Journal of Information Systems Engineering and Management
2025, 10(23s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Review on Optimizing Robotic Navigation with Deep Reinforcement Learning Algorithms

[1]Mrs. Swati Mohan Bankar, [2]Dr. Rahul Shivaji Pol
[1]*Research Scholar, Electronics and Telecommunication Department Vishwakarma Institute of Information Technology Pune, India*
swati.221p0079@viit.ac.in
[2]*Associate Professor, Electronics and Telecommunication Department Vishwakarma Institute of Information Technology Pune, India*
rahul.pol@viit.ac.in

ARTICLE INFO

Received: 20 Dec 2024

Revised: 01 Feb 2025

Accepted: 19 Feb 2025

ABSTRACT

Robotic path planning is a critical aspect of autonomous robot navigation, enabling robots to efficiently navigate in complex environments while avoiding obstacles and reaching their intended destinations. Traditional path planning algorithms often struggle with intricate and dynamic environments due to their reliance on predefined maps assumptions about the environment's behavior. In recent few years, deep reinforcement learning (DRL) has come up as a promising approach for enhancing robotic path planning. RL techniques allow robots to learn optimal or near-optimal paths through trial-and-error interactions with their surroundings, adapting to changing environments and unforeseen obstacles.

This review paper provides overview of the progress in enhancing robotic path planning using reinforcement learning. We categorize the existing research depending on the types of RL algorithms employed, such as Q-learning, policy gradients, and actor-critic methods, among others.

By synthesizing recent research findings, this review paper offers insights into the current state of enhancing robotic path planning using reinforcement learning, identifies open challenges, and suggests potential directions for future research. The synergy between RL and robotic path planning holds great promise for revolutionizing autonomous navigation, enabling robots to travel in complex and dynamic environments with unprecedented efficiency and adaptability.

Keywords: Deep reinforcement learning, optimal path, adaptive navigation, dynamic environment.

I. INTRODUCTION

Robotic path planning is a fundamental task in the area of robotics that involves determining the optimal or feasible path for a robot to navigate from a starting point to a destination avoiding obstacles and adhering to various constraints. This task is essential for enabling robots to autonomously operate in diverse environments, ranging from factory floors to outdoor terrains and even confined indoor spaces. Traditional methods for path planning [1], such as A* and Dijkstra's algorithm, rely on pre-existing maps and geometric information, which can be limiting in dynamic and uncertain environments.

Traditional path-planning algorithms have been foundational in many applications, but they do come with limitations that necessitate more adaptive and efficient approaches:

- Computational Complexity: Traditional algorithms may struggle with scalability, especially in large and dynamic environments. As the number of nodes or obstacles increases, the computational resources required for planning grow exponentially.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

- Static Environment Assumption: Many traditional algorithms assume a static environment where the obstacles do not change over time. In dynamic environments, where obstacles may move or new obstacles may appear, these algorithms may not be able to adapt quickly enough to find an optimal or feasible path.
- Memory Intensive: Some algorithms require storing information about the entire map or grid, leading to high memory requirements, especially in large-scale scenarios. This can become impractical in resource-constrained systems or real-time applications.
- Optimality vs. Efficiency Trade-off: Traditional algorithms often prioritize finding the optimal path based on certain criteria (e.g., shortest path). However, in dynamic environments or real-time scenarios, it may be more important to find a reasonably good path quickly rather than spending excessive time searching for the absolute optimal solution.
- Limited Adaptability: Traditional algorithms typically rely on predefined heuristics or cost functions, which may not capture the complexities of real-world scenarios effectively. They may struggle to adapt to diverse terrain types, varying robot dynamics, or uncertain environmental conditions.
- Incomplete Information Handling: In scenarios where the robot has incomplete or uncertain information about the environment, traditional algorithms may struggle to make informed decisions. They often assume perfect knowledge of the environment, which is not always the case in practical applications.
- High-dimensional State Spaces: In multi-agent systems or robotics applications with high-dimensional state spaces, traditional algorithms may encounter difficulties in efficiently exploring the search space and finding feasible paths.

To address these limitations, there is a growing need for more adaptive and efficient path-planning approaches. These approaches may incorporate machine learning techniques, such as reinforcement learning or imitation learning, to adapt to changing environments and learn from experience.

