STM JOURNALS

International Journal of
Advanced Robotics and
Automation Technology

Volume 2, Issue 2, 2024
July—December
DOI (Journal): 10.37591/IJARAT

https://journals.stmjournals.com/ijarat

Research IJARAT

Improvement of Convergence Speed of Q-learning based Path Planning Algorithm

Murim Pak[1], Kangsong Ro[2,*], Choljin Wang[3], Jusong Pak[4]

Abstract

Path planning is fundamental and important task of mobile robot. There are many attempts to adopt reinforcement learning (RL) in mobile robot path planning. RL based path planning is effective in path planning of intelligent mobile robot, especially in unknown environment because it doesn't require environmental information and finds optimal path through trial-and-error process. Q-learning is one of RL algorithm widely used in path planning of mobile robots. The main drawback of Q-learning is slow convergence speed. Several methods to improve convergence speed has been proposed but remarkable success has not been achieved. In this paper, some methods to improve convergence speed are proposed. This new algorithm includes; (1) new method of initializing Q-values before learning, (2) method of using searched obstacle information on improving learning efficiency, and (3) new way of adjusting greedy factor. First, The Q-values near the target are initialized considerably higher than the Q values far from the target. Second, we decreased the number of bumping into obstacles iteratively in the early stages of learning by modifying state values around obstacles by using partially searched obstacle information. Finally, we decreased greedy factor exponentially based on number of episodes reached at target correctly. In the early stages of learning, greedy factor is relatively high but it decreases dramatically as number of episodes reached at target correctly increases. It is expected that proposed algorithm has better convergence speed and learning efficiency than existing algorithms. Experiment results show that proposed algorithm has higher convergence speed and better learning efficiency than existing algorithms.

Keywords: Reinforcement learning (RL), Q-learning, path planning, convergence speed

INTRODUCTION

Nowadays, with rapid development of Artificial Intelligence technology, there are many progresses in the field of mobile robot navigation and control.

Path Planning is the most fundamental and very important task of mobile robot and number of path planning algorithms are introduced. Typical path planning algorithms include Dijkstra algorithm [1], A* algorithm [2], genetic algorithm [3], Artificial Potential Field Method [4] and RL based method.

RL based path planning algorithms are widely used in mobile robot path planning because they don't require environmental information and finds optimal path through interacting with environment. Especially in unknown environment RL based path planning algorithms are superior to traditional ones [5]. Q-learning is typical RL algorithm widely used in mobile robot path planning [6].

***Author for Correspondence**
Kangsong Ro
E-mail: GS.RO@star-co.net.kp

[1-4]Professor, Department of Mechanical Engineering, Kim Chaek University of Technology, Pyongyang, DPR Korea

Received Date: July 12, 2024
Accepted Date: October 10, 2024
Published Date: December 16, 2024

Citation: Murim Pak, Kangsong Ro, Choljin Wang, Jusong Pak. Improvement of Convergence Speed of Q-learning based Path Planning Algorithm. International Journal of Advanced Robotics and Automation Technology. 2024; 2(2): 19–28p.

The main drawbacks of Q-learning algorithm are; (1) too many invalid iterations in the early stages of learning because it explores the environment blindly, (2) exploration-exploitation dilemma.

In this paper, we proposed several methods to improve convergence speed of Q-learning based path planning algorithm; Initialization of Q-values based on range to the target position, Modification of state values based on searched obstacle information and Adjustment of greedy factor. These subsystems will be explained in details in a following section.

The rest of this paper is organized as follows. In the following section, we briefly survey the relevant related work. Following comes Q-learning, typical RL algorithm definition and general methods to improve convergence speed of Q-learning based path planning algorithm. The proposed methods to improve convergence speed of Q-learning based path planning algorithm is described next. Next, application of proposed methods with simulation discussions are presented. Finally, we conclude the paper.

