Proceedings of the 5th WRC Symposium
on Advanced Robotics and Automation 2023
Beijing, China, August 19, 2023

2023 WRC Symposium on Advanced Robotics and Automation (WRC SARA) | 979-8-3503-0732-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/WRCSARA60131.2023.10261849

Deep Reinforcement Learning Integrated RRT Algorithm for Path Planning

Huashan Liu[1,2] , Yuqing Gu[1,2] , Xiangjian Li[1,2,*] and Xinjie Xiao[1,2]

***Abstract*— Rapidly-exploring random tree (RRT) algorithm, featured with strong exploration capability, is widely used in path planning tasks. However, it is difficult for RRT to find the optimal path due to its inherent characteristics of random sampling. In this paper, we propose an improved RRT algorithm integrated with deep reinforcement learning (IRRT-DRL), which can effectively search for feasible paths and exploit previous experience for optimization. It includes a unique reward function and a dynamic waypoint selection mechanism that automatically adjusts the interval between adjacent waypoints to help the agent bypass obstacles. Experimental results have verified the feasibility and superiority of the proposed approach.**

***Index Terms*— Deep reinforcement learning (DRL), rapidly-exploring random tree (RRT), path planning, robot.**

I. INTRODUCTION

Path planning is an important part of the robotic tasks and has been extensively investigated [1]-[4]. Its primary objective is to search for a feasible node sequence that connects the initial and the target states in finite time. If obstacles are present in the environment, the path planning method must also have the ability to detect and avoid collisions, enabling safe and efficient robotic navigation.

There are many kinds of path planning algorithms, such as probabilistic roadmap method (PRM) [5], A-star [6], Dijkstra and RRT [7]. PRM requires pre-sampling of nodes randomly in the environment, which means that the paths planned by it are heavily dependent on the quality of these random points. Although A-star can search for appropriate paths through heuristic functions, it becomes increasingly challenging to find a suitable heuristic function as the size of the scene grows significantly. As a result, the computational complexity of the operation also increases drastically, making it impractical for large-scale scenarios. The search rate of Dijkstra is influenced by a large number of nodes that need to be traversed. s RRT algorithm does not need to divide the environment geometrically, and the operational complexity is slightly affected by the scale of the scenario. It also features node expansion, and the paths planned by it are traceable. But the paths generated by RRT are usually lengthy due to its numerous and superfluous nodes (or edges). Several state-of-the-art path planning algorithms based on RRT, such as RRT-star and RRT-connect [8] [9], provide different optimization schemes for the paths produced by RRT. RRT-star is capable of reselecting parent nodes by replacing the original ones with less costly nodes in expanded regions, which can simplify the paths effectively. Meanwhile, RRT-connect uses two trees to search for paths simultaneously, which can improve the planning efficiency. However, the optimizing results of these algorithms still rely heavily on random sampling. While the randomness of samspling assists the planning algorithms in exploring the environment extensively to find feasible paths, it is difficult for them to approach the optima since each sampling step is independent. Besides, previous planning experience is not sufficiently exploited, which leads to a lack of guidance in the direction of optimization.

DRL can assist the agent to interact with the environment autonomously to find feasible paths from the initial to the target states, and the techniques based on DRL are widely used in many fields [10]-[13]. The agent equipped with DRL follows the principles of exploration and exploitation. Mechanisms such as time difference error, soft updating, and experience replay facilitate the agent to approach the optimal policy by reusing previous experience. Moreover, DRL does not require a pre-prepared dataset, as the data for the training process is generated automatically when the agent interacts with the environment. However, algorithms of DRL kind may not be as effective at exploration when compared to the ones of other kinds. Although adding random noise can help the agent explore the environment, these techniques become less effective as the environment becomes more complex.