Reinforced learning is a machine learning third paradigm that allows agents, here, robots, to learn best actions through interactions with surrounding environment. Unlike traditional methods, RL-based path planning enables robots to adapt and learn from experience, making them better suited to handle complex and changing surroundings [6]. The integration of RL into robotic path planning addresses several limitations of conventional methods. RL-based approaches can handle dynamic environments where obstacles might move or new obstacles might appear. They can also adapt to variations in robot dynamics and sensory information, thus offering a level of flexibility and adaptability that was previously hard to achieve [10]. By learning from trial and error, RL-based path planning can guide to more efficient and intelligent robot navigation, even in scenarios where the optimal path is not explicitly known [14, 15].

Through this review, we seek to offer insights into the current state of RL-enhanced robotic path planning, identify key research trends, and present potential directions for future exploration. The integration of reinforcement learning into path planning has the potential to revolutionize autonomous robotic navigation, making robots more adaptable, efficient, and capable of navigating complex and dynamic real-world environments [18, 19, 20].

II. REINFORCEMENT LEARNING FOR ROBOTIC PATH PLANNING:

The main premise of reinforcement learning lies in its ability to allow robots to learn best actions through interactions with surrounding environment. RL algorithms consist of agents, actions, states, and rewards, where the agent learns to take different actions in different states to get maximum reward over time.

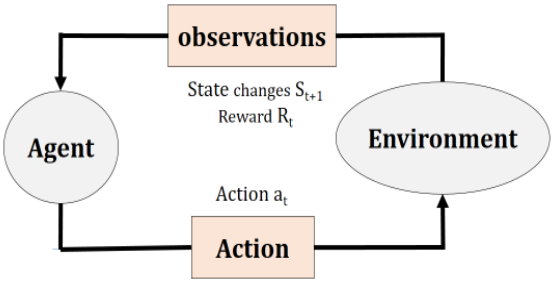

Fig 1. Block diagram of Reinforcement Learning*

*Note: Figure is adapted from MIT 6.S191: Reinforcement Learning lecture

Action: It is a move that agent can make in environment. It can be discrete or in continuous space.

Action Space: It is a set of possible actions that agent can take in environment.

Observations: It means how environment responses back to the agent.

State: It is the situation where agent find itself after taking action. It refers to the representation of the current situation or condition of an environment at a given time step. It encapsulates all the relevant information about the environment that an RL agent needs to make decisions and take actions to achieve its goals.

Reward: In reinforcement learning, rewards are used to guide the learning process of an artificial agent. The agent interacts with an environment and receives positive or negative rewards based on the actions it takes. The goal is to learn a policy that increases the cumulative rewards eventually

$$R_t = \sum_{i=t}^{\infty} r_i = r_t + r_{t+1} + - - + r_{t+n} + - - \quad (1)$$

$$R_t = \sum_{i=t}^{\infty} \gamma^i r_i \quad (2)$$

Where, γ =discount factor $0 < \gamma < 1$

In the context of robotic path planning, RL enables robots to explore their surroundings, learn from their experiences, and adapt their navigation strategies accordingly

. III. CATEGORIES OF RL ALGORITHMS FOR PATH PLANNING:

This section delves into the categorization of RL algorithms commonly employed in robotic path planning. Classification of Reinforcement learning Algorithm can be done in different ways. Dong Han et al., [25] categorized them into A. Value-based,

B. Policy-based, and

C. Actor–critic algorithms.

In a same way algorithms are arranged in this paper. Whereas Keerthana Sivamayi et al., [26] classified RL algorithm based on three approaches: 1. Monte Carlo 2. Temporal difference 3. Dynamic Programming.

A. Value-Based Methods:

These algorithms focuses on estimating the value function, which gives the expected reward starting from a state and following a certain policy.

These algorithms obtains Q function which captures the expected total future reward an agent in state S, can receive by executing a certain action. Ultimately the agent needs action value function which gives maximum reward. We will select policy

$$\pi(s) = \max_a Q(S, a) \quad (3)$$

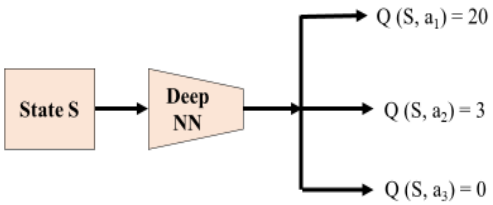

Fig* 2. Value based algorithm training policy

*NOTE: FIGURE IS ADAPTED FROM MIT 6.S191: REINFORCEMENT LEARNING LECTURE

In fig 2, we will select action from action a_1, a_2 or a_3 depending upon which action will give maximum reward. As action a_1 has maximum Q value, it will be selected.

$$\therefore \pi(s) = a_1$$

Here are enlisted few algorithms from this category.

