Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4384954, 12 pages
https://doi.org/10.1155/2022/4384954

Research Article

Supervised Reinforcement Learning for ULV Path Planning in Complex Warehouse Environment

Shoulin Li and Weiya Guo

Qingdao Agricultural University, Qingdao, China

Correspondence should be addressed to Weiya Guo; weiya_guo@126.com

Received 31 July 2022; Revised 19 September 2022; Accepted 21 September 2022; Published 14 October 2022

Academic Editor: Lei Liu

The rapid development of the logistics industry leads to an urgent need for intelligent equipment to improve warehouse transportation efficiency. Recent advances in unmanned logistics vehicles (ULVs) make them particularly important in smart warehouses. However, the complex warehouse environment poses a significant challenge to ULV transportation path planning. Multiple ULVs need to transport cargoes with good coordination ability to overcome the low efficiency of a single ULV. The ULVs also need to interact with the environment to achieve optimal path planning with obstacle avoidance. In this paper, we propose a supervised deep reinforcement learning (SDRL) approach for logistics warehouses that enables autonomous ULVs path planning for cargo transportation in a complex environment. The proposed SDRL approach is featured by (1) designing the supervision module to imitate the behaviors of experts and thus improve the coordination ability of multiple ULVs, (2) optimizing the generator of the imitation learning based on the proximal policy optimization to boost the learning performance, and (3) developing the policy module via deep reinforcement learning to avoid obstacles when navigating the ULVs in warehouse environments. The experiments over dynamic and fixed-point warehouse environments show that the proposed SDRL approach outperforms its rivals regarding average reward, training speed, task completion rate, and collision times.

1. Introduction

The past few years have witnessed the emerging applications of mobile edge computing in smart industries with the convergence of sensing, communication, and computing [1–3]. With the rapid development of the intelligent logistics industry, smart devices are introduced in smart warehouses [4, 5]. Inspired by the success of unmanned vehicles in the field of agriculture [6], object detection [7], and IoT [8], unmanned logistics vehicles (ULVs) are employed to transport goods in a warehouse environment [9]. Since the ULVs need to be driven autonomously, path planning is of great importance in improving transportation efficiency [10].

Some warehouses adopt a fixed path solution for simplicity, but it is vulnerable to complex warehouse environments due to the inflexible movement of the ULVs. And the low efficiency of a single unmanned logistics vehicle makes it difficult to satisfy the requirements of intelligent warehouses. Moreover, in a real logistics warehouse environment, multiple ULVs are often needed to cooperate to complete transportation tasks. Therefore, it is worth designing the promising approach to make multiple unmanned vehicles cooperatively carry out the logistics and transportation tasks of the entire warehouse. For example, Faigl et al. model the multigoal path planning as a generalized traveling salesman problem with neighborhoods and design a feasible solution via heuristic algorithms [11]. Zuo et al. combine the artificial fish swarm algorithm and particle swarm optimization algorithm to address multiagent cooperative work and path planning problem [12]. Hu et al. propose a multiobjective optimization approach based on the COLREGs and Hi-NDS rules for path planning of autonomous surface vehicles [13]. Zhao et al. focus on the software-defined vehicular networks and propose a prediction-based temporal graph routing algorithm [14] and an intelligent digital twin-based hierarchical routing scheme [15]. Recently, reinforcement learning (RL) [16] has attracted increasing interest in multiagent collaboration and path planning problems.

Wang et al. map the raw sensory measurements of unmanned aerial vehicles (UAVs) into control signals for autonomous navigation based on the RL framework, which enables the UAVs to execute navigation tasks in large-scale complex environments [17]. Semnani et al. focus on the problem of distributed motion planning in dense and dynamic environments and develop a hybrid algorithm by integrating the advantages of deep reinforcement learning (DRL) and force-based motion planning [18]. In [19], the proximal policy optimization (PPO) is utilized to address the multiagent formation control with obstacle avoidance. Phiboonbanakit et al. develop a hybrid optimization model via RL and a complementary tree-based regression method to solve the vehicle routing problem in transportation logistics [20]. An improved Dyna-Q algorithm is proposed to deal with the mobile robot path planning in an unknown environment [21], in which the action-selection strategy, ε -greedy policy, and heuristic reward function and actions are utilized to enhance the performance. The RL-based approaches provide practical solutions to the path planning problem of unmanned vehicles. However, they fail to perform well in complex warehouse environments due to the inability to respond to dynamic changes in complex warehouse environments adaptively. Therefore, developing an optimal path planning approach in a complex warehouse environment is urgently in demand.

In this paper, we propose a supervised deep reinforcement learning approach for the ULVs path planning, termed SDRL for short, which enables the multiple ULVs to complete the delivery task via interactive cooperation. By introducing imitation learning, the proposed SDRL approach imitates the behavior of experts through the positive guidance of the expert data, making the multiple ULVs cooperate and identify task targets quickly. Also, the generator of imitation learning is optimized by the PPO to enhance the learning performance of the SDRL. In addition, the policy module based on DRL is designed to offer an optimization strategy for ULVs' movement with obstacle avoidance by capturing the feedback from the warehouse environment.

The remainder of this paper is organized as follows. Section 2 discusses the proposed SDRL approach, including the problem definition and challenges and the SDRL model. Section 3 gives the experimental results and analysis, which is followed by the conclusion in Section 4.

2. The Proposed Approach

This section details the problem definition and challenges and the SDRL model with the supervision and policy modules.

2.1. Problem Definition and Challenges. The path planning of the ULVs is greatly affected by the environment in a complex warehouse. In a warehouse with a fixed environment, for example, the ULVs perform the picking task through a manually preset path, which leads to increased labor costs and weak robustness. And rigid predesigned paths make the ULVs not accomplish cargo transportation tasks or handle emergencies in a dynamic warehouse environment. In addition, the coordination of multiple ULVs involves the problem of scheduling and avoiding collisions with obstacles or other ULVs.

The path planning of multiple ULVs in warehouse environment can be described by a turple $<\mathbf{s}, \mathbf{a}, e, w, \mathfrak{R}>$, where $\mathbf{s} = (s_1, \cdots, s_n)$ is the observations of n ULVs, $\mathbf{a} = (a_1, \cdots, a_n)$ is the actions, e is the warehouse environment, w is the rewards during the task processing, and $\mathfrak{R}$ means the optimization model. The proposed SDRL aims to learn a value function $V_c^{\rho}(\mathbf{s}, \mathbf{a}; \theta)$ that enables multiple ULVs to achieve the maximum rewards when completing the task of cargo transportation, that is

$$\underset{\mathbf{a}\sim\rho}{\operatorname{argmax}} V_c^{\rho}(\mathbf{s}, \mathbf{a}; \theta), \tag{1}$$

where $\boldsymbol{a} \sim \rho$ and θ means the parameters of the value function.