As stated in soft actor-critic (SAC) [14], under the maximum entropy framework, the aim of the agent is not only to maximize the expected reward over time, but also to maintain a balance among the selections of actions during the exploration. When the policy updates, entropy evaluation of the policy can reduce the bias and encourage exploration. Since the main aim of the agent is to achieve the maximum returns of the epochs, the temperature parameter can be set relatively low to prioritize exploitation over exploration, which leads to a slight weighting of the entropy evaluation in the policy updates. While entropy evaluation of the policy can increase the exploration capability of the agent, it is unable to resolve the challenges presented by the exploration-exploitation dilemma inherent in DRL algorithms. As a result, the agent may still face difficulties in crossing complex obstacles and selecting optimal actions in high-dimensional state and action spaces.

This work was sponsored in part by the National Natural Science Foundation of China under Grant xxxxxxxx, the Natural Science Foundation of Shanghai under Grant 21ZR1401100, Discipline Innovation Cultivation Program under Grant XKCX202304 and the Fundamental Research Funds for the Central Universities under Grant 2232022G-09.

[1]College of Information Science and Technology, Donghua University
[2]Engineering Research Center of Digitized Textile & Fashion Technology, Ministry of Education Shanghai, China
*Corresponding author is X.Li (E-mail:xjli@mail.dhu.edu.cn)

In order to cope with complex path planning tasks, an efficient path planning method should possess strong capabilities in both exploration and optimization. A feasible scheme is to combine the merits of the RRT and the DRL. Some researchers have explored relevant concepts and proposed some combining approaches. For example, [16] has proposed an online motion planning algorithm based on RRT* and Q-learning to locally recalculate the path. In [17], PRM-RL has been proposed as a hierarchical robot navigation method that uses intensive learning agents to solve short-distance problem of path planning tasks with obstacle avoidance. PRM, a sampling-based planner, has been proposed to map the position for reliable navigation and guide the robot in seeking the shortest path. As shown in [18], based on the basic path planned by RRT*, the agent equipped with DRL algorithm can efficiently explore and generate a better path, benefiting from the mechanism of experience replay.

In response to the above discussions, an improved RRT algorithm integrated with DRL (IRRT-DRL) is proposed. The dynamic waypoint selection mechanism designed can select the key path points from the nodes of the basic path planned by RRT, which are conducive to the path optimization of DRL agents. Unique reward function is designed for each individual point-to-point (P2P) task to guide agents to learn the optimal policy.

The remainder of this paper is organized as follows. Section II reviews the fundamental of RRT and DRL. In Section III, the structure of IRRT-DRL algorithm is described in detail. Section IV shows the experimental results. Finally, some conclusions are given in Section V.

II. FUNDAMENTALS

In this section, main principle of RRT algorithms is stated first, then brief fundamentals of Markov decision making and SAC are given.

A. RRT algorithms

The RRT algorithms use tree nodes and edges to find feasible paths in the environment, while randomly generating nodes to explore the search space, resulting in strong exploration capability over the entire environment. When exploring the next node, RRT samples the surrounding nodes randomly and tries to find the nearest node to the sampled point. If enough attempts are made, the algorithm can converge to the optimal path. Due to these characteristics, the RRT algorithms (such as RRT-star, RRT-connect, etc.) have been proven to possess probabilistic completeness and a high success rate in solving path planning tasks in complex environments. However, owing to the uncertainty and irregularity of the random sampled nodes, they are challenged to optimize the paths in successive iterations. Moreover, the valuable experience in previous attempts is difficult to be reused.

Compared to the RRT algorithms, the DRL algorithms excel in automatic learning ability and adaptability. Utilizing neural networks and other advanced techniques, they can iteratively optimize the basic path, thus significantly improving the accuracy and adaptability of path planning. Therefore, the RRT algorithms can be employed to generate the initial path swiftly and conveniently, which can subsequently be fine-tuned using the DRL to obtain more optimized paths. This method not only takes into account computational efficiencys but also improves accuracy and reliability of path planning.

