Journal of Computers Vol. 34 No. 5, October 2023, pp. 265-284
doi: 10.53106/199115992023103405021

Local Path Planning via Improved Fuzzy and Q(λ)-learning Algorithms for the Mobile Robot

Li Song, Da-Zi Li*

College of Information Science and Technology, Beijing University of Chemical Technology,
Beijing, 100029, China

lidz@mail.buct.edu.cn

Received 26 October 2022; Revised 6 March 2023; Accepted 1 May 2023

Abstract. With the complexity of the robot operating environment increases, there becoming higher demands on the optimal path planning for robots. Most of the path planning is performed in known environments and static models. However, there are still challenges for robots to perform path planning in complex unknown or dynamic environments, which will suffer from deadlock problems and obstacle avoidance failures. Reinforcement learning (RL) can help fuzzy algorithm to optimize the strategy. However, the difficulty of designing the rewards in RL makes the algorithm require a large number of samples to learn the strategy, resulting in computational complexity. To solve these problems, a new local path planning based on the improved fuzzy and Q(λ)-learning algorithms is proposed, aiming to plan the shortest path and avoid obstacles. For solving the problems of breaking through and avoiding obstacles, a fuzzy controller is designed. The distance of nearest obstacle in front of the mobile robot and the distance between the obstacles in the two breakout directions are regarded as the two inputs for this controller. And the two fuzzy quantities of the mobile robot's running angle and the safe step length are outputted. In the path planning, the Q(λ)-learning algorithm are used to optimize the weights of the running angle and the safe step, obtaining a more accurate robot position and speeding up path planning efficiency. Furthermore, to solve the overlap problems among the starting point, end point, and obstacles, a safer running environment is designed considering radiuses of these objects. Besides, the mobile robot breakout scheme and sustainable obstacle avoidance scheme are designed to solve the deadlock problem and "large obstacle" avoidance problem, respectively. Simulation results in the sparse and complex operating environment show that our proposed algorithm can plan a relatively optimal and safe path, improving the success rate of path planning.

Keywords: mobile robot, local path planning, improved fuzzy algorithm, Q(λ)-learning, fuzzy controller

1 Introduction

With the continuous development of artificial intelligence technology, the field of mobile robotics has received a lot of attention from researchers at home and abroad. Path planning is one of the essential core elements in robotics research and development, and is the basis for robots to accomplish a given task in a certain environment [1-3]. The path planning capability of the robot intuitively reflects its operational and risk management capabilities, and its main goal is to allow the target object to find a collision-free and safe path from the starting point to the end point within a defined area [4]. Path planning has been widely used in high technology, daily life, logistics management and other fields, such as: manipulator path planning [5, 6], aircraft trajectory planning [7], cruise missile path planning [8], traveling salesman problem [9], path planning based on road networks [10], electronic maps GPS navigation path search and planning [11], and other areas [12, 13]. However, most of the current path planning algorithms are for known environments or static models, which are less robust to environmental changes, errors and noise. And there are few researches on challenging path planning in unknown environment or dynamic model, which limits the application and development of mobile robots to a certain extent.

Compared to known operating environments, complex unknown or dynamic environments place higher demands on path planning. There are many methods of path planning, and their scope of application varies depending on their own advantages and disadvantages. Traditional path planning algorithms include simulated annealing algorithm, artificial potential field method, fuzzy logic algorithm, tabu search algorithm, etc. [14-17]. Fuzzy control algorithm is a classical algorithm of path planning. The fuzzy logic algorithm simulates the driver's driving experience by combining physiological perception and action to obtain planning information through look-up ta-

* Corresponding Author

bles based on the system's real-time sensor information to achieve path planning. The algorithms conform to human thinking habits and facilitate the conversion of expert knowledge into control signals with good consistency, stability and continuity [18, 19]. Fuzzy control is a computer control technique based on natural language control rules and fuzzy logic reasoning. This technique relies on fuzzy rules converted from operational experience and expressive knowledge to achieve intelligent control [20, 21]. Fakoor et al. [22] proposed a decision-making method based on fuzzy Markov decision process for path planning in an unknown environment. Xiang et al. [23] proposed an improved DWA algorithm with an improved dynamic windowing method to achieve local path planning for robots, and added a fuzzy controller to achieve weight coefficient self-adaptation for adapting to more complex environments. Sun et al. [24] proposed an optimized fuzzy control algorithm to plan paths in complex 3D underwater environments, making AUVs automatically avoid dynamic obstacles. The fuzzy algorithm does not require an accurate mathematical model and has good robustness. Xiang et al. [25] proposed an improved DWA algorithm to make the obstacle avoidance path of the robot smoother. In this model, a fuzzy controller was added to achieve the adaption of weight coefficients to adapt to more complex environments. For the path tracking problem of automatic ground vehicles, Hwang et al. [26] proposed a hierarchical improved fuzzy dynamic sliding mode control designed to handle system uncertainties such as different payloads. However, in the environment of unknown dense obstacles and polygonal obstacles, the path planning using fuzzy logic algorithm is prone to deadlock problem and obstacle avoidance failure problem.

In recent years, reinforcement learning (RL) algorithms with decision performance have received wide attention to solve the challenges in path planning. RL algorithms are an important branch of machine learning, the essence of which is to learn strategies through trial and error of interaction between an intelligence and its environment to maximize the reward or achieve a specific goal. Segato et al [27] proposed a safe and effective intraoperative planning framework for flexible neurosurgical robotic lock-hole, which integrates an inverse reinforcement learning path planning algorithm to optimize surgical criteria. Xi et al [28] proposed an AUV path planning scheme using integrated ocean information and reinforcement learning, by carefully designing the state transition function and rewards to construct a 3D grid model. Zhao et al [29] proposed a formation control model for unmanned surface ships based on deep reinforcement learning to formulate a new stochastic braking mechanism, preventing the training of the decision network from falling into a local optimum. Zhang et al [30] proposed a new algorithm based on risk-sensitive learning with stability guarantees to train the strategies for motion planning of self-driving cars. However, the RL algorithm itself has some problems, such as dimension disaster, exploration and utilization difficulties. To solve the above problems, the researchers proposed that input should be fuzzy first, then RL is used for path planning. Liu et al. [31] proposed a multi-controller model combining dueling DQN with fuzzy control, and used fuzzy control to provide a large number of positive samples, improving the generalization ability of the model. Chen et al [32] proposed a path planning method based on conditional deep Q-networks, which applied end-to-end neural networks to autonomous driving to reduce the dependency between different motion commands and improve the stability of path planning. Soares et al [33] developed an autonomous driving path planning method based on deep reinforcement learning model using a set of IF-THEN rules to approximate the deep reinforcement learning model. RL can theoretically be used to optimize strategy, including environments where the world model is unknown. However, the difficulty of designing the reward function in RL makes the algorithm difficult to use for learning specific problems, which makes it necessary to use a large number of samples to learn, thus causing computational complexity.

To address the above problems, a local path planning method based on improved fuzzy algorithm and Q(λ)-learning algorithm is proposed in this paper. Firstly, the kinematic model of the improved fuzzy algorithm is established for the first time in the operating environment of the mobile robot. In this model, a new fuzzy controller is designed, which inputs the distance of the nearest obstacle in front of the mobile robot and the distance between two breakthrough direction obstacles, and outputs the running angle and safety step of the mobile robot. In the unknown dense obstacle environment, we analyze the overlap problem, deadlock problem and "big obstacle" avoidance problem that may occur in the path planning. The corresponding solutions are proposed for these problems, and the simulation is verified in the robot path planning environment. Additionally, the Q(λ)-learning algorithm are used to optimize the weights of the running angle and the safe step in the process of robot path planning, obtaining more accurate robot positions and improving the efficiency of robot path planning. The high-density obstacle environments and rectangular environments are designed to verify the performance of the proposed algorithm.

The main contributions of this paper are summarized as follows:

(1) In the robot operating environment, a new local path planning algorithm is proposed to achieve the goal of shortest path planning and safe obstacle avoidance, which is based on the improved fuzzy and Q(λ)-learning algorithm in the framework of kinematic model.

(2) To solve the overlaps problem between the starting point, the end point, and the obstacles, a safer running environment is designed. To solve the deadlock problem and safety "large obstacle" avoidance problem, the mobile robot breakout scheme and sustainable obstacle avoidance scheme are designed, respectively.

(3) In the path planning, Q(λ)-learning algorithm is used to optimize the weights of the running angle and the safe step, obtaining a more accurate robot position, accelerating the path planning efficiency, and avoiding complex calculations. In addition, the sparse and complex operational environments are designed to validate the proposed local path planning algorithm based on the improved fuzzy and Q(λ)-learning algorithm.

The rest of this paper is organized as follows. Section 2 introduces the basics of traditional fuzzy algorithms, reinforcement learning, and Q-learning that are relevant to this paper. Section 3 introduces the kinematic framework based on improved fuzzy and Q(λ)-learning algorithm for mobile robot path planning, and gives a corresponding scheme for the problems in the path planning process. Section 4 presents the simulation and experimental results in sparse obstacle and dense obstacle environments. Section 5 draws the conclusion and discusses the future works.

2 Preliminaries

In this section, the fuzzy algorithm and reinforcement learning are reviewed to make the theoretical foundation for the following.

