A Path Planning Algorithm for Space Manipulator Based on Q-Learning

Taiguo Li[1*] ,Quanhong Li[2] ,Wenxi Li[1] ,Jiagao Xia[1] , Wenhua Tang[1],Weiwen Wang[1]
1. Lanzhou Institute of Physics, Lanzhou, China
2. College of Resources and Environment, Gansu Agricultural University
ltgwhu@126.com

***Abstract*—an improved Q-Learning autonomous learning algorithm is proposed to solve the problem of the adaptive path planning of the space manipulator in the unknown environment. After simplification of the manipulator and obstacle model, the grid model of the environment is established, and the position of the manipulator and obstacles are randomly deployed in the grid map. Based on the analysis of the basic principle of reinforcement learning and the state generalization method, the improved Q-Learning algorithm is used to carry out the path planning. In this algorithm, the reward and punishment strategies in the path planning of the manipulator are designed, and the approximate greedy and continuous micro Botlzmann distribution behavior selection strategy is adopted. According to the autonomous learning of Q-table, the manipulator can guide its follow-up action selection and path planning, reduce the number of manipulator movement, and reduce the blindness of the learning process. The results show that the algorithm has the advantages of simple calculation, strong self-learning ability, and can successfully complete the adaptive path planning in unknown environment.**

Keywords— Space Manipulato; Grid Model; Q-Learning; Reinforcement Learning; Path Planning

I. Introduction

With the development of space exploration, the application of space manipulator has become an important research direction of space technology. A lot of space tasks can be completed by the space manipulator instead of astronauts, such as the release and recovery of the satellite, space equipment maintenance and complete the space science experiment. The risk of astronaut extravehicular operations can be greatly reduced, so the space manipulator application technology by domestic and foreign experts pays attention to [1]. Among them, the security of human computer interaction in the complex environment of the space manipulator has become the focus of attention. How to keep the continuous motion of the manipulator in a safe and non-collision way is particularly important. Path planning is one of the key technologies to achieve the control of the space robot arm safely[2]. The goal is to find an optimal or sub optimal security collision free path from the starting position to the target position in a given initial and target pose requirements and the surrounding environment obstacle information [3].

At present, the common method of manipulator obstacle avoidance and path planning is artificial potential field method based on free space method [7] and heuristic method based on artificial intelligence such as genetic algorithm, ant colony algorithm, fuzzy logic algorithm and neural network method.

Lazona-Perze[4] proposed free space method based on C space planning method. Although it can realize the collision free path planning of the robot arm, but because of the complex obstacle modeling method, the real-time performance of the planning is reduced. On the basis of Lazona-Perze, the path planning algorithm of obstacle avoidance for six degree of freedom space manipulator is studied by Kondo, but the method of solving the free space of obstacles is too complex [5].

FIORINI proposed C space method based on speed, can realize the mobile obstacle environment in the robot arm obstacle avoidance. But the method is only applicable to the mobile robot arm of the obstacle avoidance path planning [6][7].

Khabitl[8] proposed a repulsive potential field and attract potential field, the resultant force by repulsive potential field and attract potential field to determine the movement of the manipulator. The advantage of dynamic processing is very effective in global path planning and obstacle avoidance. The disadvantage is that it is easy to fall into local minimum point.

Connolly and Akishita proposed a potential method using reconcile function as potential function; Hwang raised the minimum potential valley of the path searching to solve local minimum problem; Sun Zengqi proposed the neural network simulated annealing algorithm which is introduced in the hypothesis and test method of penalty function, through the optimization of the penalty function to find the optimal path [7].

In the literature 9~ 16, respectively, the genetic algorithm, ant colony algorithm, fuzzy control, neural network based on artificial intelligence algorithms in the robot or manipulator path planning application [9]~[16]are researched.

Each path planning method has its own characteristics and shortcomings, its application will be subject to certain restrictions. In addition, this kind of path planning belongs to the NP-hard question. Therefore, it becomes a hot research topic in the domestic and overseas to find the effective method of the robot arm path planning in unknown complex environment. However, from the perspective of the development of the path planning algorithm, the development of the bionic algorithm and intelligent algorithm is a trend.