B. DRL algorithms

The path planning by DRL can be considered as a Markov decision process, which can be described as an tuple $\boldsymbol{\mathcal{M}} = (\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{A}}, R, \gamma, \mathrm{T})$, where $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{A}}$ are state and action spaces, espectively. R is the reward function, which is an intrinsic element of the environment. γ is the decay factor, which can adjust the impact of previous experience in policy updates. T represents the total number of times in the entire Markov decision process.

SAC algorithm is based on the actor-critic structure. It operates under the maximum entropy framework and extends the goal of DRL to maximize cumulative reward with entropy. The objective function of SAC can be expressed as

$$J(\pi) = -\sum_{t=0}^{\mathrm{T}} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t)\sim\pi(\cdot|\boldsymbol{s}_t)} \Big[r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \alpha \mathcal{H}\big(\pi(\cdot|\boldsymbol{s}_t)\big) \Big] \tag{1}$$

where α is the temperature coefficient of entropy and $\mathcal{H}(\pi(\cdot|\boldsymbol{s}_t) = -\sum \pi(\cdot|\boldsymbol{s}_t) \log \pi(\cdot|\boldsymbol{s}_t)$ is the evaluation function of the policy.

SAC is stably trained using a value network, whose update function can be expressed as

$$J_V(\boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{s}_t\sim\mathcal{D}} \Big[\frac{1}{2} \big(V_\psi(\boldsymbol{s}_t) - \mathbb{E}_{\boldsymbol{a}_t\sim\pi(\phi)} [\min_{i=1,2} Q_{\theta_i}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \log \pi_\phi(\boldsymbol{a}_t|\boldsymbol{s}_t)] \big)^2 \Big] \tag{2}$$

where $\mathcal{D}$ is the replay buffer. In addition, a form similar to the dual Q networks is adopted, the minimum of soft Q value is taken by two Q functions parameterized by θ_1 and θ_2. Soft Q function parameters can be obtained by minimizing soft Bellman residual

$$J_Q(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t)\sim\mathcal{D}} \Big[\frac{1}{2} \big(Q_{\theta_{i=1,2}}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \hat{Q}(\boldsymbol{s}_t, \boldsymbol{a}_t) \big)^2 \Big] \tag{3}$$

where $\hat{Q}(\boldsymbol{s}_t, \boldsymbol{a}_t) = r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \mathbb{E}_{(\boldsymbol{s}_{t+1}\sim p)}[V_{\bar{\psi}}(\boldsymbol{s}_{t+1})]$, $V_{\bar{\psi}}$ is the target value network in deep Q network.

The policy parameters can be obtained by minimizing the Kullback-Leibler (KL) divergence

$$J_\pi(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{s}_t\sim\mathcal{D}} \Big[D_{KL}\big(\pi_{\boldsymbol{\phi}}(\cdot|\boldsymbol{s}_t) || \frac{exp(Q_{\theta_{i=1,2}}(\boldsymbol{s}_t, \cdot))}{Z_\theta(\boldsymbol{s}_t)} \big) \Big] \tag{4}$$

DRL algorithms do not require prior knowledge, can effectively use the exploration experience of the replay buffer and exhibit better generalization performance. However, it is difficult for the agents equipped with DRL in searching the entire environment and bypassing complicated obstacles. When this happens, it may become unfavourable for policy learning when the environment becomes complex, and the

agent requires waypoints for guidance under such circumstances [15].

In order to solve this problem, nodes on the basic path can be arranged to guide the agent to implement the path planning. Nevertheless, nodes on the basic path planned by RRT are too numerous and unevenly distributed, resulting in high path complexity and poor smoothness. Hence, based on the characteristics and requirements of the actual scenario, this paper proposes a dynamic waypoint selection mechanism to select suitable waypoints to guide DRL to optimization routing tasks.

III. OUR METHOD

As shown in Fig. 1, the proposed approach can be divided into two parts: the RRT part and the DRL part. The former part generates a basic path by constructing a search tree and filtering out redundant path nodes, while the latter part utilizes previous experience and memory to guide the completion of P2P tasks between every two adjacent nodes on the basic path. Optimal path can be attained through continuous iterative updates.

