Proceedings of the 11th RSI International Conference on Robotics and Mechatronics (ICRoM 2023), Dec. 19-21, 2023, Tehran, Iran

Investigating the Performance and Reliability, of the Q-Learning Algorithm in Various Unknown Environments

Amirhossein Nourian
School of Mechanical Engineering
Tarbiat Modares University
Tehran, Iran
ORCID: 0000-0002-1457-000X
Email: a_nourian@modares.ac.ir, ahn.paf@gmail.com

Dr. Majid Sadedel
School of Mechanical Engineering
Tarbiat Modares University
Tehran, Iran
Email: majid.sadedel@modares.ac.ir

***Abstract*— Self-reinforcement algorithms, especially Q-learning and value iteration have been popular and effective for navigating mobile robots. To provide a deeper insight through their performance and usage, this paper investigates solving a simple Q-Learning problem for various working environments for mobile robots. No matter how quick or reliable they are, all temporal difference algorithms employ iterative methods to produce desired results. For this reason, in this paper, we looked at the relationship between episode steps, mesh sizes, and—most importantly—the impact of the environment on both convergence rate and solution reliability. As previously indicated, the majority of the conclusions of this study about the relationship between computation cost and environment and also dependability can be transferred to more sophisticated temporal difference based algorithms because all methods are iterative. As many robotic applications for reinforcement learning have two or three-dimensional state space for path planning, knowing the approximate convergence episode count would be a useful tool for researchers to tackle non-optimal episodes, especially on embedded systems with computational constraints. The content also discussed the relationship between three mesh sizes and tried to guide researchers to implement their reinforcement learning pipeline more effectively. As shown by this work, the performance and reliability of iterative reinforcement learning algorithms depend deeply on the environment. In certain cases, increasing the number of episodes would not alter the results entirely. Therefore, when solving these kinds of maps, researchers would refrain from using Q-learning techniques.**

Keywords*— *Reinforcement Learning, Q-Learning, Path Planning

I. INTRODUCTION

Mobile robot navigation is one of the most interesting topics for researchers. Autonomous robots and cars use various algorithms, sensors, and processes to avoid crashing into obstacles, reach the goal, follow the wall, or combine all these actions to find a policy that can fulfill all these aims simultaneously. Reinforcement learning algorithm has been used for this purpose widely in both academic and industrial context.

In reinforcement learning (RL), agents learn to make good decisions through interaction with their environment to get maximum reward. Such methods are used in object tracking, games, and recommendation systems and often involve online learning in which observations arrive with volume and variety. Both value iteration and Q-learning algorithms have been used in various applications as a robust way to solve many common problems such as path planning and control systems.

Q-learning is one of the model-free reinforcement learning algorithms that is implemented to find out the value of an action in a particular state. Q-learning also requires a finite Markov decision process to find an optimal policy in the sense of maximizing the expected value of the returned reward. However, one disadvantage of Q-Learning is that it needs a massive amount of memory, and it grows exponentially with each extra feature introduced to the state space [1]. The primary distinguishing characteristic of model-free reinforcement learning algorithms, like Q-learning, is that the agent lacks prior knowledge of state transition probabilities or rewards. Instead, the agent learns these aspects through trial and error, receiving rewards or penalties as feedback when transitioning between states.

In the field of reinforcement learning, the computational costs associated with training algorithms are a significant consideration. As such, researchers have delved into exploring the impact of varying environmental conditions on the convergence rate and reliability of these algorithms. In particular, this investigation aimed to understand how different numbers of episodes played out within specific circumstances can affect the overall performance and effectiveness of reinforcement learning algorithms. By analyzing these factors, researchers sought to shed light on the optimal conditions for achieving reliable convergence rates while mitigating computational costs.

The Q-Learning algorithm relies on two primary factors for optimal performance: carefully tuning the hyperparameters during implementation and selecting an appropriate number of training episodes to achieve the best solution. In many robotic pathfinding scenarios, it is commonly observed that using a two or three-dimensional grid world setup is standard practice. Our team decided to assess the performance of Q-Learning on a two-dimensional grid. Using the number of elapsed episodes as our primary performance metric, we believe that the results can be generalized and applied to any Q-Learning or similar algorithms, regardless of hardware and software optimizations.

979-8-3503-0810-5/23/$31.00 ©2023 IEEE

This paper shed light on how different aspects of the robot working environment can affect convergence rate and reliability. For instance, robots performing tasks in farm fields containing many pathways to a goal as well as many scattered obstacles can deploy Q-Learning or other temporal difference algorithms most effectively, whereas, robots whose working environments contain a single, long, narrow path to a goal would suffer lowest or no convergence to goal.

The novelty of this work is to evaluate the convergence rate, number of episodes, reliability, and the key factors contributing to performance in 2D environments, which are commonly used in path planning for mobile robots. Our findings can be extended to similar iterative temporal difference reinforcement learning methods that share the same characteristics. In addition, 3D environments may exhibit similar behavior in terms of convergence rate and influential factors, such as the number of ways to reach the goal or its width. To cover all possible situations and factors in a 2D mesh grid and even their combinations, we have chosen three mesh grids: 4x4, 5x5, and 6x6 to cover all situations and factors possible in grid 2D mesh and even their combinations.

II. RELATED WORKS:

Researchers have been attempting to utilize various machine learning and AI algorithms in recent years to create fully autonomous agents that can be used in applications such as robotics and self-driving cars [2]–[4]. Reinforcement learning is one the most crucial ways of pathfinding and navigation in mobile robots [5]–[7]. Q-learning and value iteration methods are straightforward and well-known algorithms to accurately implement the navigation systems.

