EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Indoor robot path planning assisted by wireless network

Xiaohua Wang[1*], Tengteng Nie[1] and Daixian Zhu[2]

Abstract

Indoor robot global path planning needs to complete the motion between the starting point and the target point according to robot position command transmitted by the wireless network. Behavior dynamics and rolling windows in global path planning methods have limitations in their applications because the path may not be optimal, there could be a pseudo attractor or blind search in an environment with a large state space, there could be an environment where offline learning is not applicable to real-time changes, or there could be a need to set the probability of selecting the robot action. To solve these problems, we propose a behavior dynamics and rolling windows approach to a path planning which is based on online reinforcement learning. It applies Q learning to optimize the behavior dynamics model parameters to improve the performance, behavior dynamics guides the learning process of Q learning and improves learning efficiency, and each round of intensive learning action selection knowledge is gradually corrected as the Q table is updated. The learning process is optimized. The simulation results show that this method has achieved remarkable improvement in path planning. And, in the actual experiment, the robot obtains the target location information by wireless network, and plans an optimized and smooth global path online.

Keywords: Behavior dynamics, Rolling windows, Q learning, Path planning

1 Introduction

Autonomous navigation in an unknown environment should require that the robot finds a suitable, safe, smooth, and even optimal path from the starting point (position and attitude) to the end point (position and attitude). Researchers have performed a large amount of research, using artificial neural networks [1],ant colony algorithm[2], and so on combined with fuzzy logic to achieve understanding and rapid classification of current environmental perceptions, the artificial potential field method [3], behavior dynamics [4], Firefly algorithm [5], full coverage path planning algorithm [6], lidar acquisition data and RBPF-SLAM [7] to construct maps, and other methods to solve the autonomous navigation problem in unknown environments for global planning of the robot path or for a combination of global and local planning [8–10]. Researchers combine behavior dynamics and rolling windows to perform path planning [11, 12]; the local sub-objective is optimized by using a heuristic function according to the local information in the rolling window obtained by the robot; the behavior dynamics model is used to perform autonomous path planning [13] in the rolling windows; and the planning trajectory of a series of windows is connected end to end to realize the global path planning. Behavior dynamics uses point attractor and point repulsor to build robot behavior, and the robot's heading angle and motion speed are used as behavior variables to describe robots moving in a plane [14], but in the application, the line speed is limited by the heading angular velocity control and causes a deadlock; in addition, because of the non additivity of the virtual forces, there are also pseudo attractor problems. The literature [15, 16] uses the linear velocity control method to solve this problem, but it is required that the linear velocity ensures that the robot's heading angle is always near the heading angle of the attractor. At the same time, there is also a situation in which the path between windows in the rolling windows method is not smooth because of the different directions of the robot motion.

* Correspondence: 19970704@xpu.edu.cn
[1]College of Electrics and Information, Xi'an Polytechnic University, Xi'an, China
Full list of author information is available at the end of the article

© The Author(s). 2019 **Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Reinforcement learning is a machine learning algorithm that adjusts its own action strategy by interacting with the environment and ultimately finds the optimal strategy to achieve the goal. The adaptive navigation and obstacle avoidance algorithm based on reinforcement learning can solve the abovementioned problems well and has strong adaptive ability. However, the intensive learning method takes too long to train, and the convergence is slow. The general idea of the application of reinforcement learning in robot path planning is as follows [17]: robots roam in the environment, learn obstacle avoidance movements, and gain experience; when robots enter new unknown environments, they rely on previously learned experience to plan obstacle avoidance paths. This kind of approach which involves blind search makes the target reward and punishment spread very slowly, so it is almost impossible to solve the problem of robot navigation in the complex environment with a large state space; in addition, the offline learning method does not apply to the real-time changing environment. Researchers optimize from the aspect of reducing the state space, the action selection, and so on, but they have lost useful information, so the learning cannot be adequately optimized. Some scholars introduce the theory of hierarchical reinforcement learning, and they decompose complex reinforcement learning problems into sub-problems that are easy to solve. By solving these sub-problems separately, the original reinforcement learning problems are finally solved [18, 19]. It has also been proposed to use neural network structures to automate the hierarchical abstraction and learning of problems [20]. The literature [21] has proposed a reinforcement learning method that is based on path knowledge enlightenment, which uses the knowledge gained by online learning to reduce the blindness of the search, but it needs to set the robot action selection probability fictitiously.

Taking the above research methods into consideration, an idea comes to our mind. If the robot can choose the action itself according to the current state and the experience gained in the learning process, it can not only apply knowledge to enhance the learning efficiency but also enhance the robot's autonomous navigation ability.

Q learning with behavior dynamics and rolling windows is proposed in this paper to solve the problems in robot autonomous navigation while applying the above methods. On the one hand, the online knowledge obtained by applying behavior dynamics improves the robot learning efficiency in this window, and it continuously corrects the knowledge obtained in the previous window to optimize the subsequent learning process. On the other hand, the Q-learning behavior dynamics model parameters enable the robot to adaptively change the motion scheme online to adapt to the state in the process of interacting with the environment, thereby obtaining a good running state and planning an optimal path.