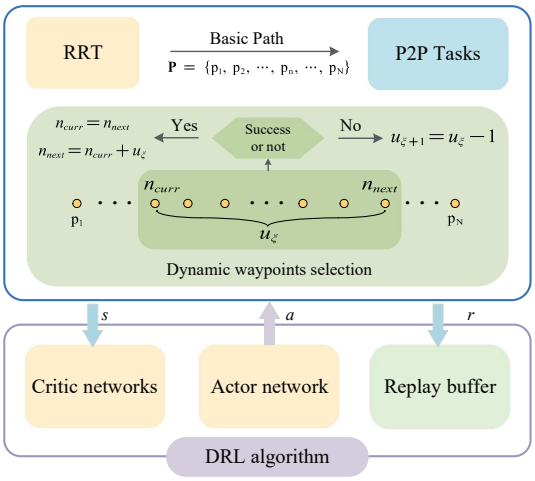

Fig. 1. The schematic of IRRT-DRL.

In the RRT part, firstly a basic path is planned by RRT, which can be denoted as $\mathbf{P}=\{\mathrm{p}_1, \mathrm{p}_2, \cdots, \mathrm{p}_n, \cdots, \mathrm{p}_\mathrm{N}\}$, where p_n is the node on the basic path and N is the total number of these nodes. Then, DRL is employed to efficiently complete the P2P tasks between the nodes along the path, where a dynamic waypoint selection mechanism is proposed. It can be described as

$$n_{\mathrm{next}} = n_{\mathrm{curr}} + u_\xi \tag{5}$$

$$\begin{cases} u_{\xi+1} = \mathrm{U}_{\max}, & \text{Successful or initial} \\ u_{\xi+1} = u_\xi - 1, & \text{Failed} \end{cases} \tag{6}$$

where n_{next} and n_{curr} represent the target and current waypoint number of a P2P task, respectively. ξ refers to the ξ-th stage of the P2P tasks, u_ξ denotes the interval between two adjacent waypoints and $\mathrm{U}_{\max}$ is the initial value of u_ξ.

Choosing the right waypoints is critical for DRL to complete the task with high quality. As described in (5) and (6), when initializing, interval u_ξ is first set to the maximum value $\mathrm{U}_{\max}$ to facilitate agent to optimize basic path quickly. Nevertheless, in a complex environment, if the interval between waypoints becomes extremely large, the agent may find it challenging to overcome the obstacles that lie between them, potentially leading to failures. Hence, the interval u_ξ will be adjusted according to the results of the optimization in the current phase. If the optimization fails, then reduce u_ξ and try again, otherwise, the agent will reset $\mathrm{U}_{\max}$ to the maximum value and proceed to the next phase of the optimization task. By dynamically setting the goals of P2P tasks according to the complexity of the environment, the agent can complete the optimization tasks more effectively.

Based on the characteristics and requirements of the actual scenario, dynamic waypoint selection mechanism selects the path points that can help agents complete P2P tasks as the screening requirements, and then finds the path points that can realize the limit position of path planning as the key waypoints. Subsequently, it divides the basic path into multiple P2P tasks, which is conducive to the DRL to complete the path optimization task. Meanwhile, for DRL, the design of reward function directly affects the learning performance.

Appropriate reward function can provide effective feedback for the training, so that the agent can learn from the environment more efficiently. In the P2P task, the reward function is designed as

$$r_{\mathrm{curr}} = \mathrm{A}\exp(-\mathrm{C}\mathrm{dis}(p_{\mathrm{curr}}, p_{\mathrm{targ}})) + \mathrm{colli}(p_{\mathrm{curr}}) \tag{7}$$

$$\mathrm{colli}(p_{\mathrm{curr}}) = \begin{cases} 0, & \text{Collision} \\ \mathrm{D}, & \text{Else} \end{cases} \tag{8}$$

where A, C and D are constants. p_{curr} and p_{targ} are the current position of the agent and the target waypoint in the current P2P task. $\mathrm{dis}(p_{\mathrm{curr}}, p_{\mathrm{targ}})$ denotes the distance between nodes p_{curr} and p_{targ}. $\mathrm{colli}(p_{\mathrm{curr}})$ is a collision function that the reward will be 0 if the agent collides with obstacles, otherwise the reward is a constant.