A.1 Q-learning

It is a fundamental RL algorithm, facilitates learning of action-value functions and can be used to find optimal paths. Q-Learning is a model-free RL algorithm that learns action-value functions iteratively. It updates the Q-values depending on the observed rewards and the highest Q-value of the next state. Q-Learning is effective for discrete action spaces and can handle environments with a small number of states.

Mihai et al., [2] combined Q learning algorithm with NN for robot path planning in static and dynamic obstacle environment. Results are checked in virtual as well as real time environment. It achieved good convergence ratio. Problem of local minima is also avoided.

Zhen Shi et al [20] proposed an improved QL algorithm which combines the previous knowledge for initiating the starting Q-value. In this way agent was able to easily reach towards target earliest, eliminating number of invalid iterations. The greedy factor ε was adjusted at run time.

A.2 Deep Q-Networks (DQN):

DQN is one of the pioneering algorithms in DRL, which combines Q-learning with deep neural networks. It learns to estimate the action-value or Q-function of an agent, which represents the predicted reward of taking a specific action in a given state. DQN extends Q-Learning to handle high-dimensional state spaces using deep neural networks. It employs a target network and experience replay to stabilize and improve the learning process.

DQN is suitable for path planning tasks with large state spaces and continuous action spaces.

Jing Xin., [3] et al., approximated the state-action value Q function using DQN. This network has four frames of preprocessed image as input, and the output of DQN is the Q value of every single possible action of the mobile robot. Algorithm is evaluated virtually on environment of Deep Mind Lab. This paper introduced a novel end-to-end path planning approach using DRL to enable the robot to directly derive optimal actions from raw visual perception without the need for handcrafted features or feature matching.

Hyansu Bae et al., [8] compensated shortcomings of classical learning algorithms by applying DQN for multi robot path planning. The learning algorithm is made more robust by combining it with CNN algorithm, which is made analysis of situation more effective. In both static and dynamic obstacle environments simulated in the study, the number and placement of obstacles are randomized. The proposed algorithm is then evaluated against A* in static environment and against D* algorithm in dynamic environment to assess its performance. Based on the simulation findings, the method consistently exhibits a narrower search range compared to alternative approaches. Furthermore, there's a likelihood of approximately 34.76% that the path generated by this method matches or closely resembles the path generated by the A* algorithm

Yang Yang et al., [12] done path planning for multirobot. DQN algorithm is used with some modifications. DQN Algorithm has advantage over Q algorithm of having less dimensionality of input data. But it faces problem of sluggish convergence rate and more randomness. This two shortcoming is avoided using previous knowledge to initialise the Q-value table.

Tomoaki Nakamura et al., [23] proposed DQN for two purposes. A path planning method with turnabouts finds hard to design a proper controlling policy depending on environmental information. The RL-based algorithm is used to obtain the control strategy by interacting the robot with its surrounding environment, without previous training datasets. It is possible that using neural networks, DQN is able to handle high dimensional input data. This method takes continuous values as input data which has sensor data and velocity of robot. Simulations are performed successfully on Gazebo simulator with Robot Operating System (ROS).

Vo Thanh Ha et al., [31] compares Q-learning and Deep Q-learning algorithms for obstacle avoidance in dynamic environments. Author presented quantitative results comparing the performance of QL and DQL algorithms in terms of their effectiveness in obstacle avoidance in dynamic environments. Experimental setup was developed in Gazebo simulator. Robot has to reach the specified destination in the environment where some static as well dynamic obstacles were present. For different experimental environment time required to reach destination by robot

implemented with DQ learning was less. The implementation of the DQL showcased enhancements in various aspects, including reduced computation time, faster convergence time, improved accuracy in trajectory planning, and more effective obstacle avoidance capabilities

A.3 Double Deep Q-Networks (Double DQN):

It is an enhancement to the traditional DQN algorithm that addresses the issue of overestimation of Q-values in the Q-learning process. In standard DQN, the same network is used to both select the best action and estimate its Q-value. This can lead to overestimation of Q-values, which can result in suboptimal policies. Double DQN mitigates this issue by decoupling the action selection and Q-value estimation processes.