This paper is organized as follows. In Section 2, the robot behavior dynamics model and rolling window path planning method is introduced. Q learning algorithm and the parameters of the Q learning algorithm correspond to the motion state of the robot are introduced in Section 3. Afterward, behavior dynamics and rolling windows path planning based on Q learning method is proposed in Section 4. Simulation and experiments are implemented to verify the proposed method in Section 5. Section 6 gives conclusions.

2 Behavior dynamics and rolling window path planning

2.1 Robot behavior dynamic model

Behavior dynamics uses point attractor and point repulsor to build robot behavior, and it uses the robot's heading angle and motion speed as the behavior variables to describe robots that move in the plane. The description of the robot and its environment is shown in Fig. 1.

The behavior of the robot is determined by both moving toward the target and performing obstacle avoidance. The behavior dynamics model is coupled by a behavior state dynamics model and a behavior pattern dynamics model. It is a nonlinear differential equation group, and the dynamic state model of robot navigation behavior can be expressed as follows:

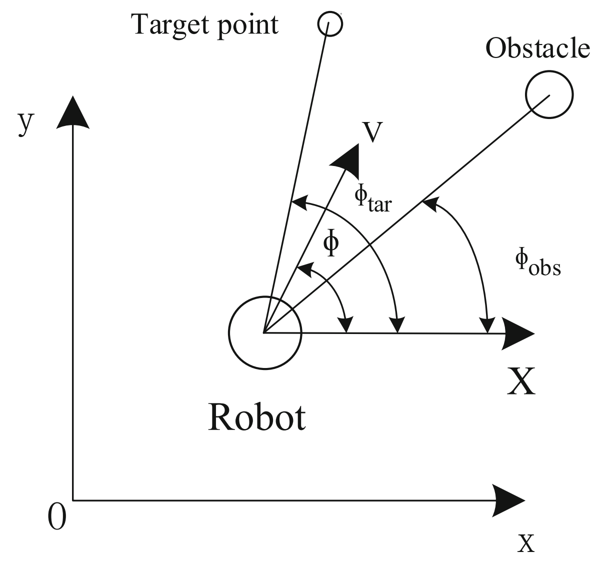


Fig. 1 Robot model. The coordinate system OXY is the world coordinate system, the robot coordinate system takes the robot position as the origin, the *X*-axis is parallel to the *X*-axis of the world coordinate system, and the *Y*-axis is perpendicular to the *X*-axis (not drawn)

$$\phi = |w_{tar}|f_{tar} + |w_{obs}|\sum_n f_{obsti} \tag{1}$$

where f_{tar} is toward the target behavior function, f_{obsti} is the obstacle avoidance behavior function, n is the number of obstacles in the environment, and w_{tar} and w_{obs} are the weights toward the target behavior and the obstacle avoidance behavior. This equation constitutes the behavior dynamics model:

$$\begin{cases} w_{tar} = \alpha_1 w_{tar}\left(1-w_{tar}^2\right)-\gamma_{21} w_{tar} w_{obs}^2 \\ w_{obs} = \alpha_2 w_{obs}\left(1-w_{obs}^2\right)-\gamma_{12} w_{obs} w_{tar}^2 \end{cases} \tag{2}$$

Where α_1, α_2, γ_{12}, and γ_{21} are environment-related parameters; α_1 and α_2 respectively, represent the competitive advantage of the toward the target behavior and the obstacle avoidance behavior; γ_{12} is the coefficient of the constraining ability of the running toward target behavior to the obstacle avoidance behavior in competition; and γ_{21} is the coefficient of obstacle avoidance behavior's constraining ability to target behavior in competition. These four parameters describe the extent of coupling between the robot and the environment, and its value changes as the robot's position changes.

If $(w_{tar}, w_{obs})^T = (0,0)^T$; then, the fixed points $(w_{tar}, w_{obs})^T$ of Eq. (2) are $(0,0)$, $(0,\pm1)$, $(\pm1,0)$, and $(\pm A_1, \pm A_2)$, where:

$$A_1 = \sqrt{\frac{\alpha_2(\alpha_1-\gamma_{12})}{\alpha_1\alpha_2-\gamma_{12}\gamma_{21}}} \quad A_2 = \sqrt{\frac{\alpha_1(\alpha_2-\gamma_{21})}{\alpha_1\alpha_2-\gamma_{12}\gamma_{21}}}$$

According to Lyapunov's first approximation theorem [22], as long as the parameters α_1, α_2, γ_{12}, and γ_{21} are properly designed so that all of the eigenvalues of the Jacobi matrix that satisfy Eq. (2) have negative real parts, then the fixed point is stable; otherwise, the fixed-point is unstable.

