sensors

MDPI

Article

Toward Energy-Efficient Routing of Multiple AGVs with Multi-Agent Reinforcement Learning

Xianfeng Ye [1], Zhiyun Deng [1], Yanjun Shi [2] and Weiming Shen [1,*]

[1] School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; xfye@hust.edu.cn (X.Y.); dengzy@hust.edu.cn (Z.D.)
[2] Department of Mechanical Engineering, Dalian University of Technology, Dalian 116023, China; syj@dlut.edu.cn
* Correspondence: shenwm@hust.edu.cn

Abstract: This paper presents a multi-agent reinforcement learning (MARL) algorithm to address the scheduling and routing problems of multiple automated guided vehicles (AGVs), with the goal of minimizing overall energy consumption. The proposed algorithm is developed based on the multi-agent deep deterministic policy gradient (MADDPG) algorithm, with modifications made to the action and state space to fit the setting of AGV activities. While previous studies overlooked the energy efficiency of AGVs, this paper develops a well-designed reward function that helps to optimize the overall energy consumption required to fulfill all tasks. Moreover, we incorporate the ϵ-greedy exploration strategy into the proposed algorithm to balance exploration and exploitation during training, which helps it converge faster and achieve better performance. The proposed MARL algorithm is equipped with carefully selected parameters that aid in avoiding obstacles, speeding up path planning, and achieving minimal energy consumption. To demonstrate the effectiveness of the proposed algorithm, three types of numerical experiments including the ϵ-greedy MADDPG, MADDPG, and Q-Learning methods were conducted. The results show that the proposed algorithm can effectively solve the multi-AGV task assignment and path planning problems, and the energy consumption results show that the planned routes can effectively improve energy efficiency.

Keywords: automated guided vehicles; multi-agent reinforcement learning; task assignment; path planning; energy consumption

Citation: Ye, X.; Deng, Z.; Shi, Y.; Shen, W. Toward Energy-Efficient Routing of Multiple AGVs with Multi-Agent Reinforcement Learning. *Sensors* **2023**, *23*, 5615. https://doi.org/10.3390/s23125615

Academic Editor: Alex Alexandridis

Received: 26 April 2023
Revised: 4 June 2023
Accepted: 11 June 2023
Published: 15 June 2023

1. Introduction

Automated guided vehicles (AGVs) are autonomous portable robots that navigate predetermined paths using various sensing technologies [1,2]. They play a vital role in modern manufacturing and logistics systems by facilitating the transportation of raw materials and finished products [3,4]. To optimize their performance, it is necessary to integrate the scheduling control system of AGVs with existing production management systems, such as manufacturing execution systems (MES), enterprise resource planning (ERP), warehouse management systems (WMS), logistics control systems (LCS), and Radio-Frequency IDentification (RFID) [5–8]. The AGV scheduling system receives request messages from the MES, dispatches AGVs to transport raw materials or finished products, and designs routes for AGVs to follow. However, the task assignment and path planning problem for multiple AGVs is challenging, as the number of decision variables and safety-related constraints grows significantly with the number of AGVs [9].

To address this issue, several optimization-based approaches have been proposed, such as integer programming [10–12], heuristic algorithms [13–15], and metaheuristics [7,16–19]. However, these methods have limitations in dealing with the dynamic and uncertain nature of the industrial environment. Therefore, machine learning-based methods, particularly reinforcement learning (RL), have emerged as a promising approach to solving the problem [20,21]. RL is a subfield of machine learning that involves an agent learning from its

interactions with the environment to maximize a cumulative reward signal. Multi-agent reinforcement learning (MARL) is an extension of RL that involves multiple agents learning to coordinate with each other to achieve a common objective [22]. MARL has been shown to be effective in solving complex problems that involve coordination and competition among multiple agents [8].

Compared with existing studies, the key contributions of this paper are summarized as follows: (1) We propose an ϵ-greedy MADDPG algorithm which is able to converge faster and achieve better performance during training by balancing exploration and exploitation. (2) Modifications are made to the action and state space to fit the setting of AGV activities, while a well-designed reward function is incorporated into the proposed algorithm that optimizes energy consumption while fulfilling all tasks. (3) The effectiveness of the proposed algorithm is demonstrated through numerical experiments, which show that it outperforms other methods in improving energy efficiency while addressing the multi-AGV task assignment and path planning problem.

The remainder of this paper is organized as follows. Section 2 presents a literature review about AGV scheduling algorithms with MARL. Section 3 presents the background of reinforcement learning, followed by a description of the proposed algorithm and model in Section 4. Section 5 presents the simulation results and analyses. Section 6 concludes this paper and discusses some open issues and future work.

2. Literature Review

Reinforcement learning (RL) has become a promising solution to the AGV scheduling and routing problems, while many researchers have carried out much pioneering work with the application of RL [23].

For example, the Markov decision process (MDP) formulation was combined with the asynchronous deep Q network (DQN) to solve the routing problem in real time and obtain high-quality solutions [24]. A decentralized framework for multiple AGVs was proposed in [25] for multi-task allocation with attention (MTAA), which uses the DNN network and the A3C and MTAA-DQN path planning techniques to achieve task assignment equilibrium. Aside from this application, RL was used to solve the routing problem in a bidirectional transport network for the purpose of avoiding deadlocks and obtaining collision-free trajectories [26]. The deep Q network (DQN) was used in [27] to learn a transportation strategy with breakpoint continuation and hierarchical feedback, which can calculate and further modify a transportation schedule in a short time to accommodate dynamic factors. That aside, the authors of [28] tried to teach a neural network to allocate transportation duties to AGVs and design routes for them in accordance with the rewards computed by the network. An enhanced DQN was suggested in [29] to find appropriate navigational approaches for certain current road circumstances, which limits the Q output of specific actions and incorporates their outcomes using calculations based on experience-based pooling.

Moreover, a state space filter was proposed in [30] to improve the negotiation rules between different agents that adjust their routes when probable collisions are identified. Li et al. [31] proposed a deep learning approach that concurrently addresses task assignment and path planning concerns, and it uses the Markov decision chain to formulate the challenge of finding the shortest path without running afoul of other AGVs. In addition, De Ryck et al. [1] gave a general overview of the control algorithms and methods applied to the first-generation and latest AGV systems. Xue et al. [32] used an RL approach to solve a multi-AGV flow-shop scheduling problem, where AGVs communicate comprehensive data about each machine's current state and running jobs. In other words, users are able to make decisions based on knowledge of the entire flow shop. Nagayoshi et al. [33] presented a decentralized autonomous strategy for controlling a large number of AGVs in response to ambiguous delivery requests, where the AGVs are equipped with transportation route plans that are intended to save travel time while avoiding collisions. Sierra-Garcia et al. [34] presented an intelligent hybrid control scheme that combines RL-based control (RLC) with

conventional PI regulators, where the RLC allows the AGVs to learn how to improve trajectory tracking adaptively.