RELATED WORKS

There are several path planning algorithms for mobile robots such as Dijkstra algorithm [1], A* algorithm [2], genetic algorithm [3], Artificial Potential Field Method [4] and RL based method

Dijkstra's algorithm is a famous algorithm that is used for finding the shortest path, from starting node to target node in a weighted graph. This algorithm makes a tree of the shortest path from the starting node to all other nodes in the graph. It makes use of weights of the edges for finding the path that minimizes the total distance from the source node to all other nodes [1].

The A* algorithm is just like Dijkstra's algorithm. The only difference is that A* algorithm tries to look for a better path by using a heuristic function H which gives priority to nodes that are supposed to be better than others while Dijkstra's algorithm just explores all possible paths [2].

Genetic algorithm is a very popular meta-heuristic technique for solving optimization problems. In path planning of mobile robot, genetic algorithm is used efficiently. Toolika Arora et al. proposed a path planning method based on genetic algorithm for finding path for mobile robot in dynamic environment. Here the genetic algorithm is applied at a point in the problem space not at the complete space [3].

The artificial potential field method is a path planning method for mobile robot to move to the target correctly, using interaction between attractive force caused by the target and repulsive force caused by the obstacles. There are several improved artificial potential field methods [4].

Li S. et al. compared performance between traditional path planning algorithms and RL based path planning algorithm [5]. RL based path planning algorithms are widely used in mobile robot path planning because they don't require environmental information and finds optimal path through interacting with environment. Especially in unknown environment RL based path planning algorithms are superior to traditional ones [5].

But, as mentioned above Q-learning algorithm, which is one of the RL algorithm, has long convergence speed.

Wen et al. proposed a method of initializing Q-values based on fuzzy logic before learning, resulting decrease of invalid iterations in the early stages of learning [7]. Zhen Shi et al. proposed a method of initializing Q-values integrating prior knowledge about environment-initial position of target and agent and adjusting greedy factor as number of episodes reached at target correctly increases [8].

DEFINITIONS & GENERAL METHODS

Reinforcement Learning

Reinforcement Learning (RL) is one section of Machine Learning (ML) algorithm but it is totally different with other ML algorithms such as supervised and unsupervised learning which learn from data. RL is essentially the learning through interacting with the environment. RL is represented by its own concepts such as Agent, Environment, Reward and so on. RL process can be formulated by Markov Decision Process (MDP). Agent takes an action in each state and gets reward or penalty as a result of interacting with Environment. Policy is the method for agent to take action. Agent should find optimal policy maximizing reward through interacting with environment. In RL, future reward is one of the important concepts. Agent must consider not only immediate effect of an action but also future effect to select best action in each state. For example, during climbing on the mountain, we maybe go down to find best way to go up to the mountain. For a given policy, total reward is sum of immediate reward and discounted future reward.

$$R = r_0 + \sum_{k=1}^{\infty} \gamma^k r_k \tag{1}$$

Where, r_i is immediate reward at step i, and $\gamma \in [0,1]$ is discount factor indicating how much we pay attention to future reward. High γ means that future reward is more important, and low γ means that future reward is less important.

Q-function is adopted to evaluate state-action pairs. All state-action pairs in each state are configured and a Q-function which can reflect reward corresponding to each state-action pair is formulated as follows:

$$Q(s_t, a_t) = E[\sum_{k=1}^{\infty} \gamma^k r_k \mid s_t, a_t] \tag{2}$$

High $Q(s_t, a_t)$ means action a_t in state s_t can raise reward r.

Above equation is represented by expected value. Its Bellman equation is as follows:

$$Q(s_t, a_t) = E[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) \mid s_t, a_t] \tag{3}$$

Optimal Q-function can be represented as follows:

$$Q^*(s_t, a_t) = \max_{\pi} Q^{\pi}(s_t, a_t) \tag{4}$$

Where, π is policy. Optimization problem of RL can be formulated as follows:

$$\pi^*(a_t \mid s_t) = \arg\max_{\pi} Q^{\pi}(s_t, a_t) \tag{5}$$

Thus, it is to find $\pi^*(a_t \mid s_t)$ to maximize Q-function of each state-action pair.