Based on the definition above, there are several challenges when designing the SDRL model. First, the logistics warehouse is more complex than the conventional reinforcement learning training environment, which makes the path planning of multiple ULVs more difficult. The DRL cannot deal with the dynamic changes of the entire environment adaptively. Secondly, complex warehouse space interferes with ULVs' task recognition, which causes the ULVs to stagnate in the corner due to the long-term inability to obtain positive rewards during the training process. It is challenging for the agent to learn the task target quickly and efficiently through the DRL. Therefore, it is necessary to introduce the supervised information to provide positive guidance for pretraining and thus enable the ULVs can quickly identify task targets. In addition, when multiple ULVs perform tasks together in the same logistics warehouse environment, the difficulty of path planning is further increased due to the expansion of the joint action space.

2.2. The SDRL Model. To address the challenges described above, we employ the DRL to cope with dynamic changes in a complex warehouse environment. In the pretraining process, positive guidance should be introduced to teach the ULVs how to accomplish a task, just like teaching kids to imitate our behavior to perform an assignment. Therefore, we design a supervised DRL model by integrating the merits of the DRL and imitation learning. Figure 1 shows the architecture of the proposed SDRL model which consists of the supervision module and policy modules. Given the recorded successful path data as expert data, the supervision module offers internal rewards by imitating the expert behaviors. The ULVs interact with the environment to generate external rewards, and the policy module combines the internal and external rewards to provide optimization strategies for path planning.

2.2.1. The Supervision Module. To offer positive guidance to help the ULVs identify task targets, we design the supervision module based on generative adversarial imitation learning (GAIL) [22]. The GAIL is to compare the imitation data with the expert data through the generative adversarial

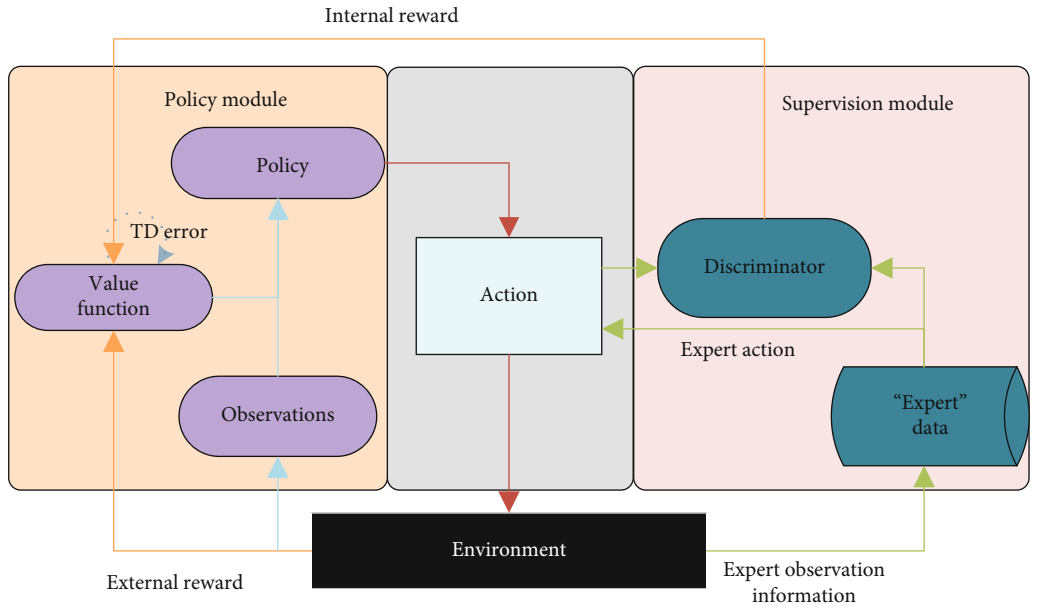

Figure 1: The architecture of the proposed SDRL model.

network (GAN) [23, 24] so that the agent can learn the policy directly from the expert data.

Given the generator $\mathscr{G}$ and the discriminator $\mathscr{D}$, the network calculates the loss function between the imitation data generated by $\mathscr{G}$ and the expert data. Thus, a well-trained generator can generate imitation data with the same characteristics as the expert data. According to the study in [25], the PPO-based generator performs well, so we optimize the GAIL by using the PPO as the generator, and the value function of the discriminator $\mathscr{D}$ can be described by

$$\mathbb{E}_{\rho}[\log(D(\mathbf{s},\mathbf{a}))] + \mathbb{E}_{\rho_E}[\log(1 - D(\mathbf{s},\mathbf{a}))] - \gamma H(\rho), \quad (2)$$

where D and ρ are approximated by the discriminator function $D_{\mu} : \mathscr{S} \times \mathscr{A} \longrightarrow (0,1)$ with the weight μ, and the policy ρ_{θ}, respectively. ρ_E means the expert policy, $H(\rho)$ represents the causal entropy of ρ, and γ is the discount factor of H. The optimization of the value function is achieved by performing gradient boosting on μ by the optimizer in the discriminator and gradient descent on θ by the policy in the policy module.

2.2.2. The Policy Module. In addition to imitating expert data, the proposed SDRL model reacts to dynamic environmental feedback and optimizes the ULVs generation path continuously. The policy module is designed based on deep reinforcement learning, which offers an optimization strategy for the ULVs' movement in a complex warehouse environment.

The proposed policy model consists of a decision maker and a value function. The decision maker generates actions based on feedback, and the value function processes the collected internal and external rewards. Given the policy ρ, the value function of the policy module is given by

$$V_c^{\rho}(\mathbf{s},\mathbf{a};\theta) = \mathbb{E}\left[w_{\sigma}(\mathbf{s},\mathbf{a}) + \kappa \mathbb{E}_{\mathbf{a}' \sim \rho}\left[V_c^{\rho}\left(\mathbf{s}', a'\right)\right]\right], \quad (3)$$

where θ is the parameter of function V_c^{ρ}, $\kappa \in (0,1]$ is the discount factor for the rewards, and w_{σ} is defined by

$$w_{\sigma}(\mathbf{s},\mathbf{a}) = \eta \cdot w_{\text{in}}(\mathbf{s},\mathbf{a}) + (1-\eta) \cdot w_{\text{ex}}(\mathbf{s},\mathbf{a}), \quad (4)$$

where $\eta \in (0,1)$ is the confidence weight of the expert data, w_{ex} is the reward function from the external environment, and w_{in} is the reward function of the imitation learning model, which is used to measure the similarity between the imitation data and the expert data.

Algorithm 1 gives the training process of the proposed SDRL model. During the training, the SDRL model recursively optimizes the discriminator $\mathscr{D}$ in the supervision module to reduce the difference between the imitation data and the expert data. With the rewards of internal and external, the value function V_c^{ρ} of the policy module can be updated. Afterward, the policy ρ is updated to achieve the highest cumulative value while completing the given task. Based on this, a well-trained model enables the multiple ULVs to accomplish the cargo path planning in warehouse environments.