Xiaoyun Lei et al., [4] successfully implemented Double DQN for dynamic path planning in unknown environment. Design of reward and punishment function is done to deal with training stage instability and environment state space sparsity. Training method also designed in a same way. Starting position and target position is adjusted dynamically in unknown environment. By updating NN and the increasing greedy rule probability, the local space searched by the agent is increased. Dynamic environment is created using Pygame module in Python.

Huiyan Han et al., [25] introduced an enhanced Double DQN path planning algorithm. Here high dimensionality problem of data is solved by discretizing the high-dimensional Li DAR data into a state space to minimize redundant information. Convergence speed is increased using Epsilon Greedy algorithm. ROS is used to real-world implementation

A.4 Dueling Deep Q-Networks (Dueling DQN):

It is an extension of the traditional DQN algorithm that aims to improve the efficiency and stability of Q-learning, particularly when dealing with environments where the values of different actions might vary significantly. Dueling DQN separates the Q-value estimation into two streams: one for estimating the value of the state V(s) and another for estimating the advantage of each action A(s, a). This separation allows the algorithm to better understand the importance of each action independently of the state value. Q Value function becomes:

$$Q(s, a) = V(s) + A(s, a) \quad (4)$$

Dueling DQN is particularly useful when navigating complex environments where different actions have varying impacts on the Q-values. It can help agents learn more efficiently and effectively by disentangling state values from action advantages.

Xiaogang Ruan et al., [7] combined Algorithm of Double DQN and Dueling DQN. With this combination performance is improved and problem of estimation is eliminated. The performance of Algorithm is checked in simple as well as complex indoor environment. This is done with the help of Gazebo simulator.

A.5 Rainbow DQN

It is an advanced variant of the Deep Q-Network (DQN) algorithm that integrates several improvements to address specific challenges associated with DQN. It combines various enhancements from different DQN variations to create a more robust and efficient algorithm for reinforcement learning.

The name 'Rainbow' in Rainbow DQN stands for 'A Unified Architecture for DQN' highlighting its integration of multiple techniques into a cohesive framework. By combining the strengths of various DQN variants, Rainbow DQN aims to achieve better learning stability, improved performance, and more efficient convergence.

Miguel Quinones-Ramırez et al., [23] implemented variants of the deep Q networks method to achieve maples navigation of mobile robot. The Dueling DQN and rainbow algorithms are used in obstacle avoidance as well as in goal-oriented navigation task. They found that Rainbow DQN allowed to reach more targets and collide less during training than the D3QN agents. Results achieved collision rate of 41.83% and target-reaching rate of 96.9% for the goal-oriented navigation task.

B. Policy gradient methods:

These methods, on the other hand, optimize the policy directly and are well-suited for continuous action spaces. It doesn't check the value of action but it checks the probability of selecting action which will give best performance value. In fig 3, we will have policy for selection as,

$$\pi(s) \sim P(a|s) \quad (5)$$

As probability of action a_1 is more,

$$\pi(s) = a_1$$

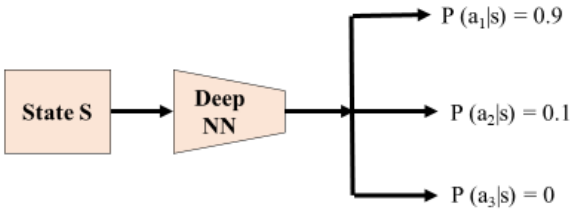

Fig 3. Policy Gradient algorithm training policy

Few algorithms from this category are described below.

B.1 Reinforce (Monte Carlo Policy Gradient):

REINFORCE is a foundational policy gradient algorithm that uses Monte Carlo sampling to estimate the gradient of the expected reward as per the policy parameters. It updates the policy parameters in the direction that increases the expected reward. REINFORCE is simple and intuitive but can have high variance in gradient estimates.

B.2 Proximal Policy Optimization (PPO):

PPO is a policy optimization algorithm that aims to find policy updates while staying within a certain "trust region" to ensure stable learning.

B.3 Trust Region Policy Optimization (TRPO):

TRPO optimizes policies while ensuring that the policy updates do not deviate significantly from the current policy, helping to maintain stability during learning.