Considering the overall behavior equation of the robot (1), the stability of the above four abovementioned types of fixed points in Eq. (2) can be seen:

1. The first type of fixed point $(0,0)$ means that the weights of the target behavior and the obstacle avoidance behavior are zero, and both are in the state of complete suppression (i.e., off), which indicates that the robot is in a roaming disorder state. In the robot navigation process of this paper, this state is not needed. As long as $\alpha_1 > 0$ and $\alpha_2 > 0$ are satisfied, the fixed point is unstable, and this situation can be avoided.
2. The second type of fixed point $(0,\pm1)$ means that in the overall behavior of the robot, there is only obstacle avoidance behavior without target behavior; this situation occurs in the case of robot emergency obstacle avoidance, and as long as $y_{12} >$ 0, $\alpha_2 > 0$ is satisfied, this fixed point is stable.
3. The third type of fixed point $(\pm1,0)$ means that when there is no obstacle in front of the robot, the robot moves directly to the target point, as long as $\gamma_{21} > \alpha_2$ and $\alpha_1 > 0$ are satisfied, the fixed point is a stable fixed point.
4. The fourth type of fixed point $(\pm A_1, \pm A_2)$ means that the weights of the two behavior modes are not 0 or 1 but are between 0 and 1. At this time, both of them act simultaneously,and the navigation behavior of the robot emerges. It is necessary to move to the target while avoiding obstacles and being in a competitive state. Since the parameters α_1, α_2, γ_{12}, and γ_{21} are descriptions of the coupling state between the robot and the environment, the value changes as the position of the robot changes. Therefore, the fixed point is changed. The fixed point is stable as long as $\alpha_1 > 0$, $\alpha_2 > 0$, $\alpha_1 > \gamma_{12}$, $\alpha_2 > \gamma_{21}$, $\gamma_{12}\gamma_{21} < 0$, or $\gamma_{12} > 0$, $\gamma_{21} > 0$ are satisfied.

It can be seen from the above stability analysis that the stability of various fixed points is mutually exclusive.-When a fixed point is stable at a certain time, other fixed points are unstable, as long as there are reasonable design parameters α_1, α_2, γ_{12}, γ_{21} in such a way that the parameters can accurately reflect the relationship between the robot and the environment; then, the correct navigation of the robot can be realized.

In the robot autonomous navigation task, the robot exhibits a mutually coordinated target behavior and obstacle avoidance behavior. In interactions with the environment, the robot needs to independently determine the direction and speed of the navigation. The robot's heading angle dynamics model is:

$$\ddot{\phi} = f_{\phi d}(\dot{\phi}) + |w_{tar}|f_{\phi t}(\phi, d_t) + |w_{obs}|\sum_n f_{\phi o}(\phi, d_o) \tag{3}$$

where $f_{\phi d}(\dot{\phi})$ is related to the damping coefficient of the system, the distance between the robot and the target point is d_t, and d_o is the distance between the obstacle and the robot. The speed dynamics model of the robot is expressed as follows:

$$\dot{v} = |w_{tar}|\left(v-v_{\exp\_t}\right) + |w_{obs}|\left(v-v_{\exp\_o}\right) \tag{4}$$

where $V_{\exp\_t}$ is the desired speed of the target behavior and $V_{\exp\_o}$ is the desired speed of the obstacle avoidance behavior.

When the robot travels in an unknown environment, it changes its heading angle and speed in real time according to the information obtained from the environment, and it autonomously avoids obstacles.

2.2 Rolling window path planning

The rolling window path planning method is to use the local environment information acquired by the sensor in real time near the current position of the robot and to perform local path planning under each sub-window to realize the robot behavior control. As the robot advances, the sub-window scrolls forward. This kind of repeated scrolling window local planning realizes a combination of optimization and feedback to complete the global planning [23].

The robot acquires environmental information within a certain range of its current location. Regarding this environment as a window, the heuristic function is used to determine the local sub-target point of the window based on the location of the feature point and the obstacle. Starting from the current position, the path to reaching the local sub-goal is planned. During the entire navigation process, the window is scrolled forward accordingly, and the prior information in the local environment is corrected. To determine the sub-objects in the window is the premise of the local path planning. The sub-objectives are usually judged by a heuristic function. The heuristic function in this paper is as follows [16]:

$$F(P) = G(P) + H(P) + J(P) \tag{5}$$

The $G(P)$ value represents the distance from the sub-target in the set P to the global target guide point G, and the set P always guarantees that the sub-target exists in the window. $H(P)$ is the cost of P to travel to the global target guidance point, ensuring that the sub-target will not be located in the robot forbidden zone of existing obstacles; $J(P)$ guarantees that the sub-targets of two adjacent windows at moment i and moment $i-1$ are different. The point at which $F(P)$ is the minimum must be included in the point set at which $H(P)$ and $J(P)$ are simultaneously minimized, and if the point at which $G(P)$ is minimized is also selected at the point set, then the sub-target point of this window is ascertained. The location of the global target guidance point is the main factor that determines the target of each window. The sub-target point determined by the heuristic function is always close to the global target guidance point.

In each window, the experience acquired by the robot does not form the knowledge of applying subsequent navigation behavior, which results in problems such as the path of the robot's window junction place not being smooth and the path not being optimal.