The exponential function $y=\exp(-x)$ approaches zero as the input x tends to positive infinity, making it a suitable choice for describing the uneven distribution of distance-based rewards. The implication of this uneven reward distribution is that the variation in the reward received by the agent should be significant as the agent gradually approaches the target waypoint. The agent can be inspired by this uneven reward function to approach the target waypoint. The constant A is the amplitude, which is used to regulate the range of the maximum return of the epochs obtained by the agent. The collision function $\mathrm{colli}(p_{\mathrm{curr}})$ can describe the state of the agent during path exploration. If the agent is able to avoid colliding with obstacles, it will receive an additional constant reward D. Conversely, there is no additional reward. The collision detection function should have the ability to detect the state between the agent and obstacles and provide additional rewards or prompts to the agent to avoid obstacles.

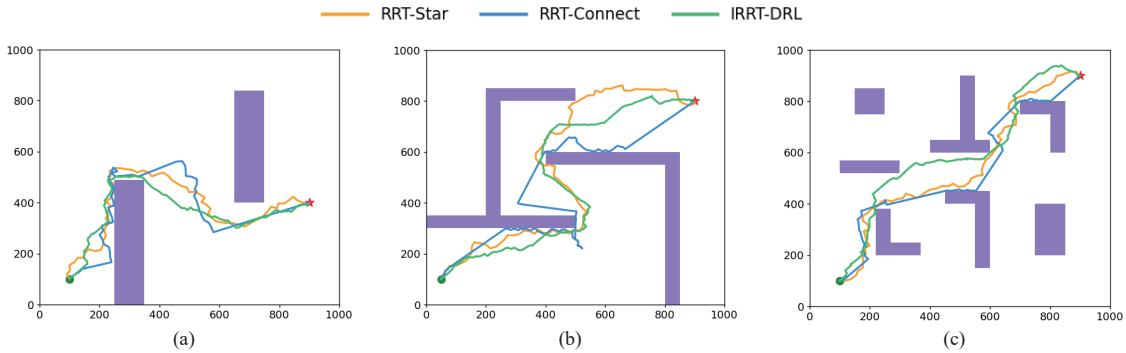

Fig. 2. Experimental results in three different 2-D scenarios. Each path is the optimal result chosen from 15 attempts. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

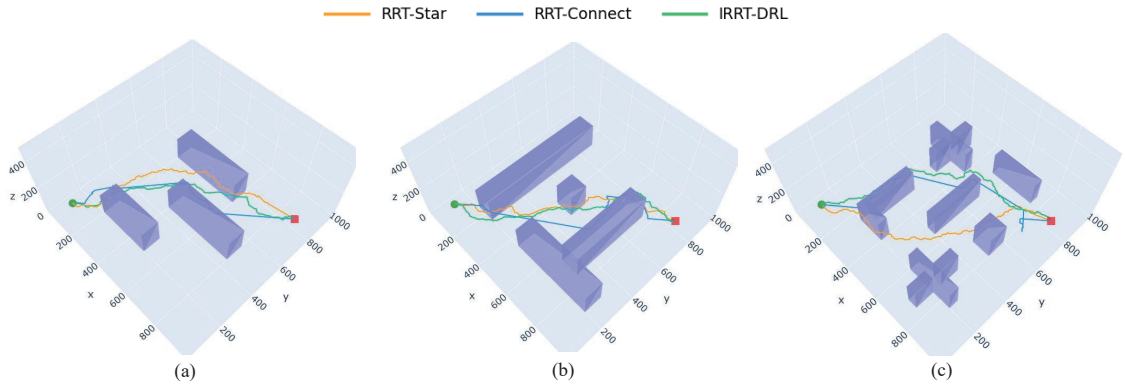

Fig. 3. Experimental results in three different 3-D scenarios. Each path is the optimal result chosen from 15 attempts. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