The reinforcement learning method is very suitable for autonomous manipulator path planning in complex environment. In recent years, many researchers have carried out in the field deep research.

978-1-5386-8178-7/19/$31.00 ©2019 IEEE

II. Environment modeling

Environment-based modeling is all-important for path planning of manipulator. The general situation is that it is very difficult to get the environment information of whole manipulator. Considering grid approach is simple, convenient and efficient in representing 2D environment information, the paper plans to model environment using grid approach [9].

The grid approach is to break the work environment of manipulator into a series of grid cells, plan the 2D operating space of manipulator using the same-sized grids. The size of grid depends on the size of manipulator. The grid within whose scope there is no obstacles is called free grid, otherwise it is called obstacle grid. The grid not fully occupied by the obstacle is counted as one grid. Hence both free space and obstacle can be expressed via set of grid blocks.

At any moment, manipulator can detect the information of the environment with current position as the center and r as radius. Set the travel step of manipulator as R, established rectangular coordinate system $\sum$ on plane S, took upper left corner of S as coordinate origin, took horizontal direction as X axis and longitudinal direction as Y axis, then the lengths of S in directions of X and Y were L_x and L_y. Divided X and Y with R as step to form a grid net. The number of grids on each line $N_x = L_x / R$, and the number of grids on each column $N_y = L_y / R$. Assumed the divided grid diagram was rectangular or square, and each obstacle b_i ($i = 1,2,...,n$) occupied one or more grids. The grid not fully occupied was counted as one grid [12].

Noted $g \in S$ as arbitrary grid, A was the set of g in S, noted $O = \{o_1, o_1, ..., o_n\} \subseteq A$ as obstacle grid set, where $b_i \in O$. $\forall g$ had definite coordinate (x, y) in coordinate system $\sum$, where x is its line No., and y is the column No. Noted the No. corresponding to the grid whose coordinates were (x, y) as $ID_{Gridx,y}$, as shown in figure 1. According to above, the coordinates (x, y) of $g_i \in A$ made up mapping relation with the ID of grid. The calculation equation is as follows:

$$x = ID_{Gridx,y} \bmod NUM$$
$$y = ID_{Gridx,y} / NUM \quad (1)$$

Where, mod represents the operation of taking remainder, and / represents the operation of taking integer.

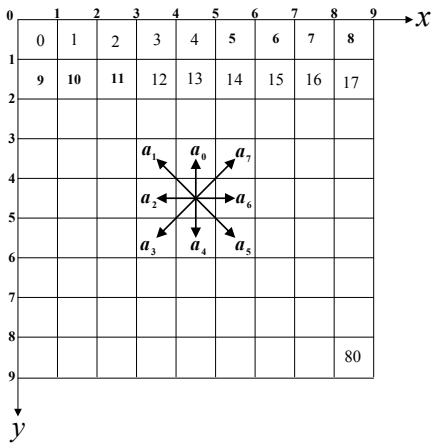

Fig 1 Grid coordinate figure

The paper assumed the manipulator moved within the convex polygon area on 2D plane, with a limited number of obstacles distributed therein. The origin position of path planning was arbitrary position $s_i \in A$, and $s_g \notin O$, terminating point $s_g \in A$, and $s_i \notin O$ was also an arbitrary position. Other given conditions were $s_i, s_g \in A$, and $s_i \neq s_g$. The purpose of planning the path for manipulator to evade obstacle is to enable the manipulator too safely and collision less go along a path with the least price from arbitrary original point s_i to arrive at the designated terminating point s_g. The objective function of path planning for manipulator is to make the total length L of collision less path from the original point to the terminating point the shortest, i.e.:

$$\min L = \sum d(t) \quad (2)$$

And meeting following condition:

$$d(t) = \begin{cases} 1, x(t) = x(t+1) \text{ or } y(t) = y(t)+1 \\ \sqrt{2}, \text{othres} \end{cases} \quad (3)$$

The equation (3) means distance of one step the manipulator moves along the neighboring directions of left and right, up and down is one unit, and the distance of one step the manipulator moves along the oblique line is $\sqrt{2}$ unit.