3. Experimental Results and Analysis

In this section, we first present the experimental configuration, including the simulation environment, parameter setting, and evaluation metrics. Then, the experimental results are discussed to demonstrate the effectiveness and efficiency of the proposed approach.

3.1. Experimental Configuration. In this paper, we employ the Unity3D simulation platform to build the warehouse environment for performance evaluation. As shown in Figure 2, the white balls represent the target cargoes, the green squares represent the ULVs, and the dark grey rectangles represent the obstacles. Based on the path optimized by the proposed

Input: Expert data, initial parameters μ and θ;
for episode $i = 1$ to ψ **do**
  Update the discriminator $\mathcal{D}$ by ascending the stochastic gradient;
  Update the internal rewards w_{in} and external rewards w_{ex};
  Update the value function V_c^{ρ} by w_{σ};
  Update the policy ρ of the DRL by V_c^{ρ};
end

Algorithm 1: The Training Procedure of the SDRL.

SDRL model, multiple ULVs aim to efficiently complete the task of picking up cargoes with obstacle avoidance.

Table 1 summarizes the external rewards configuration used for setting the model's environmental scores. Considering that the ULVs have bumpers and the walls have sponges or protectors, we set a low negative value for the ULV hitting the wall. Since the collision of the ULVs with each other may cause damages to the transported cargoes, the penalty is twice that of the wall collision. In addition, the rewards from the target can motivate the ULVs to learn the target task quickly, so we set positive rewards for the collected cargoes and the ULVs approaching the target. Similarly, for the ULV to achieve the goal in the shortest path, we set a negative reward for each step of the ULVs, so that the ULVs can take the least steps to accomplish the task under negative feedback. When all tasks are completed, the ULV will receive a positive reward.

Four existing algorithmic models are included in the experiment as performance comparison baselines, namely, the GAIL model [22], the PPO network model [26], the soft actor-critic (SAC) network model [27], and the behavior cloning (BC) network model [28]. In the experiments, we utilize the same parameter setting for each model, including the maximum number of steps per agent, the reward value, and the simulation environment.

During the training process, we set the maximum number of steps per agent at 100,000 because the ULVs could not find the task direction and would stagnate in the corner in the early training process. The simulation environment will be reset if all target cargoes are picked up. In addition, we build ten replicates of the simulation environment so that each agent can learn from all the replicates simultaneously, thus significantly increasing the training speed. In the experiments, four metrics are defined to evaluate the training performance of the models.

(i) Average reward. Given a unit time, we record the rewards per episode of each ULV during a unit time. The average reward can be calculated by $w_{\text{avg}} = w_{\text{acc}}/\lambda$, where w_{avg} is the accumulative rewards in a unit time, and λ is the number of episodes in a unit time. The higher the w_{avg} is, the better the training performance is

(ii) Training steps to complete each episode. In the experiments, we monitor the number of steps the agent moves in each training set. When the agent reaches the maximum number of steps or completes all mission objectives, reset the number of steps. In this way, the training effect of the agent can be observed by changing the number of steps in each episode

(iii) Task completion rate. In one episode, the task completion rate is defined as the ratio of the number of cargoes picked to the total number of cargoes, which is used to see if the agent can complete the task well

(iv) Collision times. We monitor the number of agent collisions per episode in the simulation environment, including collisions with walls, obstacles, and other agents, which is used to see if the agent can learn obstacle avoidance

In the following sections, we give the experimental results in the dynamic environment and fixed-point environment to evaluate the performance.

3.2. Experimental Results in Dynamic Environment. In this section, we build a dynamic environment for performance evaluation, in which one ULV, two obstacles, and ten cargo targets are scattered. The cargo locations are randomly generated, and the entire environment will be reset if either all cargoes are picked up every time or the ULV reaches the maximum number of steps. It is noted that the initial positions of the ULV and cargoes are random in a dynamic environment, which means that the optimal route for each episode is uncertain. We prefer that the ULV can learn how to obtain the maximum reward value, so the system will generate a penalty when a collision occurs instead of resetting the environment.

The average rewards are given in in Figure 3. We can see that the proposed SDRL, PPO, and BC finally converge, but the GAIL and SAC do not. And obviously the SDRL has a higher convergence speed and stronger stability in comparison with its rivals. It is noted that the GAIL model only copies the expert path to complete the task and fails to choose an optimal path to increase the reward of each episode. Since the SAC model [27] is designed for continuous action settings and does not apply to discrete action settings [29], it fails to complete the cargo transportation task in any episode, leading the moving step to reach the maximum number of steps, i.e., 100,000, in each episode, as shown in Figure 4. Because we optimize the generator of the GAIL model with the PPO in the proposed approach, the SDRL not only imitates the expert paths as GAIL does but also

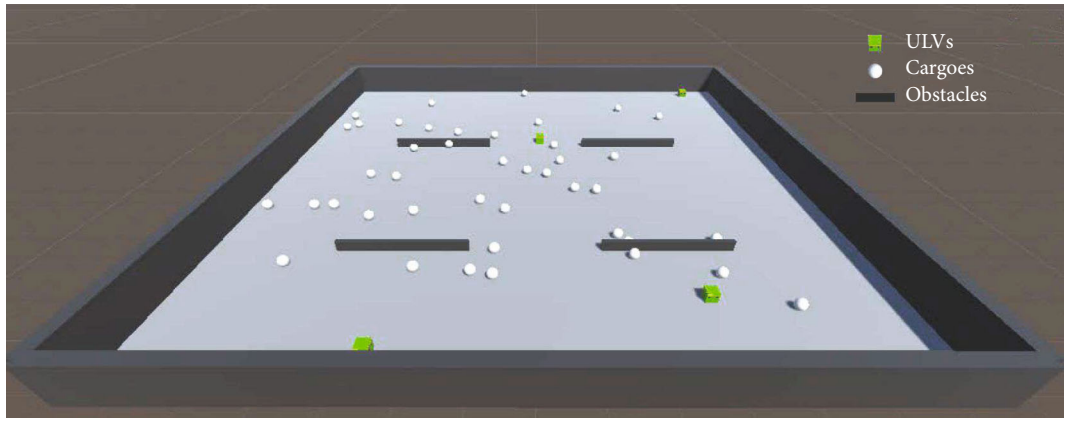

Figure 2: Example of warehouse simulation environment.

Table 1: External rewards configuration.

Reward item	Reward value
The ULV reaches the goal	+30
The ULV completes all tasks	+30
The ULV collides with an obstacle	-15
The ULV collides with a wall	-15
The ULV collides with another agent	-30
The ULV moves a step	-0.1
The ULV moves (a step) closer to the target	+0.6

takes the influence of the external environment of the PPO network into account, leading to a higher average reward than the baseline models.