2.1 Fuzzy Algorithm

For complex systems that are difficult to describe with existing laws, the intelligent control is realized via qualitative, inaccurate and fuzzy conditional sentences of the fuzzy algorithm. The main idea of fuzzy control is that the machine realizes automatic control by imitating people's operation experience, which is described as fuzzy rule [34-35]. The process of fuzzy control is shown in Fig. 1.

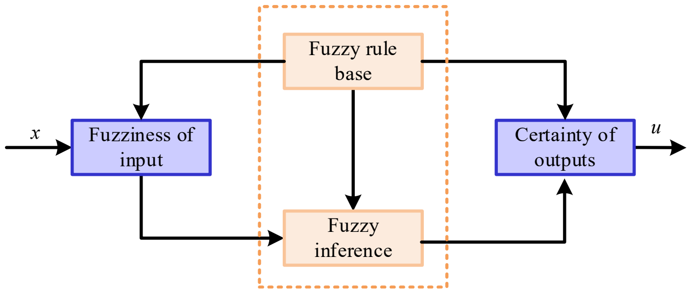


Fig. 1. Principle and structure of fuzzy controller

Given the input variable $x = [x_1, x_2, \dots, x_i, \dots, x_n]^T$, each component x_i is a fuzzy language variable. $T(x_i)$ is set in Eq. (1),

$$T(x_i) = \{A_i^1, A_i^2, \cdots, A_i^{m_i}\}, \quad i = 1, 2, \cdots, n, \tag{1}$$

where A_i^j ($j = 1, 2, ..., m_i$) is the jth language variable value of x_i, which is a fuzzy set defined on the domain x_i, corresponding to membership function $\mu_{A_i}^j(x_i)$($i = 1, 2, ..., n; j = 1, 2, ..., m_i$).

Given the set $T_u = \{B^1, B^2, ..., B^{m_\mu}\}$, where the output u is a fuzzy language variable, and B^j ($j = 1, 2, ..., m_\mu$) is the jth language variable value of u, corresponding to the membership function $\mu_{B^j}(\mu)$.

If the input variable adopts the fuzzy method of single point fuzzy set, the rule fitness of the given input x is given,

$$\alpha_i = \mu_{A_1^i}(x_1)\mu_{A_2^i}(x_2)\cdots\mu_{A_n^i}(x_n). \quad \textbf{(2)}$$

Through fuzzy inference, the membership function of the fuzzy set B_i for the output of each fuzzy rule is calculated,

$$\mu_{B_i}(\mu) = \alpha_1 \mu B^i(\mu), \quad \textbf{(3)}$$

Therefore, the total fuzzy set of quantity is calculated,

$$\mu_B(\mu) = \vee_{i=1}^{m} \mu_{B_i}(\mu) = \vee_{i=1}^{m} \mu_{\bigcup_{i=1}^{m} B_i}(\mu). \quad \textbf{(4)}$$

Weighted average method is adopted to make the certainty of outputs,

$$u = \frac{\int_{U_u} u\mu_B(u)du}{\int_{U_u} \mu_B(u)du}, \quad \textbf{(5)}$$

Where U_u represents the domain of the total fuzzy set $\mu_B(u)$, and $\mu \in U_u$.

2.2 Reinforcement Learning

The problem of RL is described as an intelligence that continuously learns from its interaction with the environment to accomplish a specific goal [36]. RL aims to find an optimal strategy that allows the intelligence to obtain as many rewards from the environment as possible. Discounted rewards at time t are calculated by Eq. (6),

$$R_t = \sum_{\tau=0}^{\mathcal{T}} \gamma^{\tau} r_{t+\tau+1}, \quad \textbf{(6)}$$

where $r_{t+\tau+1}$ indicates the immediate rewards, γ represents the discount factor, $\mathcal{T}$ represents the total time.

The value function and state value function are defined to evaluate the expected return of a strategy, and the state value function $V^{\pi}(s)$ is the expectation of the state-action value function Q^{π}(s, a) with respect to the action a by Markov property.

Definition 1 (Value function). The expected return in state s is $V^{\pi}(\mathrm{s}) = E_{\pi}[R_t \mid s_t = s] = E_{\pi}\left[\sum_{\tau=0}^{\infty} \gamma^{\tau} r_{t+\tau+1} \mid s_t = s\right]$.

Definition 2 (State value function). The expected return after performing the action a in the state s is $Q^{\pi}(\mathrm{s,a}) = E_{\pi}[R_t \mid s_t = s, a_t = a] = E_{\pi}\left[\sum_{\tau=0}^{\infty} \gamma^{\tau} r_{t+\tau+1} \mid s_t = s, a_t = a\right]$.

Q-learning is a time-sequence difference learning algorithm with different strategies, and its Q value is updated as follows [37],

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left(R_{t+1} + \xi \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right), \quad \textbf{(7)}$$

where α $(0 < \alpha \leq 1)$ represents the learning rate, ξ represents the decay factor.

The main idea of Q-learning is to form a Q-table with state and action to store Q values, and then select the actions that can obtain the maximum return according to the Q values.

3 Local Path Planning with the Improved Fuzzy and Q(λ)-learning

Based on the kinematics analysis of the local path planning, the new proposed algorithm uses the fuzzy controller to find the optimal or sub-optimal path of the safe obstacle avoidance. To obtain a more accurate robot position, Q(λ)-learning algorithm is used to optimize the weights of the running angle and the safe step, accelerating the path planning efficiency, and avoiding complex calculations. In the local path planning, both the safe obstacle avoidance of the mobile robot, and the shortest path from the starting point to the end point are considered.

3.1 Kinematics Model based on the Improved Fuzzy and Q(λ)-learning

Fig. 2 shows the kinematics model of the mobile robot. (x_{rob}, y_{rob}) indicates the location of the mobile robot, (x_{obs}, y_{obs}) represents the position of obstacles, (x_{goal}, y_{goal}) indicates the position of goal. And $Goal_1$, $Goal_2$, $Goal_3$ are the positions of the goals that are vertical, normal, and horizontal to the starting point of the mobile robot, respectively.

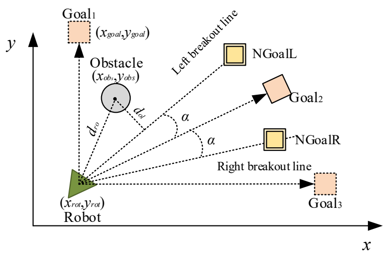


Fig. 2. The kinematics model of the mobile robot

Based on the normal position $Goal_1$, NGoalL and NGoalR are the positions corresponding to the new goals in the left and right breakout directions, respectively. The angles to the left and right breakout directions both are α, respectively. d_{ro} indicates the distance from the current position of mobile robot to the obstacle, and d_{ol} represents the distance from the obstacle to the right detection line.

In Fig. 2, if the goal is in the vertical direction of the mobile robot, the linear equation between the mobile robot and the goal is $x = b_{line}$. If the goal is in the horizontal direction of the mobile robot, the linear equation between the mobile robot and the goal is $y = b_{line}$. Therefore, the distance d_{ol} from the obstacle to the left or right detection lines are calculated, respectively,

$$\begin{cases} d_{ol} = \left| b_{line} - x_{obs} \right| & x = b \\ d_{ol} = \left| b_{line} - y_{obs} \right| & y = b \end{cases} . \tag{8}$$

If the goal is in the normal direction of the mobile robot, the linear equation between the mobile robot and the goal is $y = k_{line}x + b_{line}$. With $\frac{y - y_{rob}}{y_{goal} - x_{rob}} = \frac{x - x_{rob}}{x_{goal} - x_{rob}}$, the equation of the line d_{ro} is calculated by Eq. (9),

$$y = \frac{y_{goal} - y_{rob}}{x_{goal} - x_{rob}} x - \frac{y_{goal} - y_{rob}}{x_{goal} - x_{rob}} x_{rob} + y_{rob} , \tag{9}$$

where k_{line} indicates the slope of the line connecting the current position of mobile robot to the obstacle, b_{line} indicates the intercept of line connecting the current position of mobile robot to the obstacle.

The distance from each obstacle to the line between the current position of the mobile robot and the goal is calculated,

$$d_{ol} = \frac{\left| \frac{y_{goal} - y_{rob}}{x_{goal} - x_{rob}} x_{obs} - y_{obs} - \frac{y_{goal} - y_{rob}}{x_{goal} - x_{rob}} x_{rob} + y_{rob} \right|}{\sqrt{\left(\frac{y_{goal} - y_{rob}}{x_{goal} - x_{rob}} \right)^2 + 1}} . \tag{10}$$

The distance between the current position of the mobile robot and each obstacle is calculated,

$$d_{ro} = \sqrt{\left(x_{rob} - x_{obs}\right)^2 + \left(y_{rob} - y_{obs}\right)^2} . \tag{11}$$

In the direction connecting the current position of the mobile robot and the end point, the distance from the mobile robot to the obstacle is calculated,

$$d_{rol} = \sqrt{d_{ro}^2 + d_{ol}^2} . \tag{12}$$

Because the number of the obstacles is set to N_{obs}, the smallest distance d_{ro} is taken as the distance of the obstacle detected by the mobile robot,

$$d_{ro} = \min\left(d_{rol}\right) . \tag{13}$$

The distance of the nearest obstacle d_{ro} in the detection direction is obtained by Eq. (13). Algorithm 1 is the pseudocode to judge whether the front of the mobile robot is blocked, which can help the mobile robot to complete the objective of safe obstacle avoidance. d_{lro} indicates the distance d_{ro} in the left detection direction, d_{mro} indicates the distance d_{ro} in the middle detection direction, d_{rro} indicates the distance d_{ro} in the right detection direction, and S_p indicates the standard one-step running distance of the mobile robot.