Q-learning as one of the most well-known algorithms in reinforcement learning has been implemented in many applications such as UAVs [6], [8], [9], traffic control [10], and various robots with many different dynamics [4], [11]. Authors in [1] proposed a low-cost Q-learning approach for a decentralized multi-agent environment that uses the novel method for a new robust Q-learning base model. The proposed model was able to handle a continuous space problem and does not require the model of the environment. Moreover, authors in [12], introduced Q-Learning-based path planning being able to shift policy according to priorities such as safety. In addition, authors in [13] developed an improved Q-learning algorithm with an accelerated convergence rate. This algorithm also explored the map more efficiently. Additionally, authors in [14] studied the experimental safety response mechanism for autonomous mobile robots using a Q-Learning algorithm and speech recognition. The speech recognition algorithm uses a neural network to identify the user's voice and trigger safe mode. Authors in [15] have also implemented the Q-Learning algorithm for robots in a digital assembly twin system. In the virtual space, a modified Q-Learning algorithm was proposed to solve the path planning problem in product assembly in three-dimensional space. Authors in [16] introduced a new improved Q-Learning algorithm that had enhanced parameters and convergence speed improvements. Authors in [17] studied accumulated errors caused by Q-Learning path planning algorithms using the error estimation method. Furthermore, authors in [18] implemented the Q-Learning algorithm for a mobile robot with a custom environment. In addition, authors in [19] studied wheeled mobile robots in partially unknown uneven trains. They first modeled the environment respecting terrain features including steps, slopes, and unevenness thereafter They used the A* algorithm to plan a global path for the robot based on the partially unknown map. Finally, the Q-learning algorithm was employed for obstacle avoidance and local path planning. Authors in [20] used computer vision-based path planning in robotic arms. The proposed algorithm was fed by two images from two views to obtain accurate spatial coordinates of objects in real time. Next, the Q-Learning algorithm is used to determine the sequence of simple actions. Finally, a trained neural network is used to determine a sequence of joint angles. There are too many review papers available regarding Q-learning and motion planning for mobile robots in the literature [21]–[23].

OpenAI Gym is considered a great tool to implement reinforcement learning research that has gained popularity in the machine learning community. This tool can even be combined with ROS and Gazebo environment [24]. Authors in [25] simulated two gym environments, FrozenLake and TrafficLight, using gym environment API. Using a deep Q-Learning algorithm with a stable baseline, they have investigated the evolution of Q values for individual states and thus the performance. Authors in [26] used four Gym environments namely HalfCheetah, Ant, Walke2d, and humanoid to benchmark self-imitation learning via generalized lower bound Q-Learning. Moreover, Authors in [27] benchmark CQL to prior online methods using Gym online domains such as HalfCheetah-random, Hopper-random, and Walker2d-random. Authors in [28] used adaptive Q-Learning and single partition adaptive Q-Learning which are two efficient model-free episodic reinforcement learning in two Gym classical environments, Pendulum and Cartpole. Authors in [29] optimized agent training with deep Q-Learning in a Gym CarRacing environment. Authors in [30] had also taught the BipedalWalker robot in Gym using a deep Q-Learning algorithm. Furthermore, Authors [31] investigate the effect of learning rate on overestimation of connectionist Q-Learning which may lead to adverse effects such as poor performance and unstable learning. For this purpose, they combined Q-learning with a multilayer perceptron (MLP). Later, they implemented the algorithm in the Gym 2D Maze environment. There are too many similar researches that mostly focused on improving reinforcement algorithms in recent years [32]–[39].

As illustrated in Table 1, none of the recent papers investigate the Q-learning convergence by examining the effect of different environments.

Table 1 List of recent publication and their research focus in recent years

Application scenario	Ref	Algorithm	Simulation	real system	mesh	gym
Solving simple grid world	[40]	Value-iteration, and deep reinforcement using Keras-rl	Y	N	N	N
Moving cart pole and pendulum	[41]	Value-iteration	Y	N	N	Cartpole, Furuta pendulum
path planning	[37]	Deep Q-Learning, Q-Learning and SARSA	Y	Y	N	N
Obstacle avoidance	[36]	Neural Network, Q-Learning Algorithm	Y	Y	N	N

path planning	[32]	Q-learning, CQ-learning	Y	N	N	MUJUCO
path planning	[33]	Double Q-learning	Y	N	N	MUJUCO
Controlling another gym environment	[39]	Q-learning, random forests	Y	N	N	Blackjack, inverted pendulum
Controlling bipedal robot	[30]	Deep Q-learning	Y	N	N	Bipedal robot
path planning	[29]	Deep Q-learning	Y	N	N	CarRacing
path planning	[31]	Q-learning, double Q-learning	Y	N	N	Maze
gym scenarios	[35]	Q-learning, value iteration, deep Q-learning	Y	N	N	Cartpole
gym scenarios	[27]	CQ-learning	Y	N	N	Maze, Atari games
Following the wall	[3]	FLC_R-IDS	Y	Y	N	N
Collision avoidance	[2]	Deep neural network	Y	Y	N	N
path planning	[2]	Q-learning, IDQ	Y	Y	N	N
Navigating roundabouts	[42]	Q-learning	Y	N	N	N
path planning	[8]	Q-learning	Y	N	N	N
Obstacle avoidance	[7]	Q-learning	Y	N	N	N
path planning	[43]	Deep reinforcement learning	Y	Y	N	N
path planning	[44]	Q-learning	Y	N	N	N
path planning	[45]	Deep Q-learning	Y	Y	N	N
path planning	[1]	Q-learning	Y	N	N	N
Emergency path planning	[14]	Q-learning	Y	Y	N	N
path planning	[46]	EQ-learning	Y	Y	N	N
path planning	[47]	Q-learning	Y	N	N	N
path planning	[48]	Q-learning, deep learning	Y	N	N	N
path planning	[11]	Q-learning	Y	Y	N	N
path planning	[49]	EMQ-learning	Y	Y	N	N
path planning	[50]	DQN	Y	N	N	N
This work	-	Q-Learning, value iteration	Y	N	Y	FrozenLake

As said earlier, in robotic applications it is common to use the 2D and 3D settings for many robotic tasks whether it is manipulation, path planning, or even autonomy. Q-Learning and other similar temporal difference base algorithms are deployed widely to solve these complications. Until now, none of the recent papers investigated the performance of these algorithms based on the robot deployment environment.