As suggested in [35], the multi-agent reinforcement learning (MARL) policy is capable of (1) scaling to a large number of agents in a real-world setting with an offline response time within acceptable levels and (2) outperforming existing algorithms with lower path lengths and faster solution times. Takahashi et al. [36] provided a multi-agent deep deterministic policy gradient (MADDPG) approach for managing several AGVs using DRL, where simulated experiments demonstrated that the suggested method learns optimal or nearly optimal solutions from prior knowledge. Aside from that, several numerical tests were carried out in [8] to confirm the effectiveness of the RL method. The authors of [37] used the MARL method to deal with the increased flexibility and complexity introduced by the increased use of AGVs. In addition, Li et al. [38] proposed a reward-shaping technique based on the potential information field which offers stepwise incentives and implicitly directs the AGVs to various targets to address the problem of reward sparsity. Moreover, Lu et al. [24] presented a DRL technique to address the AGV routing issue, where the conflict vectors are created from the retrieved embeddings and then processed using the LSTM network.

From the above work, it can be seen that RL is very effective for solving the AGV scheduling and routing problem. However, the existing DQN algorithm in RL has some limitations, since it cannot solve continued questions directly. Moreover, it does not consider how to solve the path planning problem with the application of RL.

3. Background

3.1. Single-Agent Reinforcement Learning Model

RL is a framework for learning how an agent can take action in an environment to maximize a cumulative reward signal. This framework can be expressed as a system consisting of an agent and an environment, as illustrated in Figure 1 [39]. The environment produces information that describes the state of the system, while the agent interacts with the environment by observing the state and then selecting an action to perform. Subsequently, the environment accepts the action and transitions into the next state while returning a reward to the agent. This reward denotes the feedback signal from the environment, indicating whether it is beneficial for the agent to adopt a certain strategy at a certain step. The agent's objective is to learn a policy that maps states to actions to maximize the expected future cumulative reward. That is to say, the agent outputs the action A_t, observes the system's state S_t, and receives the reward R_t from the system.

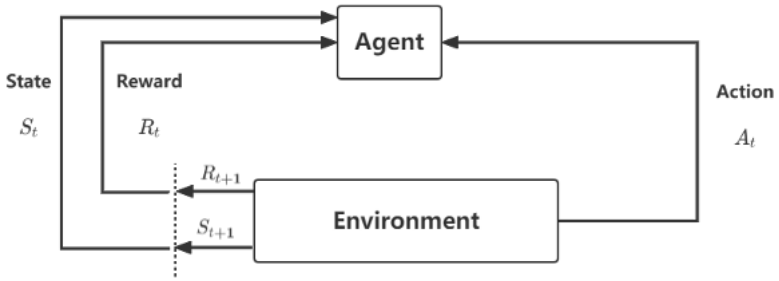


Figure 1. The single agent of the reinforcement learning system.

In the context of AGVs, the vehicle can be formulated as an agent, since it can capture information with onboard sensors and perform an action with its actuators. The environment can be the map of a manufacturing factory or an automated warehouse where the AGV operates. The action space of the AGV agent can be represented by go forward, turn left, turn right, and stopping operations, for example. During the training phase, the agent interacts with the environment until the terminal conditions are met. After that, the agent can learn to perform actions without human guidance to maximize its expected future reward in a certain state.

3.2. Multi-Agent Reinforcement Learning Model

In the MARL model, there are at least two agents existing in the same environment and interacting with each other as shown in Figure 2 [39]. We take into account the multi-agent Markov decision process extension known as Markov games [40]. A set of states S, action sets for each of N agents $A_1, \ldots, A_N$, a state transition function $T : S \times A_1 \times \ldots A_N - > P(S)$ which specifies the probability distribution over the possible next states, given the current state and actions for each agent, and a reward function for each agent that also depends on the overall state and actions of all agents $R_i : S \times A_1 \ldots A_N - > R$ define them. This means that all agents choose actions a_i simultaneously after watching the system's state s and receiving each agent's individual reward r_i. In the case of multiple AGVs in a warehouse, each AGV can be considered an agent, and the entire warehouse map can be viewed as the environment.

To enable effective cooperation and competition between AGVs, researchers have developed various MARL algorithms that can learn the best strategies for multiple agents in the same environment. For example, one approach is to use a centralized training and decentralized execution (CTDE) architecture in which a central controller learns a joint policy for all agents during training, and each agent executes its own policy during execution. This approach has been shown to be effective in scenarios where there is a strong interdependence between agents, such as in a convoy of AGVs transporting a large item. Another approach is to use independent reinforcement learning (IRL), in which each agent learns its own policy independently without any communication or coordination with other agents. This approach can be useful when the actions of different agents do not have a significant impact on each other, such as in scenarios where AGVs are used to transport different items to different locations. MARL has the potential to improve the efficiency and effectiveness of AGV systems in various industrial applications, and ongoing research in this area is expected to lead to even more sophisticated and effective algorithms in the future.

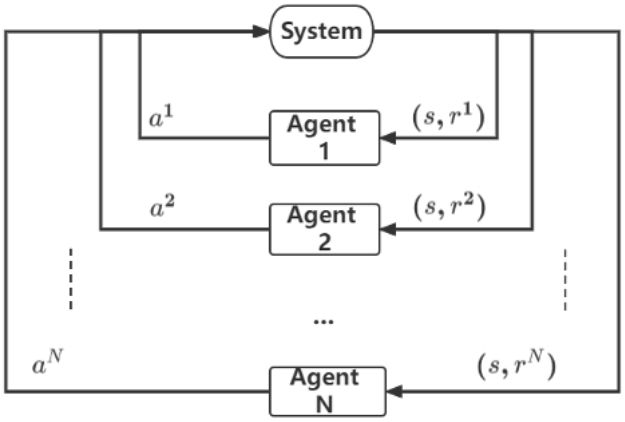


Figure 2. The multi-agent architecture in the reinforcement learning system.

4. Methodology

4.1. The Multi-Agent Deep Deterministic Policy Gradient (MADDPG) Algorithm

In this paper, we propose an energy-efficient scheduling and routing algorithm based on the multi-agent deep deterministic policy gradient (MADDPG) algorithm and the path planning D* Lite algorithm. The D* Lite algorithm's basic idea is to plan the global optimal path from the destination point to the beginning point based on available environmental information, treating the unknown portion as free space [41]. However, in this paper, we combine the D* Lite algorithm with energy consumption computation to optimize energy efficiency.