Figure 5 compares the training steps for completing one episode of the four models. It can be seen that the proposed SDRL reduces the average steps below 10,000 steps after 350 episodes and tends to converge gradually. In comparison, the average steps of the other three models can slowly decrease with the increase of episodes, but their stability is poor. The SDRL shows a small fluctuation at the beginning because the target locations of the simulated environment are randomly generated, and target cargoes far away require more steps to complete one episode.

Figure 6 shows the comparison results of the task completion rate. Compared with the other three models, the SDRL can reach convergence quickly and maintain good stability. Due to the designed supervision module, its agent can select the path by imitating the recorded expert path in the early stage and then accelerate the learning speed. The GAIL model can imitate the expert path; however, since there is no interaction with the environment, the agent cannot obtain any benefit from the feedback of the environment. As shown in Figure 6, the completion rate of the GAIL model is high at the beginning of training, but it shows a big fluctuation with the increase in training times, leading to poor convergence. The PPO model generates action paths based on the environmental reward feedback through continuous training. The agent can summarize the experience of the previous operation after a couple of training times. As a result, the completion rate of the PPO is very low at the beginning, but it gets bigger as the number of training increases. The BC model shows the poorest performance in task completion rate.

The collision times per episode are shown in Figure 7, in which the SDRL and PPO models achieve low collision times. This is because both models consider the feedback from the external environment. In contrast, the GAIL model only imitates the expert path without the environmental feedback, so its collision time is significantly larger. Similarly, the BC model is only based on the path recorded by imitation, and the agent fails to adjust the training procedure according to the feedback of the environment. Through multiple training, although the collision time can be reduced with the increase of training episodes, its convergence speed is still relatively low.

3.3. Experimental Results in Fixed-Point Environment. In this section, we evaluate the performance in a fixed-point warehouse environment where the initial positions of the target cargoes and ULVs are fixed. This environment mimics the real warehouse scenario, where the cargoes are placed in the designated locations, and the logistics vehicles need to be trained many times to obtain the optimal path. In the experiment, we prefer that the ULVs can quickly find the unique optimal path without any collision. Once a collision appears, the environment should be reset to restart a new trial. Therefore, when one of the following conditions satisfies, the warehouse environment will be reset: (1) a collision occurs, and (2) all the target cargoes are picked up.

3.3.1. Simple Warehouse Environment. Different from the dynamic cargo experiments in previous sections, in the fixed-point experiment, the position of each cargo is fixed. After multiple training, the ULV path will be continuously optimized, and an optimal fixed path can be obtained. As shown in Figure 8, this section builds a simple warehouse environment, in which one ULV, four cargoes, and several obstacles are distributed in the space.

According to the results shown in Figure 9, both the SDRL model and the PPO model tend to converge eventually. The GAIL model starts with the highest average reward,

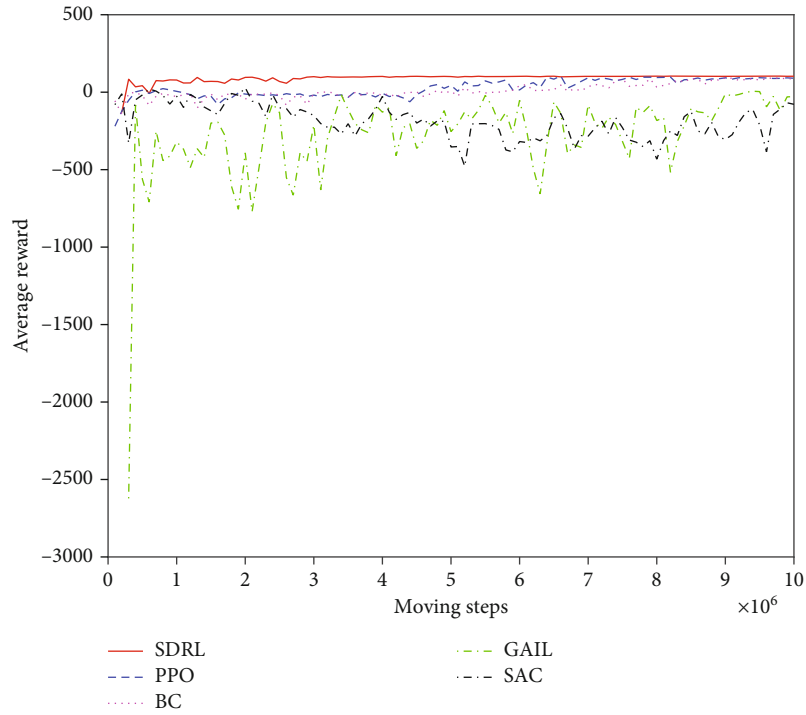

Figure 3: Comparison of average rewards in a dynamic warehouse environment.

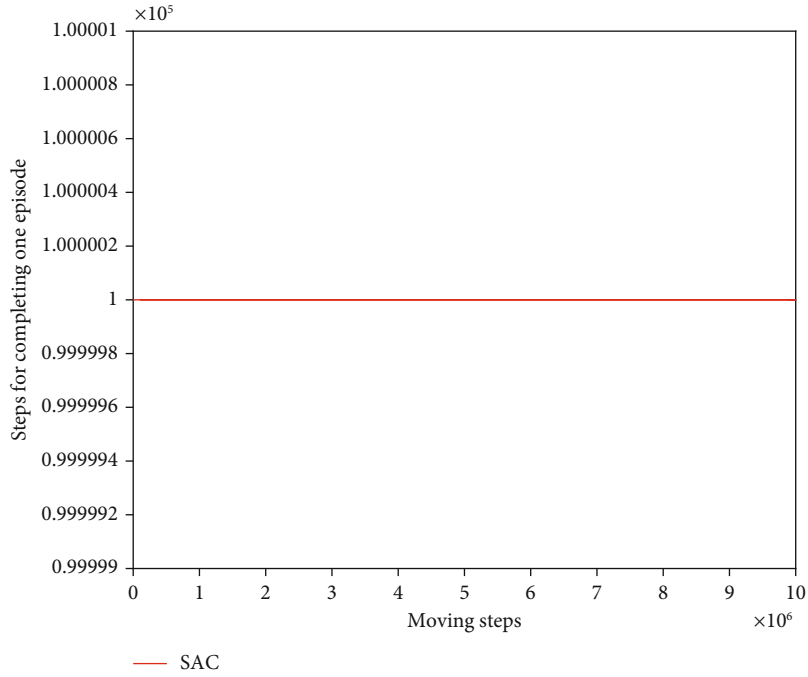

Figure 4: The steps for completing one episode of the SAC model.

but the training effect worsens as the number of training steps increases. This is because the GAIL model is greatly affected by expert paths in the early stage. However, when the collision with the obstacles happens, the GAIL cannot get any feedback to optimize the path even if the environment is continuously reset. By contrast, the SDRL model and PPO model can summarize previous experiences to avoid obstacles according to the feedback of the environment and thus obtain a path with a higher reward.