III. THEORETICAL BACKGROUND:

This section aims to enhance the comprehension of the proposed approach by providing a concise review of relevant theoretical concepts. First, we will cover path and trajectory planning, followed by an explanation of reinforcement learning, including the Q-Learning algorithm. Additionally, key terminology will be introduced to ensure a comprehensive understanding of the topic.

A. Path Planning:

In robotics, path planning, and trajectory planning are becoming increasingly important. Using path planning, a robot can travel in an environment with obstacles by calculating a path free of collision between a starting point and a goal [11,12]. The solution to this problem is crucial to the design of an autonomous robot navigation system. During navigation, the algorithms read a map of the environment and establish free paths for the robot to follow, avoiding obstacles and objects. In path planning, there are several algorithms to choose from, including road maps, optimal search, heuristic algorithms, bug algorithms, and randomized algorithms. To successfully solve the trajectory planning problem, reference inputs will be generated for the robot's control system, ensuring that desired motions can be performed [13].

B. Q-Learning Method:

Q-learning is an algorithm that operates without a predefined model. It is a value-based approach, meaning that it determines the optimal sequence of actions based on the current state of the agent. The "Q" in Q-learning stands for quality, which signifies the value of an action in maximizing future rewards.

The algorithm applied in this paper is the base Q-Learning Algorithm. There must be a discrete simulation environment and a discrete action space for this to work. A value known as Q is estimated, representing the expected reward for taking a certain action. Expected reward increases with Q. As the algorithm explores the environment, it is rewarded with positive or negative feedback. When an algorithm learns a positive behavior, it is learning good behavior, while when it learns a negative behavior, it learns to avoid it. Longer paths were penalized by the incentives system, which encouraged speedier fixes. The value of the positive incentives for good behavior was placed at the same level as the benefits of achieving the goal. If it had a greater value, the agent may try to figure out how to go to the objective even if it meant colliding with something.

The algorithm has 3 hyperparameters which are α alpha, γ gamma, and ε epsilon:

- α is the learning rate. It plays a crucial role in the algorithm as it determines the magnitude of influence that rewards have on updating the new value of Q.
- γ is the discount factor that refers to the value assigned to the expected future reward for selecting the next optimal course of action.
- ε is the ratio of exploration to exploitation refers to how frequently the algorithm attempts new strategies compared to utilizing the information it has already acquired.

The equation below accounts for Q-Value estimation in the Q-Learning algorithm:

$$\mathrm{Q_{new}}(\mathrm{s_t}, \mathrm{a_t}) = \mathrm{Q_{old}}(\mathrm{s_t}, \mathrm{a_t}) + \alpha\,[\mathrm{r_t} + \gamma \times \left(\max\left(\mathrm{Q}\,(\mathrm{s_{t+1}}, \mathrm{a})\right) - \mathrm{Q_{old}}(\mathrm{s_t}, \mathrm{a_t})\right)] \quad (1)$$

In this equation:

- Q is the estimated expected value or reward for acting in a given state.
- Q_{new} is the new predicted value of Q
- Q_{old} is the previous step value of Qs_t, a_t, r_t are the state, action, and reward for a given timestep t

- $\max(\mathrm{Q}\,(s_{t+1}, a))$ is the maximum expected reward of the next state for the current action.

Table 2 shows how the Q-Learning algorithm is implemented in Python.

Table 2 Q-Learning algorithm and steps [51]

```
Algorithm 1 Q-Learning
Set Hyper Parameters
LOOP episodes from 1 to episode_count
        Reset the Agent's Starting Position and the Environment
        Measure The current State of the Agent
        WHILE The goal is not Reached
                Randomly choose whether to exploit or explore the environment through epsilon.
                IF Exploring The environment
                        Randomly Choose an Action
                ELSE
                        Choose the Best Action
                END IF
                Perform the Action and Measure the Next  State and The Reward
                Log the Data
                Measure New Q based on old Q and Reward
        END WHILE
        Log the Q-Table
END LOOP
```

IV. IMPLEMENTATION:

Autonomous vehicles are complex sets of tasks that continuously perform in a loop including the perception of the environment, planning of the path and the trajectory, and finally execution and control of the trajectory. This paper investigates the required episodes for each scenario and the reliability of global route planning as well as a nice offline maneuver and trajectory planning locally which can provide a nice benchmark of real problem implementation and also optimal use of the Q-Learning algorithm in various environments.

A. Modeling:

With three distinct grid sizes (6 by 6, 5 by 5, and finally 4 by 4), we constructed seven environment types that are frequently used in robot route finding to give varied environments for the Q-Learning algorithm that would be identical to any unknown map. These settings would be able to closely match actual path planning in the real world. Figure 1 essentially depicts a mix of many environments for a robot's path-finding assignment in a real-world issue.