C. Actor-Critic methods:

These methods combine policy-based and value-based approaches, keeping a balance between stability and exploration. Model-based RL techniques involve learning a model of the environment dynamics to aid decision-making.

C.1 Advantage Actor-Critic (A2C):

This is a synchronous variant of the Actor-Critic algorithm where multiple agents learn in parallel to improve sample efficiency.

C.2 Asynchronous Advantage Actor-Critic (A3C):

This is an asynchronous variant of A2C, where multiple agents learn asynchronously and share their experiences to accelerate learning.

C.3 Deep Deterministic Policy Gradient (DDPG):

DDPG is a reinforcement learning algorithm used for solving continuous action space problems in the field of artificial intelligence and machine learning. It is an extension of the actor-critic framework and is particularly well-suited for tasks where the action space is continuous, such as robotics control, autonomous driving, and continuous control in games.

Jinglun Yu et al., [9] found that DDPG, which is based on neural networks and hierarchical reinforcement learning, performed better in each aspects as compared to any other path planning algorithms. With this algorithm convergence time is lessened by 91% in comparision with the Q-learning algorithm and smoothness of the path got better by 79%. The simulation environment is created using the multimedia framework pyglet under Python.

Yuansheng Dong et al., [11] proposed a path planning method which is based on an improved DDPG reinforcement learning algorithm. In this algorithm little amount of a previous knowledge is used to improve the training of DRL. This will minimize number of trial and error steps. Here adaptive exploration method based on the ε greedy algorithm is used.

C.4 Twin Delayed Deep Deterministic Policy Gradient (TD3):

While primarily used for continuous action spaces, TD3 is also an example of an Actor-Critic algorithm. It optimizes a deterministic policy using twin value networks and delayed policy updates.

Papers which are reviewed in this paper are selected from 2016 as implementation of RL algorithm has just started. The first implementation of RL Algorithms was in ATARI games in 2015. All papers are compared on the basis of three points:

1. Simulation environment used for implementation of RL algorithm and for comparing results obtained from each algorithm.
2. Improvement in result after applying the algorithm
3. Limitations of implemented algorithm

TABLE 1: A COMPARATIVE SUMMARY OF PREVIOUS WORKS BASED ON ABOVE MENTIONED POINTS

Paper	***RL Algorithm***	***Simulation Environment***	***Achievements***	***Limitations***
2	Q learning & NN	C++/ VRML and MATLAB	It achieves good convergence ratio. Robot control at desired speed is possible	Unable to self-tune parameters Can't deal with dynamic obstacles
3	Deep Q Network	Seekavoid arena 01 environment on DeepMind Lab platform	Effective, optimal and end-to-end robotic path planning technique.	Only three actions left, right and forward are possible which won't give solution in real environment.
4	Double Q-network (D3QN)	Pygame module for creating dynamic environment, ROS	able to navigate in unknown dynamic environment. flexible for local path planning with the LiDAR data input	Need to upgrade for more complex environment
5	(SA-CADRL)	Python	Fully autonomous navigation near about to human walking speed in a dynamic surroundings having many pedestrians. Designed for multi-Agent system.	Not implemented in real environment

Paper	***RL Algorithm***	***Simulation Environment***	***Achievements***	***Limitations***
7	Dueling Double DQN (D3QN)	Gazebo simulator	Combination of Double DQN and Dueling DQN increased. performance and problem of estimation is removed	Unable to navigate in dynamic environment
8	DQN with CNN algorithm	Python and C++.	Multi-robot path planning Applicable in a static as well as in a dynamic environment	Not implemented in real environment
9	Deep Deterministic Policy Gradient (DDPG)	multimedia framework pyglet under Python	Minimizes the path planning time, decreases the path steps, Lessens the convergence time and enhances smoothness path	Performance of algorithm for dynamic environmental changes is not verified Practical implementation is not verified
11	Improved Deep Deterministic Policy Gradient (DDPG)	Matlab platform to compare results	Works in dynamic obstacle environment Performs better than Q-learning and DDPG algorithm.	Practical implementation is not done
13	Deep Q-network (DQN)	TensorFlow deep learning tools, Python	Path planning for multi-robot system. Converges faster and excessive randomness is eliminated	Not applicable for dynamic environment
16	Layered-RQN	Nvidia GTX 1080 Ti using CuDNN & Caffe	Outperform in dynamic environment, Robust algorithm	Not applicable for multiple UAV. Real-time implementation is not done
17	Mobile robot Collision Avoidance Learning (MCAL)	Robot Operating System (ROS),	Excellent performance in static as well as in dynamic environment. Can work in multi robot	It cannot drive in a straight line. Unable to accelerate/decelerate while driving