Figure 10 shows the task completion rate in a simple fixed-point environment. As shown, the task completion rate of the three models can reach 100%. When the training episode is about up to 200, the SDRL accomplishes the goal, but other models do not. In terms of convergence speed and

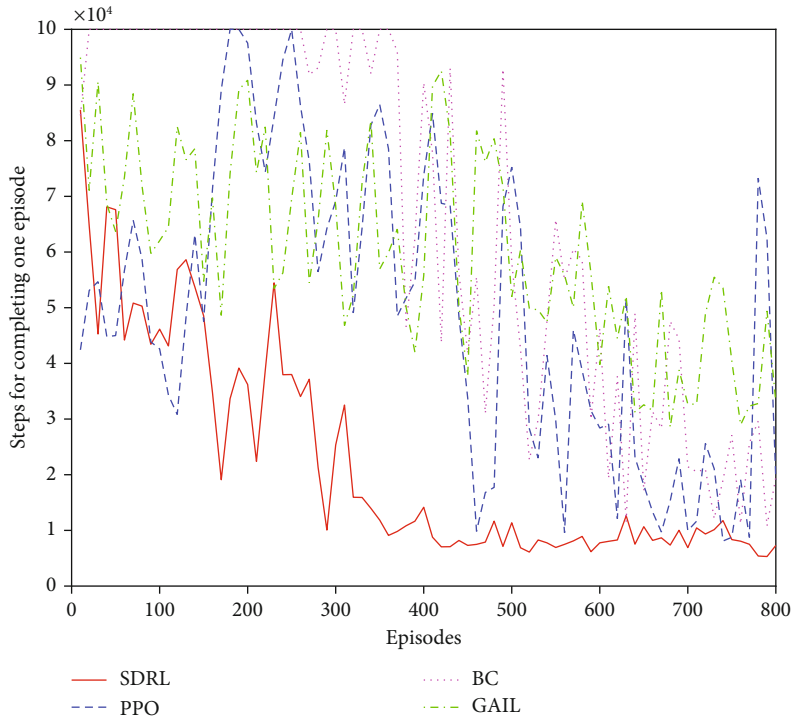

Figure 5: The steps for completing one episode in a dynamic environment.

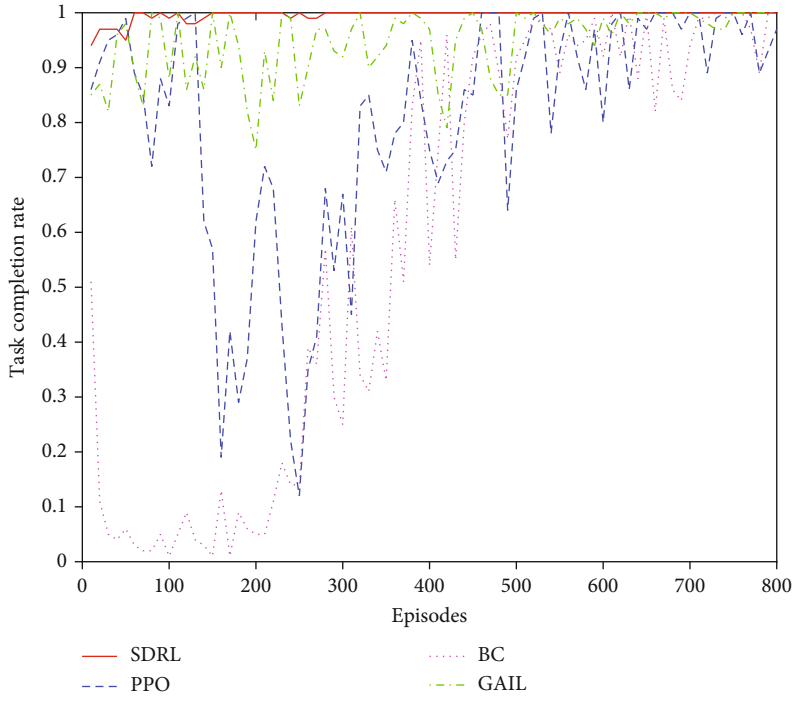

Figure 6: The task completion rate in a dynamic environment.

stability, the SDRL model is significantly better than the other two models.

3.3.2. Complex Warehouse Environment. Figure 11 shows the simulation environment of a fixed-point complex warehouse scenario, in which 5 ULVs, 20 target cargoes with fixed positions and many obstacles are scattered. By adding the number of the obstacles, we aim to evaluate the performance in complex warehouse scenario. Here, we give the results of the SDRL, PPO, and GAIL models.

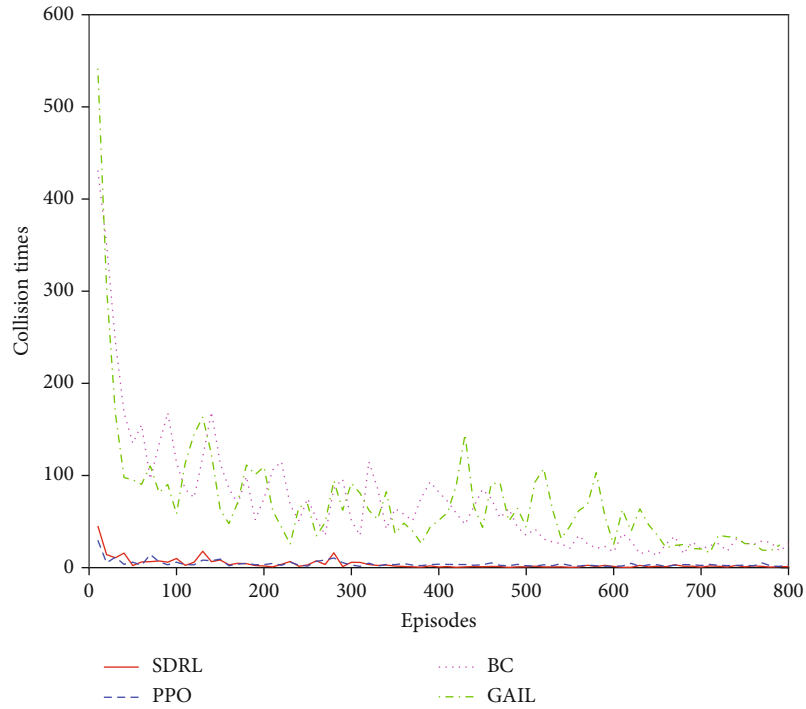

Figure 7: The collision times per episode in a dynamic environment.