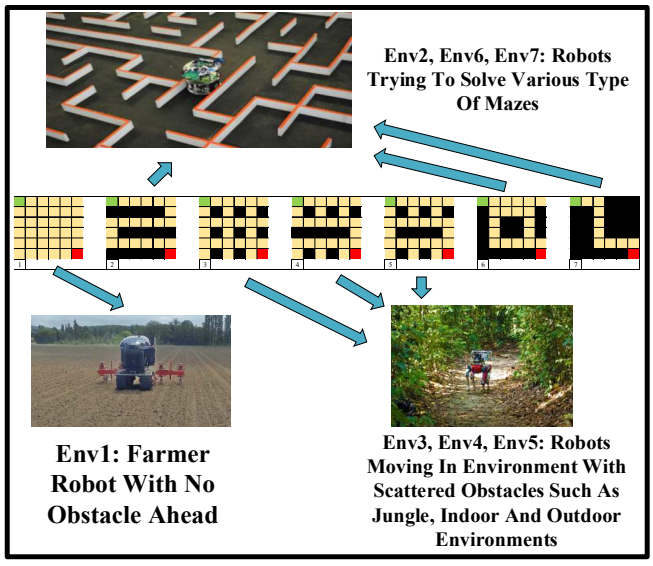

Figure 1 illustration of various unknown environments

Additionally, because mobile robots may get sensory information from their surroundings and operate in a variety of work settings, the resolution of the grid may change. To maintain the information required for path planning, it is also important to consider meshing with higher resolution in some areas with many obstacles or narrow paths. For this reason, we have implemented two additional sets of maps that are similar to the 6x6 maps in Figure 2 in terms of the placement of their obstacles. We would be able to examine the impact of having greater density meshing on the dependability and convergence rate by using lower-resolution equivalent maps.

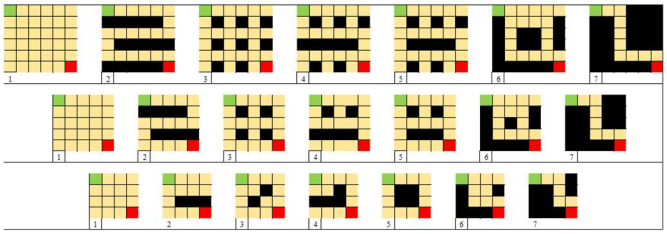

Figure 2 Different environment setup for each category of unknown two-dimensional grid (6x6, 5x5, 4x4)

B. State Space, Action Space, and Rewards:

The number of states in each grid globe is equal to the product of the number of columns and rows; as a result, the 6x6, 5x5, and 4x4 maps contain 64, 25, and 16 states, respectively. In this work, mobile robots move right, up, down, and left in a frozen lake environment created by Open AI. It is important to note that the lake is not slick, indicating that the robot's choice of action is assured.

The reward function is straightforward +1 for reaching a goal and termination for either reaching a goal or obstacles. The maximum number of steps in each episode is set to be 100. Figure 3 shows agent actions and grid properties such as obstacles and goals.

Figure 3 Agent actions, states, obstacles, and goal placement in Frozen-lake

C. Implementing code and algorithms:

As was already established, since all temporal difference learning techniques are iterative, it seems to sense that hardware configuration has no impact on the number of episodes. This would also make it possible to somewhat apply the existing results to alternative reinforcement methods, regardless of whether they are quicker or more dependable. For instance, even if the convergence rate may differ in these algorithms, the SARSA algorithm and DQN employ a similar mechanism to extract the solution, therefore the conclusions of this research would also apply to these algorithms.

The method may use the Q-learning algorithm for a specific episode interval to solve the path-planning problem. A certain amount of incidents would be resolved each time. These findings may then be utilized to determine the number of policies in each episode and, in turn, determine how many efforts were successful. Investigating the convergence rate

and dependability under various episode-number-related conditions may be aided by this.

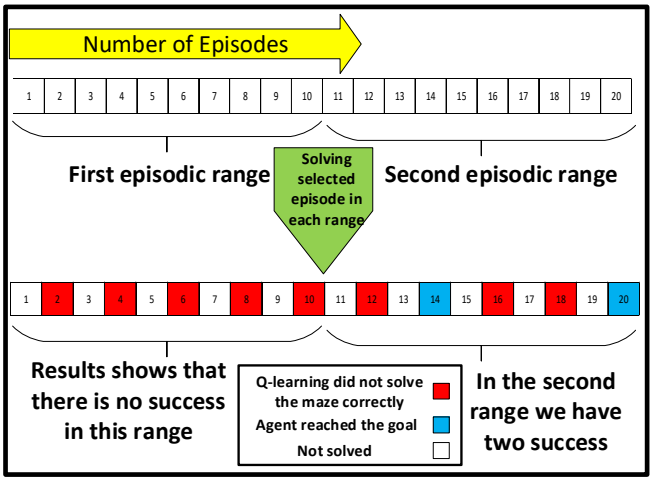

Figure 4 Success rate evaluation approach

D. hyperparameters identification:

It is worth mentioning best hyperparameter is essential to maintain the best combination of behaviors in the Q-learning algorithm. The three hyperparameters are set as follows shown in Table 3.

Table 3 3 hyperparameters of Q-Learning algorithms

Hyper Parameter	Value
α (the learning rate)	0.9
γ (discount factor)	0.95
ε (ratio of exploration to exploitation)	Decaying Value (0.9 to 0.3)

The most important parameter would encourage exploitation and exploration in epsilon. The best implementation for epsilon is being highest at the first of each episode to maintain the highest exploration and later decay to focus more on exploitation; therefore, we hardcoded the epsilon as it would be extracted for each episode as a hyperbolic decaying function regardless of the number of episodes which is apparent in Figure 5.