III. Reinforcement learning algorithm design

The dynamic system learning control based on Markov decision process (MDP) is mainly designed to enable multi-stage optimal control based on data drive when model is complicated or uncertain. So, reinforcement learning theory and method is among the main content studied in this domain and the frontier hotspot in machine learning and intelligent control domain in recent years.

A. Fundamentals and optimal value function of reinforcement learning algorithm

The reinforcement learning method of manipulator is the learning mode from environment state to action mapping, which is a strategy learning method in the course of interaction between manipulator and dynamic environment. The basic model of reinforcement learning in path planning of manipulator is as shown in figure 2.

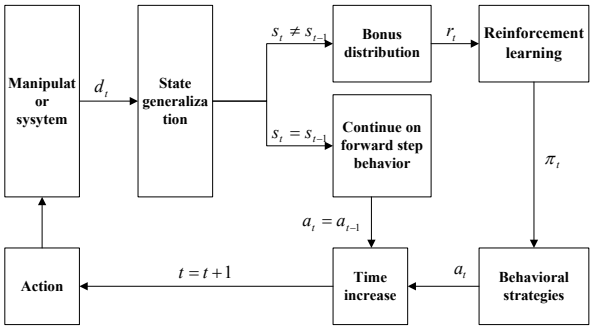

Fig 2 Basic model figure of reinforcement learning in path planning of manipulator

The theoretical frame base on MDP is composed of following 4 parts: state set S, action set A, reward value function $R: S \times A \to R$, state transfer function $T: S \times A \to \pi(S)$.

According to basic model in figure 2, in the reinforcement learning of manipulator path planning, the manipulator continuously interacted with the environment with following elements and events sequence occurring in a cyclic way in the course of learning to get the optimal path:

Step 1: The manipulator obtained environment state x_i (being the distance between manipulator and obstacle in path planning) at time t via sensor, and the obtained environment state was expressed as inner state of manipulator $s_i \in S$ (S is space set of intelligent agent state) via a proper generalization mode.

Step 2: Targeting current state and strategy $\pi: S \to A$, the manipulator chose an action $a_t \subset A(s_t)$, (where $A(s_t)$ is action space set under state s_t)

Step 3: When the action chosen by manipulator acted on the environment, the environment state was transferred to new state s_{t+1}, and reward $r_t \subset R$ was granted according to the action result of manipulator (R is the rewarding rule established according the objective of path planning).

Step 4: Way of calculating reward value for state-action was: $R(s,a) = \sum_{s' \in S} p(s,a,s')R(s,a,s')$

Where, s is the current state, s' is the next state after manipulator executes action a under state s.

B. Q-Learning algorithm

Being the foundation for various reinforcement learning methods, random reinforcement learning has Dyna-Q algorithm, TD algorithm, sarsa algorithm, Q-learning algorithm, and other learning modes. Q-Learning is a learning strategy evaluating the value function of state-action pair. As the most efficient algorithm unrelated to environmental model, it is characterized by online learning, no need for too much offline learning and the widest practical application. The paper planned to adopt Q-Learning algorithm to plan the path for space manipulator [18][19].

B.1 Action selection strategy of Q-Learning algorithm

The manipulator needs to consider selecting different actions to identify the optimal strategy, and also needs to consider selecting action with the maximal Q value to get the maximal reward. As Q-Learning algorithm spots the searched target through the process of exploration-utilization, it is required to design reasonable action selection strategy and control the balance of exploration-utilization.