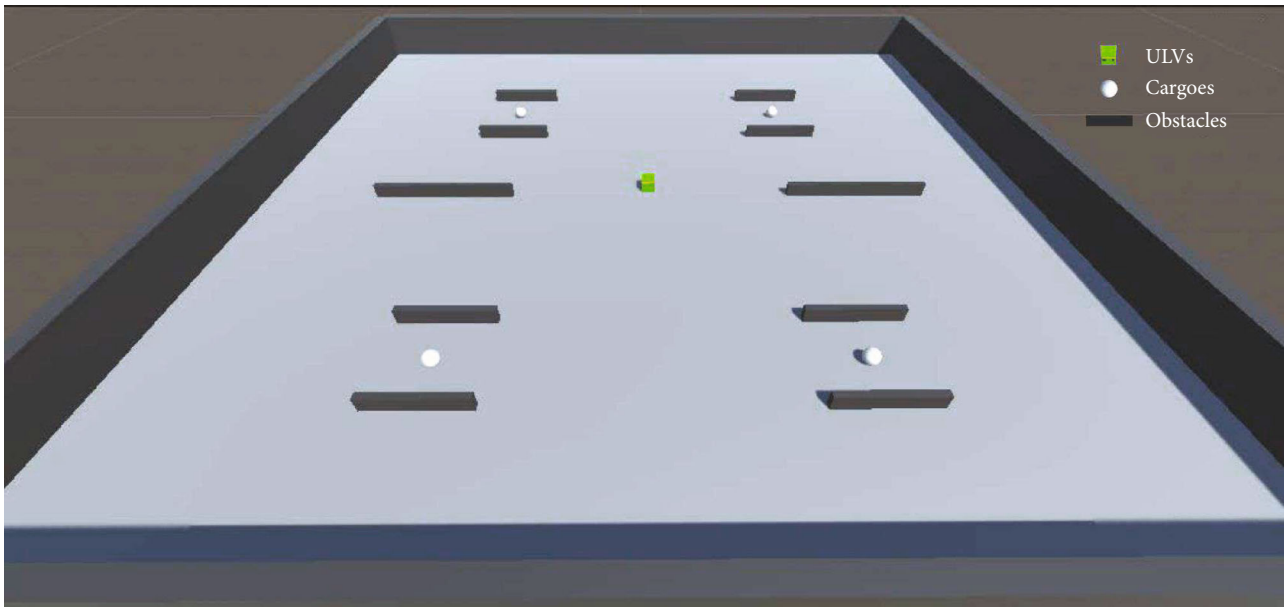

Figure 8: Simulation environment of a simple fixed-point warehouse scenario.

Figure 12 shows the results of average reward in a complex fixed-point environment. There are 5 ULVs in the complex environment, and the average reward is the average of 5 ULVs rewards. Due to the fixed positions of target cargoes, the ULVs can choose an optimal fixed path after training episodes, leading to more stable average rewards in comparison with the results in a dynamic environment. It can be seen that the average rewards of the three models increase as the number of moving steps increases. Compared with the results in previous scenarios, the PPO and GAIL models need much more steps to achieve the convergence. However, the SDRL model performs well with the highest average reward values and the fastest convergence speed.

In addition, the task completion rate is shown in Figure 13. When the episode is up to 5000, the SDRL model

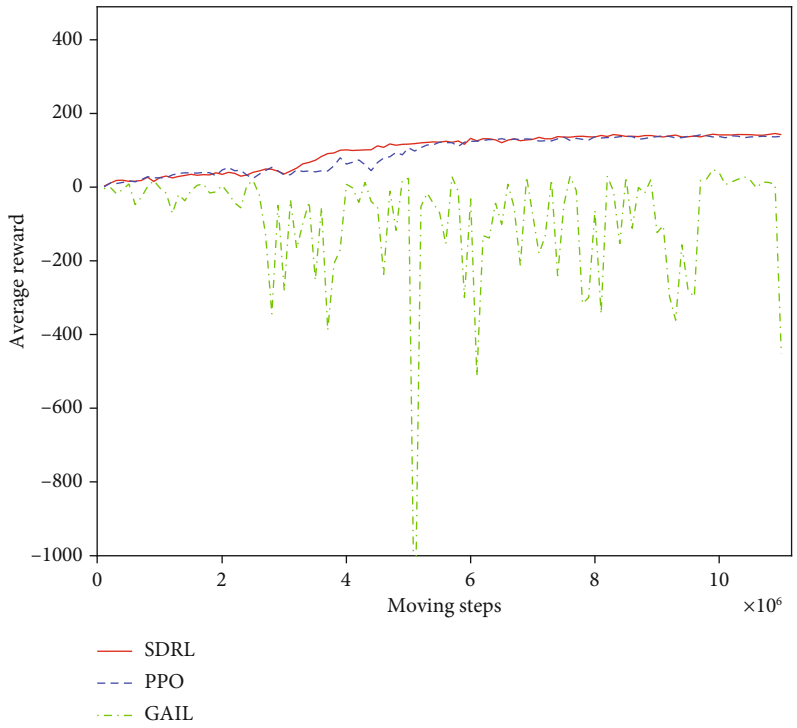

Figure 9: Average reward in a simple fixed-point environment.

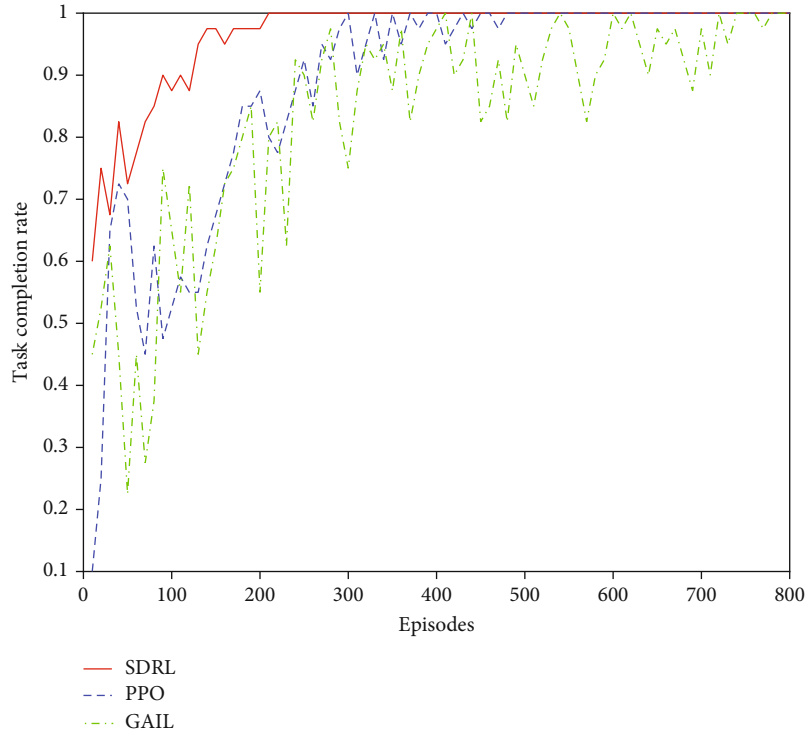

Figure 10: Task completion rate in a simple fixed-point environment.