Algorithm 1. Judging whether the front of mobile robot is blocked

```
1: Given $d_{lro}$, $d_{mro}$, $d_{rro}$, $S_p$
2: Compute $Dis = 2 \times S_p$
3: If $Dis > d_{lro}$, $Dis > d_{mro}$, or $Dis > d_{rro}$
4:   Mobile robot meets with obstacles at the front.
5: else
6:   Mobile robot does not meet with obstacles at the front.
7: End
```

Moreover,

$$Df = d_{lro} - d_{rro} , \tag{14}$$

where *Df* indicates the distance between the obstacles in the two breakout directions of the mobile robot.

If the mobile robot is blocked, it will enter the breakout state, and then invokes the fuzzy controller for routine path planning; otherwise, it will call the fuzzy controller for routine path planning directly. With the inputs of the distance d_{ro} and *Df*, the scale of running angles N_a of the next action and the number of safe steps N_s are obtained that can avoid obstacles. The running angle A_r and safe step length of the next step S_r is calculated,

$$\begin{cases} A_r = N_a \times A_p \\ S_r = N_s \times S_p \end{cases} , \tag{15}$$

where A_p indicates the standard angle of each rotation for mobile robot avoiding obstacles, S_p indicates the standard single-step running distance of the mobile robot.

To complete the objective of shortest path planning from the starting point to the end point, in the coordinate system, the angle A_r of the line between the current starting point and the end point is obtained. If the line l_{cre}, connecting the current position of the mobile robot before moving and the end point, is horizontal, $A_r = A_r$ will be obtained. If the line l_{cre} is vertical, $A_r = \frac{\pi}{2} + A_r$ will be obtained. If the line l_{cre} is normal, $A_r = \arctan(k_{line}) + A_r$ will be obtained.

$$\begin{cases} A_r = A_r & Horizontal \\ A_r = \frac{\pi}{2} + A_r & Vertical \\ A_r = \arctan\left(k_{line}\right) + A_r & Normal \end{cases} . \tag{16}$$

The new coordinate points (x_{rn}, y_{rn}) of the mobile robot is calculated,

$$\begin{cases} x_{rn} = x_{rob} + \omega_1 S_r \times \cos\left(\omega_2 A_r\right) \\ y_{rn} = y_{rob} + \omega_1 S_r \times \sin\left(\omega_2 A_r\right) \end{cases}, \tag{17}$$

where ω_1 indicates the weight of the running angle A_r, helping to achieve the objective of planning the shortest path. And ω_2 indicates the weight of the safe step, which helps to achieve the objective of safe obstacle avoidance.

In Eq. (17), to get better weights ω_1 and ω_2, the Q(λ)-learning is utilized to learn the weight parameters. Q(λ)-learning is a kind of Q-learning. The algorithm uses eligibility trace to record the track of each episode, which makes the node memory closer to the end stronger. The mobile robot's memory of the trajectory diminishes with taking a step. Thus, Q(λ)-learning can strengthen the weight near the end point and speed up the learning process.

In the learning process of the weights ω_1 and ω_2, the Q(λ)-learning inputs the distance d_{ro} and the difference Df, and outputs the actions. The TD error $\delta_t = r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$ is utilized to calculate gradient used for parameter updating. To simplify the calculation, Q value corresponding to the maximum probability behavior in q-table is used to estimate $\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$,

$$Q_{t+1}\left(s_t, a_t\right) = Q_t\left(s_t, a_t\right) + \alpha \delta_t e_t, \tag{18}$$

where $e_t = \gamma \lambda e_{t-1} + \frac{\partial Q_t\left(s_t, a_t\right)}{\partial \omega}$. And ω is the updated parameters ω_1 or ω_2, which is updated by Eq. (19):

$$\omega\left(t+1\right) = \omega\left(t\right) + \eta \delta_t \left\{ \gamma \lambda e_{t-1} + \frac{\partial Q_t\left(s_t, a_t\right)}{\partial \omega} \right\}, \tag{19}$$

where η represents the learning rate, γ is the discounted factor, $\lambda (0 \le \lambda \le 1)$ indicates the trace-decay parameter.

$$\eta = 0.1 - 0.09\left(\frac{i}{Max.Episodes}\right), \tag{20}$$

$$\lambda = 0.01 - 0.009\left(\frac{i}{Max.Episodes}\right), \tag{21}$$

where i indicates the current episode and *Max.Episodes* represents the number of episodes. The trace-decay parameters decrease with episodes increase, speeding up the learning process.

Table 1. Initial setting parameters of the algorithm

Parameter	Value
Initial value of Q	0
Eligibility trace *e*	0
Decay factor ξ	0.9
Discounted factor γ	0.95

After experimental analysis, the optimal solutions of ω_1 and ω_2 are set to 0.5 and 0.5, respectively. After the mobile robot completes one step of operation, it enters the next step of path planning.

3.2 Design of Fuzzy Controller based on Q(λ)-learning

The obstacle avoidance control of path planning for the mobile robot based on fuzzy algorithm and Q(λ)-learning algorithm is shown in Fig. 3, which consists of the fuzzy controller and learning of weighting parameters. In Fig. 3, the fuzzy controller is composed of four parts: fuzzification, fuzzy rules, fuzzy reasoning and clarity. And the Q(λ)-learning algorithm is used to optimize the weights ω_1 of the running angle and the weight ω_2 of the safe step. Additionally, with N_a, N_s, ω_1, and ω_2, the new coordinate points of the mobile robot can be obtained. By the interaction between the mobile robot and operating environment, the obstacle distance in front of robot and distance difference can be calculated.

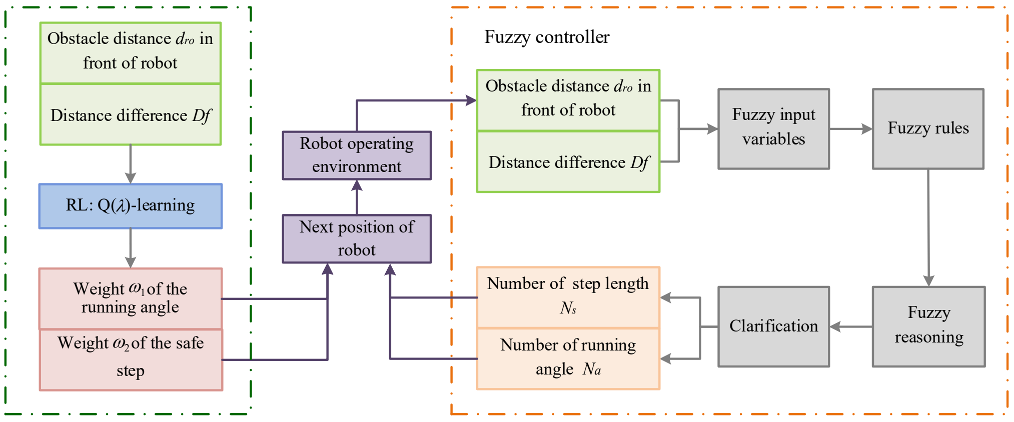

Obstacle distance d_{ro} in front of robot
Distance difference Df
RL: Q(λ)-learning
Weight ω_1 of the running angle
Weight ω_2 of the safe step
Robot operating environment
Next position of robot
Fuzzy controller
Obstacle distance d_{ro} in front of robot
Distance difference Df
Fuzzy input variables
Fuzzy rules
Fuzzy reasoning
Clarification
Number of step length N_s
Number of running angle N_a

Fig. 3. Obstacle avoidance control of the local path planning for the mobile robot

Given the current position of the mobile robot and the position of the obstacle, the distance d_{ro} of the nearest obstacle in front of the mobile robot, and the difference Df of the obstacle distances in the left and right breakout directions are obtained. Taking d_{ro} and Df as the inputs of the fuzzy controller, after fuzzy inference, the scale of the running angle N_a for the mobile robot and the number of safe step length N_s are obtained, respectively. Furthermore, with N_a and N_s, the angle and distance of the mobile robot's next operation are obtained after the clarification process.

1) Fuzzy set, fuzzy domain, membership function

Inputs of the fuzzy controller are the distance d_{ro} of the nearest obstacle in front of the mobile robot and the distance difference Df between the obstacles in the left and right breakout directions. The outputs are the scale of the running angle N_a for the mobile robot and the number of safe step length N_s.

The maximum detection distance of the mobile robot is set to 5 times as long as the standard one-step length S_p. The domain range of the distance d_{ro} is set to [0 5], and its fuzzy subset is set to {VN, N, M, F, VF} (VN: Very Near, N: Near, M: Medium, F: Far, VF: Very Far). The domain range of the difference Df is set to [-5 5], and its fuzzy subset is {NB, NS, ZO, PS, PB} (NB: Negative Big, NS: Negative Small, ZO: Zero, PS: Positive Small, PB: Positive Big). To obtain better results, Gaussian function is selected as the membership function, and the width and center of the membership function of each fuzzy subset are shown in Table 2.