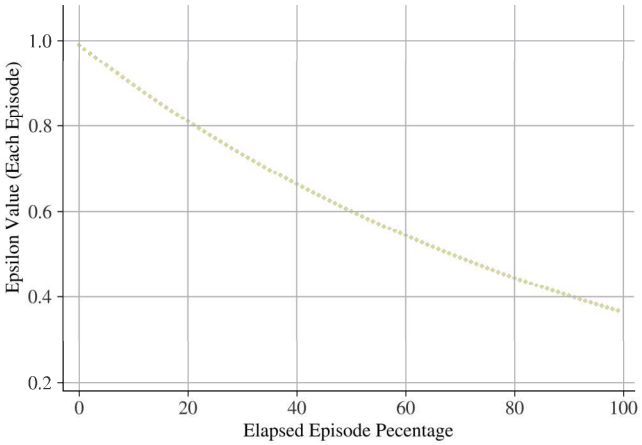

Figure 5 Epsilon hyper-parameter distribution over the number of episodes

V. SIMULATION AND RESULTS:

After solving three maps of different sizes and environments, it is clear that there is a significant difference between each environment. Furthermore, it is worth noting that in certain environments, Q-Learning was unable to provide accurate answers even after a significant number of episodes. As a result, it may not be considered reliable enough for use in these particular scenarios. Additionally, more episodes may be required to obtain the correct policy.

Firstly, we investigate the performance on a 4x4 map. Figure 6 shows that Env1, which is blank, converged in approximately 200 episodes. This is expected since there are no obstacles to hinder the performance.

After the convergence of Env1, Env3, and Env4, which had two paths leading to the goal, it becomes apparent that the number of ways to reach the goal is of utmost importance. The only outlier here is Env2, which, despite its heavy resemblance to Env4, converged faster with just one obstacle difference. The presence of the L-shaped obstacle in Env4 and the positioning of the goal seemed to cause the Q-learning agent to remain stuck within the borders until episode 4,000. In contrast, for Env2, this occurred in just 200 episodes. This demonstrates how the location of the goal can affect the convergence in a 4x4 map to the extent equal to having one or two pathways to the goal.

After that, it becomes clear that maps with just one path to converge may require more work to solve, depending on the route's width or narrowness. As in Env7, where the only difference between the two is the narrowness of the road to the goal and the fact that they both have the same number of pathways to the goal, we can see that the number of episodes required to get a reliable answer would rise to three times higher than in Env4 and Env6, where the only difference is the number of pathways to goal. As a result, the agent may run into obstacles repeatedly to offer a solution to the path planning problem.

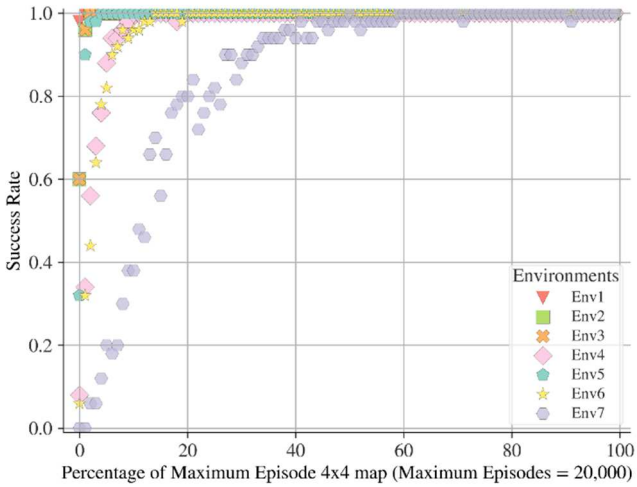

Figure 6 Performance and reliability of Q-Learning algorithm for 4x4 environment

Moreover, in the 5x5 map, the same pattern is recognizable, Env1 has the most convergence rate, and following this environment Env3 with multiple pathways to the goal. Following Env4 then we have Env5 with two pathways to the goal. Env4 is similar to Env5 despite having just one pathway and thus the agent performs more poorly. The outlier of several pathways is Env6 which contains two pathways but with shorter paths and more obstacles. The

length of the path and several ways to goal can affect the convergence rate equally.

Env2 and Env7 exaggerate the length of the journey to the destination and its significance on bigger maps. In Env2, the agent was unable to complete the task due to a single, lengthy pathway and several barriers in the vicinity. In contrast, in Env7, despite having only one short pathway, the agent was able to partially resolve the issue thanks to a shorter approach. It is crucial to remember that while many more episodes would be required for Env2 if we doubled or tripled the number of episodes for this environment, we may converge on a valid answer for Env7.

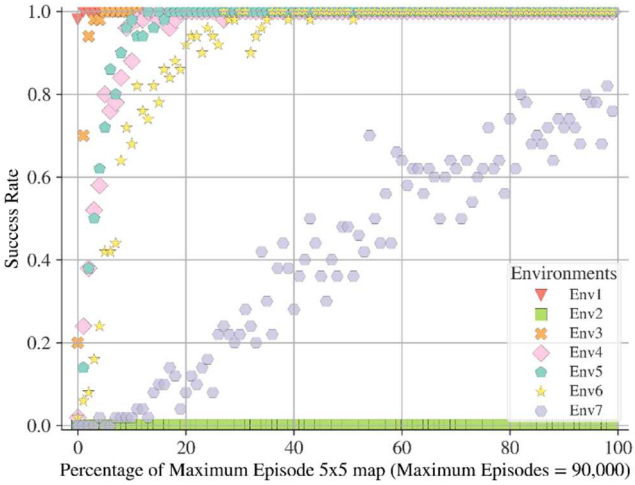

Figure 7 Performance and reliability of Q-Learning algorithm for 5x5 environment

In Figure 8, again the same pattern is recognizable for the 6x6 map. We Env3 with multiple pathways to be slower than blank Env1. Thereafter, we have Env5 with two pathways to the goal and scattered obstacles performing similarly to Env3. By reducing the pathways to only one, we have Env4 which converges more slowly than Env5 which was expected.

