









# Path Planning Algorithm Based on A\_star Algorithm and Q-Learning Algorithm

Xiaodong Zhao<sup>1</sup> , Mengying Cao<sup>1</sup> , Jingfang Su<sup>1</sup> , Yijin Zhao<sup>2</sup> ,  
Shuying Liu<sup>1</sup> , and Pingping Yu<sup>1</sup> 

<sup>1</sup> College of School of Information Science and Engineering, Hebei University of Science and Technology, Shijiazhuang, Hebei, China  
sujingfang1980@hebust.edu.cn

<sup>2</sup> College of Feduni Information Engineering Institute, Hebei University of Science and Technology, Shijiazhuang, Hebei, China

**Abstract.** The path planning algorithm is one of the most important algorithms in indoor mobile robot applications. As an integral part of ground mobile robot research, the path planning problem has greater research and application value. Based on machine learning, the mobile robot is continuously tried and trained in the simulation environment to eventually achieve the optimal path planning requirements for real-time obstacle avoidance, resulting in a new path planning algorithm. To make the planning goal smoother, after optimizing the global path planning A\_star algorithm, it is necessary to combine the Q-learning algorithm, so this paper proposes the HA-Q algorithm. Under the HA-Q algorithm, the mobile robot can smoothly move from the specified starting point to the target point where the specified function is designated, to realize the functions of obstacle avoidance and path selection. After some simulation experiments, the HA-Q algorithm is more consistent with the ground mobile robot movement in the actual scene compared to the traditional algorithm. At the same time, these experimental results also show that the algorithm can be used to obtain a short and smooth path, avoid obstacles in real time, and effectively avoid the problem of falling into a locally optimal solution.

**Keywords:** Path planning · Mobile robot · A\_star · Q-learning

## 1 Introduction

With the rapid development of artificial intelligence, indoor mobile robots have entered people's lives [1]. Path planning is an indispensable part of the autonomous navigation process of mobile robots [2]. Path planning means designing the desired path for mobile carriers to avoid obstacles and arrive at the designated destination when there are obstacles in the space [3–5]. It can be divided into global path planning and local path planning.

The A\_star algorithm is a very effective path optimization algorithm, which is faster than the Dijkstra algorithm [6]. It first appeared in 1968, and the overall framework of

the algorithm is a graph traversal search algorithm [7]. Unlike most other graph search algorithms, it uses a heuristic function to estimate the distance between any point on the map and the target point. Through this heuristic, it is possible to coordinate the search in the best direction. In an unknown environment, the reward and punishment mechanism of the Q-learning reinforcement learning algorithm can apply incentives at the target node, so that the incentives are transmitted along the path. Rewards and punishments are interdependent throughout the learning process [8]. Finally, the mobile robot will get closer and closer to the target point.

The A\_star algorithm is well mature and has the advantage that it is fully capable of capturing the solution. However, the resulting disadvantage is that the algorithm is complicated in the dynamic moving obstacle environment. Therefore, the improved A\_star algorithm has been widely studied. Xin Yu et al. proposed the improved A\_star algorithm that adjusts the number of searchable neighborhoods from eight discrete ones to infinite ones [9]. Chen Guangrong et al. [10] proposed a path planning method based on the combination of convex optimization [11] and the A\_star algorithm. This method allows the A\_star algorithm to use a large-scale grid to plan the general trend of an optimal path to improve efficiency. Then the ir-SDP method is used to solve A maximum convex polygon barrier-free area for the position of the mobile robot so that the robot can carry out motion planning and obstacle avoidance processing in this area.

Zhu Zhibin et al. used system data to iteratively solve the control method that minimizes a given objective function in order to achieve consistency in multi-intelligent systems [12]. Later Osmanovic D et al. [3] proposed a fuzzy neural network so that the mobile robot can avoid static and dynamic obstacles autonomously after making full use of sensor information and information collection of the surrounding environment for navigation. Feng Shuo et al. [13] nested deep learning into the Q-learning framework for robot path planning in 3D disaster relief environments. But the A\_star algorithm cannot guarantee that the search path is optimal when there are multiple minima, and its space growth is exponential, which will cause too many redundant points. The method of adding the direction cost to the cost function of the A\_star algorithm is adopted to reduce the number of raster searches. Therefore, it is necessary to use the improved A\_star algorithm. At the same time, to optimize the effect of path planning, this paper takes the environmental information based on learning prior knowledge as the research condition and performs reinforcement learning on this.