Table 2. The width and center of membership function

	d_{ro}		Df
VN	[1.100 -0.30]	NB	[1.679 -5.31]
N	[0.425 1.49]	NS	[0.944 -2.22]
M	[0.425 2.49]	ZO	[0.943 -0.00]
F	[0.425 3.57]	PS	[0.944 2.22]
VF	[1.295 4.73]	PB	[1.964 5.43]

The domain range of N_a is set to [0 2], and its fuzzy subset is {VL, L, ME, M, VM} (VL: Very Less, L: Less, ME: Medium, M: More, VM: Very More). The number of safe step length N_s has a domain ranged between [-2 2], and its fuzzy subset is set to {NL, NS, ZO, PS, PL} (NL: Negative Large, NS: Negative Small, ZO: Zero, PS: Positive Small, PL: Positive Large). To obtain better results, Gaussian function is selected as the membership function of the scale factor, and the width and center of the membership function of each fuzzy subset are shown in Table 3.

Table 3. Width and center of the membership function

	N_a		Df
VL	[0.237 0.12]	NL	[0.639 -1.95]
L	[0.170 0.60]	NS	[0.340 -0.80]
ME	[0.170 1.00]	ZO	[0.340 0.00]
M	[0.170 1.40]	PS	[0.340 0.80]
VM	[0.406 1.90]	PL	[0.486 1.78]

2) Fuzzy rules and fuzzy reasoning

Fuzzy control rule is the core of fuzzy controller, and also is the main content of designing control system. There are two methods for obtaining fuzzy control rules: one is to obtain from experts' actual operating experience and knowledge of the control system, the other is to summarize from the input-output data of the test system. Our improved fuzzy algorithm uses the latter to generate fuzzy control rules.

The path planning rules of the mobile robot based on the improved fuzzy algorithm are as follows.

Firstly, the principle of the step length of mobile robot for avoiding obstacles is designed. When the fuzzy subset of the nearest obstacle distance in front of the mobile robot is {VN, N, M, F, VF}, the corresponding outputs of the scale of mobile robot running angle are {VL, L, ME, M, VM}.

And then the corner principle of avoiding obstacles for the mobile robot is designed. The line between the starting point of the mobile robot to the end point is regarded as the navigation line. When the distance of obstacles detected by the mobile robot on the right detection line is larger than that detected on the left detection line, the mobile robot will move to the right next. When the distance of obstacles detected by the mobile robot on the left detection line is larger than that detected on the right detection line, the mobile robot will move to the left next. Otherwise, the mobile robot will move in the direction of the navigation line.

The control rules of the path planning based on the improved fuzzy with Q(λ)-learning algorithms are summarized using the input-output data, as shown in Table 4.

Table 4. Fuzzy control rules

d_{ro}	Df				
	NB	NS	ZO	PS	PB
VN	[NB VN]	[PB VN]	[ZO VN]	[PB VN]	[PB VN]
N	[NB N]	[NS N]	[NB M]	[NB F]	[NB VF]
M	[NS M]	[NS M]	[ZO M]	[PS M]	[PS M]
F	[NS F]	[NS F]	[ZO F]	[PS F]	[PS F]
VF	[NS VF]	[NS VF]	[ZO VF]	[PS VF]	[PS VF]

3) Clarification

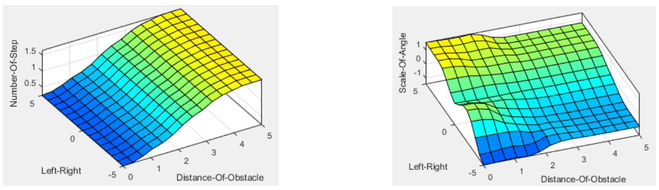
After clarification processing, the fuzzy quantity obtained by the approximate reasoning becomes the clear quantity, which can drive the subsequent actuators. The area barycentric method in Eq. (22) is used to find the center of the area surrounded by the fuzzy set membership function curve and the abscissa. With Eq. (5), the abscissa of this center is used as the representative value of the fuzzy set to obtain the amount of clarity,

$$u_{cen} = \frac{\sum_{j=1}^{n} u_j u_B\left(u_j\right)}{\sum_{j=1}^{n} u_B\left(u_j\right)}, \tag{22}$$

where $u_B(u_j)$ is the membership function at u_j. For the scale of the running angle N_a, the domain is set to [0 2], thus $u_j \in$ [0 2] and n will be set to 5. And for the number of safe step length N_s, the domain is set to [-2 2], thus $u_j \in$ [0 2], and n will be set to 5.

4) Effectiveness of fuzzy controller

In the designed fuzzy controller, the curved observation window of the output is obtained, as shown in Fig. 4.



(a) Number of step (b) Scale of angle

Fig. 4. The surface observation window of the outputs

In Fig. 4(a), x-coordinate represents the distance of the nearest obstacle in front of the mobile robot, y-coordinate represents the difference of the obstacle distances in the left and right breakout directions, z-coordinate denotes the number of safe step length. In Fig. 4(b), z-coordinate represents the scale of the running angle for the mobile robot. Fig. 4 shows the surface observation window of the outputs is a plane in the three dimensions, indicating the output is a nonlinear function of the input. Because the smoother the surface, the better the system performance, our proposed fuzzy algorithm has good performance for path planning.

3.3 Path Planning Process based on Improved Fuzzy and Q(λ)-learning

Based on MATLAB 2016a simulation platform, the control value of mobile robot operation is obtained with the input and output variables of fuzzy controller.

The steps of local path planning for the mobile robot based on the improved fuzzy and Q(λ)-learning algorithm are as follows.

Step 1: A coordinate system Σs is established.

Step 2: In the coordinate system Σs, the starting point, the size of the environment, and the number and location of obstacles are randomly set.

Step 3: The parameters of the improved fuzzy algorithm are initialized: l_{sg} indicates the line between the current starting point to the end point, and M_s is the maximum number of steps for the mobile robot from the starting point to the end point. (k, b, f) are the slope, intercept, and position relationship of the line l_{sg} (f = 0 means the direction of the line l_{sg} is normal, f = 1 means it is vertical, and f = 2 means it is horizontal). And d_{sg} is the distance from the current starting point to the end point, S_p is the standard one-step running distance of mobile robot, A_r is the running angle at the current position, S_r is the running step length at the current position; (x_{rn}, y_{rn}) is the current starting point.

Step 4: The mobile robot starts from the current starting point. And the (k, b, f) of the line l_{sg} can be calculated.

Step 5: Judging whether the mobile robot reaches the goal, if S_p is greater than d_{sg}, the mobile robot reaches the goal, and the path planning is completed; otherwise, the mobile robot will take the next step.

Step 6: Calculate the (k, b, f) of the detection line on the left and right sides of the measurement line l_{sg}, and the distances *disl*, *dism*, *disr* of the nearest obstacle in the left, middle, and right detection directions for the mobile robot.

Step 7: Calculating whether the front of the robot is blocked with Algorithm 1, if yes, $flag_{block}$ = 1, the mobile robot will enter step8; otherwise, $flag_{block}$ = 0, the mobile robot will enter step9.

Step 8: When the mobile robot enters the breakout state, it will constantly explore the left and right sides, looking for a feasible exit. If a feasible exit is not found, the path planning fails this time; otherwise, it goes to the next step.

Step 9: Taking the distance of the obstacle that is closest to the mobile robot dis = min ($disl$, $dism$, $disr$), the fuzzy controller is called, which can calculate and output the scale of running angle and the number of the safe step (N_a, N_s) with the inputs (dis, $disl - disr$).

Step 10: The Q(λ)-learning algorithm is used to optimize the weights ω_1 of the running angle and the weight ω_2 of the safe step.

Step 11: Calculating the coordinates (x_{rn}, y_{rn}) of the new starting point of this path planning with Eq. (17), the mobile robot will reach a new starting point.

Step 12: Judging whether the number of the current steps is greater than the maximum number of steps M_s, if yes, the mobile robot will end this path planning; otherwise, it returns to step 4 for the next path planning.

4 Analysis of Problems in the Path Planning

In a sparse obstacle environment, the mobile robot can use the improved fuzzy and Q(λ)-learning algorithms to plan path. However, in the complex running environments, the mobile robot will encounter a series of problems. To improve the efficiency of the mobile robot for path planning, specific problems are analyzed and corresponding specific solutions are proposed.

4.1 Overlap Problems between the Starting Point, End Point, and Obstacles

The obstacles are randomly set in the simulation environment of the mobile robot in Fig. 5. Because the radius $r_{obstacle}$ of the obstacles, the radius r_{goal} of the end point, and the radius r_{robot} of the mobile robot are ignored, the starting point or end point may overlap with obstacles, and the starting point may overlap the end point. Therefore, considering the radiuses of these objects in the environment, the pseudo code of the improved designed environment is designed as shown in Algorithm 2, which can calculate the coordinate (X_{obs}, Y_{obs}) and the number of the obstacles N_{obs}.