Interestingly, Env6, despite its two pathways setting performs similarly to Env4. We have Env4 with one pathway to the goal but structurally it is very similar to Env3 as the goal is not far from scattered obstacles. Here the length of the path length and number of pathway effects have the same impact on their convergence.

Finally, we can see how Env7 and Env2 are affected by longer and shorter narrow single pathways. The length impact here was greatly magnified due to the bigger map, similar to our prior findings in the 4x4 and 6x6 maps.

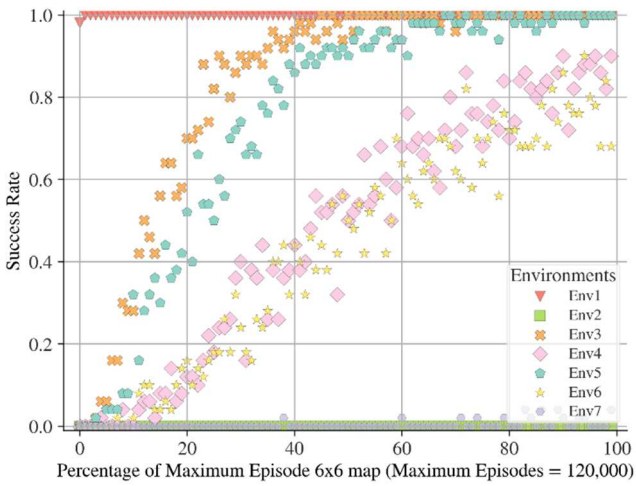

Figure 8 Performance and reliability of Q-Learning algorithm for 6x6 environment

VI. CONLUSION

First and foremost, it's critical to comprehend what influences distinct maps' rates of convergence and reliability the most. Table 4 displays what we discovered. It is important to remember that there are numerous ways to lessen the impact of harmful environmental consequences, such as adjusting hyperparameters or even raising epsilon greed to the maximum level to promote exploitation in the grid world.

Table 4 Different Unknown environments and their impact on the performance

Characteristics of Environment	Impact	considerations
Path width	Low to Medium	largely dependent on other characteristics
Number of Routes to Goal	Low to Medium	Unusual barriers could affect the impact
Length of Routs	Medium to High	The cost of calculation increases exponentially with the number of cells.
Grid number of Cells	High	The cost of calculation increases exponentially with the number of cells.

Although numerous studies have pointed out the differences between traditional path planning and the Q-Learning approach [52], none have looked into the impact of the environment on reliability and solution, as this work does. After dedicating the influential factors on convergence rate and reliability of the Q-Learning algorithm in various environment, we can depict the convergence rate for any situation.

Table 5 Convergence rate for 80% reliability for each map type/ grid size

Environment	4x4	5x5	6x6
Scattered (Env 3)	400	4,500	28,800
Mixed (Env 4, Env 5)	1,000	7,200	120,000
Mixed Maze (Env 6)	1,000	13,500	120,000
Pure Maze (Env 2, Env 7)	4,400	90,000>	120,000>>

The Table 5 indicates the number of episodes needed for convergence with 80% reliability in each case. This

demonstrates that, despite the fact that there are more obstacles from Env1 to Env7, respectively, the type of environment has a significant impact on the number of convergence episodes required for each map type. In some cases, we are unable to reach convergence even after a large number of episodes, so researchers should avoid using Q-Learning in these situations.

REFERENCES

[1] V. B. Ajabshir, M. S. Guzel, and E. Bostanci, "A Low-Cost Q-Learning-Based Approach to Handle Continuous Space Problems for Decentralized Multi-Agent Robot Navigation in Cluttered Environments," *IEEE Access*, vol. 10, pp. 35287–35301, 2022, doi: 10.1109/ACCESS.2022.3163393.

[2] J. Li, M. Ran, H. Wang, and L. Xie, "A behavior-based mobile robot navigation method with deep reinforcement learning," *Unmanned Syst.*, vol. 9, no. 3, pp. 201–209, 2021, doi: 10.1142/S2301385021410041.

[3] C. H. Chen, S. Y. Jeng, and C. J. Lin, "Mobile robot wall-following control using fuzzy logic controller with improved differential search and reinforcement learning," *Mathematics*, vol. 8, no. 8, 2020, doi: 10.3390/MATH8081254.

[4] F. Real, A. Batou, T. Ritto, and C. Desceliers, "Motion Control for Mobile Robot Navigation Using Machine Learning: a Survey," *Int. J. Rob. Res.*, p. 107754631982824, 2020, doi: 10.1177/ToBeAssigned.

[5] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 1478–1483, 2011, doi: 10.1109/ICRA.2011.5980479.

[6] H. Wang, B. Wang, B. Liu, X. Meng, and G. Yang, "Pedestrian recognition and tracking using 3D LiDAR for autonomous vehicle," *Rob. Auton. Syst.*, vol. 88, pp. 71–78, 2017, doi: 10.1016/j.robot.2016.11.014.

[7] Y. Zhao, Z. Zheng, X. Zhang, and Y. Liu, "Q learning algorithm based UAV path learning and obstacle avoidence approach," *Chinese Control Conf. CCC*, pp. 3397–3402, 2017, doi: 10.23919/ChiCC.2017.8027884.

[8] O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud, "Q-learning based Routing Scheduling for a Multi-Task Autonomous Agent," *Midwest Symp. Circuits Syst.*, vol. 2019-Augus, pp. 634–637, 2019, doi: 10.1109/MWSCAS.2019.8885080.