Q-Learning algorithm adopts the strategy iteration mechanism, i.e. action selection strategy will change with change of Q value function estimate. The "utilization" course selection has been learned, being selecting the action with the maximal action value function from action set with the largest probability. When learning is incomplete, its Q value may be unreliable and unreliable Q value impacts future learning course. The "exploration" course refers to selecting selectable action as per designated rule. This selection is random or blind. The too small proportion of exploration action is not favorable for finding potential optimization result, thereby falling in local optimum; the too large exploration action leads to too blind action selection, slow rate of convergence, or even convergence failure.

If greedy action strategy is selected, i.e. selecting current optimum action with probability 1, then the change of action selection probability for value function estimate is not continuous. Accordingly the paper considered using approximately greedy and continuously differentiable action selection strategy (Soft-Max strategy) of Boltzmann distribution, and introduced random change of certain probability in action selection. In the stage of beginning to learn, the manipulator knew little about environment information; hence action selection was random. As learning went deep and it gained more knowledge about environment, the action selection became less random. Set action set of MDP as $A = \{a_1, a_2, a_3 \cdots a_n\}$, and $Q(s,a)$ as estimate of action value function, then action selection probability determined by Soft-Max strategy was:

$$p(s,a_i) = \frac{\exp^{Q(s,a_i)/T}}{\sum_{a \in A} \exp^{Q(s,a)/T}} \quad (4)$$

Where, T represents temperature constant, generally $T > 0$. The above equation obeys Boltzmann distribution, describing the probability of object being at different states of

energy E (corresponding to action a). The probability of object being at microscopic state with lower energy (corresponding 0 value is large) is large [20].

B.2 Action heuristics of Q-Learning algorithm

With guide of prior knowledge, the blind search can change into purposeful search which provides guide in terms of search direction to shrink search space and raise learning efficiency [21].

To speed up learning course, the paper presented a heuristic search strategy (HSS), which is met when random action is being selected. Assumed (x_c, y_c) was current state, (x_g, y_g) was target state, where a_x is as shown in figure 1. The rules of heuristic search strategy are as follows:

Table I Rules of heuristic search strategy

condition	action
$x_c < x_g$ and $y_c < y_g$	$a = rand(a_4, a_5, a_6)$
$x_c > x_g$ and $y_c > y_g$	$a = rand(a_0, a_1, a_2)$
$x_c < x_g$ and $y_c > y_g$	$a = rand(a_0, a_7, a_6)$
$x_c > x_g$ and $y_c < y_g$	$a = rand(a_2, a_3, a_4)$
$x_c = x_g$ and $y_c > y_g$	$a = a_0$
$x_c = x_g$ and $y_c < y_g$	$a = a_4$
$x_c > x_g$ and $y_c = y_g$	$a = a_2$
$x_c < x_g$ and $y_c = y_g$	$a = a_6$

B.3 Improved Q-Learning algorithm (IQLA)

The improved Q-Learning algorithm adopted action heuristics mode described in section B.2 and action selection strategy in section B.1 to select the optimal action. There were m states and 8 actions in the course of path planning for manipulator, then the algorithm needed 3 $m \times 8$ matrixes spaces, Q-table and Lock table to store reward values, Q values and state locking table respectively. The course of the whole improved Q-Learning algorithm is as follows:

Definition of parameters:

s_i: Represents initial state of beginning to learn;

s_g: Represents target state;

s': Represents next state;

α: Represents learning rate（$0 \le \alpha < 1$）;

γ: Represents discount parameter（$0 \le \gamma < 1$）;

ξ: Represents threshold value of Q difference value;

ε: Represents probability threshold value;

P: Represents reward parameter;

Steps of the algorithm:

Step 1: Initialization;

1.1 Gave parameters α, γ and reward matrix R;

1.2 Initiated the value of each element in Q-table to 0;

Step 2: Selected initial state $s = s_i$;

Step 3: Initiated Lock table. If it exceeded the scope or is obstacle, initiated the value of Lock table to 1 (state locked), otherwise initiated to 0 (state unlocked)

Step 4: One exploration course;