3 Q learning algorithm

3.1 Original Q learning algorithm

Most reinforcement learning systems apply a theoretical framework of the Markov decision process (MDP) [24]. The MDP includes the following four parts: S is the environmental state set, A is the set of actions, $W: S \times A \rightarrow T(A)$ is the set of environmental state transitions, and $R: S \times A \rightarrow R$ is the feedback signal of the environment state to action which is also a reward value. At each time t moment, robot obtains an environmental state s_t and executes a corresponding action a_t according to a certain strategy, and then, the environment state changes according to the action. At the next moment, robot receives the reward value r_{t+1} from the environmental state; that is, for each group given (s, a), here is a reward value R_s^a; at the same time, robot moves to new environment state s_{t+1}, and the transition probability is p_s^a. In this learning process, experience knowledge $(s_t, a_t, s_{t+1}, a_{t+1})$ is obtained. Based on this empirical knowledge, the $Q(s, a)$ value is modified. Robot's goal is to seek in each discrete state S such that $Q(s, a)$ is maximal.

The Q learning algorithm is as follows:

```
Initialization $Q(s, a)$
For each round of learning {
Selection status S = source;
While (Does not satisfy the end condition)
In the current state $s$, we select the behavior $a$ according to a certain strategy;
Performing action $a$, obtaining reward $r$, entering state $s_{t+1}$, and updating the $Q$ value:
```

$$Q(s_t, a_t) = \begin{cases} (1-\beta)Q(s_t, a_t) \\ +\beta[r_t + \gamma \max Q(s_{t+1}, a_t) \quad s = s_t, a = a_t\} \\ Q(s_t, a_t) \quad \text{otherwise} \end{cases}$$

Among these, $0 < \gamma < 1$ is called a discount coefficient and β is called a learning rate.

In Q learning, the robot selection action is usually random or blind. In the single-step reinforcement learning process, the reward obtained after searching for the target can only be propagated one step in the next round, and most of the experience results gained in each round of the search is lost and not fully utilized.

3.2 Q learning about robot

The parameters of the robot behavior dynamics model parameters α_1, α_2, γ_{12}, and γ_{21} are in the range (0–1). Therefore, the set of environmental statuses S is defined as the range of values of the parameters that need to be reinforced for learning, that is (0,1). The set A of corresponding actions is the turning angle of the robot. When accounting for the harsh environment of the robot during its movement, the robot's turning angle regards the current forward direction as zero degrees; expanding to the left and right has an angle range of 60°, which is divided into seven discrete actions, − 60, − 40, − 20, 0, 20, 40, and 60. During the movement of the robot, when it is closer to obstacles, the robot decelerates, and it turns the specified angle in a timely manner according to the angle between the robot and the obstacles.When the robot is farther away from the obstacles, it will accelerate or turn the specified angle in a timely manner. Therefore, determining the reward function needs to consider two factors: the distance and angle. The value of the reward function for different distances and angles should be different, specifically as follows:

$$r = \begin{cases} \dfrac{1-e^{-x}}{1+e^{-x}} & (d_{\min} > 2R) \quad (\phi_t < \phi_{obs}) \\ -1 & \text{otherwise} \end{cases} \tag{6}$$

where ϕ_t is the robot's heading angle and x is the distance that the robot has traveled at this moment relative to the previous moment.

Since the robot system does not satisfy the constraint conditions that the state/action must be finite discrete in the Q learning algorithm, there is a need for generalization. In this paper, the neural network is applied to generalize the Q learning algorithm [25]; because the value of the kinetic parameter is a real number between zero and one, we take 0.2 as the opening degree, and the state-discrete is expressed as $s = \{s_1, s_2, s_3, s_4, s_5\}$, where $0 < s_1 \leq 0.2$, and so on, $0.8 < s_5 \leq 1$. The input of the back propagation (BP) network corresponds to the five discrete states of the behavior dynamics parameters.The output of the BP network corresponds to the Q value of each turning angle action, and there are seven output values in total.

4 Path planning of behavior dynamics and rolling window based on Q learning

4.1 Q learning action search

Original Q learning usually follows certain rules and selects actions, and the choice of actions is random, which results in decreasing the convergence speed of the learning. Applying behavior dynamics knowledge in the rolling window to guide Q learning can reduce the search space and improve the learning efficiency. At the same time, Q learning adjusts the behavior dynamics parameters in such a way that the robot can better adjust its own running status based on the environmental information.

The Q learning set of actions in this paper is the robot turning angle. The behavior dynamics model also calculates the robot's heading angle at any time.The current heading angle of the robot is used to guide the Q learning search for the action space to reduce the search scope. The current heading angle of the robot is recorded as A; this angle corresponds to the action space B, and the two discrete action ranges on the left and right, which are similar to this angle, are taken as possible performing actions of the robot, thereby shrinking the action space. The probability of execution of each robot action is:

$$P(a_i|s) = \frac{e^{Q(s,a_i)}}{\sum_j e^{Q(s,a_j)}} \tag{7}$$

This equation indicates the probability of selecting action i from j actions in state s.

