International Journal of Control and Automation
Vol. 9, No. 5 (2016), pp. 351-362
http://dx.doi.org/10.14257/ijca.2016.9.5.34

A Hybrid Path Planning Algorithm for Indoor Mobile Robot Using Hierarchy Reinforcement Learning

Shen Cheng'en and He Jun*

College of Computer Science & College of Software Engineering,
Sichuan University, China
sce@stu.scu.edu.cn

Abstract

This paper focuses on the path planning for a mobile robot which is operated in indoor environment. Since the layout of indoor environment is a hybrid structure of known and unknown, this paper presents a hybrid algorithm which uses the Max-Q method and the option method together. Firstly, a novel task graph and high level definition are presented to divide sub-tasks. Then, the appropriate definitions of states, actions and options could let a robot fulfill a task. Finally, an angle parameter is employed in the reward function to ensure a robot select a shorter path and adjust orientation timely. In the series of simulations, a robot can arrival any position successfully with random initial positions and directions. Moreover the results show that a robot can overcome the local minimal problem with our hybrid method.

Key-words: *mobile robot, Hierarchical Reinforcement Learning, path planning, hybrid method*

1. Introduction

With the development of indoor mobile robot, the indoor navigation which plan a free-collision path from an initial location to a goal, become one of the most popular tasks. There are many algorithms, such as A*, Dijkstra, genetic algorithm[1-2] and artificial potential field(APF) method[3-4]. A* can design an optimal path, but it is dependent on the known environment information. Artificial potential field, which is widely used in path planning because of its straightforward and efficient mathematical analysis, could have some trap situations due to local minima[4]. Therefore, the reinforcement learning method is raised as a new way to plan path[5-8], because the reinforcement learning method as an artificial intelligence method has strong adaptability to the environment.

Recent years some scholars already used the reinforcement learning in path planning. R.Karthikeyan, B.SheelaRani [7] used the standard reinforcement learning algorithm to plan path .They assumed that the information of indoor environment are known ,and used the grid mapping method to express the information. Unfortunately, their study may be more reasonable if they considered the computational time when the information of indoor environment is huge.

Andrea and Fernando[8] used the option method which is one of the hierarchical reinforcement learning methods , and they thought that the layout of indoor environment is totally unknowable. However this approach could cause some local minimal problems when a robot is locating in some specific situations. They also only focused on some fixed directions of a robot, such as $30^\circ, 90^\circ$, but a real robot could fulfill a task with random directions.

However we believe that the layout of indoor environment is a mix structure of known and unknown, and any individual reinforcement learning algorithms could not deal with the path planning task in indoor environment commendably. In this paper, we use a hybrid method of hierarchical reinforcement learning method to generate a path. We consider a mobile robot just move in 2-dimesion situation and do not encounter any dynamic objects.

Section 2 present the background knowledge of the reinforcement learning algorithm .The algorithm's details are proposed in section 3.The simulation results and discussion are talked in section 4.The final section covered summarizations and the future works.

2. Related Knowledge

2.1. Reinforcement Learning

Reinforcement learning (RL)[9] is a method that an agent optimize its tasks by interacting with environment, even it begins with minimal information of its environment. In order to fulfill a task, a robot must detect the environment by sensors. According to perceived information, a robot takes an action and gets a reward by reward function. Q-learning is one of the broadest reinforcement learning algorithms by Watkins[10], and it is a model-free reinforcement learning algorithm. A robot could choose an action from current state by a Q matrix, which is a two-dimensional matrix about the mapping of states and actions. The matrix Q update formula is presented as blow.

$$Q_n(x,a) = (1-\partial_n)Q_{n-1}(x,a) + \partial_n\left[r_n + \gamma \max(Q_{n-1}(y_n, b))\right] \tag{1}$$

Where the $\partial \in [0,1]$ is learning rate, $\gamma \in [0,1]$ is the discount rate.