4.1 If Q-table is null, $a = Random(A)$, meeting HSS;

4.2 If Q-table is not null, generate p randomly, $p \in (0,1)$

4.2.1 If $p < \varepsilon$, $a = Random(A)$, meeting HSS;

4.2.2 If $p \ge \varepsilon$, $a = Boltzmann(A)$;

4.2.3 If $Locktable[s][a] = 1$, i.e. state was locked, continued to execute Step4.2;

4.3 Used selected action a to get the next state s'

4.3.1 If s' was an obstacle or exceeded current environment defined, $Q(s,a) = Q(s,a) - P$

4.3.2 If s' was target state s_g, $Q(s,a) = Q(s,a) + P$

4.3.3 In other cases, calculate Q value as per following equation:

$$Q_t(s,a) = \begin{cases} (1-\alpha)Q_t(s,a) + \alpha[R(s,a) + \gamma \max_{a \in A}(s',a)]; s = s_t, a = a_t \\ Q_{t-1}(s,a); others \end{cases}$$

4.4 $Locktable[s][a] = 0;$ (state locked)

4.5 $s = s'$;

Step 5: Updated Q-table

Step 6: If $|Q_t - Q_{t-1}| > \xi$ and $Num < MaxIterations$ (largest iterations), continued to execute Step2;

Step 7: Output Q-table

C. Algorithm of obstacle avoidance path planning

The obstacle avoidance path planning for space manipulator is designed to enable the manipulator to safely go along a path with the least price from arbitrary origin s_i to arrive at the terminating point s_g designated. The optimal Q-table can be generated via improved Q-Learning algorithm, and the maximum value of the Q-table can be selected from the current state to next state. If there were a lot of points which had the same maximal value in the course of transferring to the next state, selected the action with the smallest price from the group of same maximal value. The course of algorithm for whole obstacle avoidance path planning is as follows:

Definition of parameters:

S: Represents initial state of path planning;

G: Represents target state

s_c: Represents current state of manipulator

s_r: Represents last state of manipulator

P: Represents reward parameters

$L(s_c, s_r)$: Represents state transfer price function between current state s_c and the last state s_r

Algorithm steps:

Step1: Invoke IQLA algorithm to get the optimum Q-table

Step 2: Initialization: $s = S$， $s_0 = s$；

Step 3: If $s \neq G$ (current state is not the target state);

3.1 If current state was not obstacle;

3.1.1 If $\forall r, Q_n > Q_r$

$$s' = \arg\max Q(s_r, a)$$

3.1.2 If $\forall r, Q_{n1} = Q_{n2} = ... Q_{ni} > Q_r$,

$$s' = \arg\min(L(s_c, s_r))$$

3.1.3 $s = s'$

3.2 If current state was obstacle,

$$Q(s) = -P, s = s_0$$

Step 4: If $s \neq G$, continued to execute Step 3;

Step5: Output the optimal action sequence.

IV. Simulation and verification

The simulation scene can be established as a $n \times n$ grid diagram, as shown in figure 3. Therefore, each grid represents a state of manipulator. The manipulator started from original point S, and target point is G. Black area represents obstacle. Assumed the obstacle and target in the environment were static. The manipulator knew nothing about the environment (i.e. positions of obstacle, boundary and target). 8 movement directions were evenly distributed in 2D space centered on manipulator, representing its 8 action options. Immediately reward r was $\{-300, -100, 100, 0\}$, and corresponding conditions were {exceeding scope, collide with obstacle, reach target state, others}. Success episodes refer to the learning cycle during which the manipulator begins from the original position, and successfully reaches target point after learning. The success step number refers to the step number of successfully learning of manipulator in each episode. The smaller success step number indicates the action strategy of manipulator is more optimal, and the efficiency of path planning is higher.

The hardware environment of simulation is PC with Intel core Duo, basic frequency of 2.93 GHz, and memory of 2G. The simulation software used is Visual C++ 6.0.