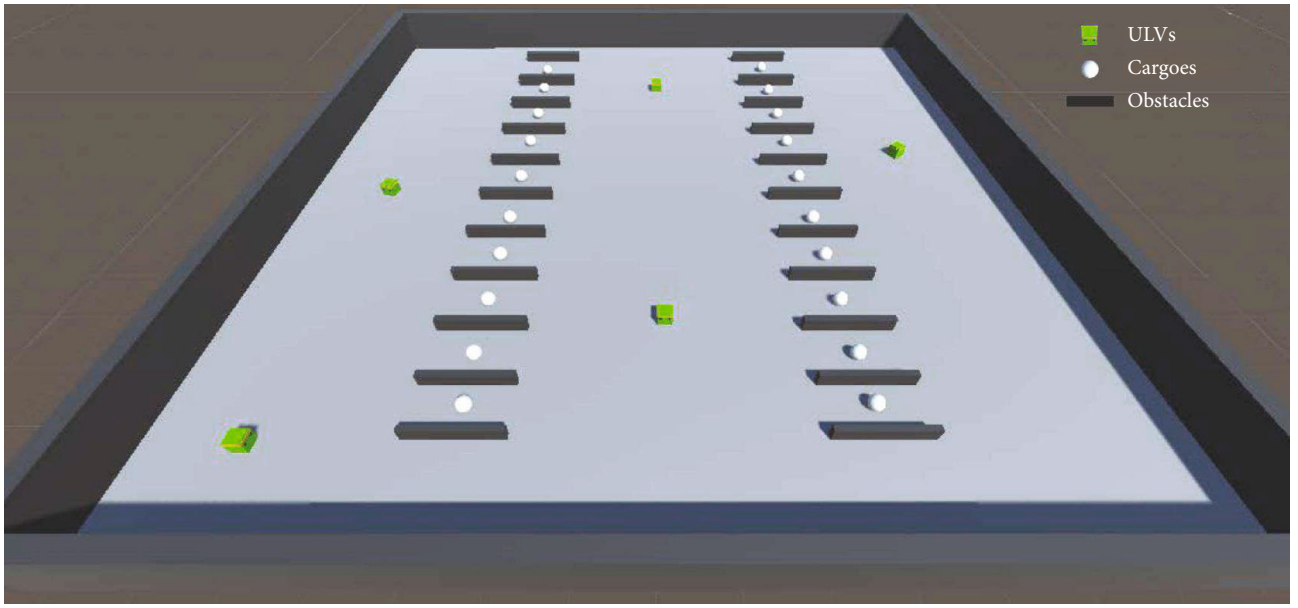

Figure 11: Simulation environment of a complex fixed-point warehouse scenario.

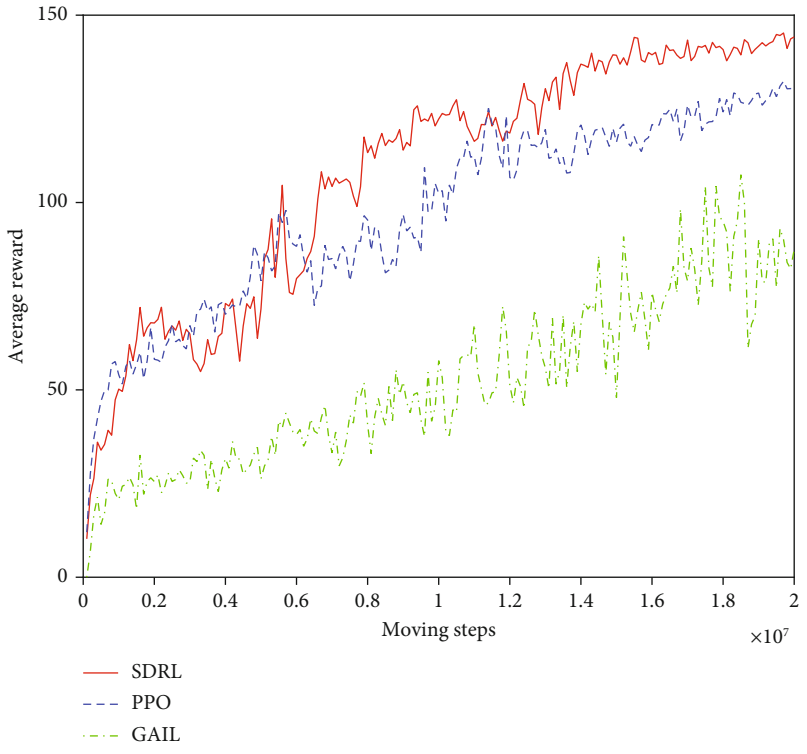

Figure 12: Average reward in a complex fixed-point environments.

almost completes the tasks, but the PPO needs more training episodes to achieve the 100% task completion rate. Moreover, the PPO model entirely relies on the summary of past experience to optimize the paths. At the beginning of the movement, the PPO needs to keep trying to gain experience, resulting in the lowest completion rate. The GAIL model lacks interaction with the environment, and the agent cannot optimize the path through the feedback, showing the poorest performance in task completion rate. Based on this, we can see that the proposed SDRL model outperforms its rivals in complex fixed-point environment.

In summary, the proposed SDRL model shows a better performance in both dynamic and fixed-point warehouse environments in comparison with the baselines.

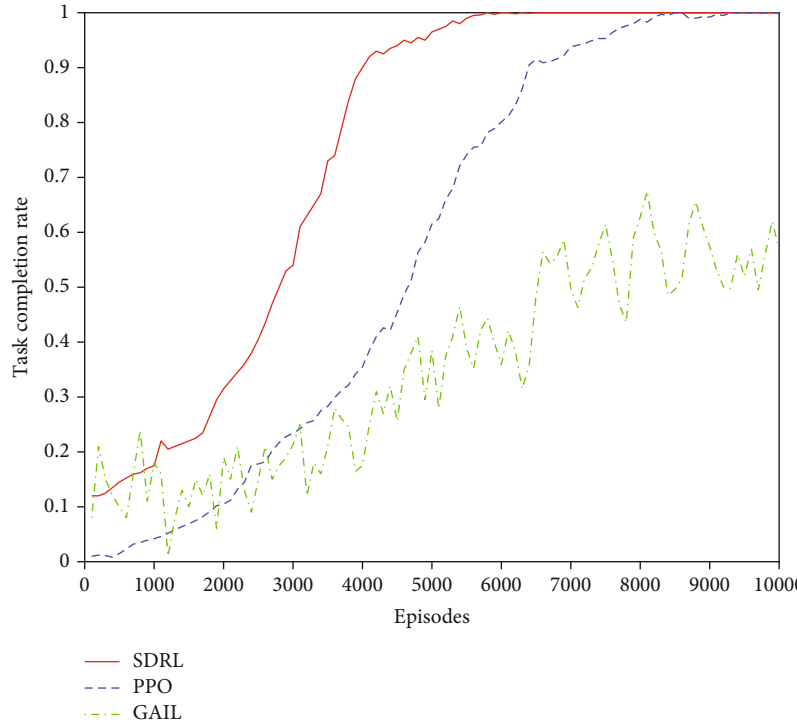

Figure 13: Task completion rates in a complex fixed-point environments.