In RL, the deep deterministic policy gradient (DDPG) algorithm is a model-free, off-policy, and policy-based method suitable for solving such problems [42,43]. The DDPG

algorithm uses a deterministic policy, which means that when the policy and observed state are given, the action is uniquely determined. This is in contrast to classical RL algorithms, which use a stochastic policy that performs actions based on a probability distribution. DDPG follows the idea of fixing the target network that is used in the DQN algorithm, resulting in only two networks that need to be learned: the policy network and the value network [44,45]. In DDPG, each network is subdivided into a current network and a target network, and the updating process for these two networks is different. Under the actor-critic framework, the policy network is referred to as an actor network that outputs a deterministic action, while the value network is referred to as a critic network that fits the value function $Q_\pi(s, a)$. Multi-agent in DDPG, an extension of DDPG, means that decentralized agents learn a centralized critique based on their collective observations and actions, resulting in a multi-agent policy gradient algorithm. It generates learned policies that, during execution, only use local information (i.e., their own observations), does not require a differentiable model of the dynamics of the environment or any particular structure on the method of communication between agents, and is applicable to competitive or mixed interactions involving both physical and communicative behavior. The critic possesses additional knowledge about the practices of other agents, but the actor just has access to local information. Once trained, only locally based actors who work independently are used throughout the execution phase.

The estimated policy network of the actor is $\theta(s)$, where θ is the parameter of the neural network. The actor also has another target network that is used to update the value of the critic network. Both networks have the same structure and output corresponding actions, but the parameters within the neural networks are different. In terms of the critic network, there are also two networks: an estimation network and a target network. Both networks output the Q value of the current state, but they differ in terms of their input. For instance, the input of the critic's target network has two parameters, which are the observation of the current state and the action of the actor's target network output. In contrast, the input of the critic's estimation network is the action of the current actor's estimation network output.

The target network is used to calculate Q_{target}. The update of the value network is based on the gradient descent of the TD-error. The critic, which acts as a judge, does not initially know whether the actor's action is good enough and needs to learn step by step to provide accurate scoring. With the help of the value Q_π in the next moment, fitted by the target network and the actual gain r, we can obtain Q_{target}, which is then subtracted from the current Q to find the mean squared deviation, allowing us to construct the loss function. In terms of the policy network, its update is based on gradient ascent. Since the goal of the actor is to find an action A that maximizes the value Q of the output, optimizing the gradient of the policy network is for maximizing this Q value of the output of the value network. The loss function then adds a negative sign to facilitate minimizing the error.

The parameters of n agents are identified as $\theta = [\theta_1, \ldots, \theta_n]$, and the policies of n agents are identified as $\pi = [\pi_1, \ldots, \pi_n]$ [42]. Therefore, the accumulated reward for a certain agent i and the expected reward gradient for a deterministic policy μ_{θ_i} can be represented as follows:

$$J(\theta_i) = E_{s\sim\rho^\pi, a\sim\pi_{\theta_i}}[\sum_{t=0}^{\infty} \gamma^t r_{i,t}] \tag{1}$$

$$\mathcal{L}(\theta_i) = \frac{1}{s}\sum_j (y^j - Q_i^\mu(x^j, a_1^j, \ldots, a_N^j))^2 \tag{2}$$

$$\nabla_{\theta_i} J(\mu_i) = E_{x,a\sim D}[\nabla_{\theta_i}\mu_i(a_i|o_i)\nabla a_i Q_i^\mu(x, a_1, \ldots, a_n)|a_i = \mu_i(o_i)] \tag{3}$$

where o_i is the observation of the agent i, $x = [o_i, \ldots, o_n]$ is the observation value, $Q_i^\mu(x, a_1, \ldots, a_n)$ is the action and state function, $\nabla_{\theta_i}\mu_i(a_i|o_i)$ is the gradient of the policy network at θ_i, and $\nabla_{a_i} Q_i^\mu(x, a_1, \ldots, a_n)$ is the gradient of the value network at x and action sets $(a_1, \ldots, a_n)$.

4.2. Multi-Agent Model for AGV Operations

This paper presents a formal definition of the action space of the automated guided vehicle (AGV) agent, denoted by $a(v, \omega)$. The velocity of the AGV v is variable and can range from -1 m/s to 1 m/s, while the angular velocity ω is restricted to values between -1 rad/s and 1 rad/s. Consequently, the AGV agent is capable of performing five distinct actions, namely moving forward, moving backward, turning left, turning right, and halting. The reward function is defined as follows:

$$r_i = k_{i1} \times D_{position} + k_{i2} \times C_v \times v \times cos(\omega) + k_{i3} \times C_e \times (E_{target} - E_i) + k_{i4} \times C_{AGV} + k_{i5} \times C_{obstacles} \quad (4)$$

In the above equation, k_{i1}, k_{i2}, k_{i3}, k_{i4}, and k_{i5} are the weight parameters, while $D_{position}$ represents the reward value based on the current position relative to the previous position. A positive reward is given if the current position is closer to the destination than the previous position, and vice versa. This incentivizes the AGV to approach the target site, using the distance reward as guidance. The equation $D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ calculates the distance between the current position and the destination point. The value of the award will be negative if $D_{current}$ is greater than $D_{previous}$. The value of the award will be negative if there is an AGV collision. Let us define the previous and current position for AGV_i as $Pos1(x_0, y_0)$ and $Pos2(x_1, y_1)$ for AGV_j, respectively; that is, $Pos1(x_2, y_2)$ and $Pos2(x_3, y_3)$ if $\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2)} < 0.5$ are satisfied by $(y_1 - y_0) \times (y_3 - y_2) < 0$ or $(x_1 - x_0) \times (x_3 - x_2) < 0$, which means that if they travel in the opposite direction, then they will collide, and the reward value will be negative. The result of $\sqrt{((x_i - x_j)^2 + (y_i - y_j)^2)}$ is less than 0.1 if an AGV collides with an obstruction due to the line's narrow width. C_{AGV} and $C_{obstacles}$ are both defined as -50 when there is a collision; otherwise, they are 50.

The speed reward guides the AGV to complete the task with the least number of rotations and the most significant amount of linear speed achievable. C_v is the velocity coefficient and is used to scale this reward item. The third reward item is related to energy consumption, denoted by E_i, which is computed differently based on whether the AGV is stationary or in motion. When the AGV is not stationary, the energy consumption is computed as the average energy consumption per time step. Otherwise, the corresponding energy consumption is multiplied by a parameter factor of 0.3, which signifies that the AGV consumes less energy when in a stationary state compared with when it is in a normal driving state. The value of E_{target} is computed using the route path determined by the D* Lite algorithm, while C_e is the energy coefficient, which is used to fit this reward item with other items.