2.2. Hierarchical Reinforcement Learning

A question of the normal reinforcement learning method is that the amount of calculations will become huge if state spaces are large. This problem is called dimension disaster which could make the algorithm spent a lot of time and memories on calculation. To solve this problem, hierarchical reinforcement learning algorithm was proposed, and the option[11] and Max-Q[12] are two common ways.

2.2.1. Option: The option algorithm is based on the model that algorithm use option as a high level item to control a robot and the option is defined as a three-tuple $< \Pi, \beta, \mathrm{I} >$. A policy π : $S \times A \to [0,1]$, termination conditions β: $S^+ \to [0,1]$ and initiation sets. The state transform s_t to s_{t+1} when a robot select an action according to π(s).Then termination conditions is checked, if the termination condition β(s_{t+1})satisfies the threshold ,then the option is terminated and select another option. Otherwise the option continue take an action according to π(s_{t+1}).This process is repeated until the goal is finished. This model is computed by Bellman equations for general policies and options.

$$Q_{k+1}(s,o) = (1-\partial_k)Q_k(s,o) + \partial_k\left[r + \gamma \max_{o' \in s'} Q_k(s', o')\right] \tag{2}$$

Where r is the reward function, α is learning rate, k is the time step, γ is reward discounted rate. It has been proved that for all s∈S, o∈O, the Q(s,o) converge to the optimal value function $Q^*(s,o)$,under the conditions which similar to the general Q-learning.

2.2.2. Max-Q: According to prior environmental information , a whole task M is divided into a set of sub-tasks { $m_0, m_1 \ldots\ldots m_m$ },a sub-task is defined as a three-tuple $< T_i, A_i, \tilde{R}_i >, T_i$ is a termination prediction which comprise s set of active and terminal states, A_i is a set of actions which could be chosen when a robot running in a sub-task m_i, $\tilde{R}_i$ is the pseudo-reward function. Correspondingly, there is a hierarchical policy $\pi = \{\pi_1, \pi_2 ......\pi_m\}$, the π_i is the sub-task m_i's policy. The purpose of Max-Q is that pick up an action (composite or primitive) to maximize the value function $V^{\pi}(i,s)$.The value function is updated according to the equation:

$$V^{\pi}(i,s) = \begin{cases} Q^{\pi}(i,s,\pi_i(s)) & \text{if i is composite} \\ \sum_{s'} P(s' \mid s,i) R(s' \mid s,i) & \text{if i is primitive} \end{cases} \tag{3}$$

i is the action both composite action and primitive action, different kind have different way to update.

Those two methods have their own merits and limitations .The option way could automatic partitions sub-tasks very well in unknown environment ,and the size of each option could be control very well, but it is not ideal when the pre-knowledge is used. The Max-Q method is good at online learning, and it is very clearly to show the structure of sub-tasks with task graph under environmental information. However the ability of automatically partition without environment information is not well and the size of those sub-tasks is not of uniform size.

3. The Design of the Hybrid Method

In the previous works, the layout of indoor environment is regarded as known or unknown. However those viewpoints are too absolute, because the layout of indoor environment actually is a mix structure of known and unknown. For example, considering a house presented as Figure 1(a), the over-all layout is known. It is means that the number of rooms and topology showed in Figure 1(b) are already knew, but the circumstance of one specific room is unknowable. People do not know the location of a specific object because the object is changed constantly. The amount of calculations could become huge using standard reinforcement learning when the information of the house is large. So neither option way nor Max-Q method could cope with navigation task perfectly in indoor environment.