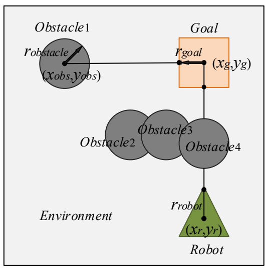


Fig. 5. The running environment of the mobile robot

Algorithm 2. Improved running environment

```
Input: Starting point $(x_r, y_r)$, end point $(x_g, y_g)$, obstacles $(x_{obs}, y_{obs})$, number of obstacle
    $N_{obs}$, maximum detect distance $D_m$ of the mobile robot, standard one-step running
    distance $S_p$, and $[S_p]$ which indicates the largest integer no more than $S_p$.
Output: New obstacles $(x_{obs}, y_{obs})$ and number of obstacle $N_{obs}$.
1: Design $(x_r, y_r)$, $(x_g, y_g)$, and $N_{obs}$
2: Calculate $D_m = 5([(x_g, x_r)^2+(y_g, y_r)^2] +1)$
3: In the intervals $(D_m, x_g - D_m)$ an $(D_m, y_g - D_m)$, $N_{obs}$-dimensional coordinate matrix $(X_{obs}, Y_{obs})$ of the
   obstacles is randomly generated, respectively, which can prevent the end point from overlapping with
   the obstacles or the starting point;
```

```
4: Repeat
5:   $d_i = [(X_{obs,i} - x_r)^2 + (Y_{obs,i} - y_r)^2]$
6:   if $d_i < 3$
7:     Omitting obstacles points $(X_{obs,i}, Y_{obs,i})$ from the matrix $(X_{obs}, Y_{obs})$
8:     $N_{obs} = N_{obs} - 1$
9:   else
10:    $N_{obs} = N_{obs}$
11:  end
12: Until $i = N_{obs}$
13: Preventing the starting point from overlapping with the obstacles.
```

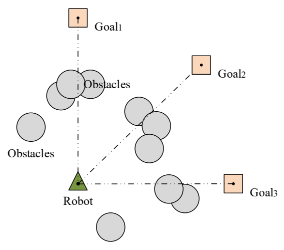
The radiuses of the obstacles, end point, and mobile robot are all set to 1 unit. To prevent the end point from overlapping the obstacles, the maximum detect distance D_m of mobile robot needs to meet Eq. (23),

$$D_m > r_{obstacle} + r_{goal}\,, \tag{23}$$

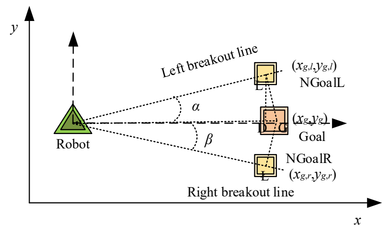
where $r_{obstacle}$ indicates the radius of obstacle, r_{goal} indicates the radius of goal. $D_m = r_{obstacle} + r_{goal}$ means that the obstacle is exactly tangent to the goal. Therefore, in Algorithm 2, if $D_m \leq 2$, D_m will be set to 3; otherwise, $D_m = D_m$.

4.2 Deadlock Problem of Obstacle Avoidance for the Mobile Robot

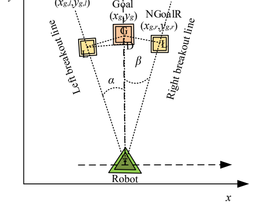
In the complicated environment with dense obstacles, many obstacles may form the "semi-circular obstacle". In this case, the mobile robot will trap in a deadlock problem of left-right circular motion, and the mobile robot is difficult to find the optimal path for safe obstacle avoidance. Therefore, a breakout scheme is designed to guide the mobile robot to plan the optimal path. According to the positions of the mobile robot and the end point, the positions can be divided into three situations: horizontal, vertical, and normal, as shown in Fig. 6(a). And the three cases are analyzed as shown in Fig. 6(b), Fig. 6(c), and Fig. 6(d), respectively. Therefore, a mobile robot breakout scheme is designed.



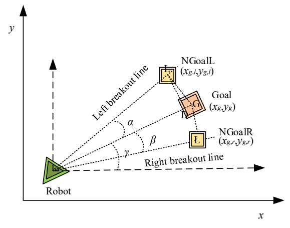
(a) Shows the positions of the mobile robot and the end point



(b) Shows that the goal is in the horizontal direction of the mobile robot



(c) Shows that the goal is in the vertical direction of the mobile robot



(d) Shows that the goal is in the normal direction of the mobile robot

Fig. 6. Analysis of the deadlock problem

In Fig. 6, the triangle represents the mobile robot and the rectangle represents the goal. When the mobile robot detects an obstacle ahead, it takes a left and right breakout along the direction of the line between the mobile robot and the end point. The angle of left breakout is *α*, and the angle of right breakout is $\beta = -\alpha$. In the direction of the left side of Fig. 6, the line *LG* and line *RL* (*RL* represents the line between the mobile robot and point *L*) are perpendicular to each other with the intersection point *L*. And at the point *G*, the line *LD* is perpendicular to the line *RG* (*RG* represents the line between the mobile robot and point *G*).

In Fig. 6(b), the position of the new goal ($x_{g,l}$, $y_{g,l}$) in the direction of left breakout is calculated,

$$\begin{cases} x_{g,l} = x\cos\alpha\cos\alpha \\ \square_{g,l} = \quad + \quad \cos\alpha\sin\alpha \end{cases}. \tag{24}$$

In Fig. 6(c), the position of the new goal ($x_{g,l}$, $y_{g,l}$) in the direction of left breakout is calculated,

$$\begin{cases} x_{g,l} = x - y\cos\alpha\sin\alpha \\ y_{g,l} = y\cos\alpha\cos\alpha \end{cases}. \tag{25}$$

In Fig. 6(d), the position of the new goal ($x_{g,l}$, $y_{g,l}$) in the direction of left breakout is calculated,

$$\begin{cases} x_{g,l} = \left(\dfrac{x}{\cos\gamma}\right)\cos\alpha\cos(\alpha+\gamma) = x\cos^2\alpha - y\cos\alpha\sin\alpha \\ y_{g,l} = \left(\dfrac{x}{\cos\gamma}\right)\cos\alpha\sin(\alpha+\gamma) = x\cos\alpha\sin\alpha + y\cos^2\alpha \end{cases}, \tag{26}$$

where *γ* indicates the angle the line *RG*.

Similarly, substituting *β* into the above formulas, the position of the new goal in the right breakout direction can be obtained (see Appendix).

4.3 Failure Problem of Breaking through Obstacles in Path Planning

In a complex environment with a lot of obstacles, many obstacles may partially overlap to form a "large obstacle". Because the "large obstacle" has long sides, which will increase the complexity of planning path, as shown in Fig. 7. When the mobile robot repeatedly falls into the complex environment of "large obstacle", the mobile robot will eventually fail to break through the obstacles, hitting the obstacles.

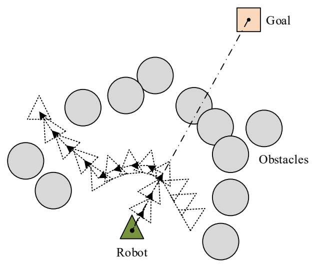


Fig. 7. The obstacle avoidance of "large obstacle" for the mobile robot

To overcome the failure problem of avoiding "large obstacles" for the mobile robot, a sustainable obstacle avoidance scheme is designed to prevent the mobile robot from failing to avoid obstacles when repeatedly breaking in the complex environment of "large obstacles". The flow chart of sustainable obstacle avoidance scheme is shown in Fig. 8.

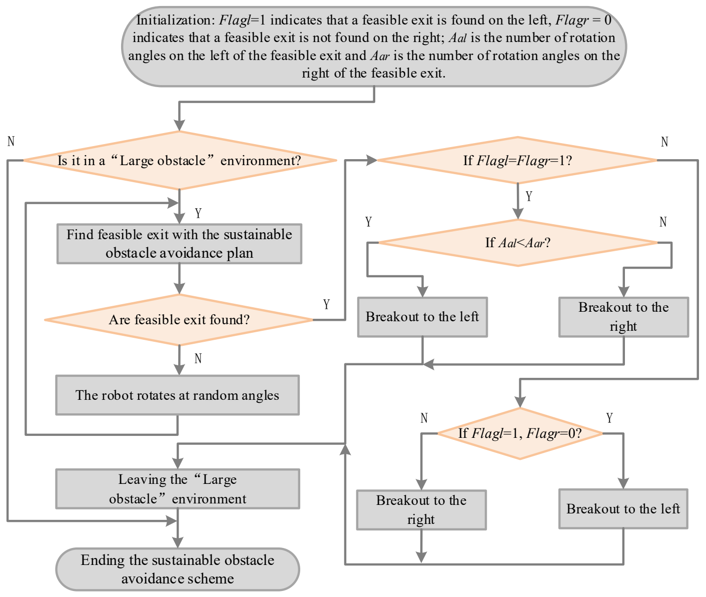

Initialization: Flagl=1 indicates that a feasible exit is found on the left, Flagr = 0 indicates that a feasible exit is not found on the right; Aal is the number of rotation angles on the left of the feasible exit and Aar is the number of rotation angles on the right of the feasible exit.
Is it in a "Large obstacle" environment?
N
Y
Find feasible exit with the sustainable obstacle avoidance plan
Are feasible exit found?
Y
N
The robot rotates at random angles
Leaving the "Large obstacle" environment
Ending the sustainable obstacle avoidance scheme
If Flagl=Flagr=1?
N
Y
If Aal<Aar?
Y
N
Breakout to the left
Breakout to the right
If Flagl=1, Flagr=0?
N
Y
Breakout to the right
Breakout to the left