4.2 Shortest path guided robot action selection

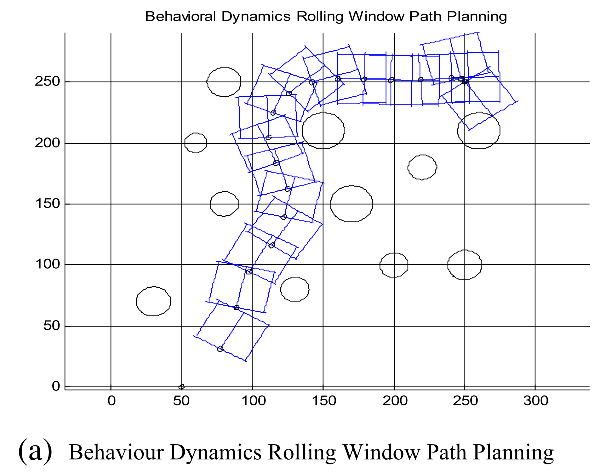
In each scroll window, the robot is learning while moving to the sub-target point as the termination condition. The action of the robot is initially calculated by the behavior dynamics model; with the gradual establishment of the Q value table, in a certain state s, the environmental state also corresponds to a certain action. At this point, there are two possibilities for the robot action. One option is to select the action based on the size of the Q value, and the other option is to use the angle action calculated from formula (3).

This paper is based on the principle in favor of gaining the global shortest path, and it uses the shortest path to determine the robot's choice of action. At moment t, the robot obtains an environmental state s_t that corresponds to the parameters of the behavior dynamics, calculates the distance between the robot and the end point when the robot moves to a new location according to this parameter, and calculates the distance between the robot and the end point when the robot that executes the corresponding action a_t reaches the new position according to the Q learning strategy; then, the robot selects an action according to the short distance between the two distances, and at the same time, the Q value table is updated. If the robot performs actions according to the behavior dynamics parameters, this parameter is compared to the corresponding state value in the Q value table; then it obtain rewards and updates the Q value table. If the robot performs actions according to the Q learning strategy, the Q value table is directly updated.

4.3 Q learning knowledge acquisition

One round of learning of the robot will gain some experience, retain the result of this experience, and be able to optimize the later learning process. This paper uses the value of the reward function to make the knowledge in the previous round of learning be applied.

The specific approach is as follows: in the first sub-window, the robot moves from the starting point to the sub-target point, and the first learning is over. Recording each state and the corresponding turning angle action and then saves the information to table T; in each round of learning after the first window, when the robot performs an action, it traverses the table T. If the state value that corresponds to this action in the table is the same as the current state value, the reward



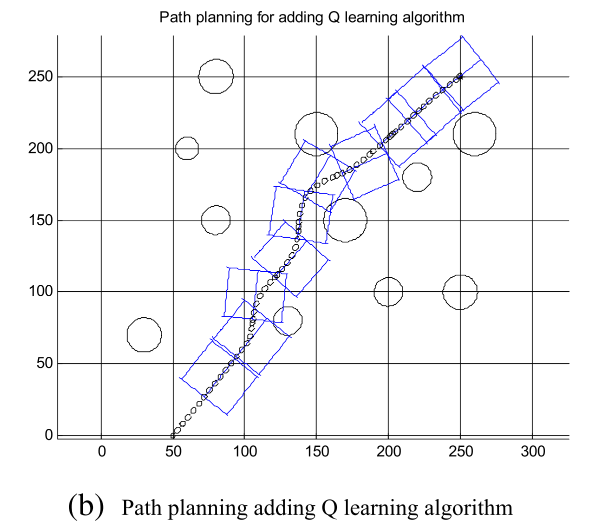


Fig. 2 MATLAB simulation results. The robot is set to run from the origin (50, 0) to the endpoint (250, 250), and the maximum speed does not exceed 2 m/s. There are multiple obstacles of different sizes in the environment. The size of the rectangular window is 40 × 34. In the scroll window, the small "。" indicates the position of the robot and forms the robot's movement trajectory. **a, b** The same environment, in **b** robot can walk from small space to target point. This result show that Behavior Dynamics Rolling Window Path Planning adding Q learning algorithm is much suitable than Behavior Dynamics Rolling Window Path Planning

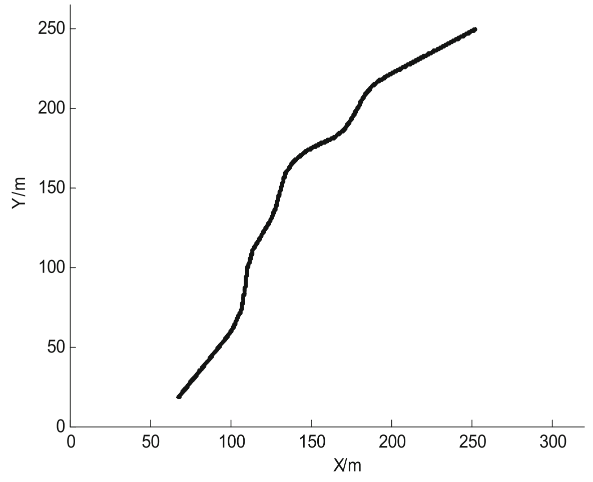
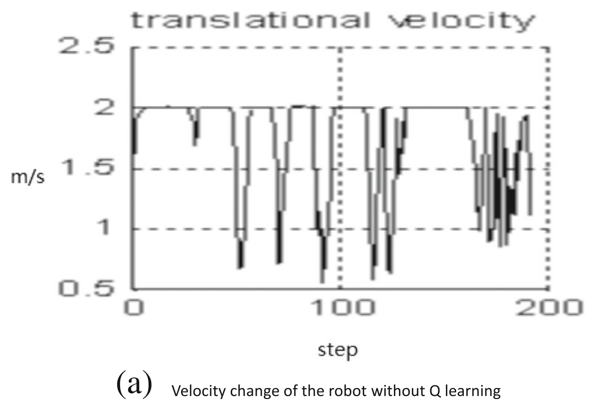