[9] D. Ma, N. Shlezinger, T. Huang, Y. Liu, and Y. C. Eldar, "Joint Radar-Communications Strategies for Autonomous Vehicles," no. 646804, pp. 1–21, 2019, doi: 10.1109/MSP.2020.2983832.

[10] C. Guindel, D. Martín, and J. M. Armingol, "Modeling Traffic Scenes for Intelligent Vehicles Using CNN-Based Detection and Orientation Estimation," *Adv. Intell. Syst. Comput.*, vol. 694, pp. 487–498, 2018, doi: 10.1007/978-3-319-70836-2_40.

[11] N. Altuntas, E. Imal, N. Emanet, and C. N. Öztürk, "Reinforcement learning-based mobile robot navigation," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 24, no. 3, pp. 1747–1767, 2016, doi: 10.3906/elk-1311-129.

[12] K. B. De Carvalho, I. R. L. De Oliveira, D. K. D. Villa, A. G. Caldeira, M. Sarcinelli-Filho, and A. S. Brandao, "Q-learning based Path Planning Method for UAVs using Priority Shifting," in *2022 International Conference on Unmanned Aircraft Systems, ICUAS 2022*, 2022, pp. 421–426. doi: 10.1109/ICUAS54217.2022.9836175.

[13] T. Luo, "Improved reinforcement learning algorithm for mobile robot path planning," *ITM Web Conf.*, vol. 47, p. 02030, 2022, doi: 10.1051/itmconf/20224702030.

[14] K. S. Kiangala and Z. Wang, "An Experimental Safety Response Mechanism for an Autonomous Moving Robot in a Smart Manufacturing Environment Using Q-Learning Algorithm and Speech Recognition," *Sensors*, vol. 22, no. 3, Feb. 2022, doi: 10.3390/s22030941.

[15] X. Guo, G. Peng, and Y. Meng, "A modified Q-learning algorithm for robot path planning in a digital twin assembly system," *Int. J. Adv. Manuf. Technol.*, vol. 119, no. 5–6, pp. 3951–3961, 2022, doi: 10.1007/s00170-021-08597-9.

[16] C. Chen and D. Wang, "PATH PLANNING OF MOBILE ROBOT BASED ON THE IMPROVED Q-LEARNING ALGORITHM," *Int. J. Innov. Comput. Inf. Control*, vol. 18, no. 3, pp. 687–702, 2022, doi: 10.24507/ijicic.18.03.687.

[17] F. Zhang, C. Wang, C. Cheng, D. Yang, and G. Pan, "Reinforcement Learning Path Planning Method with Error Estimation," *Energies*, vol. 15, no. 1, 2022, doi: 10.3390/en15010247.

[18] Q. Jiang, "Path Planning Method of Mobile Robot Based on Q-learning," in *Journal of Physics: Conference Series*, Feb. 2022, vol. 2181, no. 1. doi: 10.1088/1742-6596/2181/1/012030.

[19] B. Zhang, G. Li, Q. Zheng, X. Bai, Y. Ding, and A. Khan, "Path Planning for Wheeled Mobile Robot in Partially Known Uneven Terrain," *Sensors*, vol. 22, no. 14, Jul. 2022, doi: 10.3390/s22145217.

[20] A. Abdi, M. H. Ranjbar, and J. H. Park, "Computer Vision-Based Path Planning for Robot Arms in Three-Dimensional Workspaces Using Q-Learning and Neural Networks," *Sensors*, vol. 22, no. 5, Mar. 2022, doi: 10.3390/s22051697.

[21] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: a survey," *Auton. Robots*, vol. 46, no. 5, pp. 569–597, Nov. 2022, doi: 10.1007/s10514-022-10039-8.

[22] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Sci. Technol.*, vol. 26, no. 5, pp. 674–691, 2021, doi: 10.26599/TST.2021.9010012.

[23] H. Sun, W. Zhang, R. Yu, and Y. Zhang, "Motion Planning for Mobile Robots - Focusing on Deep Reinforcement Learning: A Systematic Review," *IEEE*