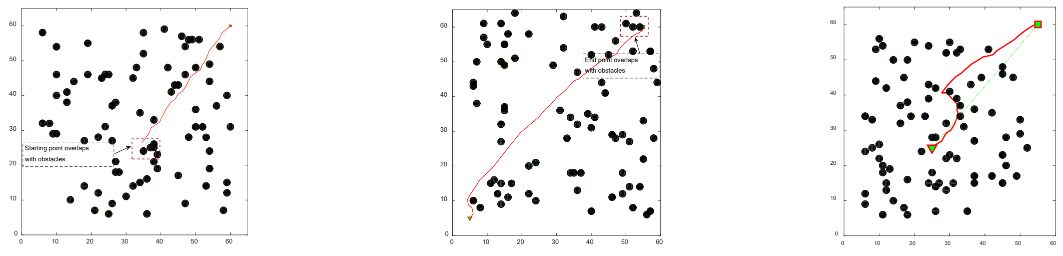
Fig. 8. Design scheme of sustainable obstacle avoidance

5 Simulation and Experiments

In the process of local path planning, the mobile robot may encounter some problems, such as the overlap problems between the starting point, end point and obstacles, deadlock problem, and obstacle avoidance failure problem. To address the existing problems, this section performs experimental simulations based on the corresponding solutions proposed in Section 4. Then different obstacles, starting point and end point for the mobile robot are randomly set for the simulation of path planning. Additionally, to better illustrate the performance of the improved fuzzy and Q(λ)-learning algorithm, it is compared with the fuzzy algorithm, the improved fuzzy algorithm, Q-learning algorithm, Q(λ)-learning algorithm.

5.1 Simulation of the Existing Problems

Because the radiuses of the starting point, end point and obstacles are ignored in setting the running environment for the mobile robot, the starting point may overlap with the obstacles or the end point, and the end point may overlap with the obstacles. These cases may make it difficult for mobile robots to find their starting point and end point with safe obstacle avoidance. In Fig. 9(a), because the starting point overlaps with the obstacles, the mobile robot starts near the starting point for path planning. In Fig. 9(b), the goal overlaps with the obstacles; thus, the mobile robot regards the obstacle that overlaps with the end point as the new end point.

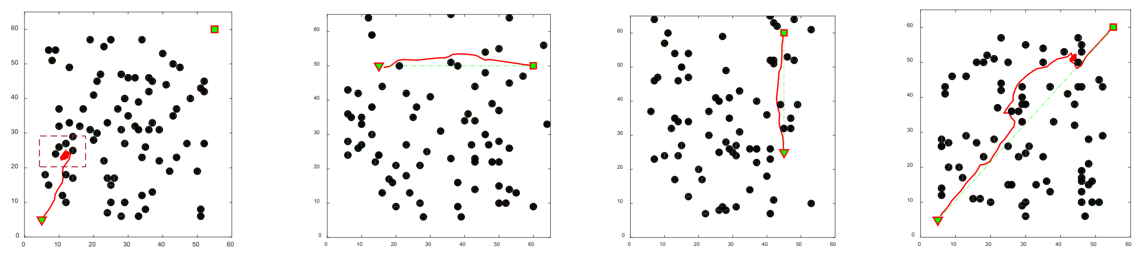


(a) Shows starting point overlaps with obstacles (b) Shows end point overlaps with obstacles (c) Shows normal path planning

Fig. 9. Overlap problems that between the starting point, end point, and obstacles

To prevent the overlap problems, the solution to this problem in subsection 4.1 considers the radiuses of these objects in the environment, for solving the overlap problems between the starting point, end point, and obstacles. Therefore, a safer simulation environment is designed than before, as shown in Fig. 9(c). In Fig. 9(c), the starting point and end point do not overlap with the obstacles. Therefore, the influences of overlap problems on path planning are avoided, increasing the success rate of mobile robot path planning.

In a complicated environment with the "semi-circular obstacle", the mobile robot may encounter the deadlock problem that the robot runs back and forth, as shown in Fig. 10(a). It is difficult for the robot to find the optimal path with safe obstacle avoidance. Thus, the breakout scheme proposed in subsection 4.2 is used to solve the deadlock problem. And the simulations of the three cases are carried out to illustrate the performance of our proposed scheme for planning path, respectively, as shown in Fig. 10(b), Fig. 10(c), and Fig. 10(d).



(a) Shows the deadlock problem

(b) Shows that the connecting line between the mobile robot and the end point is horizontal

(c) Shows that the connecting line between the mobile robot and the end point is vertical

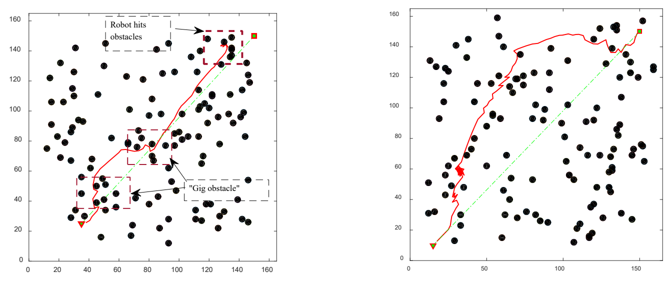
(d) Shows that the connecting line between the mobile robot and the end point is normal

Fig. 10. Mobile robot's deadlock problems

In Fig. 10(a), the mobile robot repeatedly runs in a "semicircle obstacle" composed of multiple obstacles, and is stuck in a deadlock problem. To solve this problem, a mobile robot breakout scheme is designed, which can help the mobile robot plan the optimal path in a complex environment. Fig. 10(b) to Fig. 10(d) show the path planning when the connecting lines between the mobile robot and the end point are horizontal, vertical, and normal, respectively. In Fig. 10(d), the mobile robot finds a feasible planned path after several runs in the "semi-circular obstacle" with the design scheme. Simulation results in Fig. 10(b) to Fig. 10(d) show that the mobile robot can plan a feasible optimal or sub-optimal path.

Because the obstacles are many and densely distributed, multiple obstacles will form "big obstacle", which may cause the mobile robot to fail to find a path to escape, and hit the obstacles directly. To solve the failure problem of obstacle avoidance for the mobile robot, a sustainable obstacle avoidance scheme for obstacle avoidance is proposed. The simulation results are shown in Fig. 11.

In Fig. 11(a), the mobile robot fails to avoid obstacles in a complex environment with "big obstacle". To overcome the failure problem of obstacle avoidance for the mobile robot, the feasible path from the starting point to the end point can be planned, as shown in Fig. 11(b). It can be seen from Fig. 11 that the mobile robot can find the path to break through the obstacles through repeated test runs in the "big obstacle" area. And considering the radiuses of the starting point, end point and obstacles, the end point doesn't overlap the obstacles. Therefore, the mobile robot can plan the optimal path.

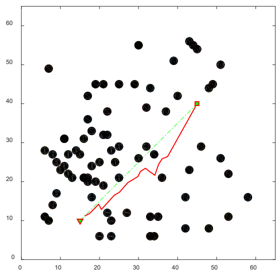


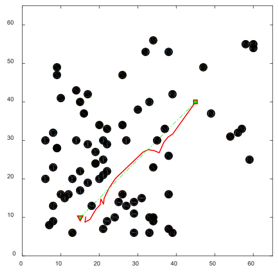
(a) Shows the failed path planning (b) Shows the successful path planning

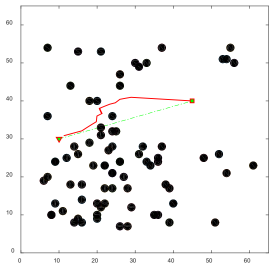
Fig. 11. Mobile robot obstacle avoidance for "big obstacle"

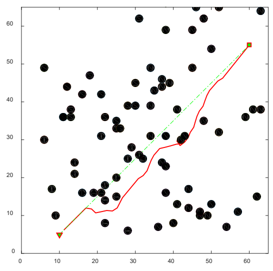
5.2 Simulation of Path Planning for Mobile Robot

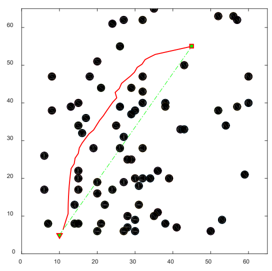
After verifying the feasibility of the solutions to the problems in subsection 4.1, the different experimental environments are randomly set up to perform simulation validation of the proposed algorithm for path planning. Some simulation results are shown in Fig. 12. In Fig. 12(a) to Fig. 12(c), relatively sparse 80 obstacles are set and the locations of the obstacles are set randomly, which increases the learning difficulty of the environment. Compared to Fig. 12(a) and Fig. 12(b), the starting and end points of the robot are changed in Fig. 12(c).