## 2 A\_star Algorithm

### 2.1 Traditional A\_star Algorithm

The traditional A\_star algorithm mainly consists of the Dijkstra algorithm and the greedy algorithm. To some extent, it retains the ability of the Dijkstra algorithm to find the shortest path, whereas the greedy algorithm can restrict its blind iterative operation. The main idea of the traditional A\_star algorithm is to select the next node by using a heuristic search method based on the global map information. At the same time, it is necessary to evaluate the cost consumed by the initial node to the current node and the cost expected to be consumed by the current node to the destination. The output of the

evaluation function is used to evaluate the input of all nodes in a neighborhood of the current node. The evaluation function can be expressed as:

$$f(n) = g(n) + h(n) \quad (1)$$

In the formula,  $g(n)$  is the rollback cost, which indicates the consumption cost generated from the initial node to the current node;  $h(n)$  is the forward cost, which is the cost expected to be consumed from the current node to the destination;  $f(n)$  is an evaluation function, which represents the global substitution value integrating two different costs. Its main function is to evaluate the size of the function and to determine whether the planned path is convenient.  $g(n)$  and  $h(n)$  are mutually restricted [14].  $g(n)$  is the traversed set in the whole process and its size will not change because of subsequent operations. Therefore, if  $f(n)$  is to be sufficiently small, the main goal of the study should be to minimize  $h(n)$ . Just like the Dijkstra algorithm, the A\_star algorithm also maintains an Open List and a Closed List [13], where the Open List stores adjacent grids which already been searched grids and they are the nodes that will be searched by the algorithm. Closed List stores nodes, which is the smallest node of  $f(n)$  in the Open List for each sort.

The implementation process of the traditional A\_star algorithm is as follows:

- (1) First, the starting point of the ground mobile robot is denoted as node  $S$ , and it is put into the Open List. At this point, Closed List is an empty table;
- (2) Nodes in the area covered by obstacles in the known environment or nodes in the unknown environment need to be recorded as obstacle nodes and stored in the Closed List;
- (3) Search for reachable nodes around  $S$ , which are also nodes whose value is 0 in the cost grid map, and place them in Open List;
- (4) Ejecting the node  $S$  in the Open List and adding the node  $S$  to the Closed List. At the same time, the value of  $f(n)$  about each reachable node needs to be calculated, and the node with the smallest value of  $f(n)$  should be taken as the next moving node of the ground mobile robot. The search will fail if the Open List is empty at this point and a viable path cannot be found;
- (5) It is necessary to judge whether there is a target node in the Closed List. The algorithm will successfully search for the path if there is one. Instead, go to step (6);
- (6) Traverse all the grids in the neighborhood of the current node  $n$ , select a grid, in which the value of  $f(n)$  is the smallest, and set the node  $m$  as the next moving node. Then pop the current node  $n$  from the Open List and put it into the Closed List;
- (7) Determine whether node  $m$  is in the Open List and Closed List:
  - 1) If the node  $m$  is not in the Open List and Closed List, add it to the Open List;
  - 2) If node  $m$  is in the Open List, it is necessary to compare the calculated value of node  $m$  in  $f(n)$  with the value of node  $m$  in  $f_{pre}(n)$  stored in the Open List. The stored value of  $f_{pre}(n)$  should be replaced by the value of  $f(n)$  if  $f(n)$  is less than  $f_{pre}(n)$ , and the  $m$  node is will be added to the Closed List;

- 3) The node will be skip if the node  $m$  exists in Closed List, which indicates that the node is on the current optimal path, then return to the previous step and continue to compare other child nodes;

(8) Repeat steps (5) to (7) until the target node is searched or the Open List is empty.

## 2.2 Hybrid A\_star Algorithm

The A\_star algorithm doesn't take into account the direction of motion of the object, but the Hybrid A\_star algorithm considers the actual situation of the object movement. The mobile robot starts from a specific position and can only reach the position where can be possible. When the A\_star algorithm is used to search, another path is finally obtained because of the algorithm itself or environmental influences. In this paper, the Hybrid A\_star algorithm is considered under the premise of considering the moving direction of the object. In the Hybrid A\_star algorithm, the actual motion constraints of the object are considered, so the mobile robot can appear at any position in each grid, which is more in line with the actual planning situation. The Hybrid A\_star algorithm takes into account an additional theta compared to the A\_star algorithm for grid search, in which case the continuous 3D state space  $(x, y, \theta)$  becomes a grid.