Access, vol. 9, pp. 69061–69081, 2021, doi: 10.1109/ACCESS.2021.3076530.
[24] A. Cioffi, A. R. Asghar, and P. Schito, "Parameters estimation of a steady-state wind farm wake model implemented in OpenFAST," *Wind Eng.*, no. 2020, 2022, doi: 10.1177/0309524X221117820.
[25] M. Andrews, C. Dibek, and K. Palyutina, "Evolution of Q Values for Deep Q Learning in Stable Baselines," pp. 1–14, 2020, [Online]. Available: http://arxiv.org/abs/2004.11766
[26] Y. Tang, "Self-imitation learning via generalized lower bound Q-learning," *Adv. Neural Inf. Process. Syst.*, vol. 2020-Decem, no. NeurIPS, 2020.
[27] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-learning for offline reinforcement learning," *Adv. Neural Inf. Process. Syst.*, vol. 2020-Decem, no. NeurIPS, pp. 1–13, 2020.
[28] J. P. Araújo, M. A. T. Figueiredo, and M. A. Botto, "Control with adaptive Q-learning," pp. 1–29, 2020, [Online]. Available: http://arxiv.org/abs/2011.02141
[29] P. Rodrigues and S. Vieira, "Optimizing Agent Training with Deep Q-Learning on a Self-Driving Reinforcement Learning Environment," *2020 IEEE Symp. Ser. Comput. Intell. SSCI 2020*, pp. 745–752, 2020, doi: 10.1109/SSCI47803.2020.9308525.
[30] J. Dibachi and J. Azoulay, "Teaching a Robot to Walk Using Reinforcement Learning," 2021, [Online]. Available: http://arxiv.org/abs/2112.07031
[31] Y. Chen, L. Schomaker, and M. Wiering, "An investigation into the effect of the learning rate on overestimation bias of connectionist q-learning," *ICAART 2021 - Proc. 13th Int. Conf. Agents Artif. Intell.*, vol. 2, no. Icaart, pp. 107–118, 2021, doi: 10.5220/0010227301070118.
[32] T. Yamagata, A. Khalil, and R. Santos-Rodriguez, "Q-learning Decision Transformer: Leveraging Dynamic Programming for Conditional Sequence Modelling in Offline RL," 2022, [Online]. Available: http://arxiv.org/abs/2209.03993
[33] Q. Li, W. Zhou, Z. Lu, and H. Li, "Simultaneous Double Q-learning with Conservative Advantage Learning for Actor-Critic Methods," pp. 1–11, 2022, [Online]. Available: http://arxiv.org/abs/2205.03819
[34] P. P. Alternative, Q. U. E. Para, O. El, and E. N. L. A. E. De, "Deep Q-Learning in Robotics : A path planning alternative," 2022.
[35] P. Sunden, "Q-LEARNING AND DEEP Q-LEARNING IN OPENAI GYM CARTPOLE CLASSIC CONTROL ENVIRONMENT," no. March, 2022.
[36] R. V. Hoa, T. D. Chuyen, N. D. Dien, and D. H. Du, *Automatic Navigation Research for Multi-directional Mobile Robots on the Basis of Artificial Intelligence Application, Q-Learning Algorithm*, vol. 366 LNNS. Springer International Publishing, 2022. doi: 10.1007/978-3-030-92574-1_21.
[37] H. Anas, W. H. Ong, and O. A. Malik, *Comparison of Deep Q-Learning, Q-Learning and SARSA Reinforced Learning for Robot Local Navigation*, vol. 429 LNNS. Springer International Publishing, 2022. doi: 10.1007/978-3-030-97672-9_40.
[38] Z. Zhou, C. Allen, K. Asadi, and G. Konidaris, "Characterizing the Action-Generalization Gap in Deep Q-Learning," 2022.
[39] J. Min and L. T. Elliott, "Q-learning with online random forests," pp. 1–8, 2022, [Online]. Available: http://arxiv.org/abs/2204.03771
[40] M. Sewak, "Coding the Environment and MDP Solution: Coding the Environment, Value Iteration, and Policy Iteration Algorithms," *Deep Learn.*, no. June 2019, 2019, doi: 10.1007/978-981-13-8285-7.
[41] M. Lutter, S. Mannor, J. Peters, D. Fox, and A. Garg, "Value Iteration in Continuous Actions, States and Time," 2021, [Online]. Available: http://arxiv.org/abs/2105.04682
[42] L. G. Cuenca, E. Puertas, J. F. Andrés, and N. Aliane, "Autonomous driving in roundabout maneuvers using reinforcement learning with q-learning," *Electron.*, vol. 8, no. 12, 2019, doi: 10.3390/electronics8121536.
[43] J. Li, M. Ran, H. Wang, and L. Xie, "A behavior-based mobile robot navigation method with deep reinforcement learning," *Unmanned Syst.*, vol. 9, no. 3, pp. 201–209, 2021, doi: 10.1142/S2301385021410041.
[44] M. A. Kareem Jaradat, M. Al-Rousan, and L. Quadan, "Reinforcement based mobile robot navigation in dynamic environment," *Robot. Comput. Integr. Manuf.*, vol. 27, no. 1, pp. 135–149, 2011, doi: 10.1016/j.rcim.2010.06.019.
[45] M. Luong and C. Pham, "Incremental Learning for Autonomous Navigation of Mobile Robots based on Deep Reinforcement Learning," *J. Intell. Robot. Syst. Theory Appl.*, vol. 101, no. 1, pp. 1–11, 2021, doi: 10.1007/s10846-020-01262-5.
[46] A. Maoudj and A. Hentout, "Optimal path planning approach based on Q-learning algorithm for mobile robots," *Appl. Soft Comput. J.*, vol. 97, p. 106796, 2020, doi: 10.1016/j.asoc.2020.106796.
[47] A. G. da Silva Junior, D. H. Dos Santos, A. P. F. de Negreiros, J. M. V. B. de S. Silva, and L. M. G. Gonçalves, "High-level path planning for an autonomous sailboat robot using q-learning," *Sensors (Switzerland)*, vol. 20, no. 6, pp. 1–22, 2020, doi: 10.3390/s20061550.
[48] X. Guoa and X. Guo, "A Modiied Q-Learning Algorithm for Robot Path Planning in a Digital Twin Assembly System A Modified Q-learning Algorithm for Robot Path Planning in a Digital Twin Assembly System," *Int. J. Adv. Manuf. Technol.*, 2021, doi: 10.21203/rs.3.rs-825772/v1.
[49] M. Zhao, H. Lu, S. Yang, and F. Guo, "The experience-memory Q-Learning algorithm for robot path planning in unknown environment," *IEEE Access*, vol. 8, pp. 47824–47844, 2020, doi: 10.1109/ACCESS.2020.2978077.
[50] I. Kim, "Reinforcement Learning for Navigation of Mobile Robot with LiDAR," 2020.
[51] P. Chintala, R. Dornberger, and T. Hanne, "Robotic Path Planning by Q Learning and a Performance Comparison with Classical Path Finding Algorithms," *Int. J. Mech. Eng. Robot. Res.*, vol. 11, no. 6, pp. 373–378, 2022, doi: 10.18178/ijmerr.11.6.373-378.