Fig. 3 Running track. The trajectory in this figure is the trajectory of the robot in Fig. 2b when Q learning method is applied in path planning



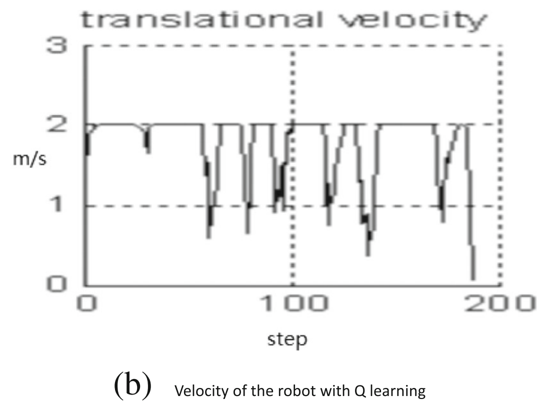


Fig. 4 Comparison chart of speed changes. The vertical coordinate represents the movement speed of the robot (m/s), while the horizontal coordinate represents the number of steps the robot takes (step). The change in the velocity in figure (**a**) of the robot is more gentle than that in figure (**b**)

function becomes 2r; that is, each round of knowledge gained strengthens the learning of the later round. In the process of the robot running, after a series of Q learning, it gradually establishes the Q value that corresponds to the action knowledge, which is used to guide the Q learning, and it also constantly perfects the knowledge itself. Q learning improved the behavior dynamics parameters, and the interaction between the robot and the environment was significantly improved.

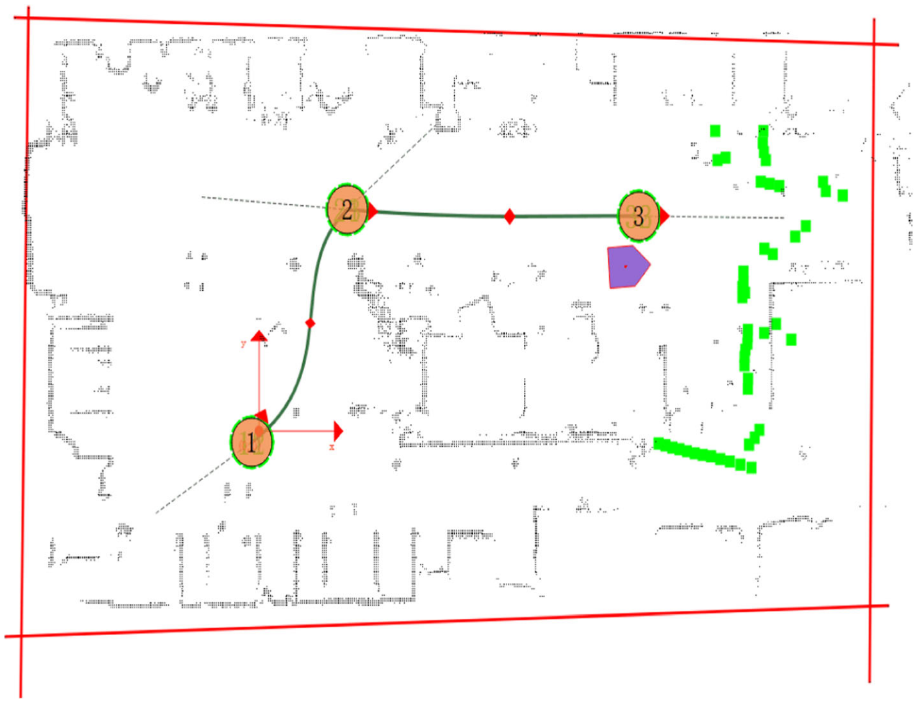


Fig. 5 Robot planning path. The gray part in this figure represents obstacles. The number 1 is the starting point for the robot, and the numbers 2 and 3 are the target points (robot receives the signal sent by the control terminal through the wireless network) of the robot respectively. The curve between the numbers is the actual path of the robot walking, the purple pentagon is the robot, and the green spot is the real-time data scanned by the laser sensor of the robot

4.4 Path planning based on Q learning

Based on the above definition of the environment state space, action definition, action search method, action selection, and design of reward function, the path planning step of the behavior dynamics rolling window based on Q learning proposed in this paper is as follows:

Step 1: System initialization; the robot detects the first child window and determines sub-goals.