The heuristic function of the A\_star algorithm search efficiency is crucial, and it also is important to have reasonable and effective research to estimate the target price of the extended node. The function is divided into two kinds in the Hybrid A\_star algorithm: no obstacle of integrity constraints inspired the cost and the integrity of the heuristic cost have obstacles.

The Hybrid A\_star algorithm is similar to the A\_star algorithm, the key difference is that the Hybrid A\_star state transformation occurred in the continuous space (extended node), rather than a discrete space. Although the Hybrid A\_star algorithm search in discrete grid map building, each path point is not limited by the grid, and the excess of similar path points are pruned with the help of the grid.

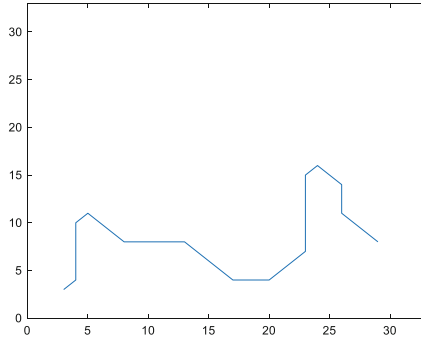
## 2.3 Simulation Experiment

This simulation experiment will be simulated in MATLAB, the rendering shown in Fig. 1 is the simulation result in the platform. The blue curve is the path planning of the traditional A\_star algorithm and the red color curve is the Hybrid A\_star algorithm in the figure. The Hybrid A\_star algorithm makes the integral path planning algorithm smooth. The traditional A\_star algorithm is shown in Fig. 1. And Fig. 2 and Fig. 3 show the application effect of the Hybrid A\_star algorithm in path planning.

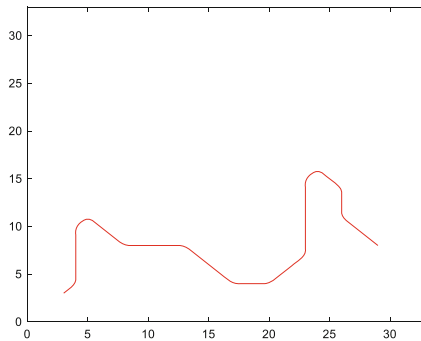
# 3 Q-Learning Algorithm

## 3.1 Q-Learning Algorithm

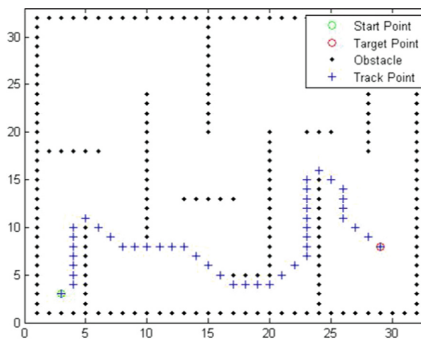
The Q-learning algorithm is a stand-alone control algorithm as one of the reinforcement learning methods [15, 16]. It can be applied to any Markovian decision process to obtain



**Fig. 1.** The traditional A\_star algorithm.



**Fig. 2.** The hybrid A\_star algorithm.



**Fig. 3.** Application of the hybrid A\_star algorithm in path planning.

an optimal policy. The Q-learning algorithm [17] is a general reinforcement learning algorithm that uses iterative computation to approximate the optimal value. The robot will tend to choose the path again if the environment gives a positive reward ( $+r$ ) for this choice; conversely, the tendency will decrease if the environment gives a negative reward ( $-r$ ) for this choice. The Q-learning algorithm continuously adjusts the previous

strategy through the feedback result information, so that the algorithm realizes a process of dynamic allocation [18].

$$r_t = r(s_t, a_t, s_{t+1}) \quad (2)$$

From Eq. 2, it can be seen that the probability distribution is determined by the state-action estimate at the beginning of the moment  $t$ . The update rule for the optimal value function obtained by iterative calculation is varied according to Eq. 3.

$$Q(s, a) = r(s, a) + \gamma \max_{a'} Q(\delta(s, a'), a') \quad (3)$$

In the above equation  $\gamma \in (0, 1)$  is the commutation factor, the reward obtained from the execution of the action  $a$  by the state  $s$  is  $r(s, a)$ , and the function to determine the state  $s$  from the next action  $a'$  and the action to be performed is  $\delta$ . The definition of the Q-function is the basis of the Q-learning algorithm [19]. The combination of the Q-learning algorithm with the robot path planning algorithm continuously selects and updates the Q-value. Also, its exploration coefficients keep decreasing with the training time of the algorithm, i.e., the optimal is selected to the maximum extent (Table 1).