The last two reward items are related to collision avoidance and incentivize the AGV to avoid path conflicts with other AGVs or obstacles. These reward items are critical to ensuring the safe and efficient operation of the AGV. In addition, it is worth noting that the parameters in the reward function play a crucial role in numerical simulation and will be discussed further in another paper.

4.3. MADDPG with ϵ-Greedy

Exploration and exploitation are very prominent problems in reinforcement learning, and they are also the focus of determining whether the reinforcement learning system can obtain an optimal solution. Exploration would allow an agent to enhance its knowledge about its action, which may lead to long-term benefits. Improving the accuracy of action value estimation enables an agent to make more informed decisions. Exploitation uses the greedy action to acquire the greatest reward through exploiting the agent's action value estimates. However, being greedy may not lead to the greatest reward and in fact may lead to suboptimal results. While exploration may find more accurate action value estimates, exploitation may obtain more rewards. However, it is not possible to do both at the same time. Exploration is the right way to maximize the expected return at the present moment, while exploitation is the right way to maximize the total return in the long run. Unfortunately, in a certain state, the agent can only perform one action: either

exploration or exploitation. The two cannot be carried out at the same time, and thus this is the contradiction that accentuates the emphasis of reinforcement learning and how to balance exploration and exploitation.

The ϵ-greedy policy is a popular strategy for balancing exploration and exploitation [46,47]. This policy selects the best action with a probability $1-\epsilon$ and a random action with a probability ϵ. The parameter ϵ determines the degree of exploration versus exploitation, where a high value of ϵ results in more exploration and a low value of ϵ results in more exploitation [48]. However, the ϵ-greedy method has a limitation in that it selects random actions uniformly, even though certain actions may be better than others. To address this limitation, softmax policies have been proposed, which select random actions with probabilities proportional to their current values [49].In the context of AGV route selection, the ϵ-greedy policy can be used to balance exploration and exploitation by randomly selecting between exploration and exploitation. When one AGV explores its action, the other AGVs can exploit that action to their advantage. However, the optimal action for one AGV may not be the optimal action for all AGVs, as route conflicts may require different actions. Therefore, the challenge in AGV route selection is to find a balance between exploration and exploitation that maximizes the overall performance of the system. The ϵ-greedy policy is a straightforward strategy for balancing discovery and exploitation, where the parameter ϵ controls the degree of exploration versus exploitation. The algorithm of MADDPG with ϵ-greedy for AGVs is illustrated in Algorithm 1.

Algorithm 1: An algorithm of MADDPG with the *ϵ-greedy* policy for AGVs.

for j=1 to max-episode **do**
  Initialization of the parameters
  for t=1 to M **do**
    for i=1 to N **do**
      n= random number
      if $n < \epsilon$ **then**
        execute any action(a)
      else
        execute the action which maximizes $Q_t(a) with 1-\epsilon$
      end if
      $a_i = \mu_{\theta i}(o_i)+N_t$
      a=$(a_i, ..., a_N)$
      $r_i = k_{i1} \times D_{position} + k_{i2} \times C_v \times v \times cos(\omega) + k_{i3} \times C_e \times (E_{target} - E_i) + k_{i4} \times C_{AGV} + k_{i5} \times C_{obstacles}$
    end for
    for agent i=1 to N **do**
      $y^j = r_i^j + \gamma Q^{\mu\prime}{}_i(x'_j, a'_1, ...a'_N)$
      $\mathcal{L}(\theta_i) = \frac{1}{s}\sum_j (y^j - Q_i^\mu(x^j, a_1^j, ..., a_N^j))^2$
      $\nabla_{\theta i} J \simeq \frac{1}{S}\sum_j \theta_i \mu_i(o_i^j) a_i Q_i^\mu(x^j, a_1^j, ...a_i, ..., a_N^j)|a_i = \mu_i(o_i^j)$
    end for
    for i=1 to N **do**
      $\theta'_i < -\tau\theta_i + (1-\tau)\theta'_i$
    end for
  end for
end for

4.4. Three Algorithms in This Experiment

In this study, we focus on evaluating the performance of three popular reinforcement learning algorithms in a specific scenario. The algorithms we considered were Q-learning, MADDPG, and enhanced MADDPG with the epsilon-greedy policy, which are described below.

First, Q-learning is a widely used algorithm that employs off-policy reinforcement learning to maximize rewards. The algorithm updates a Q table that stores the expected

reward of each state-action pair. Although Q-learning is a model-independent technique, it can be prone to taking risks in real-world applications. Our study aims to investigate the strengths and limitations of Q-learning in the given scenario.

MADDPG, on the other hand, is a centralized, critic-based actor-critic approach that allows one to consider various reward functions. The algorithm has a critic for each agent, which increases the amount of data used in the learning process. We evaluated the performance of MADDPG and compared it with Q-learning to gain insights into the strengths and limitations of these two approaches.

Finally, we introduce enhanced MADDPG with the epsilon-greedy policy, which is a straightforward strategy for balancing exploration and exploitation. By randomly selecting between exploration and exploitation, the algorithm can achieve a balance between the two strategies. We compared the performance of enhanced MADDPG with the epsilon-greedy policy with that of Q-learning and MADDPG to evaluate the effectiveness of this approach in the given scenario.

Our study contributes to the existing literature on reinforcement learning by evaluating the performance of three popular algorithms in a specific scenario. The findings of this study can help researchers and practitioners select the most appropriate algorithm for a given problem and improve the overall performance of reinforcement learning algorithms.

5. Experiments and Results

5.1. Test Scenarios

The experimental test scenarios presented in this paper aim to evaluate the performance of an AGV-based system in a warehouse environment. The state of the AGV includes the position, velocity, and distance between the starting point and the final destination. The warehouse, as illustrated in Figure 3, has a total area of 82×66 m^2 and is equipped with a variety of facilities, including a power center, a repair center, and an office room for workers. The warehouse is divided into 39 racks, denoted by BLOCK, which are strictly off limits for AGVs.

The AGVs were programmed to stop at the park center when no tasks were assigned. Additionally, there were designated halt positions, such as A02 and B12, where the AGVs could temporarily pause while processing an order. There were three types of goods in this warehouse: delivered goods, transported goods, and relocated goods. Delivered goods were loaded at the door and unloaded at another position, whereas transported goods were moved from a specific rack to the door, and relocated goods were transferred from one rack to another. These tasks were challenging, as the AGVs had to navigate through the warehouse while avoiding obstacles, as shown in Figure 3.

