Proceedings of the 2019 IEEE 16th International Conference on Networking, Sensing and Control, May 9-11, 2019, Banff, Alberta, Canada

Reinforcement Learning for Robots Path Planning with Rule-based Shallow-trial

Kaiqiang Tang*, Huiqiao Fu*†, Hao Jiang†, Canghai Liu†, Lan Wang*
*Department of Control and Systems Engineering, School of Management and Engineering
Nanjing University, Nanjing 210093, China
Email: MG1715008@smail.nju.edu.cn
†Key Laboratory of Testing Technology for Manufacturing Process of Ministry of Education
School of Manufacturing Science and Engineering
Southwest University of Science and Technology, Mianyang 621010, China
Email: liucanghai@swust.edu.cn

***Abstract*—A key skill for mobile robots is the ability to navigate efficiently through their environment, and reinforcement learning is widely used in path planning for mobile robots. However, this algorithm has a slow convergence speed and a large number of iterations. There are few studies on how to improve learning efficiently from the perspective of acquisition in rule-based shallow-trial strategy. In biological world, animals depend on their own empirical knowledge when making path planning. Humanity has transcendental knowledge, which is of great help to peoples navigation. We take the transcendental knowledge of human behavior, and express it acts as shallow-trial rules , then apply the rule-based shallow-trial reinforcement learning(RSRL) to the navigation learning of robot and improve learning efficiently.**

***Index Terms*—Reinforcement learning, Shallow-trial, Path planning, Rule-based.**

I. Introduction

The ultimate goal of robotics is to create autonomous robots, which may execute tasks without human beings' intervention, and one of the most important tasks in robotics is path planning where a mobile robot interacts with its environment and tries to find the optimal path from a starting point to an ending point [1]. In other words, efforts are underway to ascertain approaches to endow human-like capability to interact with physical environment.

Many classical, heuristic and metaheuristic methods have been proposed by researchers to solve path planning problem [2]; for example, classical methods, such as cell decomposition, road map, potential field [3], etc., are based on the concept of space configuration. Those methods require a very high execution time to solve a NP-hard problem such as path planning [4]. It is far from exhibiting human-like natural intelligence in terms of flexibility and reliability to work in dynamic scenarios.

In order to overcome that problem, reinforcement learning(RL) was widely used in path planning for mobile robots[5],[6]. Those methods [7] can be classified into model-free [8] and model-based [9], [10], [11] methods. Model-free methods obtain the optimal policy without learning the model. On the other hand, model-based methods employ the information of model which is known to obtain the optimal policy. Those methods can also be divided into on-policy and off-policy, while classified to monte-carlo update and temporal-difference update.

Application of reinforcement learning in robotics has helped making robots realize whether the desired result has been achieved by a series of certain actions, and it's aim is to obtain the optimal policy by interacting with the environment. We have designed a task of autonomous robot navigation. However, training in a real environment using conventional reinforcement learning techniques could be quite cumbersome and inefficient, and the reason is that the training requires abundant data.

To address those problems, we refer to the biological world, where animals use their own empirical knowledge when making path planning. Human being has transcendental knowledge, which is of great help to people's navigation. We take the transcendental knowledge of human beings, and many times we can express it in terms of rules. It stores this information as a rule, and when it encounters a similar situation, it tries to make a correct move. We shallow-trial this rule to navigation learning of robot and improve the learning efficiency.

In this paper, we formulate a model of a real environment scenario in the form of grid-world like simulator, and propose a novel approach for robots path planning by rule-based shallow-trial reinforcement learning(RSRL). The rest of this paper is organized as follows. Discussed the RL algorithm and the rule-based shallow-trial strategy in Section II. The algorithms of RSRL on robots path planning has been proposed in Section III. Simulations results are provided in Section IV. Conclusions are given in Section V.

II. Background

A. Reinforcement Learning

Reinforcement learning(RL) addresses the problem of how an autonomous active agent can learn to approximate the optimal behavioral policy that maps states to actions to maximize the long-term cumulative reward while interacting with its environment in a trial-and-error manner [12], [13]. In past decades, many successful RL algorithms, such as temporal difference (TD) and Q-learning, etc., have been widely used in intelligent control and industrial applications. In addition,

978-1-7281-0084-5/19/$31.00 ©2019 IEEE

recent interplay of rhythmic and discrete manipulation movements are under development: a policy-search [14], socially adaptive path planning [15], extended classifier system (XCS) [16] skills are also introduced to classical RL algorithms, so as to attain better performance for real environment applications regarding learning efficiency.

Typically, the interaction between the agent and the dynamic environment in RL is modeled as an MDP. An MDP consists of (S,A,P,R), where S is a set of system states, A is a set of actions, P is a transition function describing the probability taking an action a at state s which will lead to state s, and R denotes the reward when the agent takes action a in state s.