Paper	***RL Algorithm***	***Simulation Environment***	***Achievements***	***Limitations***
			environment	
21	Improved Q Algorithm	Python library Tkinter	Using previous knowledge number of invalid iterations avoided. Dynamic adjustment in greedy factor	Parameter setting can be improved
22	Teaching learning based optimization (TLBO)	Robotic Operating System (ROS)	Less computation time for path finding. Minimization of the path length	Unable to navigate in dynamic environment. Can't deal non-convex and irregular shapes obstacles
23	D3QN and rainbow algorithm	Robotic Operating System (ROS)	Obstacle avoidance rate and target reaching rate are improved.	More work can be done for optimal performance
24	Deep Q-network (DQN)	Robot Operating System (ROS)	Designed for takeing turnabouts on narrow roads. Minimization of the path length	Can't navigate in environments which has difference from the learning environment. Applicable for discrete velocity commands
27	an enhanced DDQN path planning approach	Robot Operating System (ROS)	Problem of high dimensionality is solved. Results are verified in real world and virtual environment	Scope of improvement in speed and stability of algorithm
31	QL And DQL	ROS	Quantitative comparison of algorithms in terms of run time in dynamic environment	Control facilities like voice control, lane identification can be added using advanced algorithms

From above table it is observed that various RL algorithm are implemented successfully for path planning in static as well as in dynamic environment. Earlier algorithm implemented were value based. Later policy based algorithm

were preferred where continuous action space was required. Implementation in simulated environment is preferred. Robot Operating System (ROS) is commonly used for simulation. MATLAB is used for result comparison. Python is also used in some papers.

TABLE 2: QUANTITATIVE COMPARISION OF PROMINANTLY IMPLEMENTED RL ALGORITHMS BASED ON DIFFERENT PARAMETERS LIKE PATH EFFICIENCY, COMPUTATIONAL EFFICIENCY AND ROBUSTNESS.

Algorithm	***Succe-ss Rate (%)***	***Path Efficiency (Avg. Path Length)***	***Training Time (Hours)***	***Computat-ional Efficiency (FPS)***	***Robustness (Dynamic Obstacles)***
Q-Learning	85	1.2x optimal path	50	30	Moderate
Deep Q-Network (DQN)	90	1.1x optimal path	100	25	High
Proximal Policy Optimizatio n (PPO)	95	1.05x optimal path	120	40	Very High
Trust Region Policy Optimizatio n (TRPO)	92	1.08x optimal path	110	35	High
Advantage Actor-Critic (A2C)	93	1.07x optimal path	90	45	Very High
Deep Deterministi c Policy Gradient (DDPG)	94	1.06x optimal path	95	50	Very High

Table 2 highlights that while simpler algorithms like Q-learning can be effective in static or less complex environments, more advanced algorithms like PPO, A2C, and DDPG offer superior performance across most parameters. These advanced algorithms are particularly suitable for dynamic and complex environments where high success rates, path efficiency, and robustness are critical.

FUTURE DIRECTIONS AND CONCLUDING REMARKS:

In summary, this review paper investigates use of DRL in enhancing robotic path planning. By categorizing RL algorithms, addressing challenges, exploring strategies, and presenting case studies, this paper offers a comprehensive understanding of the field's current state. The integration of reinforcement learning into path planning great potential to reshape the capabilities of autonomous robots, enabling them to navigate complex and dynamic environments with unprecedented intelligence and adaptability.

Through a systematic, this review has showcased the versatility of DRL in addressing the intricacies of path planning. Moreover, the paper emphasized the successful transferability of learned policies across different robotic platforms and environments suggests the potential for creating adaptable and versatile navigation systems with broad applicability.

REFERENCES

[1] B..K. Patle, Ganesh Babu L, Anish Pandey, D.R.K. Parhi, A. Jagadeesh, A review: On path planning strategies for navigation of mobile robot, Defence Technology, Volume 15, Issue 4, 2019, Pages 582-606, ISSN 2214-9147, https://doi.org/10.1016/j.dt.2019.04.011.