C.1 Simulation parameters setting

The conditions of simulation experiment are set as follows:

1) Learning rate $\alpha = 0.2$
2) Discount parameter $\gamma = 0.8$
3) Probability threshold $\varepsilon = 0.2$
4) Reward parameter $P = 100$
5) Environment 1: on 5×5 grid map, set obstacle as {4，7，11，18}，initial state of path planning $S = 0$, the target state of path planning $G = 24$; automatically generate information of reward value matrix $reward[25][8]$ by dint of obstacle information;
6) Environment 2: on 10×10 grid map, set obstacle as {10，11，36，45，46，47，77，78，87，89，97，98}，initial state of path planning $S = 0$, the target state of path planning $G = 99$; automatically generate information of reward value matrix $reward[100][8]$ by dint of obstacle information;
7) Environment 3: on 20×20 grid map, information of obstacles is as shown in figure 3，initial state of path planning $S = 0$, the target state of path planning $G = 399$; automatically generate information of reward value matrix $reward[400][8]$ by dint of obstacle information;

C.2 Simulation result

To effectively compare the algorithms, three simulation environments were set. Table 2 shows statistics about number of steps executed by the two algorithms in three different simulation environments.

Table II Number of steps executed in different environment using different algorithms

Environment	Classical Q-Lenrning algorithm	Improved Q-Lenrning algorithm
Environment 1	73458	46112
Environment 2	1068100	244899
Environment 3	2566648	1024019

Figure 3 shows a path from original point 0 to target point 399 obtained by using the improved Q-Learning algorithm presented in this paper in environment 3. It is observed that this path is collisionless path of global optimum.

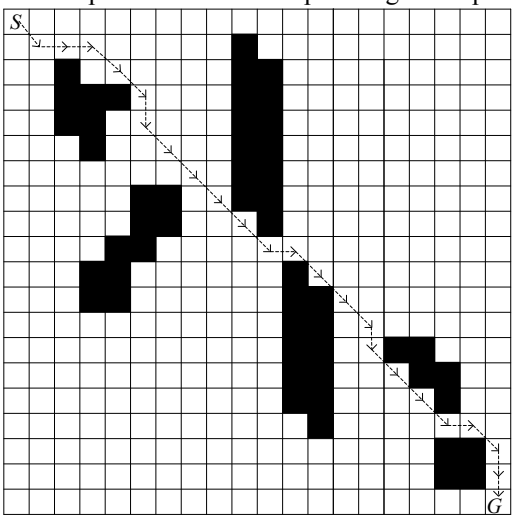

Fig 3 Path planning

Simulation result shows that compared with conventional algorithm, the improve Q-learning path planning algorithm has fewer steps to be executed and shorter path length, which are favorable for controlling manipulator and keeping relatively smooth operation, also save more energy consumption of manipulator. Meanwhile, this improves effect

of path searching, expedite convergence speed and raise real-timeliness of the whole system. It should be noted that simulation environment can be generated randomly in above simulation research on path planning of manipulator.

V. Conclusions

The state space was highly generalized via an analysis on the course and purpose of path planning of manipulator. Based on reasonable defining of action space, transfer function and reward function, an improved Q-Learning algorithm was suggested. The subsequent action selection was calculated by dint of evaluation function and reward function, through independent learning and establishing Q table, recording positions of manipulator and obstacle. The heuristic search strategy was introduced to lead the manipulator to effectively evade obstacle and approach target. In the meanwhile, with the distance between manipulator and target position as target function, the manipulator was led to find out the shortest path from the original position to the target position.

Simulation experiment was done in different obstacle environments with result showing space manipulator has strong self-teaching capacity and can finish adaptive path planning in unknown environment through online learning, which underpins effectiveness of this approach. This work provides theoretical base and data reference for real experiment of space manipulator.

References

[1] CHEN Jing-bo, ZHAO Meng, ZHANG Heng. On-line Real-time Obstacle Avoidance Path Planning of Space Manipulator[J]. Control Engineering of China, 200, 14(4): 445-450.

[2] Qu Dao-kui, DU Zhen-jun, XU Dian-guo et al. Research on Path Planning for a Mobile Robot[J]. ROBOT, 2008, 30(3): 97-101, 106.