To evaluate the effectiveness of the proposed algorithm, these tasks were selected as benchmark problems, and they are described in Table 1. The experimental scenarios aimed to examine the AGVs' ability to perform these transfer tasks efficiently and accurately. The proposed algorithm's performance was evaluated based on various metrics, including completion time, task efficiency, and AGV utilization. These experiments provided valuable insights into the AGV's performance in a real-world warehouse environment, and they may lead to improvements in AGV-based systems' efficiency and effectiveness.

5.2. Numerical Results

In this study, we analyzed the numerical results of the AGV system, which involved 30 AGVs and 3 types of goods categorized into 30 tasks, as illustrated in Table 1. Specifically, there were 12 tasks for delivered goods (Tasks 01–12), 12 tasks for transported goods (Tasks 13–24), and 6 tasks for relocated goods (Tasks 25–30). Each AGV was assigned 1 task, which required the 30 AGVs to complete all tasks simultaneously. For instance, AGV-01 was responsible for completing Task 01 by loading the cargo at the door and unloading it at position A24. The other tasks followed a similar pattern. Table 1 summarizes the three types of goods that needed to be processed. Two different types of software tools were used for the numerical simulations. A piece of the MADDPG algorithm was taken from [42]

and specifically altered for our needs to yield the transportation line for the AGVs with parameters in Table 2. Our unique Java-written AGV-scheduling program was used in the left part for visualization of transportation.

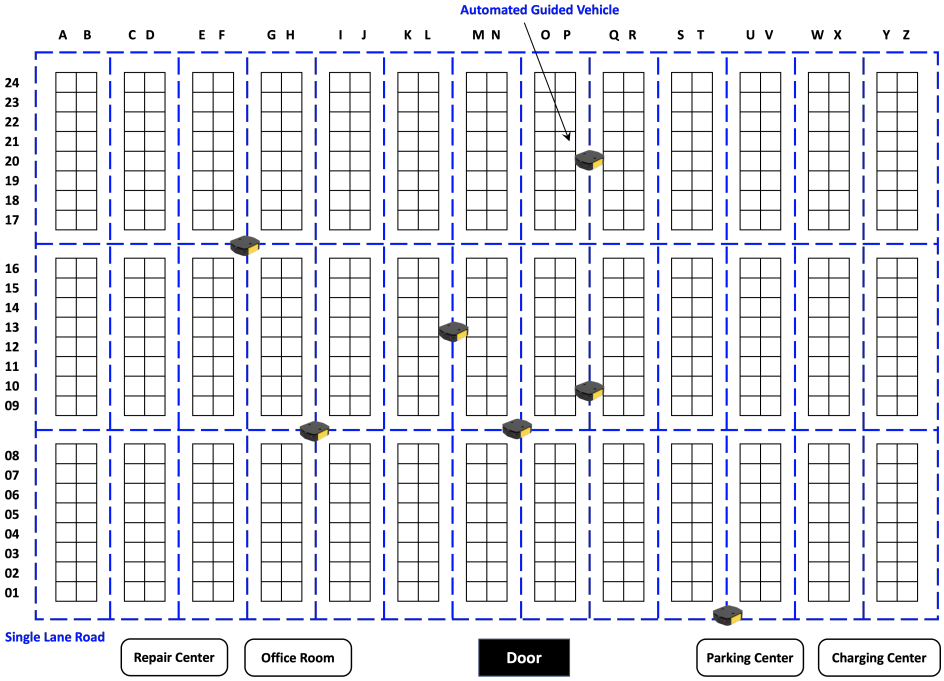


Figure 3. Map of the warehouse.

Figure 4 illustrates the convergence of the reward function in route selection based on the Q-learning technique. The function showed a gradual rise and eventual stabilization to a final state as the number of episodes increased. However, both the native multi-agent deep deterministic policy gradient (MADDPG) and ϵ-greedy MADDPG techniques converged more quickly to the steady state. When comparing the two MADDPG techniques, the ϵ-greedy strategy yielded better benefits. In the MADDPG algorithm, an ϵ value of 0.1 provided superior results to values of 0.01 and 0.05. This finding was consistent with most simulations of the ϵ-greedy approach, which demonstrated that an optimal value was attained around $\epsilon = 0.1$. However, determining the ideal value of ϵ for the current AGV environment requires further investigation.

Table 1. Thirty transfer tasks in three projects.

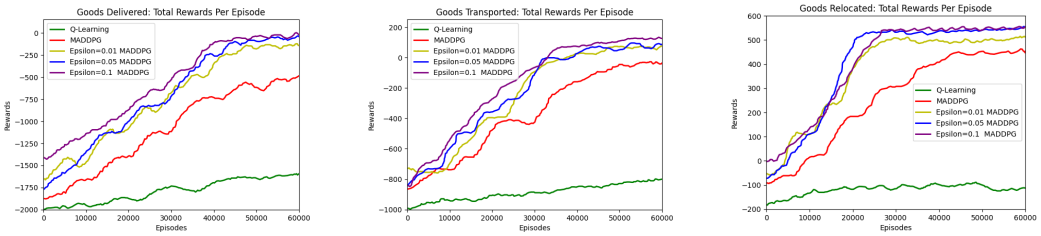
Transfer Type	Task List	Load Cargo Position	Unload Cargo Position
Goods Delivered	Task 01	Door	A24
	Task 02	Door	A17
	Task 03	Door	A16
	Task 04	Door	A09
	Task 05	Door	A08
	Task 06	Door	A01
	Task 07	Door	E24
	Task 08	Door	E17
	Task 09	Door	E16
	Task 10	Door	E09
	Task 11	Door	E08
	Task 12	Door	E01

Table 1. *Cont.*

Transfer Type	Task List	Load Cargo Position	Unload Cargo Position
Goods Transported	Task 13	I24	Door
	Task 14	I17	Door
	Task 15	I16	Door
	Task 16	I09	Door
	Task 17	I08	Door
	Task 18	I01	Door
	Task 19	M24	Door
	Task 20	M17	Door
	Task 21	M16	Door
	Task 22	M09	Door
	Task 23	M08	Door
	Task 24	M01	Door
Goods Relocated	Task 25	Q24	Z01
	Task 26	Q17	Z08
	Task 27	Q16	Z09
	Task 28	S24	W01
	Task 29	S17	W08
	Task 30	S16	W09

Table 2. Parameter settings for simulations.