Where, $\pi^*(a_t \mid s_t)$ is optimal policy for RL agent. In RL based path planning of mobile robot, agent is just robot.

Q-learning

Q-learning is off-policy algorithm, uses two policies. One is ε-greedy policy, which selects action in current state, and another one is greedy policy, which selects best action that has maximum Q-function in next state. Q-function is modified as follows:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)) \tag{6}$$

In Q-learning, the policy is modified in each step of episode and following modified Q-values, modified policy is extracted. Q-learning algorithm is as follows:

1. Initialize a Q-function $Q(s, a)$ with random values
2. For each episode:

a. Initialize state s
b. For each step in the episode:
   i. Extract a policy from $Q(s,a)$ and select an action a to perform in state s
   ii. Perform the action a, move to the next state s', and observe the reward r
   iii. Update the Q-values as

$$Q(s,a) = Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

   iv. Update $s = s'$ (update the next state s' to current state s
   v. If s is not a terminal state, repeat step (1) to (5)

Several Methods to Improve Convergence Speed of Q-learning

The main drawback of Q-learning is slow convergence speed. In traditional Q-learning, Q-values of all state-action pairs are initialized with zeros or random values.

Wen et al. initialized Q-values based on fuzzy logic before learning, resulting decrease of invalid iterations in the early stages of learning [7]. Zhen Shi et al. initialized state values based on the distance from each state to the target position and calculated Q-values with state values [8]. Here, state values are initialized linearly.

$$V(s') = \zeta(1 - \frac{\rho(s')}{\rho_{\max}}) \tag{7}$$

Where, ζ is positive coefficient, $\rho(s')$ is the distance between state s' and target, and $\rho_{\max}$ is diagonal length of the map. And Q-values are calculated as follows:

$$Q(s_t, a_t) = r + \gamma \sum_{s'} P(s' \mid s,a) V(s') \tag{8}$$

Where $P(s' \mid s,a)$ is probability of reaching s' by performing an action a in state s. Environment considered in this paper is deterministic environment, thus, $P(s' \mid s,a)$ is 1.

Another important challenge of Q-learning is exploration-exploitation dilemma. In Q-learning, ε-greedy method is used to select action in current state. Thus, with probability ε selects random action and with probability of 1-ε selects an action that has maximum Q-value. Zhen Shi et al. adjusted ε as step function according to the number of episodes reached at target correctly [8].

$$\varepsilon = \begin{cases} \varepsilon_1, & C < C_1 \\ \varepsilon_2, & C_1 \le C \le C_2 \\ \varepsilon_3, & C > C_2 \end{cases} \tag{9}$$

Where, $0<\varepsilon_3<\varepsilon_2<\varepsilon_1<1$, C is number of episodes reached at target correctly.

Above methods can improve convergence speed and learning efficiency of Q-learning algorithm.

PROPOSED METHODS

Initialization of Q-values Based on Range to the Target Position

Zhen Shi et al. proposed method of initializing state values based on the distance from each state to the target position [8]. Here, state values are initialized linearly.

But how can we assume state value changes linearly according to the range to the target? For this, we observed changes of state values during learning process of general Q-learning algorithm (Figure 1).