The goal of RL is to updates the policyπ^* based on its interaction with the environment such that the cumulative rewards over various episodes are maximized. The optimal strategy can be given by the optimal value function $V^*(s)$. Based on the optimal strategy, the formulation of $V*(s)$ can be defined as

$$V^*(s) = \max_{a \in A(s)} (\sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma V^*(s'))), \tag{1}$$

where $\gamma \in (0,1]$ is a discount factor.

Q-learning is a widely used model-free reinforcement learning algorithm . In Q-learning, the agent knows what actions can be selected in current state without model of the environment. Generally, we should build an instant reward matrix R to indicate the reward from state s to the next state s'. The one-step updating of the action-value function $Q(s_t,a_t)$ is

$$\begin{aligned} Q(s_t,a_t) =& (1-\alpha) * Q(s_t,a_t) + \alpha * (r_t + \\ & \gamma * max_a Q(s_{t+1},a)) \end{aligned} \tag{2}$$

where r_t is the reward in state s_t, $\alpha \in (0,1]$ is the learning rate that indicates to what extend the agent will learn new information, γ is discount factor denoting the importance of future rewards, $max_a Q(s_{t+1},a)$ is the estimate of the optimal future value.

B. Rule-based Shallow-trial

The ability to navigate in a crowded and dynamic environment is crucial for robots employed in indoor environment such as shopping malls, airports, schools, etc. When navigating in such environments, it is important for a robot to avoid obstacles and move towards its goal efficiently as human.

Safe and fastest navigation among people and various environments is often studied from the perspective of constructing the optimal path. Except planning to avoid collision with dynamic entities [17], recent studies started to model human's cognition and behavior. Mogan and Duguleana have developed a neural networks based on reinforcement learning for mobile robots obstacle avoidance [18].Kopp and Hemminghaus focus on adaptive social behavior generation for assistive robots using reinforcement learning [19]. Navarro and Kim have proposed a method of effective reward function in discernment behavior reinforcement learning based on categorization progress [20]. Furthermore, Yu and Chen have researched in reusable reinforcement learning via shallow trails [21]. It probes a task by running a roughly trained policy. Using rewards of the shallow trail, MAPLE automatically groups similar tasks. Moreover, when the task parameters are unknown, the rewards of the shallow trail also serve as task features.

Even though they often used models of human cognition and behavior, studies on navigation typically usually considered the problem of how the robot/agent collision can be avoided and pass through a crowd, but very few methods which based on shallow-trial rules to improve the ability to navigate efficiently through their environment has been studied, which is the topic of this research.

III. Rule-based Reinforcement learning For Robots Planing

In this section, we develop behavior rules for agents based on environmental characteristics. We use these behavior rules to conduct shallow-trial and initialize the state-action value function table, thus speeding up the convergence speed in the Q-learning iteration process. The RSRL algorithm is implemented.

The traditional Q-learning algorithm has no environmental priori information, and all the state value functions V(s) of the initial state are equal or completely random. Each action a is generated in a random state, that is to say, the transition probability of the MDP environment state is equal at this time. For an effective state-action value update, the randomness of behavior selection results in inefficient initial planning and more iterations. Especially for large-scale unknown environments, and it is easy to have huge invalid iteration search space.

In fact, in the process from the starting point to the target point, humans will combine the existing empirical knowledge with the information of the target point direction and environmental characteristics, and reach the target point depending on certain behavior rules. In the same way, when using reinforcement learning to plan the path for robots, the exploration behavior rule can be formulated with reference to human experience, and the behavior rule is used to perform shallow-trial and initialize the state-action value function table, by which unnecessary calculation cost can be reduced.

In the early stage of learning, RSRL can combine the existing behavior rules to improve the learning efficiency. When the behavior rules are cancelled, it can converge quickly in the later stage of learning. The learning processes are as follows:

1) Rules of conduct. After determining the starting point coordinate a and the target point coordinate b, according to the environmental characteristics and human experience knowledge, the appropriate behavior rules are formulated, so that the agent can avoid obstacles during the shallow-trial period and move toward the target point with a large probability, so that unnecessary exploration processes can be reduced.

2) Shallow-trial. At the beginning of the shallow-trial based on the set behavior rules, the agent gives a small positive return each step of the way, and gives a large positive return to the

end point, and updates the state-action value function table with the state-value function.

3) Cancel behavior rules. After the behavior rules are canceled, the Q-learning algorithm is used to iteratively update. In the learning process, the agent gives a small negative return every step of the way, reaches a larger positive return at the end point, updates the state-action value function table with the state-value function, ends the round iteration after reaching the target point. Subsequently, it enters the next round of learning starting from the starting point, so as to achieve rapid convergence.