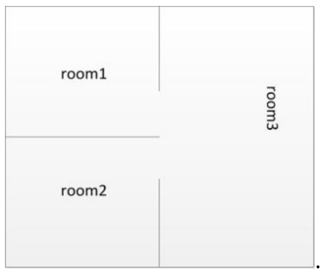

Figure 1(a). The Environment of a House

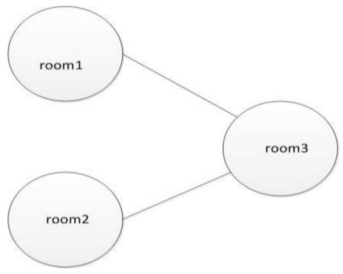

Figure 1(b). The Topology of the House

In this paper, we believe that the hybrid method using the option and Max-Q together could cope with this mix structure very well. According to the prior knowledge which is the topology of rooms, the Max-Q method could divide a root task into several sub-tasks. There are some changes about the standard Max-Q definition. The three-tuples become $< T_i, O_i, \tilde{R}_i >$ rather than $< T_i, A_i, \tilde{R}_i >$. It is means that a sub-task is comprised of some options rather than actions. Each sub-task has a sub-goal and the sub-goal of the last sub-task is the robot’s real target, the T_i (termination of sub-goal) is activated when the distance between the robot and sub-goal is smaller than a threshold. Then an option as a decider chooses a sequence of actions to fulfill its task. The task graph is Figured as Figure 2.

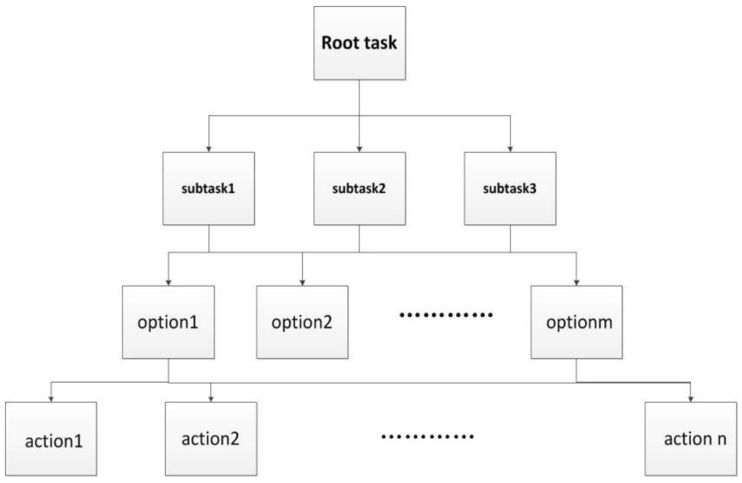

Figure 2. The Task Graph of the Hybrid Method

To generate an optimum path planning, State spaces, actions and options should be defined appropriately after sub-tasks are defined, and a right reward function is equally necessary. The details will be presented in the following sections.

3.1. States and Actions Definition

There are some features which ensure that a robot could choose an optimized planning The first one is the roughly position of a robot, it is indicate that which room the robot stay in. It is helpful for a robot to choice and sort sub-tasks, and the number of values depends on the number of rooms

The next two features is the distance and angle between a robot and the nearest obstacle respectively .In this paper, there is an assumption that a robot could detect circumstance in the frontal 180° and a robot could perceive obstacle smaller than 1000mm. The distances and angles between a robot and the nearest obstacle are discretized into several state variables separately. Table1 and Table 2 present the discretizations in detail.

Table 1. Discretization of Angle about the Nearest Obstacle

	Discretization of continuous state
(0,500]	**1**
(500,1000]	**2**

Table 2. Discretization of Angle about the Nearest Obstacle

	Discretization of continuous state
[0°,20°)	**1**
[20°,60°)	**2**
[60°,90°]	**3**
[-20°,0°)	**4**
[-60°,-20°)	**5**
[-90°,-60°)	**6**

The fourth feature is whether the robot arrive the goal. It is a binary number, which take 1 as value when the distance between the goal and a robot is smaller than a threshold, otherwise, it is choice 0.

The last feature is the angle between a sub-goal and a robot's direction. Since a robot must adjust its direction timely, this feature is a significant one for out method. It is also discretized into different variables (in Table 3).Ignore hardware problems, we think that a robot could turn around 360°. If the angle locate in range $[-10^\circ, 10^\circ]$, then we believe that a robot has a correct orientation.