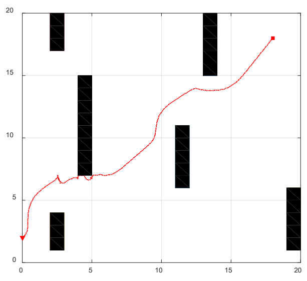

(a) Shows the sparse obstacle environment 1

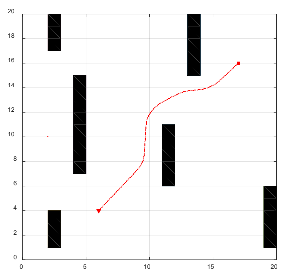

(b) Shows the sparse obstacle environment 2

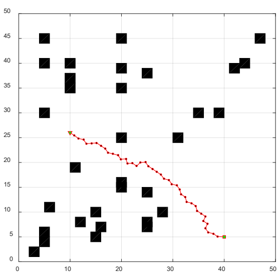

(c) Shows the sparse obstacle environment 3


(d) Shows the high-density obstacle environment 1


(e) Shows the high-density obstacle environment 2


(f) Shows the environment 1 with multiple rectangular obstacles


(g) Shows the environment 2 with multiple rectangular obstacles


(h) Shows the environment 1 with dense rectangular obstacles

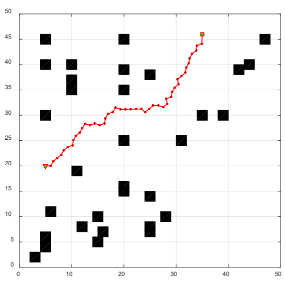

(i) Shows the environment 2 with dense rectangular obstacles

Fig. 12. Path planning of the mobile robot

It can see from Fig. 12(a) to Fig. 12(c) that the robot can plan a better path to achieve safe obstacle avoidance in the relatively sparse obstacle environment where the starting point and end point are randomly set. In Fig. 12(d) and Fig. 12(e), 120 obstacles are set, and their positions are set randomly. In this environment, the robot uses a break scheme to solve the "semi-circular obstacle" problem, which can still avoid the high-density obstacle environment and plan a better path. In Fig. 12(f) and Fig. 12(g), environment with multiple rectangular obstacles are set up. As can be seen from the Fig. 12(f) and Fig. 12(g), even if the starting point and the end point of the robot change, the robot can still safely avoid obstacles and plan the optimal or sub-optimal path from the starting point to the ending point. In addition, to increase the difficulty of the experiment, an environment with dense rectangular obstacles are set up in Fig. 12(h) and Fig. 12(i). After 200 episodes of learning, the robot's path planning also works well.

Simulation results in Fig. 12 show that the mobile robot can plan the safe optimal or suboptimal path from starting point to end point in different and complex obstacle environments.

To further illustrate the performance of our proposed algorithm, 5, 10, 15, 20, 25, …, 125 obstacles are set in the environment for planning the path, respectively. The number and ratio of successful mobile robot path planning are obtained in Fig. 13 over 50 tests. In the robot operating environments, the same starting point and end point of the robot are first set. Then, after setting the number and location of obstacles, the proposed algorithm is compared with the fuzzy algorithm, the improved fuzzy algorithm, Q-learning algorithm, and Q(λ)-learning algorithm. In the environment with set obstacles, the robot is defined as successful once if it can find a reasonable route from the starting point to the end point without colliding with the obstacles; otherwise, it is a failure. The experiment is then run 50 times and the ratio of the number of successful experiments to the total run time is defined as the success rate.

$$\text{Successful rates}=\frac{Num_s}{Num_e}, \tag{27}$$

where Num_s indicates the number of success, Num_e represents the number of the running.

Higher success rate indicates better performance of the algorithm. It can be seen from Fig. 13 that the performance of the fuzzy algorithm is the worst, mainly because the algorithm lacks the necessary obstacle avoidance rules. In a complex obstacle environment, the mobile robot gets caught in a cluster of obstacles and then loses its obstacle avoidance function, crossing the obstacles or directly hitting them. In the improved fuzzy with Q(λ)-learning algorithm, the radiuses of these objects, a break scheme for "semi-circular obstacle", and a sustainable obstacle avoidance scheme are considered. Thus, the robot can basically avoid obstacles safely and find the obstacles from the starting point to the end point, and the success rate has improved significantly. Furthermore, at a certain position, when the robot computes the next position after executing the action in Eq. (15), the Q(λ)-learning are used to compute the weights of the running angle and safe step, improving the robot's path planning efficiency. Therefore, the improved fuzzy with Q(λ)-learning algorithm has the highest success rate and the best efficiency of planning path. Furthermore, because Q-learning algorithm and Q(λ)-learning algorithm suffer from the problem of dimensional catastrophe and do not capture the task structure well, their performance of the path planning is slightly worse than the improved fuzzy with Q(λ)-learning algorithm.

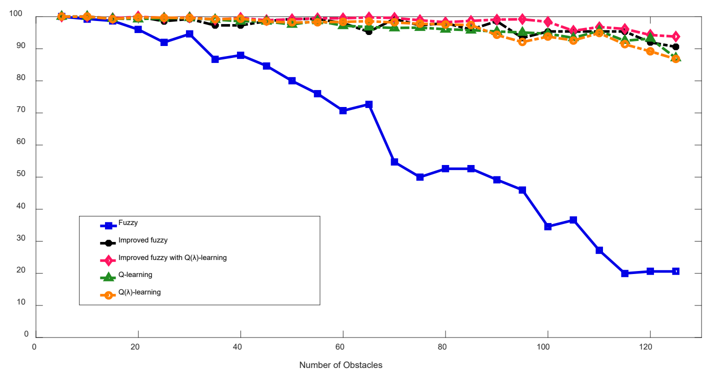


Fig. 13. Successful rates of the path planning

Additionally, 50 obstacles are set up and the average number of steps from the starting point to the end point obtained using the improved fuzzy with Q(λ)-learning algorithm after the experiments are repeated 20 times independently is shown in Table 5. It can see from Table 5 that the fuzzy algorithm uses the simplest fuzzy ideas and does not consider the overlap problems, deadlock problem, failure problem of obstacle avoidance, etc. Therefore, the mobile robot may fail in obstacle avoidance or fall into a dead loop inside the "semi-circular obstacle", which makes the robot's planning path steps significantly larger. The improved fuzzy algorithm solves the existing problems in the fuzzy algorithm so that the number of path steps from the starting point to the end point is 115. Additionally, Q-learning algorithm and Q(λ)-learning algorithm can help the mobile robot to find the better path from the starting point to the end point, but the number of steps is 128 and 146 with the constraint of the algorithm itself. The improved fuzzy with Q(λ)-learning algorithm solves the drawbacks of the fuzzy algorithm and combines the advantages of the Q(λ)-learning algorithm, so that the optimal or suboptimal path from the starting point to the end point can be planned and the minimum number of steps for path planning can be obtained.

Table 5. The average number of steps for these algorithms

Algorithms	Number of steps
Fuzzy algorithm	2342
Improved Fuzzy algorithm	115
Improved Fuzzy with Q(λ)-learning algorithm	82
Q-learning algorithm	128
Q(λ)-learning algorithm	146

In the above experiments, the path planning for the mobile robot is simulated and verified in the environment, and our proposed improved fuzzy with Q(λ)-learning algorithm is compared with the fuzzy algorithm, the improved fuzzy algorithm, Q-learning algorithm, and Q(λ)-learning algorithm. It is experimentally verified that the improved fuzzy with Q(λ)-learning algorithm can obtain optimal or suboptimal paths with better planning results.

6 Conclusions

To plan optimal path with safe obstacle avoidance, a new local path planning is proposed. Based on the construction of the kinematics model with the improved fuzzy algorithm, a fuzzy controller based on Q(λ)-learning is designed. Through fuzzification, fuzzy inference, and clarification, the control amount is obtained to control the operation of the mobile robot. For overcoming the overlap problems, the deadlock problem and the failure problem of breaking through "large obstacle", a safer operating environment, a mobile robot breakout scheme and a sustainable obstacle avoidance scheme are designed, respectively. In the robot path planning process, the Q(λ)-learning algorithm is used to optimize the weights of the running angle and safe step to improve the efficiency of the robot path planning. Then, the number and location of obstacles, the location of the starting point and the end point are randomly set, and the improved fuzzy with Q(λ)-learning algorithm is simulated and verified in simple and complex environments. Compared with the fuzzy algorithm, the improved fuzzy algorithm, Q-learning algorithm, Q(λ)-learning algorithm, simulation results show that the success rate of the improved fuzzy with Q(λ)-learning algorithm is the best and the number of steps for planning path is the least in the same planning environment. The improved fuzzy with Q(λ)-learning algorithm not only overcomes the problems in fuzzy algorithm, but also combines the better decision-making performance of Q(λ)-learning algorithm, thus, its performance of path planning is better than the existing methods.

Our proposed algorithm in this paper compensates for some shortcomings of existing path planning algorithms and improves the theoretical study of path planning to a certain extent. Furthermore, the improved fuzzy with Q(λ)-learning algorithm is tried to applied to our unmanned vehicle tracing. Our later work is to apply the proposed algorithm to the autonomous collision-free action of unmanned vehicles in a real environment, realizing the practical application of the algorithm.

Acknowledgement

The authors would like to thank anonymous reviewers for their valuable comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62273026.