The dynamic waypoint selection mechanism selects the key nodes on the path as reference points for DRL to complete the optimization, and the unique reward function guides the agent to explore the shortest path and learn the optimal policy. IRRT-DRL is formed by combining these two characteristics, and the path length can converge to the theoretical optimal with the increase of the number of iterations.The proposed IRRT-DRL is expressed in Algorithm 1.

IV. EXPERIMENTS

In this section, the proposed IRRT-DRL algorithm is compared with the state-of-the-art tree path planners (RRT-star and RRT-connect) in different scenarios of two and three dimensions. The experimental settings of the three scenes in two dimensions and three dimensions have increasing levels of difficulty for the agent to complete the path planning task. Each experiment is tried 15 times for each algorithm in each scenario, Figs. 2 and 3 show the paths generated by different algorithms. In addition, the starting and goal points are randomly set and the results are shown in Figs. 4 and 5.

As shown in Figs. 2-5, the proposed IRRT-DRL has better performance than the other algorithms, especially when the scene complexity becomes higher. The RRT part utilizes its powerful ability of random exploring to attain a feasible basic path. Based on this basic path, the key and executable

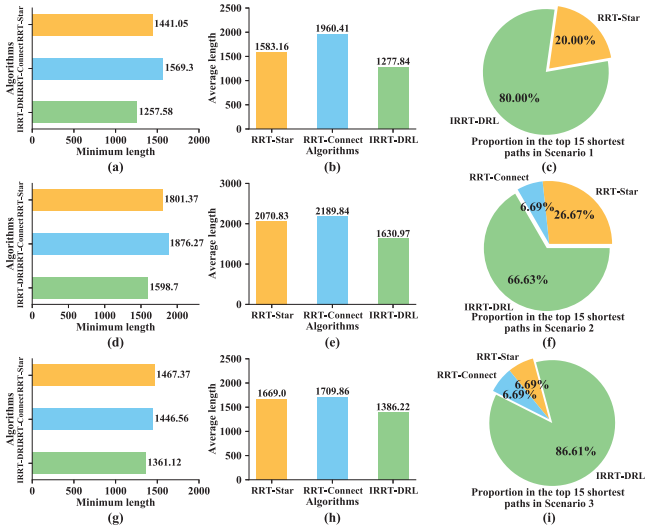

Fig. 4. Charts of algorithms in three different 2-D scenarios. (a), (b) and (c) for Scenario 1, (d), (e) and (f) for Scenario 2, and (g), (h) and (i) for Scenario 3.

waypoints are selected through the dynamic path selection mechanism. The DRL part (SAC is equipped) uses its ability of experience replay and policy update, which has the merit of constantly converging to the optimal result.

Moreover, DRL algorithm is prone to struggling with

Algorithm 1: IRRT-DRL

Initialize the parameters of DRL algorithm.
Initialize the parameters of RRT algorithm.
Set the interval u_ξ.
for nodes=1,2,$\cdots$, N **do**
  for u_ξ=$\mathrm{U_{max}}$, $\mathrm{U_{max}}-1,\cdots,1$ **do**
    Initializing the n_{next} by Equation (5)
    for t=1,2,$\cdots$, T **do**
      Obverse the environment and collect the state $\boldsymbol{s}_t$;
      Sampling the action $\boldsymbol{a}_t$ by the current policy;
      Execute the action $\boldsymbol{a}_t$ to obtain the reward r_t and observe a new state $\boldsymbol{s}_{t+1}$;
      if *the agent reaches* n_{next} **then**
        Break current loop.
      Store the transition $\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1}$ in $\mathcal{D}$
      Update the notworks of DRL algorithm;
      Delayed update the target networks.
    if *the agent reaches* n_{next} **then**
      $u_{\xi+1} = \mathrm{U_{max}}$;
      Break current loop.
    else
      $u_{\xi+1} = u_\xi - 1$.