Table 3. Discretization of the Angle between the Goal and the Robot Direction

	Discretization of continuous state
[0,10°)	**1**
[10°,30°)	**2**
[30°,90°]	**3**
[-10°,0)	**4**
[-30°,-10°)	**5**
[-90°,-30°)	**6**
else	**7**

The primitive actions are composed of turn right 10°, turn left 10°,turn right 20° ,turn left 20°,turn right 60°,turn left 60°,turn right 90° and move one step.

3.2. Options Definition

So far, the definition of states and actions is done, which is the bottom definition of our method, and the high level is options. In this paper, there are four options; each option has initiation sets, a terminal condition and a policy .Those details are showed below. These options are trained by Q-learning technique separately.

Table 4. The Division of Option

	Initiation states	Termination condition	policy
Go-straight	The distance between the goal and a robot is more than σ and robot could not perceive any obstacle	The distance between the goal and a robot is less than σ or some obstacles have be found	Allow a robot to move toward to the goal
Avoid obstacle	A robot finds some obstacles in front of itself.	A robot do not perceive any obstacle in front	Allow a robot turn right or turn left to avoid the frontal obstacle
Turn-left to parallel obstacle	A robot find some obstacles in right of itself	The wall ends or a robot perceive another direction obstacle	Allow a robot turn left and follow the right wall
Turn-right to parallel obstacle	A robot find some obstacles in left of itself	The wall ends or a robot perceive another direction obstacle	Allow a robot turn right and follow the left wall

3.3. Reward Function

A reward function is proposed as blow.

$$R = \frac{distobs}{distgoal} \mid angle \mid \tag{4}$$

Distobs is the distance between a robot and the nearest obstacle which could be detected, distgoal is the distance between a robot and sub-goal .Since we want that a robot could have a right direction timely and choose an optimum action without redundancy, the $|\mathrm{angle}|$ parameter is presented as below.

$$|angle| = |a\tan t(x\_goal - x\_agent, y\_goal - y\_agent) - direction\_agent| \tag{5}$$

The first item is the right direction and the following item is a robot actual direction, and the above variables are discretized into different values as below:

$$distobs = \begin{cases} 1000 & \text{if } distobs>1000 \\ 100 & 500<distobs<=1000 \\ 1 & 0<distobs<=500 \end{cases} \tag{6}$$

$$|angle| = \begin{cases} 4 & 0°<=|angle|<10° \\ 3 & 10°<=|angle|<30° \\ 2 & 30°<=|angle|<90° \\ 1 & \text{else} \end{cases} \tag{7}$$

The disgoal is equal to the Euclidean distance plus a constant, ensure that the value of reward do not go to infinity. Finally, the Q-algorithm is selected to train this hybrid method, with α=0.5, γ =0.5.

4. Simulation

Simulations were executed using the robot toolbox in the matlab software, and the environment is presented as blow. There are three rooms in experimental environment, and a robot is figured as a triangle.

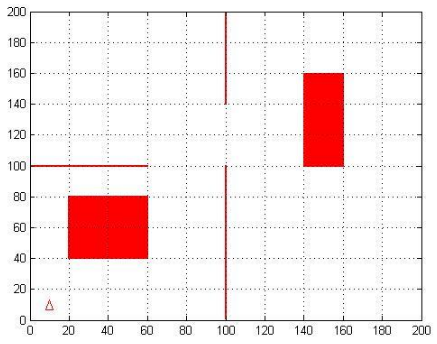

Figure 5. The Experimental Environment

4.1. Partial Simulation

Initiative states and targets were selected randomly when partial simulation was tested in a sub-task. The Figure6 are four final trajectories when Q matrixes of the option framework converged to fixed values in a room. The four trajectories have different initial positions, initial directions and targets, and all of them have good performances.