Description	Notation and Value
Weight Parameters	$k_{i1} = 0.2, k_{i2} = 0.1, k_{i3} = 0.5, k_{i4} = 0.1, k_{i5} = 0.1$
Reward Value	$D_{position}$
Velocity Coefficient	C_v
Energy Consumption	E_i
Target Coefficient	E_{target}
Energy Coefficient	C_e
Collision Parameter between AGVs	C_{AGV}
Collision Parameter between AGV and Obstacle	$C_{Obstacle}$
Learning Rate	0.15
Discount Factor	0.99
α	0.01
β	0.01
γ	0.95
τ	0.01

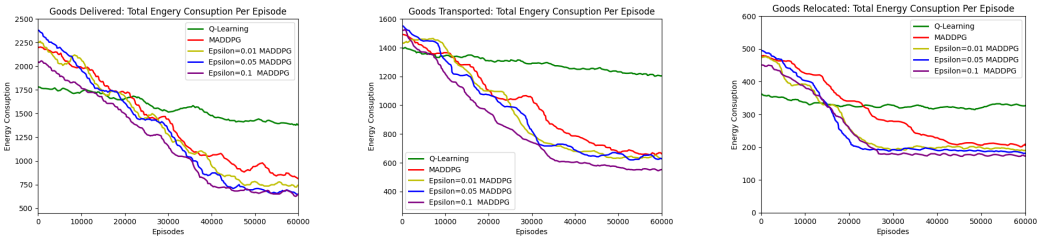


(**a**) Goods delivered. (**b**) Goods transported. (**c**) Goods relocated.

Figure 4. Three kinds of goods: rewards.

In this article, we used the AGV's energy consumption per second as a unit, the AGV's total energy once it arrived at its destination as an indicator, and the total energy of all AGV vehicles, as indicated in Figure 5. From this, it is clear that the route planned by Q-learning used a significant amount of energy, followed by the route planned by the

primeval MADDPG algorithm and the route planned by the ϵ-greedy MADDPG algorithm, which used the least amount of energy.



(**a**) Goods delivered. (**b**) Goods transported. (**c**) Goods relocated.

Figure 5. Three kinds of goods: energy consumption.

The path of the AGV is shown in Figure 6. The shelves' green tint indicates that they can hold stock. The shelf's yellow tint shows that there is merchandise there. The shelves' crimson tint indicates that they are completely full. When an AGV is in transit, it is shown by a yellow AGV, and when it is idle, it is indicated by a green AGV. From the figure, we can see that due to route conflicts, the AGV transporting items stopped for a period of time in the middle so that the transport AGV could pass smoothly. At the same time, some AGVs chose another path to avoid route conflicts.

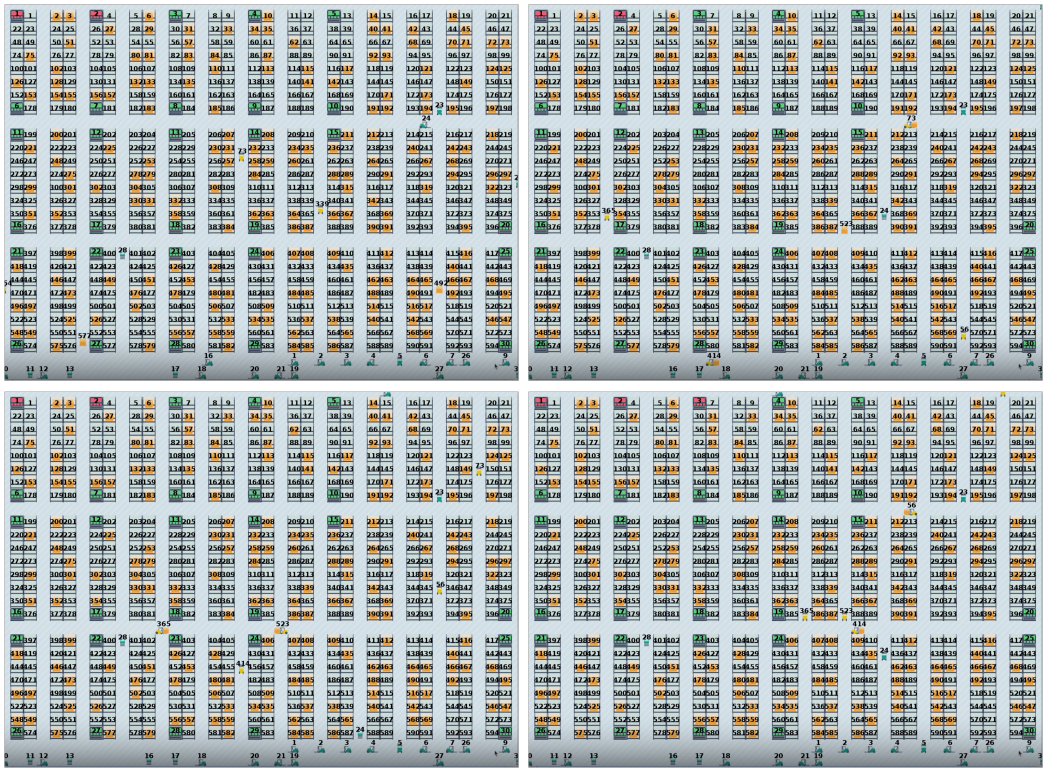


Figure 6. Illustration of AGVs' routes.

Furthermore, the simulation results indicate that energy consumption should be taken into account when selecting the path for the AGVs. It was observed that the rewards fluctuated more when the AGVs traversed multiple obstacles, whereas the rewards were more stable when the AGVs encountered fewer obstacles.

6. Conclusions

In this paper, we proposed a multi-agent reinforcement learning (MARL) algorithm to address the problem of scheduling and routing multiple AGVs with the aim of minimizing

the overall energy consumption. The proposed algorithm was built upon the multi-agent deep deterministic policy gradient (MADDPG) algorithm, with modifications made to the action space and state space to suit the specific activities of AGVs. While prior studies have overlooked the energy efficiency of AGVs, we designed a reward function that helps to optimize the overall energy consumption of the system during task completion.

To enhance the performance of our proposed MARL algorithm, we selected suitable parameters that facilitated obstacle avoidance, speedy path planning, and energy conservation. We conducted numerical experiments to evaluate the performance of the algorithm, and the results demonstrate its effectiveness in solving the multi-AGV task assigning and path planning problem. This leads to a reduction in the total energy consumption of AGV transportation, which increases as the number of operational AGVs increases.

During the simulation, the outcomes of the simulation were influenced by a range of parameters. We adopted identical settings for all AGVs in the reward function, although the parameters for each AGV's reward function were slightly different due to variations in position and the actions taken in response to those positions. These differences arose due to the unique nature of each AGV's activities. For instance, an AGV tasked with transporting heavier goods may use more energy than one carrying lighter items. Additionally, we observed that the optimal ϵ value for the MADDPG model in the current environment has not yet been established using the ϵ-greedy approach.