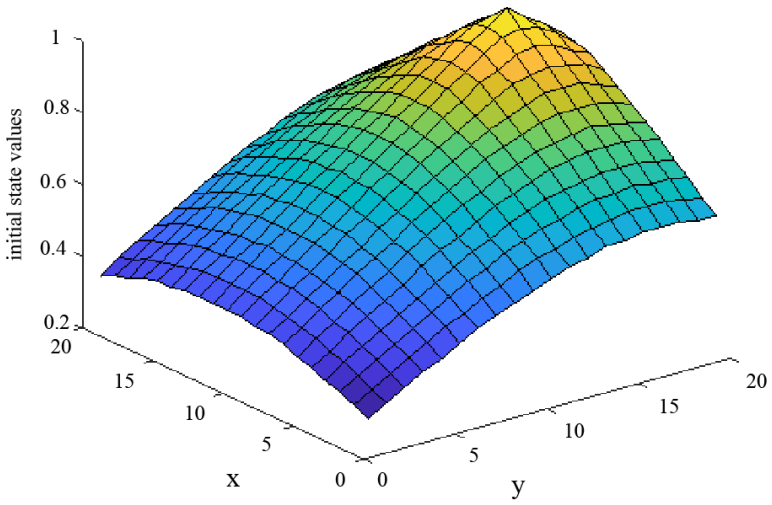

Figure 1. Initialized state values.

Observed result showed that state values in each state increase dramatically as distance between state and target is closer while Q-learning algorithm is converging. Thus, if state values before learning are initialized that it dramatically increases as distance between state and target is closer, it is expected that convergence speed and learning efficiency of Q-learning algorithm are improved.

In this paper state values of each state are initialized as follows:

$$V(s') = \zeta \cdot (1 - \sqrt{\frac{\rho(s')}{\rho_{\max}}}) \tag{10}$$

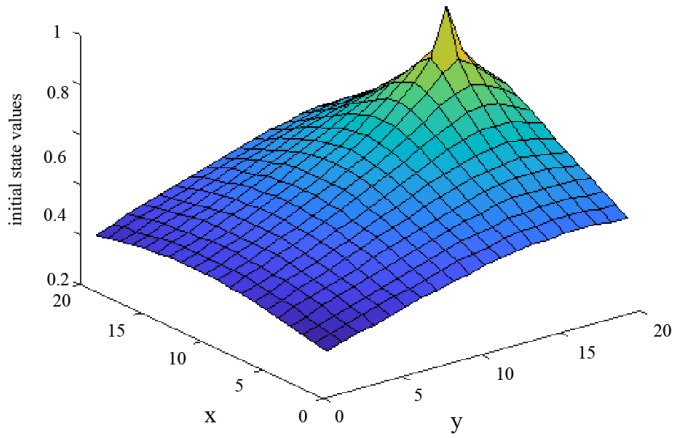

Figure 2. Initialized state values (proposed method in this paper).

As we can see above in Figure 2, state values in each state increase dramatically as distance between state and target is closer. This can improve performance of Q-learning algorithm. According to the state values in each state, Q-values are initialized as follows:

$$Q(s_t, a_t) = r + \gamma \sum_{s'} P(s' \mid s, a) V(s') \tag{11}$$

Modification of state values based on searched obstacle information

In general Q-learning algorithm, searched obstacle information is used only in configuration of reward function. Here, state value in obstacle position is decreased by getting negative reward when agent bumped into obstacle. Thus, state values around searched obstacle can't be modified enough and can't overcome iterative collision into obstacles during learning.

In this paper, a method of using partially searched obstacle information to modify state values is proposed. During learning when agent bumps into obstacle, state values of searched obstacle position and surrounding states are decreased artificially, resulting in small iterative collisions into obstacles. Figure 3. shows the modification based on searched obstacle information. During learning, we assume that agent bumped into obstacle at [10, 6]. Then we decreased 8 state values around [10, 6] according to the distance to that point, resulting reduction probability to bump into obstacle [10, 6] again and improvement learning efficiency.

