Modelling*. 23(4), 599-610. https://doi.org/10.2507/IJSIMM23-4-699.

Schultz, A. C., Parker, L. E. & Schneider, F. E. (2023) Multi-robot Systems: From Swarms to intelligent automata*. Proceedings from the 2023 International Workshop on Multi-Robot Systems*. Vol. II. Springer.

Sivaranjani, A. & Vinod, B. (2023) Artificial Potential Field Incorporated Deep-Q-Network Algorithm for Mobile Robot Path Prediction. *Intelligent Automation & Soft Computing*. 35(1), 1135-1150. https://doi.org/10.32604/iasc.2023.028126.

Tilki, U., Ölgün, M. (2023) Terminal and Backstepping Sliding Mode Control with Genetic Algorithms for Robot Manipulators. *Studies in Informatics and Control*, 32(2), 117–126. https://doi.org/10.24846/v32i2y202311.

Wang, S., Zheng, S., Ahn, C. K., Shi, P. & Jiang, X. (2023) Event-triggered cooperative control for uncertain multi-agent systems and applications. *International Journal of Robust and Nonlinear Control*. 33(12), 7221-7245. https://doi.org/10.1002/rnc.6752.

Wang, X., Zhao, C., Huang, T., Chakrabarti, P. & Kurths, J. (2023) Cooperative Learning of Multi-Agent Systems Via Reinforcement Learning. *IEEE Transactions on Signal and Information Processing over Networks*. 9, 13-23. https://doi.org/10.1109/TSIPN.2023.3239654.

Wu, B. & Suh, C. S. (2024) Deep Reinforcement Learning for Decentralized Multi-Robot Control: A DQN Approach to Robustness and Information Integration. In: *Proceedings of the ASME 2024 International Mechanical Engineering Congress and Exposition, IMECE 2024 (Vol. 88636), 17-21 November 2024, Portland, USA*. New York, USA, American Society of Mechanical Engineers. p. V005T07A035.

Yao, Y., Li, X. & Gao, L. (2024) A DQN-based memetic algorithm for energy-efficient job shop scheduling problem with integrated limited AGVs. *Swarm and Evolutionary Computation*. 87, 101544. https://doi.org/10.1016/j.swevo.2024.101544.

Yin, Y., Zhang, L., Shi, X., Wang, Y., Peng, J. & Zou, J. (2024). Improved Double Deep Q Network Algorithm Based on Average Q-Value Estimation and Reward Redistribution for Robot Path Planning. Computers, Materials&Continua, 81(2). https://doi.org/10.32604/cmc.2024.056791.

Zeng, D., Yan, T., Zeng, Z., Liu, H. & Guan, P. (2023) A Hyperparameter Adaptive Genetic Algorithm Based on DQN.*Journal of Circuits,Systems and Computers*.32(04),2350062. https://doi.org/10.1142/S0218126623500627.

Zhang, Y., Li, C., Zhang, G., Zhou, R. & Liang, Z. (2023) Research on the Local Path Planning for Mobile robot based on PRO-Dueling Deep Q-Network (DQN) Algorithm. *International Journal of Advanced Computer Science and Applications*. 14(8), 381-387. https://doi.org/10.14569/IJACSA.2023.0140842.

Zhu, C., Shen, J., Li, J., Zhang, X., Zhou, L., Zhu, D. & Zhang, Y., Zhao, J. & Sun, J. (2020) Robot Path Planning Method Based on Deep Reinforcement Learning. In: *2020 IEEE 3rd International Conference on Computer and Communication Engineering Technology (CCET), 14-16 August 2020, Beijing, China*. Piscataway, NJ, USA, IEEE. pp. 49-53.