In our future work, we plan to develop an end-to-end learning framework that focuses on the direct control inputs of AGVs, rather than utilizing the desired velocity and angular velocity as the action space. We anticipate that this approach will enhance the performance of our proposed algorithm even further. By integrating a more comprehensive understanding of AGVs' behaviors and their interactions with the environment, the algorithm will be better equipped to adapt to the diverse requirements of various tasks and settings. Ultimately, we are confident that our research will contribute significantly to the advancement of energy-efficient AGV systems, which are of growing importance in contemporary logistics and transportation applications.

Author Contributions: Conceptualization, X.Y. and W.S.; methodology, X.Y., Z.D., Y.S. and W.S.; software, X.Y.; validation, X.Y. and Z.D.; formal analysis, X.Y.; investigation, X.Y., Z.D., Y.S. and W.S.; resources, W.S.; data curation, X.Y.; writing—original draft preparation, X.Y.; writing—review and editing, Z.D., Y.S. and W.S.; visualization, X.Y.; supervision, W.S.; project administration, Z.D.; funding acquisition, W.S. All authors have read and agreed to the published version of the manuscript.

Funding: The work presented in this paper was supported in part by the National Key R&D Program of China under Grant No.2022YFE0114200.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. De Ryck, M.; Versteyhe, M.; Debrouwere, F. Automated guided vehicle systems, state-of-the-art control algorithms and techniques. *J. Manuf. Syst.* **2020**, *54*, 152–173. [CrossRef]
2. Shi, Y.; Han, Q.; Shen, W.; Zhang, H. Potential applications of 5G communication technologies in collaborative intelligent manufacturing. *IET Collab. Intell. Manuf.* **2019**, *1*, 109–116. [CrossRef]
3. Yoshitake, H.; Kamoshida, R.; Nagashima, Y. New automated guided vehicle system using real-time holonic scheduling for warehouse picking. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1045–1052. [CrossRef]
4. Shen, W.; Yang, C.; Gao, L. Address business crisis caused by COVID-19 with collaborative intelligent manufacturing technologies. *IET Collab. Intell. Manuf.* **2020**, *2*, 96–99. [CrossRef]
5. Ahmed, S.U.; Affan, M.; Raza, M.I.; Hashmi, M.H. Inspecting Mega Solar Plants through Computer Vision and Drone Technologies. In Proceedings of the 2022 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 12–13 December 2022; pp. 18–23; IEEE: Piscataway, NJ, USA, 2022.
6. Alzahrani, A.; Sajjad, K.; Hafeez, G.; Murawwat, S.; Khan, S.; Khan, F.A. Real-time energy optimization and scheduling of buildings integrated with renewable microgrid. *Appl. Energy* **2023**, *335*, 120640. [CrossRef]