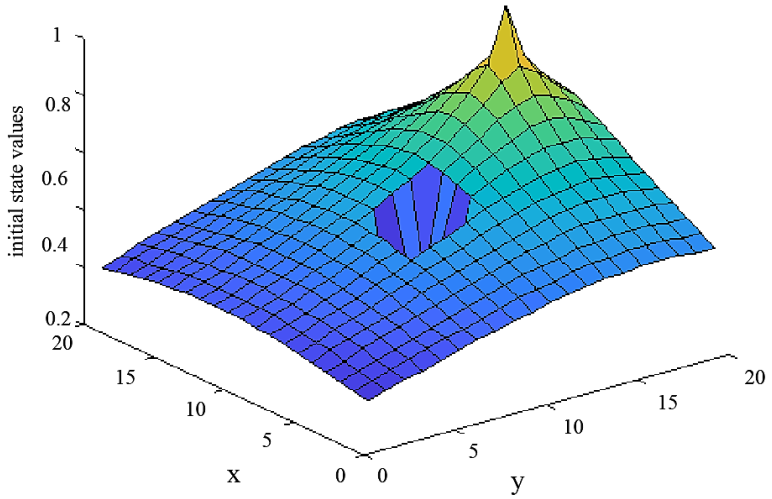

Figure 3. Modification of state values using searched obstacle information.

$$\begin{cases} V(s') = V(s') - V_{0,rep} \cdot (1 - \dfrac{\rho}{\rho_0}), & \rho < \rho_0 \\ V(s') = V(s'), & \rho \geq \rho_0 \end{cases} \tag{12}$$

Where, ρ is the distance between current state s' and searched obstacle position, ρ_0 is considering range, $V_{0,rep}$ is positive coefficient. Based on modified state values, Q-values are modified as follows:

$$Q(s_t, a_t) = r + \gamma \sum_{s'} P(s' \mid s, a) V(s') \tag{13}$$

Adjustment of greedy factor

Zhen Shi et al. adjusted greedy factor according to the number of episodes reached at target correctly [8]. But in this method, hyperparameters for adjusting greedy factor are set based on experience, so it can't ensure optimal effect.

As number of episodes reached at target correctly increases, exploitation rate is needed to set high dramatically. Thus, greedy factor should be decreased dramatically.

In this paper greedy factor is decreased using exponential function as follows:

$$\varepsilon = \max\left[\varepsilon_{\max} \cdot (e^{-C/10}),\ \varepsilon_{\min} \right] \tag{14}$$

Here, $\varepsilon_{\max}$ and $\varepsilon_{\min}$ are greedy factors in initial and final state and C is number of episodes that agent reached at target correctly. So, greedy factor will be decreased dramatically as number of episodes reached at target correctly increases and will improve convergence speed and learning efficiency of Q-learning algorithm

EXPERIMENT

Simulation Environment Construction

Simulation environment was constructed in robot experiment platform CoppeliaSim. Map size is set to 20x20. Each grid represents individual states, yellow rectangle represents obstacle, red circle represents target and pink circle is agent(robot). Initial position of the robot is (1, 1), target position is (11, 17) (Figure 4).