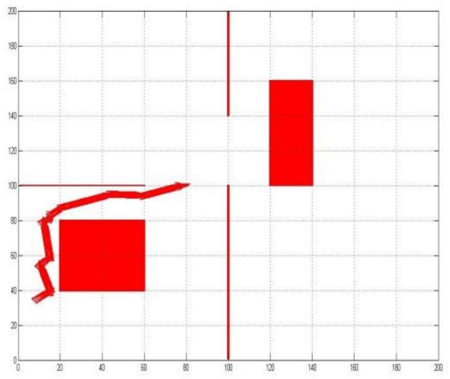

Figure 6(a). Origin State (5,35,pi/4) Target (80,100)

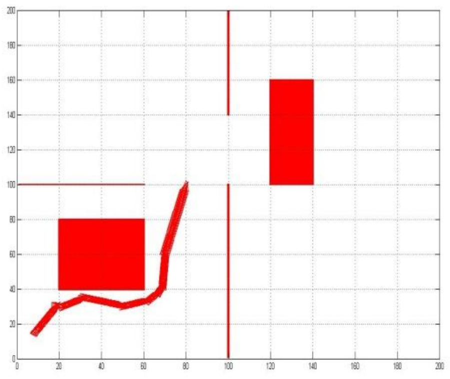

Figure 6(b). Origin State (5,10,0°)Target (80,100)

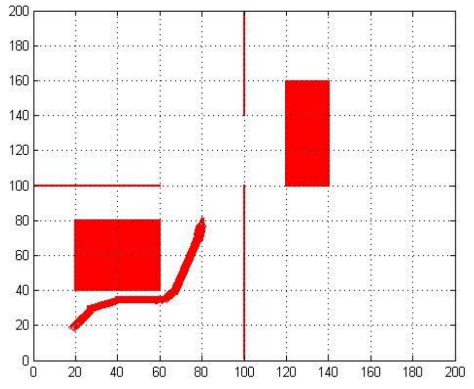

Figure 6(c). Origin State(20,20,0°) Target(80,80)

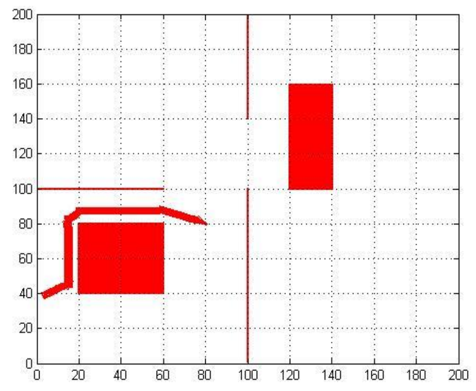

Figure 6(d). Origin State(5,40,pi/6) Target(80,80)

4.2. Whole Simulation

When all Q matrixes of the Max-Q and option framework converged to fixed values, the entire simulation used our hybrid method was tested with go through different sub-tasks.

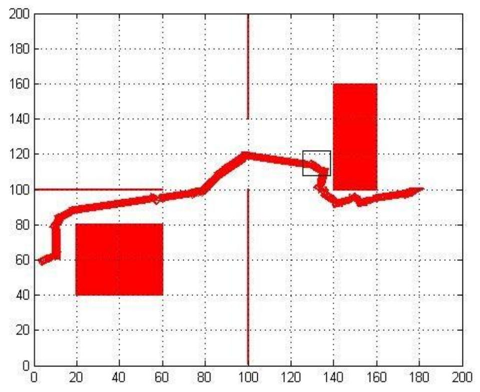

Figure 7(a). The Origin State is (5,60, 0°) and the Final Position is (180,100)

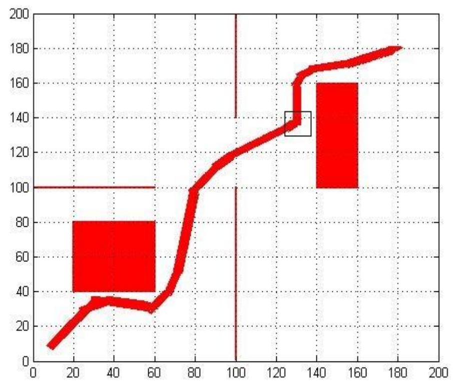

Figure 7(b). The Origin Location is (10,10,45°) and The Final Position is (180,180)