[2] Mihai Duguleana, Gheorghe Mogan, Neural networks based reinforcement learning for mobile robots obstacle avoidance, Expert Systems with Applications, Volume 62, 2016, Pages 104-115, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2016.06.021. (https://www.sciencedirect.com/science/article/pii/S0957417416303001)

[3] J. Xin, H. Zhao, D. Liu and M. Li, "Application of deep reinforcement learning in mobile robot path planning," 2017 Chinese Automation Congress (CAC), Jinan, China, 2017, pp. 7112-7116, doi: 10.1109/CAC.2017.8244061

[4] Xiaoyun Lei, Zhian Zhang, Peifang Dong, "Dynamic Path Planning of Unknown Environment Based on Deep Reinforcement Learning", Journal of Robotics, vol. 2018, Article ID 5781591, 10 pages, 2018. https://doi.org/10.1155/2018/5781591

[5] Y. F. Chen, M. Everett, M. Liu and J. P. How, "Socially aware motion planning with deep reinforcement learning," 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 2017, pp. 1343-1350, doi: 10.1109/IROS.2017.8202312.

[6] H. Nguyen and H. La, "Review of Deep Reinforcement Learning for Robot Manipulation," 2019 Third IEEE International Conference on Robotic Computing (IRC), Naples, Italy, 2019, pp. 590-595, doi: 10.1109/IRC.2019.00120.

[7] X. Ruan, D. Ren, X. Zhu and J. Huang, "Mobile Robot Navigation based on Deep Reinforcement Learning," 2019 Chinese Control And Decision Conference (CCDC), Nanchang, China, 2019, pp. 6174-6178, doi: 10.1109/CCDC.2019.8832393.

[8] Bae, H.; Kim, G.; Kim, J.; Qian, D.; Lee, S. Multi-Robot Path Planning Method Using Reinforcement Learning. *Appl. Sci.* 2019, *9*, 3057. https://doi.org/10.3390/app9153057

[9] Yu J, Su Y and Liao Y (2020) The Path Planning of Mobile Robot by Neural Networks and Hierarchical Reinforcement Learning. Front. Neurorobot. 14:63. doi: 10.3389/fnbot.2020.00063

[10] T. T. Nguyen, N. D. Nguyen and S. Nahavandi, "Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications," in IEEE Transactions on Cybernetics, vol. 50, no. 9, pp. 3826-3839, Sept. 2020, doi: 10.1109/TCYB.2020.2977374.

[11] Y. Dong and X. Zou, "Mobile Robot Path Planning Based on Improved DDPG Reinforcement Learning Algorithm," 2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 2020, pp. 52-56, doi: 10.1109/ICSESS49938.2020.9237641.

[12] L. Liu, D. Dugas, G. Cesari, R. Siegwart and R. Dubé, "Robot Navigation in Crowded Environments Using Deep Reinforcement Learning," 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 2020, pp. 5671-5677, doi: 10.1109/IROS45743.2020.9341540.

[13] Yang, Y., Juntao, L., & Lingling, P. (2020). “Multi-robot path planning based on a deep reinforcement learning dqn algorithm”, CAAI Transactions on Intelligence Technology, 5(3), 177-183. https://doi.org/10.1049/trit.2020.0024

[14] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," in Tsinghua Science and Technology, vol. 26, no. 5, pp. 674-691, Oct. 2021, doi: 10.26599/TST.2021.9010012.

[15] Sánchez-Ibáñez, J.R.; Pérez-del-Pulgar, C.J.; García-Cerezo, A. Path Planning for Autonomous Mobile Robots: A Review. Sensors 2021, 21, 7898. https://doi.org/10.3390/s21237898

[16] Tong GUO, Nan JIANG, Biyue LI, Xi ZHU, Ya WANG, Wenbo DU, UAV navigation in high dynamic environments: A deep reinforcement learning approach, Chinese Journal of Aeronautics,

[17] Volume 34, Issue 2, 2021, Pages 479-489, SSN 1000-9361, https://doi.org/10.1016/j.cja.2020.05.011. (https://www.sciencedirect.com/science/article/pii/S1000936120302247)

[18] Choi, J., Lee, G. & Lee, C. Reinforcement learning-based dynamic obstacle avoidance and integration of path planning. Intel Serv Robotics 14, 663–677 (2021). https://doi.org/10.1007/s11370-021-00387-2

[19] Zhou, C., Huang, B. & Fränti, P. A review of motion planning algorithms for intelligent robots. J Intell Manuf 33, 387–424 (2022). https://doi.org/10.1007/s10845-021-01867-z

[20] Gasteiger, N., Hellou, M. & Ahn, H.S. Factors for Personalization and Localization to Optimize Human–Robot Interaction: A Literature Review. Int J of Soc Robotics 15, 689–701 (2023). https://doi.org/10.1007/s12369-021-00811-8

[21] S. Aradi, "Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 2, pp. 740-759, Feb. 2022, doi: 10.1109/TITS.2020.3024655.