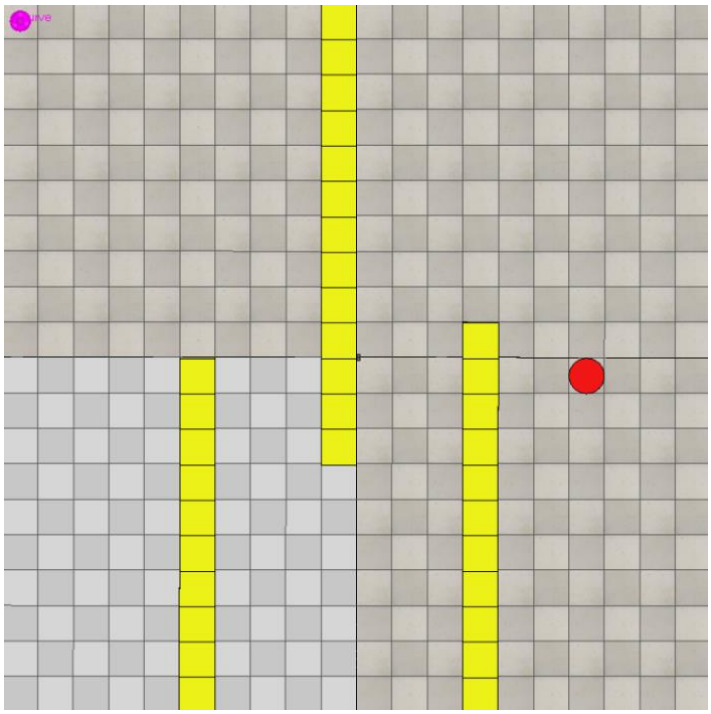

Figure 4. Simulation environment.

Experiment were performed on 8GB RAM, Intel Core-i7-1165G7 processor and NVIDIA RTX 3050 GPU. Reward function is configured as follows:

$$r=\begin{cases}1, & \text{reached at the target}\\ -1, & \text{bump into obstacle}\\ 0, & \text{else}\end{cases} \tag{15}$$

When agent reached at the target or bumped into obstacles, episode ends and agent position is initialized. In above environment, algorithms proposed in preceding works and algorithm proposed in this paper are experimented and results are compared.

Experimental parameters

In Table 1, experimental parameters are given.

Table 1. Experiment Parameters.

Parameter	Value	Parameter	Values	Parameter	Value	Parameter	Values
α	0.01	C_2	50	ε_1	0.5	$\varepsilon_{\max}$	0.2
γ	0.9	$V_{0,rep}$	1	ε_2	0.2	$\varepsilon_{\min}$	0.01
C_1	1	ζ	1	ε_3	0.05	ρ_0	2

RESULTS AND DISCUSSION

Learning results of 3 algorithms are as follows (Figures 5-7).

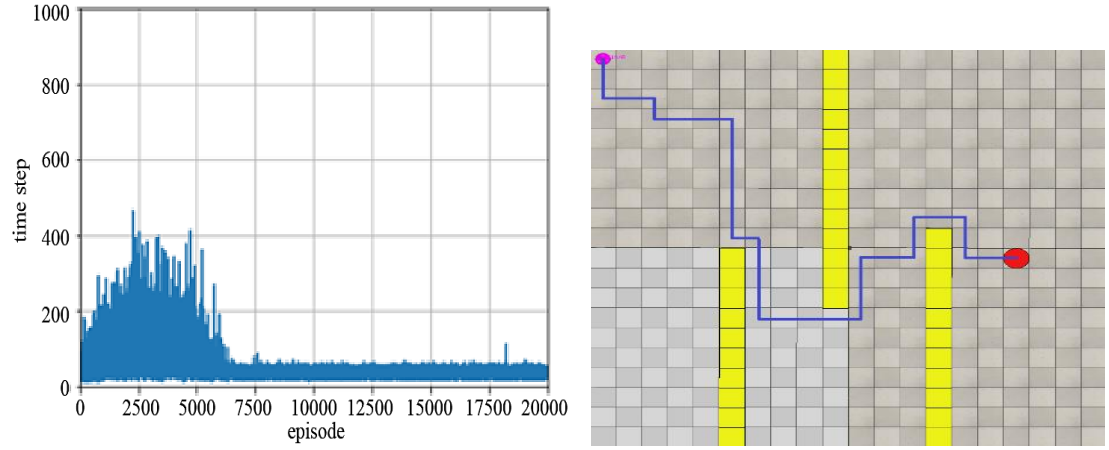

Figure 5. Algorithm 1 - diagram of time steps according to episodes(left), obtained path(right).