References

[1] U. Orozco-Rosas, O. Montiel, R. Sepúlveda, Mobile robot path planning using membrane evolutionary artificial potential field, Applied Soft Computing 77(2019) 236-251.
[2] Y.L. Li, Y. Zuo, Q.H. Shan, T.S. Li, Path planning of ship collision avoidance for minimized energy consumption, in: Proc. 33rd Chinese Control and Decision Conference, 2021.
[3] J. Zhang, J.-W. Li, H.-W. Yang, X. Feng, G. Sun, Complex environment path planning for unmanned aerial vehicles, Sensors-Basel 21(15)(2021) 1-29.
[4] T.M. Cabreira, P.R. Ferreira, C.D. Franco, G.C. Buttazzo, Grid-based coverage path planning with minimum energy over irregular-shaped areas with uavs, in: Proc. 2019 International Conference on Unmanned Aircraft Systems, 2019.
[5] D. Guo, F. Xu, L. Yan, New pseudoinverse-based path-planning scheme with PID characteristic for redundant robot manipulators in the presence of noise, IEEE Transactions on Control Systems Technology 26(6)(2018) 2008-2019.
[6] W. Liu, H. Niu, M.-N. Mahyuddin, G. Herrmann, J. Carrasco, A model-free deep reinforcement learning approach for robotic manipulators path planning, in: Proc. 2021 21st International Conference on Control, Automation and Systems, 2021.
[7] E. Andrés, D. González-Arribas, M. Soler, M. Kamgarpour, M. Sanjurjo-Rivo, Informed scenario-based RRT* for aircraft trajectory planning under ensemble forecasting of thunderstorms, Transportation Research Part C: Emerging Technologies 129(2021) 1-21.
[8] X. Liu, Q. Gu, C. Yang, Path planning of multi-cruise missile based on particle swarm optimization, in: Proc. 2019 International Conference on Sensing, Diagnostics, Prognostics, and Control, 2019.
[9] P.T. Kyaw, A. Paing, T.T. Thu, R.E. Mohan, A.V. Le, P. Veerajagadheswar, Coverage path planning for decomposition reconfigurable grid-maps using deep reinforcement learning based travelling salesman problem, IEEE Access, 8(2020) 225945-225956.
[10] Z. Cai, X.R. Cui, X. Su, Q. Mi, L. M. Guo, Z.M. Ding, A novel vector-based dynamic path planning method in urban road network, IEEE Access 8(2020) 9046-9060.
[11] M. Lee, Y.K. Lee, R.G. Rittenhouse, A genetic algorithm for PDA optimal path generation using GPS, Applied Soft Computing 12(8)(2012) 2379-2386.
[12] C.T. Brenda, R.F. Sari, A review of firefly algorithms for path planning, vehicle routing and traveling salesman problems, in: Proc. 2nd International Conference on Electrical Engineering and Informatics, 2018.
[13] T.H. Hsieh, S.Z. Wang, H.J. Gong, W. Liu, N. Xu, Sea ice warning visualization and path planning for ice navigation based on radar image recognition, Journal of Marine Science and Technology 29(2021) 277-286.
[14] W.J. Shi, Z. He, W. Tang, W.F. Liu, Z. Ma, Path planning of multi-robot systems with boolean specifications based on simulated annealing, IEEE Robotics and Automation Letters 7(3)(2022) 6091-6098.
[15] Z.H. Pan, C.X. Zhang, Y.Q. Xia, H. Xiong, X.D. Shao, An improved artificial potential field method for path planning and formation control of the multi-UAV systems, IEEE Transactions on Circuits and Systems II: Express Briefs 69(3)(2022) 1129-1133.
[16] X.Y. Yang, M. Moallem, R.V. Patel, A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 35(6)(1224) 1214-1224.
[17] K. Balan, C. Luo, Optimal trajectory planning for multiple waypoint path planning using tabu search, in: Proc. 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, 2018.
[18] P.V.S. Reddy, Generalized fuzzy logic with twofold fuzzy set: learning through neural net and application to business intelligence, in: Proc. 2021 International Conference on Fuzzy Theory and Its Applications, 2021.
[19] T. Zhao, H.Y. Cao, S.Y. Dian, A self-organized method for a hierarchical fuzzy logic system based on a fuzzy autoencoder, IEEE Transactions on Fuzzy Systems 30(12)(2022) 5104-5115.
[20] W.Q. Mo, N. Liu, L. Li, H.Q. Han, Level control system of connecting rod ultrasonic cleaning based on fuzzy algorithm, in: Proc. 2020 International Conference on Energy Environment and Bioengineering, 2020.
[21] X.B. Xiang, C.Y. Yu, L. Lapierre, J.L. Zhang, Q. Zhang, Survey on fuzzy-logic-based guidance and control of marine surface vehicles and underwater vehicles, International Journal of Fuzzy Systems 20(2018) 572-586.
[22] M. Fakoor, A. Kosari, M. Jafarzadeh, Humanoid robot path planning with fuzzy Markov decision processes, Journal of Applied Research and Technology 14(5)(2016) 300-310.
[23] L. Xiang, X. Li, H. Liu, P. Li, Parameter fuzzy self-adaptive dynamic window approach for local path planning of wheeled robot, IEEE Transactions on Intelligent Transportation Systems 3(2022) 1-6.
[24] B. Sun, D. Zhu, S.X. Yang, An optimized fuzzy control algorithm for three-dimensional AUV path planning, International Journal of Fuzzy Systems 20(2018) 597-610.

[25] L. Xiang, X. Li, H. Liu, P. Li, Parameter fuzzy self-adaptive dynamic window approach for local path planning of wheeled robot, IEEE Open Journal of Intelligent Transportation Systems 3(2022) 1-6.
[26] C.-L. Hwang, C.-C. Yang, J.-Y. Hung, Path tracking of an autonomous ground vehicle with different payloads by hierarchical improved fuzzy dynamic sliding-mode control, IEEE Transactions on Fuzzy Systems 26(2)(2018): 899-914.
[27] A. Segato, M.D. Marzo, S. Zucchelli, S. Galvan, R. Secoli, E. De Momi, Inverse reinforcement learning intra-operative path planning for steerable needle, IEEE Transactions on Biomedical Engineering 69(6)(2022) 1995-2005.
[28] M. Xi, J. Yang, J. Wen, H. Liu, Y. Li, H.H. Song, Comprehensive ocean information-enabled AUV path planning via reinforcement learning, IEEE Internet of Things Journal 9(18)(2022) 17440-17451.
[29] Y. Zhao, Y. Ma, S. Hu, USV formation and path-following control via deep reinforcement learning with random braking, IEEE Transactions on Neural Networks and Learning Systems 32(12)(2021) 5468-5478.
[30] L. Zhang, R. Zhang, T. Wu, R. Weng, M. Han, Y. Zhao, Safe reinforcement learning with stability guarantee for motion planning of autonomous vehicles, IEEE Transactions on Neural Networks and Learning Systems 32(12)(2021) 5435-5444.
[31] C. Liu, J. Xu, K. Guo, Path planning for mobile robot based on deep reinforcement learning and fuzzy control, in: Proc. 2022 International Conference on Image Processing, Computer Vision and Machine Learning, 2022.
[32] L. Chen, X. Hu, B. Tang, Y. Cheng, Conditional DQN-based motion planning with fuzzy logic for autonomous driving, IEEE Transactions on Intelligent Transportation Systems 23(4)(2022) 2966-2977.
[33] E. Soares, P.P. Angelov, B. Costa, M.P.G. Castro, S. Nageshrao, D. Filev, Explaining deep learning models through rule-based approximation and visualization, IEEE Transactions on Fuzzy Systems 29(8)(2021) 2399-2407.
[34] W.B. Xie, B. Liu, L.W. Bu, Y.L. Wang, J. Zhang, A decoupling approach for observer-based controller design of T-S fuzzy system with unknown premise variables, IEEE Transactions on Fuzzy Systems 29(9)(2021) 2714-2725.
[35] Y. Qin, J.J. Sung, J.Y. Sang, Q-learning-based fuzzy logic for multi-objective routing algorithm in flying ad hoc networks, Wireless Personal Communications 113(2020) 115-138.
[36] D. Liu, F. L. Lewis, Q. Wei, Editorial special issue on adaptive dynamic programming and reinforcement learning, IEEE Transactions on Systems, Man, and Cybernetics: Systems 50(11)(2020) 3944-3947.
[37] A. Konar, I. Goswami Chakraborty, S.J. Singh, L.C. Jain, A.K. Nagar, A deterministic improved Q-Learning for path planning of a mobile robot, IEEE Transactions on Systems, Man, and Cybernetics: Systems 43(5)(2013) 1141-1153.

Appendix

In the right direction of Fig. 6, the line $\mathcal{LG}$ and line $\mathcal{RL}$ ($\mathcal{RL}$ represents the line between the mobile robot and point $\mathcal{L}$) are perpendicular to each other with the intersection point $\mathcal{L}$. And at the point G , the line $\mathcal{LD}$ is perpendicular to the line RG (RG represents the line between the mobile robot and G point).

In Fig. 6(b), the position of the new goal in the direction of right breakout is calculated,

$$\begin{cases} x_{g,l} = x\cos\beta\cos\beta \\ y_{g,l} = y + x\cos\beta\sin\beta \end{cases}. \tag{28}$$

In Fig. 6(c), the position of the new goal in the direction of right breakout is calculated,

$$\begin{cases} x_{g,l} = x - y\cos\beta\sin\beta \\ y_{g,l} = y\cos\beta\cos\beta \end{cases}. \tag{29}$$

In Fig. 6(d), the position of the new goal in the direction of right breakout is calculated,

$$\begin{cases} x_{g,l} = \left(\dfrac{x}{\cos\gamma}\right)\cos\beta\cos\left(\gamma+\beta\right) = x\cos^2\beta - y\cos\beta\sin\beta \\ y_{g,l} = \left(\dfrac{x}{\cos\gamma}\right)\cos\beta\sin\left(\gamma+\beta\right) = y\cos^2\beta - x\cos\beta\sin\beta \end{cases}, \tag{30}$$

where γ indicates the angle the line RG.