Step 2: The robot moves in the child window according to the behavior dynamics model, and it plans the path. The first round of Q learning begins at the same time as the robot movement, and the behavior dynamics model is used to guide the search for Q learning according to the method of Section 3.1; the T table is created.

Step 3: At the starting point of the next window, a new Q learning process is started. A proper reward function value is obtained by querying the table T. The robot performs the action selection according to the method of Section 3.2, and the table T is updated.

Step 4: Repeat step 3 until the robot reaches the global end point.

5 Experiments and results analysis

In path planning based on the behavior dynamics model, the position of the obstacles detected by each rolling window is different, and then, the path walked by the robot is also different. To verify the validity of the above method combination and the continuity and smoothness of the path planning, this paper selects a representative situation in the path planning, and we conduct simulation experiments in the MATLAB.

Figure 2a shows the experiment that applies only the behavior dynamics rolling window method. It can be seen from the figure that the robot can avoid the obstacle and reach the target point. Figure 2b shows the behavior dynamics rolling window method combined with the Q learning algorithm. In contrast to the path in Fig. 2a, the robot can walk in between the obstacles, improving the planned path in Fig. 2a and minimizing the path to reach the destination. In the same external environment and exercise conditions, in Fig. 2b, the path that the robot can plan is more reasonable than that in figure (a) after applying Q learning algorithm.

The treading track of the robot using Q learning algorithm is shown in Fig. 3, and the track shown in Fig. 3 is smooth. It shows that the Q learning improves the phenomenon that the track is not smooth when the sub-windows are connected to each other, making the robot's kinematic performance perform better in the actual operation.

Figure 4 is a comparison of robot speeds in the experiments shown in Fig. 2a and Fig. 2b. It can be seen that the robot speed variation in graph (b) added to the Q learning algorithm is much more moderate than that of graph (a), especially in terms of the distance from the obstacle area after reaching the end point; in graph (b), speed variation trend is obviously better than that of graph (a), which shows that the accumulated knowledge of Q learning can instruct the robot to better interact with the environment and make the robot move more smoothly.

This method is applied to a real robot system. Mobile R robots carry laser ranging sensor urg-10 lx, accurate Odomete Odomete, gyro module GPX, absolute encoder e6b2-cwz6c 10P/R sensor, and wireless data transmission function circuit module; the wireless data transmission module is used to receive the robot target position information in the indoor environment, such as point 2 and 3 (Fig. 5). In the process of walking toward the target point (especially from point 1 to point 2), the robot can plan a reasonable route and walk a smooth trajectory. The experimental results show that the proposed method can improve the walking performance of the robot and realize the online global path smoothing.

6 Conclusions

In this paper, the online Q learning method is used to solve the accuracy problem that exists in the robot behavior dynamics model and the problem of the Not smooth path of the rolling window connection. The behavior dynamic model guided Q learning reduced the search space and improved the learning efficiency. The experience and knowledge accumulated in the previous window are beneficial to the later learning; the gradually improved Q learning online adjusts the model parameters in such a way that the robot exhibits better performance in interactions with the environment, and an optimized and smooth global path is obtained. This method described in this paper can obtain and use knowledge of the environment in real time, which can improve the practicality of the dynamic model and rolling window path planning and plan an optimal path safely and stably.

Abbreviations

BP: Back propagation; MDP: Markov decision process; RBPF-SLAM: Rao-Blackwellized particle filter (RBPF)-simultaneous localization and mapping (SLAM)

Acknowledgements

The research of this paper is supported by the National Natural Science Foundation of China and the project of ministry of education.

Funding

The authors acknowledge the National Natural Science Foundation of China (No.51607133) and Ministry of education of humanities and social science research special tasks (engineering technology personnel training research) (No.18JDGC029).

Availability of data and materials

Not applicable.

Authors' contributions

Xiaohua Wang is the main writer of this paper. She proposed the main idea, deduced the performance of Q learning algorithm completed the simulation, and analyzed the result. Tengteng Nie introduced the Q learning in robot domain. Daixian Zhu deduced the performance of rolling window path planning algorithm completed the simulation, and analyzed the result. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details

[1]College of Electrics and Information, Xi'an Polytechnic University, Xi'an, China. [2]College of Communication and Information Engineer, Xi'an University of Science and Technology, Xi'an, China.