### 3.2 Simulation Experiment

**Table 1.** Q-learning algorithm pseudo-code.

Q-learning algorithm
Initialize $Q(s, a)$ arbitrarily
Repeat (for each episode)
Initialize $s$
Repeat (for each step of the episode)
Choose $a$ from $s$ using policy derived from $Q$ (e.g., $\epsilon$ -greedy)
Take action $a$ , observe $r, s'$
$Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
$s \leftarrow s'$
Until $s$ is terminal

In the MATLAB software simulation of the robot walking room when the room number is from 0 to 5, from the beginning of the room to the target room, the output of the optimal strategy is shown in Table 2.

## 4 Mixed Algorithm

The HA-Q algorithm is guided by the global path mainly based on the A\_star algorithm, and the Q-learning algorithm is used to adjust the local path in real-time. A compound

**Table 2.** Q-learning algorithm optimal strategy.

Q-learning algorithm optimal strategy
Initialized state 1
the robot goes to 5
the robot goes to 6

path planning strategy is developed according to the ground mobile robot's attributes and motion characteristics, which effectively solves the path planning and real-time obstacle avoidance problems of mobile robots in complex indoor environments, and ensures that the robot can safely and smoothly reach the target point.

The HA-Q algorithm absorbs the advantages of global path planning algorithm and machine learning algorithm, which is more efficient and practical compared with single path planning. The following steps shows the flow chart of the HA-Q algorithm:

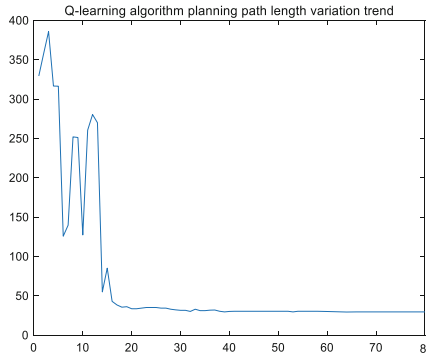
1. Initialization of the exploration factor, maximum number of iterations, termination state parameters, target state parameters, maximum count threshold, start update moment, number of iterations, the current moment, action-value function, number of visits to the state-action pair, success path, and success path storage table for the single robot system;
2. Determine whether the number of iterations is greater than the maximum number of iterations, if yes: execute step 3; if not: initialize the current state parameters and then execute step 3;
3. Generating a random number, comparing the random number with the exploration factor and selecting an action command, and calculating the parameters of the robot's running state and the reward function after executing this action command based on this action command;
4. Determine whether the run state parameter is equal to the termination state parameter, if yes: continue to determine whether the run state parameter is equal to the target state parameter. If equal, store the success path into the success path storage table, execute the iteration count self-add one, and then return to step 2; if not equal, execute the iteration count self-add one, and then return to step 2; if not: execute the next step;
5. Determine whether the start update moment is less than or equal to the current moment. If yes: store the reward function, execute the visit count of the state-action pair since plus one, and then execute the next step; if not: determine whether the visit count of the state-action pair is equal to the maximum count threshold, if yes, update the action value function, and then execute the next step, if not, then execute the next step;
6. Store the run status parameters into the success path, execute the current moment self-add one, and return to step 3;
7. After the action value function is obtained, the action command is selected from the action value function according to the preset initial state parameters, and repeated to obtain the optimal path for the single-robot system.

The experiments in this subsection will be conducted in the simulation platform MATLAB. As can be seen from the results of Fig. 4 and Fig. 5, the proposed HA-Q algorithm in this paper can obtain a complete planning path, which combines the characteristics of the improved A\_star algorithm and the Q-learning algorithm. At the same time, the HA-Q algorithm makes up for the shortcomings of these two algorithms to a certain extent, which further analyzes and verifies the obstacle avoidance technology to meet the requirements of safety and stability, and can ensure that the indoor mobile robot can quickly and smoothly reach the target point.

This experiment will be simulated in MATLAB. The Q-learning algorithm finds the trajectory process as shown in Fig. 4. And the trend of the planned path length is shown in Fig. 5.



**Fig. 4.** The Q-learning algorithm finds the trajectory process.



**Fig. 5.** Trend of the planned path length.

From the above experiments, it can be proved that the trend of path planning effect based on the Q-learning algorithm gradually becomes better, the path length for conducting search path gradually becomes shorter, and the planning process also becomes smooth under the planning of the Hybrid A\_star algorithm.