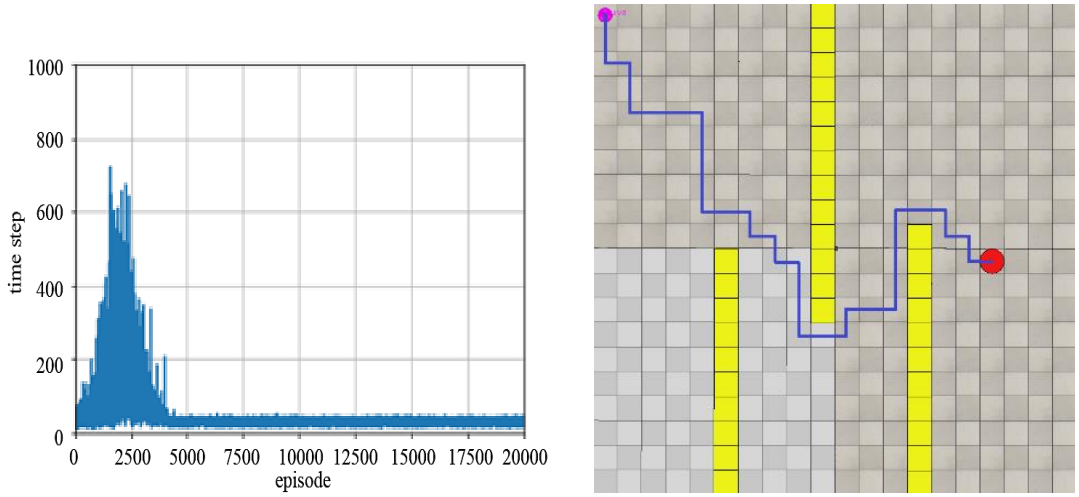

Figure 6. Algorithm 2 - diagram of time steps according to episodes(left), obtained path(right).

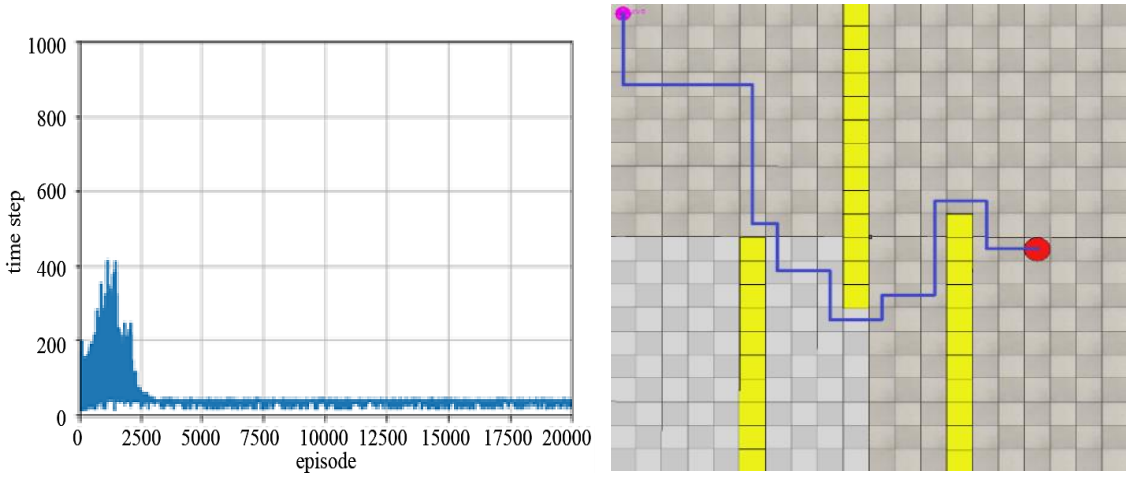

Figure 7. Algorithm 3 - diagram of time steps according to episodes(left), obtained path(right).

The diagram of time steps according to episode is shown in Figure 8.