Received: 20 February 2019 Accepted: 18 April 2019
Published online: 22 May 2019

References

1. N.H. Singh, K. Thongam, Mobile robot navigation using MLP-BP approaches in dynamic environments. Arab. J. Sci. Eng. **43**(12), 8013–8028 (2018)
2. W.B. Zhang, X.P. Gong, G.J. Han, Y.T. Zhao, An improved ant Colony algorithm for path planning in one scenic area with many spots. IEEE ACCESS. Commun. **5**, 13260–13269 (2017)
3. J. Borenstein, Y. Koren, Real time obstacle avoidance for fast mobile robots. IEEE Trans. on Systems Man and Cybernetics Commun. **19**(5), 1179–1187 (1989)
4. A. Philipp, I. Henrik, Behaviour coordination in structured environments. Adv. Robot **17**, 657–674 (2003)
5. Y. Shao, S.L. Zhu, R.J. Liu, X.C. Zhou, Diversity-guided dynamic step firefly algorithm. Int. J. of Hybrid Inf. Technol **9**(8), 95–104 (2016)
6. S.S. Mansouri, K. Christoforos, E. Fresk, D. Kominiak, G. Nikolakopoulos, Cooperative coverage path planning for visual inspection. Control Eng. Pract. Commun. **74**, 118–131 (2018)
7. J. Hyunggi, H.M. Cho, J.S. jin, K. Euntai, Efficient grid-based Rao-Blackwellized particle filter SLAM with interparticle map sharing. IEEE/ASME Transactions on Mechatronics. Commun. **23**(2), 714–724 (2018)
8. M. Galicki, Real-time constrained trajectory generation of mobile manipulators. Rob. Autom. Syst. **78**, 49–62 (2016)
9. J. Keller, D. Thakur, M. Likhachev, J. Gallier, V. Kumar, Coordinated path planning for fixed-wing UAS conducting persistent surveillance missions. IEEE Transactions on Automation Science and Engineering, Commun. **14**(1), 17–24 (2017)
10. P.K. Mohanty, D.R. Parhi, Optimal path planning for a mobile robot using cuckoo search algorithm. J. Exp. Theor. Artif. Intell. **28**(1–2), 35–52 (2016)
11. C.A. Liu, X.H. Yan, C.Y. Liu, H. Wu, Dynamic path planning for mobile robot based on improved ant colony optimization algorithm. Acta Electronica sinica. **39**(5), 1220–1224 (2011)
12. J.J. TY Zhang, G.X. Zhang, Analysis of a dual-arm mobile robot dynamics and balance compensation Int. J. Comput. Appl. Technol. **41**(3–4), 246–252 (2011)
13. L. Zhang, X.J. Zhang, K.H. Guo, C.L. Wang, L. Yang, T. Liu, Rolling window optimization for intelligent vehicle trajectory planning in unknown environment. Journal of Jilin University (Engineering and Technology Edition), Commun. **48**(3), 652–660 (2018)
14. Y.F. Zhang, W.L. Li, D. Silva, W. Clarence, RSMDP-based robust Q-learning for optimal path planning in a dynamic environment. Int J Rob Autom. **31**(4), 290–300 (2016)
15. H.D. Zhang, S.H.R. Liu, Autonomous navigation control for mobile robots based on emotion and environment cognition. Kongzhi Lilun Yu Yingyong/ Control Theory and Applications, Commun. **25**(6), 995–1000 (2008)

16. A. Gentile, A. Messina, Dynamic behaviour of a mobile robot vehicle with a two caster and two driving wheel configuration. Veh. Syst. Dyn. **25**(2), 89–112 (1996)
17. H.K. Sajad, B.S. Saeed, S.K. Soroush, Path planning of modular robots on various terrains using Q-learning versus optimization algorithms. Intell. Serv. Robot **10**(2), 121–136 (2017)
18. Z.F. Shao, J.E. Meng, A survey of inverse reinforcement learning techniques. International Journal of Intelligent Computing and Cybernetics, Commun **5**(3), 293–311 (2012)
19. L.W. Zhou, P. Yang, C.L. Chen, Y. Gao, Multiagent reinforcement learning with sparse interactions by negotiation and knowledge transfer. IEEE Transactions on Cybernetics, Commun. **47**(5), 1238–1250 (2017)
20. H. Huang, X.T. Hu, Y. Zhao, M. Milad, Q.L. Dong, S.J. Zhao, L. Guo, T.M. Liu, Modeling task fMRI data via deep convolutional autoencoder. IEEE Transactions on Medical Imaging, Commun. **37**(7), 1551–1561 (2018)
21. H.P. Zhu, F. Tian, K. Wu, S. Nazaraf, Y. Chen, Y.F. Ni, X.H. Zhang, K.M. Chao, Q. H. Zheng, A multiconstraint learning path recommendation algorithm based on knowledge map. Knowl. -Based Syst. **143**, 102–114 (2018)
22. A.J. Krener, M.Q. Xiao, Nonlinear observer design in the siegel domain. SIAM J. Control Optim. **41**(3), 932–953 (2003)
23. L.L. Yu, D.C.H. Kong, X.Y. Shao, X.X. Yan, A path planning and navigation control system design for driver less electric bus. IEEE Access, Commun. **6**, 53960–53975 (2018)
24. C.H. Jaedeug, K.E. Kim, Inverse reinforcement learning in partially observable environments. J. Mach. Learn. Res. **12**, 691–730 (2011)
25. M. Sangiil, M. Ave, Q learning regression neural network. Neural Network World, Commun. **28**(5), 415–431 (2018)

Submit your manuscript to a SpringerOpen journal and benefit from:

► Convenient online submission
► Rigorous peer review
► Open access: articles freely available online
► High visibility within the field
► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com