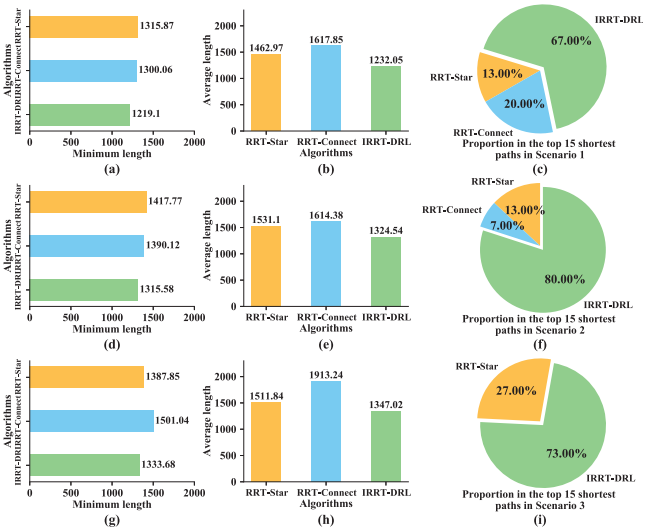

Fig. 5. Charts of algorithms in three different 3-D scenarios. (a), (b) and (c) for Scenario 1, (d), (e) and (f) for Scenario 2, and (g), (h) and (i) for Scenario 3.

complex obstacles, yet the dynamic selection mechanism can make up for this deficiency. As shown in Figs. 2a and 3a, the complexity of scenario is simple and the interval u_ξ is set to the maximum, which enables the agent to move quickly towards distant target waypoint without extra exploration. However, in Figs. 2b, 2c, 3b and 3c, when obstacles become more challenging, the agent may get stuck, which leads to expensive costs of extra path exploration or even failure. At this point, reducing the interval u_ξ and setting a more accessible target waypoint, such as coming to a hard-to-cross corner, can provide a closer waypoint, making it easier for the agent to optimize the whole path.

As shown in Figs. 4 and 5, the shortest path and average path of IRRT-DRL are shorter than the other two algorithms, and the results are closer to the theoretical optima. In Figs. 4c, 4f, 4i, 5c, 5f and 5i, the 15 experimental results of each algorithm are arranged according to the path length, and the proportion of each algorithm in the first 15 shortest paths is calculated.

Among them, RRT-connect takes a low proportion or even zero, which explores new paths using two trees at the same time, but its dependence on randomness can not be avoided and lead to generate sub-optimal path. RRT-star adopts the mechanism of re-selecting parent nodes and trimming excess edges to optimize paths, and it has a larger proportion of the results in the first 15 shortest paths than the RRT-connect. However, RRT algorithms are easy to fall into local optimization and are tough to adapt to complex scenes. The proposed IRRT-DRL algorithm addresses this issue by utilizing the experience of previous explorations as a reference, and learns from past explorations to guide the trend of further planning. Hence, it performs better than the others and dominates in these proportions, with demonstrating its superiority to other algorithms in the optimization of path length in general.

V. CONCLUSIONS

Inspired by RRT algorithm and DRL algorithm, an improved path planning algorithm (IRRT-DRL) with a dynamic waypoint selection mechanism is proposed. By combining the advantages of RRT and DRL, IRRT-DRL enables powerful spatial exploration and skilled exploitation of past experiences to continuously converge to the optimal path. Experimental results have demonstrated its feasibility and superiority, and it is promising to be applied in general robotic path planning tasks.

REFERENCES

[1] F. Ying, H. Liu and R. Jiang, "Trajectory generation for multiprocess robotic tasks based on nested dual-memory deep deterministic policy gradient,"IEEE/ASME Transactions on Mechatronics, vol. 27, no. 6, pp. 4643-4653, Dec. 2022.