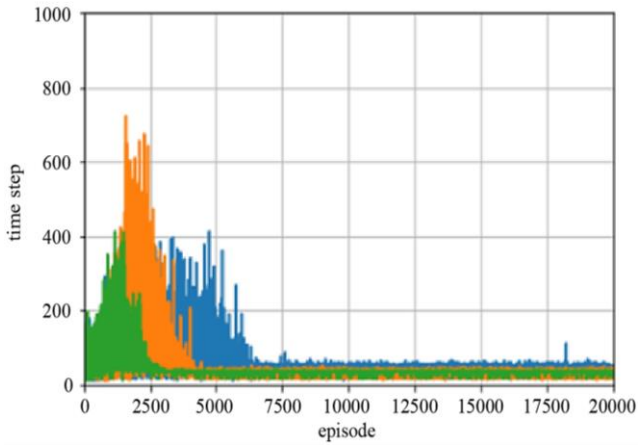

Figure 8. Performance comparison of 3 algorithms.

Convergence speed and learning efficiency are compared (Table 2).

Table 2. Performance comparation of 3 algorithms.

Algorithm	Total time steps before convergenc	Number of episodes before convergence	Convergence time(s
	567568	8692	10.90
	377837	3830	6.15
	200929	2329	4.05

As we can see above, by using proposed algorithm, we reduced total time steps before convergence as 64.6%, 46.8% and number of episodes before convergence as 73.2%, 39.2% than algorithm 1 and 2. And we reduced convergence time as 63% than algorithm 1 and 34% than algorithm 2. In conclusion, algorithm 3(proposed in this paper) has better performance than previous ones.

CONCLUSION

Q-learning is typical RL algorithm, widely used in intelligent robot path planning. The main drawback of Q-learning is slow convergence speed. In this paper several methods to improve convergence speed are proposed; (1) Q-value initialization before learning, (2) modification of state values based on partially searched obstacle information, and (3) new method to adjust greedy factor. These methods improved convergence speed and learning efficiency of Q-learning algorithm. Experiment results show that algorithm proposed in this paper has better performance than preceding algorithms in the same condition.

Conflict of Interest

The authors declare no conflicts of interest associated with this manuscript.

Author Contributions

Murim Pak contributed to the study of conception. The study was designed by Murim Pak and Kangsong Ro. Literature Review was written by Kangsong Ro. Improved methods are proposed by Murim Pak and Kangsong Ro under supervision of Choljin Wang and Jusong Pak. Data analysis was performed by all authors. The first draft of the manuscript was written by Murim Pak and all authors commented on previous versions of the manuscript.

REFERENCES

1. Chao Y. and Wang H.: Developed Dijkstra shortest path search algorithm and simulation (2010). 2010 International Conference On Computer Design and Applications. IEEE. Qinhuangdao, China.
2. Bingbing Xu, Rongfei Hao: Current Situation and Development of Robot Path Planning Technology. Electronic Technology & Software Engineering (2019)

3. Toolika Arora, Yogita Gigras, Vijay Arora: Robotic Path Planning using Genetic Algorithm in Dynamic Environment. International Journal of Computer Applications (2014). Vol 89 (11).
4. Xianxia Liang, Zhaoying Liu, Xueling Song, Yngkun Zhang: Research on Improved Artificial Potential Field Approach in Local Path Planning for Mobile Robot. Computer Simulation (2018)
5. Phalgun Chintala, Rolf Dornberger, Thomas Hanne: Robotic Path Planning by Q learning and a performance Comparation with Classical Path Finding Algorithms (2022). IJMERR 2022 Vol.11(6): 373-378
6. Li S., Xin X., and Lei Z.: Dynamic path planning of a mobile robot with improved Q-learning algorithm (2015). 2015 IEEE International Conference on Information and Automation. IEEE. Lijiang, China
7. Wen S., Chen J., Li Z., Rad A. B., and Othman K. M.: Fuzzy Q-learning obstacle avoidance algorithm of humanoid robot in unknown environment (2018). 2018 37th Chinese Control Conference (CCC). IEEE. Wuhan, China
8. Zhen Shi, Keyin Wang, Jianhui Zhang: Improved reinforcement learning path planning algorithm integrating prior knowledge. Plos One (2023).