The pseudo-code of the algorithm is shown in *Algorithm* 3. In this algorithm, the rules we formulate include the following points:

1) Try not to move backwards. Except for the only optional action in the current state, the action selected by the agent satisfies that next state s' of the agent is not equal to the last state s. The pseudo-code of this rule is shown in *Algorithm* 1.

Algorithm 1 Rule 1: Try Not to Move Backwards

```
Initialize: $'s$ (arbitrarily) and $s$.
for each state do
    $num\_obstacle =$ Number of obstacles in four directions.
    if $num\_obstacle == 3$ then
        $a =$ Direction without obstacle
    else
        for each nearest state $s'$ do
            if $s' \neq' s$ and $s' \neq obstacle$ then
                append $a$ to array $a\_enable$.
            end if
        end for
        $a =$ Random choose one action from a_enable
    end if
    Take action $a$, observe next state $s'$ and reward $r$
    Update the value function
    $'s = s,\ s = s'$
end for
```

2) Walking towards the end as far as possible. When there is no obstacle in four directions under the agent's state s, the distance between the state s and the target point is judged. When x > y, the agent moves in X direction towards the target point; when x < y, the agent moves in Y direction towards the target point; when x = y, the agent moves randomly in two directions towards the target point. The pseudo-code of this rule is shown in *Algorithm* 2.

Experiments show that better the behavioral rules are, less the number of rule-based shallow-trial required for algorithm convergence is. In fact, in a more complex environment, such as Fig.1(b), it is difficult to formulate appropriate behavior rules, to enable the agent to quickly reach the target point. Experiments show that, for such an environment, if more reasonable behavioral rules are adopted, and the rounds of the shallow-trial to train based on behavioral rules are appropriately increased, it can achieve the same effect.

IV. EXPERIMENTS

In this section, we determine the learning rate, the exploration rate, and the number of rule-based shallow-trial episodes

Algorithm 2 Rule 2: Try to Move Towards the Target Point

```
Initialize: $'s$ (arbitrarily) and $s$.
for each state do
    $num\_obstacle =$ Number of obstacles in four directions.
    if $num\_obstacle == 0$ then
        if $distance\_x > distance\_y$ then
            $a =$ Moving towards the target point in the direction of $x$
        else if $distance\_x < distance\_y$ then
            $a =$ Moving towards the target point in the direction of $y$
        else
            $a =$ Moving towards the target point in a random direction
        end if
    end if
    Take action $a$, observe next state $s'$ and the reward $r$.
    Update the value function.
    $'s = s,\ s = s'$
end for
```

Algorithm 3 Rule-Based Shallow-Trial Reinforcement Learning

```
Initialize: $Q(s,a)$ with null matrix
for each episode of shallow-trial do
    Initialize $s$
    for each step of episode do
        while true do
            Choose action $a$ under state $s$ using Rule 1 and Rule 2.
            Take action $a$, observe next state $s'$ and reward $r$.
            Update the value function
            $Q(s,a) = (1-\alpha)Q(s,a) + \alpha(r + \gamma \max_{a'} Q_i(s',a'))$
            $s = s'$
        end while until $s$ is terminal.
    end for
end for
Cancel rules.
for each episode do
    Initialize $s$
    for each step of episode do
        while true do
            Choose action $a$ under state $s$ using the policy derived from $Q$ values.
            Take action $a$, observe next state $s'$ and reward $r$.
            Update the value function
            $Q(s,a) = (1-\alpha)Q(s,a) + \alpha(r + \gamma \max_{a'} Q_i(s',a'))$
            $s = s'$
        end while until $s$ is terminal.
    end for
end for
```

on the Q-learning algorithm. At the same time, we verify that the shallow-trial training based on behavior rules can accelerate the convergence speed of Q-learning.

In the experiment, the rasterized map is used as the RSRL algorithm state-space S with a size of 22×22. The agent action space set A(s) includes four actions of moving up, down, left, and right. Takeing the lower left corner of the environment map as the coordinate origin, the horizontal direction is X axis, and the vertical direction is Y axis, so as to establish the coordinate system, while the starting point and the target point position are also set.