7. Xin, J.; Wei, L.; D'Ariano, A.; Zhang, F.; Negenborn, R. Flexible time–space network formulation and hybrid metaheuristic for conflict-free and energy-efficient path planning of automated guided vehicles. *J. Clean. Prod.* **2023**, *398*, 136472. [CrossRef]
8. Hu, H.; Yang, X.; Xiao, S.; Wang, F. Anti-conflict AGV path planning in automated container terminals based on multi-agent reinforcement learning. *Int. J. Prod. Res.* **2023**, *61*, 65–80. [CrossRef]
9. Lian, Y.; Yang, Q.; Xie, W.; Zhang, L. Cyber-physical system-based heuristic planning and scheduling method for multiple automatic guided vehicles in logistics systems. *IEEE Trans. Ind. Inform.* **2020**, *17*, 7882–7893. [CrossRef]
10. Goli, A.; Tirkolaee, E.B.; Aydın, N.S. Fuzzy integrated cell formation and production scheduling considering automated guided vehicles and human factors. *IEEE Trans. Fuzzy Syst.* **2021**, *29*, 3686–3695. [CrossRef]
11. Hu, H.; Chen, X.; Wang, T.; Zhang, Y. A three-stage decomposition method for the joint vehicle dispatching and storage allocation problem in automated container terminals. *Comput. Ind. Eng.* **2019**, *129*, 90–101. [CrossRef]
12. Yue, L.; Fan, H.; Ma, M. Optimizing configuration and scheduling of double 40 ft dual-trolley quay cranes and AGVs for improving container terminal services. *J. Clean. Prod.* **2021**, *292*, 126019. [CrossRef]
13. Fransen, K.; van Eekelen, J. Efficient path planning for automated guided vehicles using A*(Astar) algorithm incorporating turning costs in search heuristic. *Int. J. Prod. Res.* **2023**, *61*, 707–725. [CrossRef]
14. Nishi, T.; Akiyama, S.; Higashi, T.; Kumagai, K. Cell-based local search heuristics for guide path design of automated guided vehicle systems with dynamic multicommodity flow. *IEEE Trans. Autom. Sci. Eng.* **2019**, *17*, 966–980. [CrossRef]
15. Kabir, Q.S.; Suzuki, Y. Comparative analysis of different routing heuristics for the battery management of automated guided vehicles. *Int. J. Prod. Res.* **2019**, *57*, 624–641. [CrossRef]
16. Zhong, M.; Yang, Y.; Dessouky, Y.; Postolache, O. Multi-AGV scheduling for conflict-free path planning in automated container terminals. *Comput. Ind. Eng.* **2020**, *142*, 106371. [CrossRef]
17. Zou, W.Q.; Pan, Q.K.; Meng, T.; Gao, L.; Wang, Y.L. An effective discrete artificial bee colony algorithm for multi-AGVs dispatching problem in a matrix manufacturing workshop. *Expert Syst. Appl.* **2020**, *161*, 113675. [CrossRef]
18. Xiao, X.; Pan, Y.; Lv, L.; Shi, Y. Scheduling multi–mode resource–constrained tasks of automated guided vehicles with an improved particle swarm optimization algorithm. *IET Collab. Intell. Manuf.* **2021**, *3*, 93–104. [CrossRef]
19. Xie, J.; Gao, L.; Peng, K.; Li, X.; Li, H. Review on flexible job shop scheduling. *IET Collab. Intell. Manuf.* **2019**, *1*, 67–77. [CrossRef]
20. Hu, H.; Jia, X.; He, Q.; Fu, S.; Liu, K. Deep reinforcement learning based AGVs real-time scheduling with mixed rule for flexible shop floor in industry 4.0. *Comput. Ind. Eng.* **2020**, *149*, 106749. [CrossRef]
21. Melesse, T.Y.; Di Pasquale, V.; Riemma, S. Digital Twin models in industrial operations: State-of-the-art and future research directions. *IET Collab. Intell. Manuf.* **2021**, *3*, 37–47. [CrossRef]
22. Zhou, T.; Tang, D.; Zhu, H.; Zhang, Z. Multi-agent reinforcement learning for online scheduling in smart factories. *Robot. Comput.-Integr. Manuf.* **2021**, *72*, 102202. [CrossRef]
23. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Pearson Education Limited: Berkeley, CA, USA , 2016.
24. Lu, C.; Long, J.; Xing, Z.; Wu, W.; Gu, Y.; Luo, J.; Huang, Y. Deep Reinforcement Learning for Solving AGVs Routing Problem. In Proceedings of the International Conference on Verification and Evaluation of Computer and Communication Systems, Xi'an, China, 26–27 October 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 222–236.
25. Yin, Z.; Liu, J.; Wang, D. Multi-AGV Task allocation with Attention Based on Deep Reinforcement Learning. *Int. J. Pattern Recognit. Artif. Intell.* **2022**, *36*, 2252015. [CrossRef]
26. Chujo, T.; Nishida, K.; Nishi, T. A Conflict-Free Routing Method for Automated Guided Vehicles Using Reinforcement Learning. In Proceedings of the International Symposium on Flexible Automation, Virtual, Online, 8–9 July 2020; American Society of Mechanical Engineers: New York, NY, USA, 2020; Volume 83617, p. V001T04A001.
27. Yan, J.; Liu, Z.; Zhang, T.; Zhang, Y. Autonomous decision-making method of transportation process for flexible job shop scheduling problem based on reinforcement learning. In Proceedings of the 2021 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE), Chongqing, China, 9–11 July 2021; pp. 234–238.
28. Zhang, H.; Luo, J.; Lin, X.; Tan, K.; Pan, C. Dispatching and Path Planning of Automated Guided Vehicles based on Petri Nets and Deep Reinforcement Learning. In Proceedings of the 2021 IEEE International Conference on Networking, Sensing and Control (ICNSC), Xiamen, China, 3–5 December 2021; Volume 1, pp. 1–6.
29. Liu, H.; Hyodo, A.; Akai, A.; Sakaniwa, H.; Suzuki, S. Action-limited, Multimodal Deep Q Learning for AGV Fleet Route Planning. In Proceedings of the 5th International Conference on Control Engineering and Artificial Intelligence, Sanya, China, 14–16 January 2021; pp. 57–62.
30. Nagayoshi, M.; Elderton, S.; Sakakibara, K.; Tamaki, H. Adaptive Negotiation-rules Acquisition Methods in Decentralized AGV Transportation Systems by Reinforcement Learning with a State Space Filter. In Proceedings of the International Conference on Artificial Life and Robotics, ICAROB 2017, Miyazaki, Japan, 19–22 January 2017.
31. Li, M.P. Task Assignment and Path Planning for Autonomous Mobile Robots in Stochastic Warehouse Systems. Ph.D. Thesis, Rochester Institute of Technology, Rochester, NY, USA, 2021.
32. Xue, T.; Zeng, P.; Yu, H. A reinforcement learning method for multi-AGV scheduling in manufacturing. In Proceedings of the 2018 IEEE International Conference on Industrial Technology (ICIT), Lyon, France, 19–22 February 2018; pp. 1557–1561.
33. Nagayoshi, M.; Elderton, S.J.; Sakakibara, K.; Tamaki, H. Reinforcement Learning Approach for Adaptive Negotiation-Rules Acquisition in AGV Transportation Systems. *J. Adv. Comput. Intell. Intell. Inform.* **2017**, *21*, 948–957. [CrossRef]

34. Sierra-Garcia, J.E.; Santos, M. Combining reinforcement learning and conventional control to improve automatic guided vehicles tracking of complex trajectories. *Expert Syst.* **2022**, e13076. [CrossRef]
35. Zhang, Y.; Qian, Y.; Yao, Y.; Hu, H.; Xu, Y. Learning to cooperate: Application of deep reinforcement learning for online AGV path finding. In Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, Auckland, New Zealand, 9–13 May 2020; pp. 2077–2079.
36. Takahashi, K.; Tomah, S. Online optimization of AGV transport systems using deep reinforcement learning. *Bull. Netw. Comput. Syst. Softw.* **2020**, *9*, 53–57.
37. Popper, J.; Yfantis, V.; Ruskowski, M. Simultaneous production and agv scheduling using multi-agent deep reinforcement learning. *Procedia CIRP* **2021**, *104*, 1523–1528. [CrossRef]
38. Li, M.; Guo, B.; Zhang, J.; Liu, J.; Liu, S.; Yu, Z.; Li, Z.; Xiang, L. Decentralized Multi-AGV Task Allocation based on Multi-Agent Reinforcement Learning with Information Potential Field Rewards. In Proceedings of the 2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS), Denver, CO, USA, 4–7 October 2021; pp. 482–489.
39. Zhang, K.; Yang, Z.; Başar, T. Multi-agent reinforcement learning: A selective overview of theories and algorithms. In *Handbook of Reinforcement Learning and Control*; Springer: Cham, Switzerland, 2021 ; pp. 321–384.
40. Littman, M.L. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*; Elsevier: Amsterdam, The Netherlands, 1994; pp. 157–163.
41. Koenig, S.; Likhachev, M. Fast replanning for navigation in unknown terrain. *IEEE Trans. Robot.* **2005**, *21*, 354–363. [CrossRef]
42. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In Proceedings of the Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 6–9 December 2017.
43. Mordatch, I.; Abbeel, P. Emergence of Grounded Compositional Language in Multi-Agent Populations. *arXiv* **2017**, arXiv:1703.04908.
44. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic policy gradient algorithms. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 387–395.
45. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
46. Wunder, M.; Littman, M.L.; Babes, M. Classes of multiagent q-learning dynamics with epsilon-greedy exploration. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 1167–1174.
47. Dann, C.; Mansour, Y.; Mohri, M.; Sekhari, A.; Sridharan, K. Guarantees for epsilon-greedy reinforcement learning with function approximation. In Proceedings of the International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022; pp. 4666–4689.
48. Wadhwania, S.; Kim, D.K.; Omidshafiei, S.; How, J.P. Policy distillation and value matching in multiagent reinforcement learning. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 8193–8200.
49. Afsar, M.M.; Crump, T.; Far, B. Reinforcement learning based recommender systems: A survey. *Acm Comput. Surv.* **2022**, *55*, 1–38. [CrossRef]