4. Conclusion

In this paper, a supervised deep reinforcement learning (DRL) approach, i.e., SDRL, is proposed for unmanned logistics vehicles (ULVs) to automatically plan paths with obstacle avoidance when transporting cargoes in warehouse environments. By designing the supervision module, the agent imitates the behaviors of expert data and offers effective internal rewards. The policy module based on DRL evaluates the feedback from the environment via internal and external rewards. In this way, an optimized path with obstacle avoidance can be obtained. The experiments conducted in different warehouse environments show the proposed SDRL model outperforms the baselines.

Data Availability

The data used to support the findings of this study are included in the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Social Science Planning Program of Qingdao under Grant QDSKL2101218. We give warm thanks to anonymous reviewers for their critical comments and suggestions.

References

[1] J. Feng, L. Liu, Q. Pei, and K. Li, "Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 11, pp. 1–2700, 2022.

[2] L. Liu, M. Zhao, M. Yu, M. Jan, D. Lan, and A. Taherkordi, "Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2022.

[3] S. Mao, L. Liu, N. Zhang et al., "Reconfigurable intelligent surface-assisted secure mobile edge computing networks," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2022.

[4] H. Liu, Y. Deng, D. Guo, B. Fang, F. Sun, and W. Yang, "An interactive perception method for warehouse automation in smart cities," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 830–838, 2021.

[5] M. Geest, B. Tekinerdogan, and C. Catal, "Design of a reference architecture for developing smart warehouses in industry 4.0," *Computers in Industry*, vol. 124, article 103343, 2021.

[6] R. Indu, H. C. Singh, and A. Dubey, "Trajectory design for uavto-ground communication with energy optimization using genetic algorithm for agriculture application," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 17548–17555, 2021.

[7] X. Liang, J. Zhang, L. Zhuo, Y. Li, and Q. Tian, "Small object detection in unmanned aerial vehicle images using feature fusion and scaling-based single shot detector with spatial context analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1758–1770, 2020.

[8] R. Chen, Y. Sun, L. Liang, and W. Cheng, "Joint power allocation and placement scheme for uav-assisted iot with qos guarantee," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 1066–1071, 2022.

[9] H. Yoshitake, R. Kamoshida, and Y. Nagashima, "New automated guided vehicle system using real-time holonic scheduling for warehouse picking," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1045–1052, 2019.

[10] I. Draganjac, D. Miklic, Z. Kovacic, G. Vasiljevic, and S. Bogdan, "Decentralized control of multi-agv systems in autonomous warehousing applications," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 4, pp. 1433–1447, 2016.

[11] J. Faigl, P. Vana, and J. Deckerova, "Fast heuristics for the 3-d multi-goal path planning based on the generalized traveling salesman problem with neighborhoods," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2439–2446, 2019.

[12] J. Zuo, J. Chen, Y. Tan, M. Wang, and L. Wen, "A multi-agent collaborative work planning strategy based on afsa-pso algorithm," *In 2019 International Conference on Robots & Intelligent System (ICRIS)*, pp. , 2019254–257, 2019.

[13] L. Hu, W. Naeem, E. Rajabally et al., "A multiobjective optimization approach for colregs-compliant path planning of autonomous surface vehicles verified on networked bridge simulators," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1167–1179, 2020.

[14] A. Al-Dubai, G. Min, J. Li et al., "A novel prediction-based temporal graph routing algorithm for software defined vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 13275–13290, 2021.

[15] A. Hawbani-K, Y. Yu, Z. L. Zhao, Z. Bi, and M. Guizani, "Elite: an intelligent digital twin-based hierarchical routing scheme for softwarized vehicular networks," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2022.

[16] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

[17] C. Wang, J. Wang, Y. Shen, and X. Zhang, "Autonomous navigation of uavs in large-scale complex environments: a deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2124–2136, 2019.

[18] S. Semnani, H. Liu, M. Everett, A. Ruiter, and J. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3221–3226, 2020.

[19] P. Sadhukhan and R. Selmic, "Multi-agent formation control with obstacle avoidance using proximal policy optimization," in *In 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2694–2699, Melbourne, Australia, 2021.

[20] T. Phiboonbanakit, T. Horanont, V. Huynh, and T. Supnithi, "A hybrid reinforcement learning-based model for the vehicle routing problem in transportation logistics," *IEEE Access*, vol. 9, pp. 163325–163347, 2021.

[21] M. Pei, H. An, B. Liu, and C. Wang, "An improved dyna-q algorithm for mobile robot path planning in unknown dynamic environment," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, pp. 4415–4425, 2021.

[22] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[23] I. Goodfellow, J. Abadie, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[24] A. Al-Dubai, A. Zomaya, G. Min, L. Zhao, Y. Liu, and A. Hawbani, "A novel generation-adversarial-network-based vehicle trajectory prediction method for intelligent vehicular networks," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 2066–2077, 2021.

[25] J. Zhang, Z. Yu, S. Mao, S. Periaswamy, J. Patton, and X. Xia, "Iadrl: imitation augmented deep reinforcement learning enabled ugv-uav coalition for tasking in complex environments," *IEEE Access*, vol. 8, pp. 102335–102347, 2020.

[26] P. Dhariwal-A, R. J. Schulman, F. Wolski, and O. Klimov, "Proximal policy optimization algorithms," 2017. [online] Available: https://arxiv.org/abs/1707.06347.

[27] P. Abbeel, T. Haarnoja, A. Zhou, and S. Levine, "Soft actor-critic: off policy maximum entropy deep reinforcement learning with a stochastic actor," in *In International Conference on Machine Learning*, pp. 1861–1870, Stockholm, Sweden, 2018.

[28] A. Edwards, H. Sahni, Y. Schroecker, and C. Isbell, "Imitating latent policies from observation," in *In International Conference on Machine Learning*, pp. 1755–1763, Long Beach, California, USA, 2019.

[29] P. Christodoulou, "Soft actor-critic for discrete action settings," 2019. [online] Available: http://arxiv.org/abs/1910.07207.