The Figure7(a) show a complete path in the whole horse. According to the prior knowledge of the whole topological structure, the navigation task was divided into three sub-tasks with the Q matrixes of Max-Q, and the order is M1 (move in room1), M2 and M3.Firstly, the sub-task1 used the option framework to accomplish its goal. When a robot was going to the first sub-goal, the sub-task1 was finished, and the sub-task2 was activated. Through the room2, the robot moved in the room3, the sub-task3 let the robot move to the ultimate goal. The root task have not to know how exactly the option framework is work or which primitive action is taken, the root task only have to know that those sub-tasks must be finished orderly. A sub-task has taken different sequences of options to finish its task. Any option taken primitive actions with Q matrixes which trained in training step until it has been terminated. The Figure 7(b) is another path with different original states and goal. Make a general survey of the two paths, no matter what the original states and the destination are, the algorithm always have a good performance.

Look at the Figure 7(a) and 7(b) which be marked with black box. Both a robot detected the same obstacle in room 3, but they taken different actions. The robot in Figure 7(a) turned right and the other turned left, both of them taken an optimum action which could not cause redundancy problem .It shows that the robot could pick a shorter path rather than a feasible path.

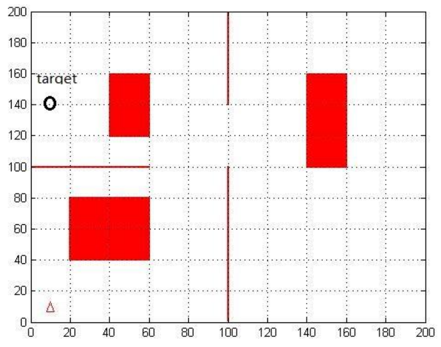

Figure 8(a). A Situation Which Could Cause Local Optimum Problem

Transfer to the circumstance presented above on the Figure 8(a) which can cause local minimal problem when standard option algorithm[8] is taken. When the method is used in this situation, a robot will go straight until it hit against the wall in the position (10,90) , then the robot turn right and follow the wall until the wall ends , later the robot adjust its direction to move toward its goal. This is a typical local minimal problem because of the lack of the prior knowledge. According to the simulation result presented as Figure 8(b), we can see that the there are two ordered sub-tasks (Move in room1 and Move in room2) were formed and a robot could overcame the local minimal problem when the hybrid method was used.

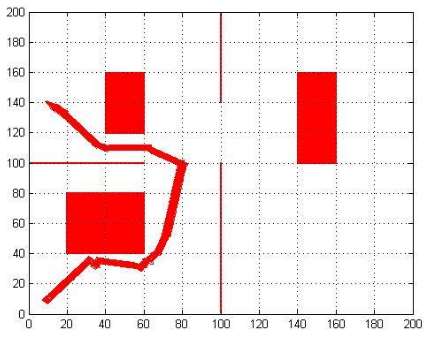

Figure 8(b) The Hybrid Method can overcome the Local Optimum Problem

All of those simulations can declare that no matter where the initial states and goal are, the hybrid algorithm could always find a feasible and a shorter path without redundancy. However, this hybrid framework has a problem called shake .When a robot and a sub-goal are very close, the value of the angle between a robot's direction and the right direction could change frequently, so the robot has to change its direction constantly to adapt the value.