## 5 Conclusion

The path planning system of ground mobile robots is mainly divided into two kinds, one is a global path planning system, and the other is a machine learning system. In this paper, a real-time obstacle avoidance strategy based on an improved A\_star algorithm and Q-learning algorithm is proposed. Experiments further verify that the HA-Q algorithm meets the requirements of obstacle avoidance technology and can ensure the indoor mobile robot reaches the target smoothly.

**Acknowledgements.** We acknowledge funding from the sub project of national key R & D plan covid-19 patient rehabilitation training posture monitoring bracelet based on 4G network (Grant No. 2021YFC0863200-6), the Hebei College and Middle School Students Science and Technology Innovation Ability Cultivation Special Project (Grant No. 22E50075D), (Grant No. 2021H010206), and (Grant No. 2021H010203).

## References

1. Zhang, H.D., Zheng, R., Cen, Y.W.: Present situation and future development of mobile robot path planning technology. *Acta Simulata Systematica Sinica* **17**, 439–443 (2005)
2. Zi, B., Lin, J., Qian, S.: Localization, obstacle avoidance planning and control of a cooperative cable parallel robot for multiple mobile cranes. *Robot. Comput. Integr. Manufact.* **34**, 105–123 (2015)
3. Osmankovic, D., Tahirovic, A., Magnani, G.: All terrain vehicle path planning based on D\* lite and MPC based planning paradigm in discrete space. In: 2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pp. 334–339. Munich, Germany (2017)
4. Liang, C.L., Zhang, X.K., Han, X.: Route planning and track keeping control for ships based on the leader-vertex ant colony and nonlinear feedback algorithms. *Appl. Ocean Res.* **101**(1), 102239 (2020)
5. Yuan, Q., Han, C.S.: Research on robot path planning based on smooth A\* algorithm for different grid scale obstacle environment. *J. Comput. Theor. Nanosci.* **13**(8), 5312–5321 (2016)
6. Kang, H.I., Lee, B., Kim, K.: Path planning algorithm using the particle swarm optimization and the improved Dijkstra algorithm. In: 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, pp. 1002–1004. Wuhan, China (2009)
7. Sudhakara, P., Ganapathy, V.: Trajectory planning of a mobile robot using enhanced A-star algorithm. *Indian J. Sci. Technol.* **9**(41), 1–10 (2016)
8. Hong, S., Zhu, J.X., Braunstein, L.A., et al.: Cascading failure and recovery of spatially interdependent networks. *J. Stat. Mech: Theory Exp.* **10**, 103208 (2017)
9. Xin, Y., Liang, H., Du, M., et al.: An improved A\* algorithm for searching infinite neighbourhoods. *Robot* **36**, 627–633 (2014)
10. Chen, G.R., Guo, S., Wang, J.Z., et al.: Convex optimization and A-star algorithm combined path planning and obstacle avoidance algorithm. *Control and Decision* **35**, 2907–2914 (2020)
11. Stephen, B., Lieven, V.: *Convex Optimization*. Cambridge University Press, England (2004)
12. Zhu, Z.B., Wang, F.Y., Yin, Y.H.Z., et al.: Consensus of discrete-time multi-agent system based on Q-learning. *Control Theory Appl.* **38**(07), 997–1005 (2021)
13. Feng, S., Shu, H., Xie, B.Q., et al.: 3D Environment Path Planning Based On Improved Deep Reinforcement Learning. *Comput. Appl. Softw.* **38**(01), 250–255 (2021)

14. Zhang, H.T., Cheng, Y.H.: Path finding using A\*algorithm. *Microcomput. Inf., Control and Decision* **24**, 238–239+308 (2007)
15. Qiao, J.F., Hou, Z.J., Ruan, X.G.: Neural network-based reinforcement learning applied to obstacle avoidance. *J. Tsinghua Univ. (Sci. Technol.)* **48**, 1747–1750 (2008)
16. Geng, X.J.: *Self-Organizing Collaborative Target Search of Mobile Multi-Agent Based on Reinforcement Learning*. Nanjing University of Posts and Telecommunications (2020)
17. Huang, B.Q., Cao, G.Y., Wang, Z.Q.: Reinforcement learning theory, algorithms and application. In: 2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC), pp. 34–38 (2006)
18. Hong, S., Yang, H., Zhao, T., et al.: Epidemic spreading model of complex dynamical network with the heterogeneity of nodes **47** (9–12), 2745–2752 (2016)
19. Watkins, C.J.C.H.: Learning from delayed rewards. *Robot. Auton. Syst.* **15**(4), 233–235 (1989)