[2] J. Guo, C. Li and S. Guo, "A novel step optimal path planning algorithm for the spherical mobile robot based on fuzzy control," IEEE Access, vol. 8, pp. 1394-1405, 2019.

[3] M. Cai, H. Liu and M. Dong. "Easy pose-error calibration for articulated serial robot based on three-closed-loop transformations," IEEE Transactions on Instrumentation and Measurement, vol. 70, pp. 1-11, 2021.

[4] A. A. Ravankar, A. Ravankar, T. Emaru and Y. Kobayashi, "HPPRM: hybrid potential based probabilistic roadmap algorithm for improved dynamic path planning of mobile robots," IEEE Access, vol. 8, pp.221743-221766, 2020.

[5] R. Geraerts and M. H. Overmars, "A comparative study of probabilistic roadmap planners," Algorithmic Foundations of Robotics V, vol. 8, pp. 43-57, 2004.

[6] J. Peng, Y. Huang and G. Luo, "Robot path planning based on improved A* algorithm," Cybernetics and Information Technologies, vol. 15, no. 2, pp. 171-180, 2015.

[7] S. M. LaValle and J. J. Kuffner, “Randomized kinodynamic planning,” International Journal of Robotics Research, vol. 20, no. 5, pp. 378-400, 2001.

[8] J. J. Kuffner and S. M. LaValle, “RRT-connect: an efficient approach to single-query path planning,”, in Proc. IEEE International Conference on Robotics and Automation(ICRA), pp. 995-1001, 2000.

[9] H. Yang, H. Li, K. Liu, W. Yu and X. Li, “Research on path planning based on improved RRT-connect algorithm,” in Proc. Chinese Control and Decision Conference (CCDC), pp. 5707-5712, 2021.

[10] J. Rückin, L. Jin and M. Popović, “Adaptive informative path planning using deep reinforcement learning for UAV-based active sensing,” in Proc. International Conference on Robotics and Automation (ICRA), pp. 4473-4479, 2022.

[11] Y. Han, I. H. Zhan, W. Zhao et al, “Deep reinforcement learning for robot collision avoidance with self-state-attention and sensor fusion,” IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 6886-6893, 2022.

[12] T. Kulvicius, S. Herzog, M. Tamosiunaite et al, “Finding optimal paths using networks without learning—unifying classical approaches,” IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 7877-7887, 2022.

[13] Z. Rao, Y. Wu, Z. Y et al, “Visual navigation with multiple goals based on deep reinforcement learning,” IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 12, pp. 5445-5455, 2021.

[14] T. Haarnoja, A. Zhou, A. Abbeel et al, “Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor,” International Conference on Machine Learning, pp. 1861-1870, 2018.

[15] J. Gao, W. Ye, J. Guo et al, “Deep reinforcement learning for indoor mobile robot path planning,” Sensors, vol. 20, no. 19, pp.5493, 2019.

[16] G. P. Kontoudis and K. G. Vamvoudakis, “Kinodynamic Motion Planning With Continuous-Time Q-Learning: An Online, Model-Free, and Safe Navigation Framework,” IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 12, pp.3803-3817, Dec. 2019.

[17] A. Francis, A. Faust, H.-T. L. Chiang et al, “Long-range indoor navigation with PRM-RL,” IEEE Transactions on Robotics, vol. 36, no. 4, pp.1115-1134, Aug. 2020.

[18] X. Li, H. Liu and M. Dong, “A General Framework of Motion Planning for Redundant Robot Manipulator Based on Deep Reinforcement Learning,” IEEE Transactions on Industrial Informatics, vol. 18, no. 8, pp. 5253-5263, Aug. 2022.

[19] F. Ying, H. Liu, M. Dong and Rongxin Jiang. “Extensively explored and evaluated actor-critic with expert-guided policy learning and fuzzy feedback reward for robotic trajectory generation,” IEEE Transactions on Industrial Informatics, vol. 18, no. 11, pp. 7749-7760, Nov. 2022.