[3] LUO De-lin, WU Shun-xiang. Ant colony optimization with potential field heuristic for robot path planning[J]. Systems Engineering and Electronics, 2010, 32(6) : 1277-1280.

[4] Lozano-Perez T. Automatic planning of manipulator transfer movement[J]. IEEE Transaction on Systems, Man and Cybernetics, 1981, 11(10): 681-698.

[5] Kondo K.Simple motion planning algorithm using heuristic free space enumeration[C]. IEEE International Workshop on Intelligent Robotics and Systems. New York: IEEE, 1988: 751-756.

[6] JIA Qing-xuan, CHEN Gang, SUN Han-xu et al. Path Planning for Space Manipulator to Avoid Obstacle Based on A* Algorithm[J]. JOURNAL OF MECHANICAL ENGINEERTNG, 2010, 46(13): 109-100.

[7] Robotics Collision Avoidance Planning [J]. AEROSPACE CONTROL, 2002, 20(1): 34-46.

[8] Khabit. Real-time obstacle avoidance for manipulators and mobile robots[J]. The International Journal of Robotics Research, 1986, 5(1): 90-98.

[9] ZHU Lei, FAN Ji-zhuang, ZHAO Jie et al. Global path planning and local obstacle avoidance of searching robot in mine disasters based on grid method[J]. Journal of Central South University, 2011, 42(11): 3422-3424.

[10] LI Qing, FENG Jin-ling, LIU Yan-ling et al. Application of adaptive genetic algorithm to optimum path planning of mobile robots[J]. Journal of University of Science and Technology Beijing, 2008, 30(3): 317-321.

[11] WANG Ke-jun, XU Jing, WANG Lei et al. The path planning for robots based on the genetic extension algorithms[J]. JOURNAL OF HARBIN INSITUTE OF TECHNOLOGY, 2006, 38(7): 1135-1138.

[12] ZHU Qing-bao. Ant Algorithm for Path Planning of Mobile Robot in a Complex Environment [J]. ACTA AUTOMATICA SINICA, 2000, 32(4): 587-593.

[13] CHEN Wei-dong, ZHU Qi-guang. Mobile Robot Palh Planning Based on Fuzzy Algorithms[J]. ACTA ELECTRONICA SINICA, 2011, 39(4): 971-974.

[14] SONG Yong, LI Yi-bin, LI Chun et al. Path planning methods of mobile robot based on neural network [J]. Systems Engineering and Electronics, 2008, 30(2): 316-319.

[15] Qian Kui, Song Ai-guo, zhang Hua-tao etl. Path planning for mobile robot based on adaptive fuzzy neural network [J]. 2012, 42(4): 637-642.

[16] LIANG Jin, SONG Ke-pu. The Application of Neural Network in Mobile Robot Path Planning [J]. Journal of System Simulation, 2010, 22(1): 269-272.

[17] XU Xin, SHEN Dong, GAO Yan-Qing et al. Learning Control of Dynamical Systems Based on Markov Decision Processes : Research Frontiers and Outlooks [J]. ACTA AUTOMATICA SINICA, 2012, 38(5): 673-682.

[18] TONG Liang, WANG Zhun. Research on Application of Reinforcement Learning in Robot Path-Planning [J]. Computer Simulation, 2013, 30(12): 351-355.

[19] Chang Bao-xian, Ding Jie, Zhu Jun-wu et al. Robot Q-learning coverage algorithm in unknown environments [J]. Journal of Nanjing University of Science and Technology, 2013, 37(6): 792-798.

[20] XU Xin, HE Han-Gen. A Gradient Algorithm for Neural-Network-based Reinforcement Learning[J]. 2003, 26(2): 1-6.

[21] LIU Zhi-bin, ZENG Xiao-qin. A M ethod of Heuristic Reinforcement Learning Based on Acquired Path Guiding Knowledg[J]. JOURNAL OF SICHUAN UNIVERSITY, 2012, 44(5): 136-141..