4. Conclusion and Future Work

In this paper, a hybrid hierarchy reinforcement learning for path plan using the option method and the max-q method is proposed to solve the navigation task in indoor application. The option way can automatic divide task very well in unknown environment and the Max-Q method is good at task partition with prior knowledge. The hybrid algorithm picks up both the advantages of the two methods, and copes with the indoor environment properly. A novel task graph and definition of the algorithm are presented to generate sub-tasks. The appropriate definitions of states, options and actions let a robot move precisely. The |angle| variable between the right direction and a robot's direction is proposed in reward function to make sure a robot can pick up an optimum action, and could adjust its orientation timely with any original directions. The results of simulations show that a robot choice a feasible and non-redundancy path regardless of initial states and target. However the variable of |angle| could produce the shake problem when the distance between a robot and a goal is close .One solution is that a threshold function of the angle could be used when the distance of a robot and a sub-goal is smaller than a fixed value. Another way is to pick up a punishing item in the reward function to confine the alteration.

In future works, we consider add another robot, and both robots could move to their own target separately without collision. This idea will has bigger state spaces, so the definition of states, actions and options will be more elaborate. With this improvement, the dimension of Q matrixes also will become huge. Since the neural network has nature advantage about calculation, so we will consider use the neural network to training the hybrid algorithm with our improvement.

References

[1] G. D. Liu, H. B. Xie and C.G. Li, "Method of mobile robot path planning in dynamic environment based on genetic algorithm", Robot, vol. 25, no. 4, **(2003)**, pp. 327-323.

[2] Y. K. Hwang and N. Ahuja, "Gross motion planning-a survey", ACM Computing Surveys, vol.24, no.3, **(1992)**, pp. 219-291.

[3] S. S. Ge and Y. J. Cui, "New Potential Functions for Mobile Robot Path Planning", IEEE Transactions on robotics and automation,vol.16, no. 5, **(2000)**, pp. 615-620.

[4] Y.Koren and J.Borenstein, "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation", Proceedings of the IEEE conference on Robotics and Automation, **(1991)** April 7-12; Sacramento, California

[5] K. Macek, I. Petrovic and N. Peric, "A reinforcement learning approach to obstacles avoidance of mobile A robot", proceedings of the 7th International Workshop on Advanced Motion Control, Maribor, Slovenia, **(2002)**, July 3-5.

[6] Y. Li, C. Li and Z. Zhang, "Q-learning based method of adaptive path planning for mobile robot", proceedings of the IEEE International Conference on Information Acquisition, Shandong, China, **(2006)**, August 20-23.

[7] R. Karthikeyan, B. SheelaRani and K. Renganathan, "An Instant Path Planning Algorithm for Indoor Mobile A robot Using Adaptive Dynamic Programming and Reinforcement Learning", International Journal of Engineering and Technology, vol.6, no.2, **(2014)**, pp. 1224-1231.

[8] Andrea Buitrago-Martinez, Fernando De la Rosa R., Fernando Lozano-Martinez, "Hierarchical Reinforcement Learning Approach for Motion Planning in Mobile Robotics", IEEE Latin American Robotics Symposium,**(2013)**, pp.83-88

[9] Leslie Pack Kaelbling and Michael L. Littman , "Reinforcement Learning: A Survey", Journal of Artificial Intelligence Research,vol.4,**(1996)**,pp.237-285

[10] C. J.C.H. Watkins and P. Dayan, "Q-learning", Machine learning, vol.8, no.3-4, **(1992)**, pp.279-292.

[11] R. S. Sutton, D. Precup and S. Singh, "Between MDPs and Semi-MDPs: A Framework for Temporal

Abstraction in Reinforcement Learning", Artificial Intelligence,vol.112, no.1-2, **(1999)**, pp.181-211
[12] T. G. Dietterich, "Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition", Journal of Artificial Intelligence Research,vol.13, **(2000)**, pp.227-303.

Authors

Shen Cheng'en, he is currently a student at College of Computer Science & College of Software Engineering, Sichuan University, China. His research interests include intelligent control.

He Jun, he is an associate professor at College of Computer Science & College of Software Engineering, Sichuan University,China. His research interests include computer network, softeware engineering, computer architecture.