[22] Shi Z, Wang K, Zhang J (2023) Improved reinforcement learning path planning algorithm integrating prior knowledge. PLoS ONE 18(5): e0284942. https://doi.org/10.1371/journal. pone.0284942

[23] Sabiha, A.D., Kamel, M.A., Said, E. et al. Real-time path planning for autonomous vehicle based on teaching–learning-based optimization. Intel Serv Robotics 15, 381–398 (2022). https://doi.org/10.1007/s11370-022-00429-3

[24] Quinones-Ramirez, Miguel, Jorge Ríos-Martínez and Víctor Uc Cetina. “Robot path planning using deep reinforcement learning.” ArXiv abs/2302.09120 (2023): n. pag.

[25] T. Nakamura, M. Kobayashi and N. Motoi, "Path Planning for Mobile Robot Considering Turnabouts on Narrow Road by Deep Q-Network," in IEEE Access, vol. 11, pp. 19111-19121, 2023, doi: 10.1109/ACCESS.2023.3247730.

[26] Han, D.; Mulyana, B.; Stankovic, V.; Cheng, S. A Survey on Deep Reinforcement Learning Algorithms for Robotic Manipulation. Sensors 2023, 23, 3762. https:// doi.org/10.3390/s23073762

[27] Sivamayil, K.; Rajasekar, E.; Aljafari, B.; Nikolovski, S.; Vairavasundaram, S.; Vairavasundaram, I. A Systematic Study on Reinforcement Learning Based Applications. Energies 2023, 16, 1512. https://doi.org/10.3390/ en16031512

[28] Han, H.; Wang, J.; Kuang, L.; Han, X.; Xue, H. Improved Robot Path Planning Method Based on Deep Reinforcement Learning. Sensors 2023, 23, 5622. https:// doi.org/10.3390/s23125622

[29] Qin, H.; Shao, S.; Wang, T.; Yu, X.; Jiang, Y.; Cao, Z. Review of Autonomous Path Planning Algorithms for Mobile Robots. Drones 2023, 7, 211. https://doi.org/ 10.3390/drones7030211

[30] Mohsen Soori, Behrooz Arezoo, Roza Dastres, Artificial intelligence, machine learning and deep learning in advanced robotics, a review, Cognitive Robotics, Volume 3, 2023, Pages 54-70, ISSN 2667-2413, https://doi.org/10.1016/j.cogr.2023.04.001.

[31] Saeid Nahavandi, Roohallah Alizadehsani, Darius Nahavandi, Chee Peng Lim, Kevin Kelly, Fernando Bello, Machine learning meets advanced robotic manipulation, Information Fusion, Volume 105, 2024, 102221, ISSN 1566-2535, https://doi.org/10.1016/j.inffus.2023.102221. (https://www.sciencedirect.com/science/article/pii/S1566253523005377)

[32] Ha, V.T.; Vinh, V.Q. Experimental Research on Avoidance Obstacle Control for Mobile Robots Using Q-Learning (QL) and Deep Q-Learning (DQL) Algorithms in Dynamic Environments. Actuators 2024, 13, 26. https://doi.org/10.3390/act13010026

[33] S Karambelkar, R Khaire, R Ghule, P Hiray, S Kulkarni, A Somatkar, Design and Analysis of Automatic Tripod Style Horizontal Multi Bobbin Wire Winder, 2022 6th International Conference On Computing, Communication, Control And Automation (ICCUBEA)

[34] K Kolhe, AA Somatkar, MS Bhandarkar, KB Kotangale, SS Ayane, Applications and Challenges of Machine Learning Techniques for Smart Manufacturing in Industry 4.0, 2023 7th International Conference On Computing, Communication, Control And Automation (ICCUBEA)