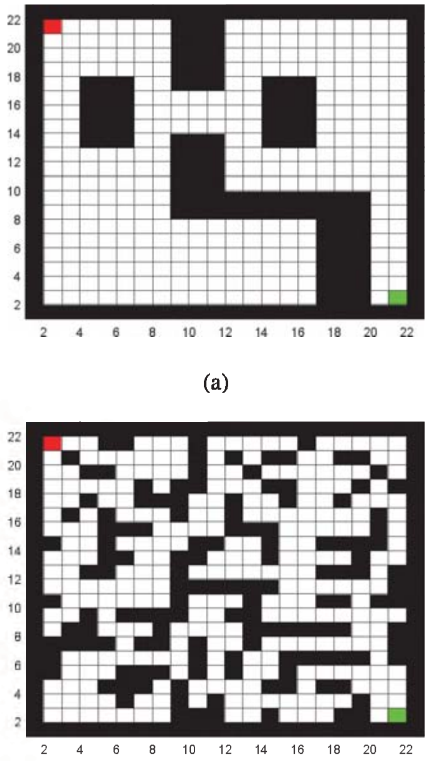
The number of training episodes based on behavioral rules requiring algorithm convergence depends on the quality of the rules, but in a more complex environment, it is difficult to formulate appropriate rules of behavior. In order to verify the effect of behavior rules on the number of shallow-trial episodes, we have established two types of maps as shown in Fig.1(a), (b). For map(a), its characteristics are clear. Using rule 1 and rule 2, it can make the agent reach the target point quickly. For map(b), its features are more complicated. It is difficult to make the agent reach the target point quickly by using only rule 1 and rule 2. In the map, the red dot represents the initial point, the initial coordinates (2, 21), and the green dot represents the target point, and the target coordinates are (21, 2). The correlation experiments were adjusted by using α, ε, and Rule-episodes(Ep_{rule}). Each experiment was performed 100 times, and the final return was averaged. The effects of various parameters on RSRL were compared by comparison.

In order to further test the performance of our algorithm, we provide more results about different learning parameters, including the learning rate, exploratory rate and shallow-trial episodes.

For learning rate, learning rate (map(a): $\alpha = 0.4 \sim 0.7$, map(b): $\alpha = 0.08 \sim 0.14$, exploration strategy (map(a): $\varepsilon = 0.01$, map(b): $\varepsilon = 0.01$), shallow-trial episodes (map(a) $= 40$, map(b) $= 150$) were used. As shown in Fig.2, the learning process tends to accelerate as alpha gradually increases from a smaller value to a larger value, and all those learning processes converge steadily. However, huge learning rate may inevitably make the learning process unstable or even divergent.

For exploration rate, learning rate (map (a): $\alpha = 0.6$, map (b):$\alpha = 0.12$), exploration strategy (map (a): $e = 0.01 \sim 0.2$, map (b): $e = 0.01 \sim 0.2$), shallow-trial episodes (map (a) $= 40$, map (b) $= 150$) were used. As shown in Fig.3, the probabilities of exploration affect the degree of agent breaking away from the rules. The higher the probabilities of exploration, the more difficult it is to borrow rules to iterate to the optimal value. Therefore, in this algorithm, after the shallow-trial episodes, it is not appropriate to adopt a larger exploration rate.

For the number of shallow-trial episodes, learning rate (map (a): $\alpha = 0.6$, map (b): $\alpha = 0.12$), exploration strategy (map (a): $\varepsilon = 0.01$, map (b): $\varepsilon = 0.01$), shallow-trial episodes (map (a) $= 20 \sim 50$, map (b) $= 50 \sim 200$) were adopted. As can be seen from Fig.4, for maps with relatively simple features, a small number of shallow-trial episodes can make the algorithm converge quickly. For maps with more complex features, under the same rules of behavior, in order to make the algorithm converge quickly, the number of shallow-trial episodes can be increased appropriately.



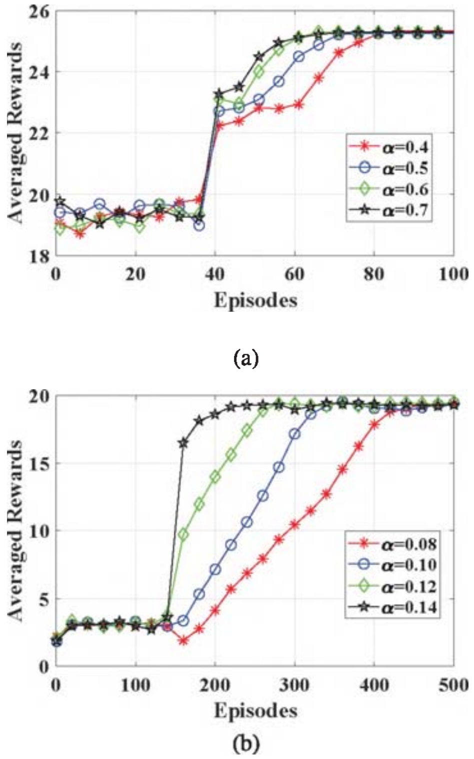
(a)

(b)

Fig. 1: Two types of maps.

In order to verify the effectiveness of the proposed scheme, two groups of comparative experiments were carried out in two different environments, using RSRL and Q-learning algorithm for comparative experiments, using the same γ, α and ε. Each method was trained 100 times, 600 episodes each time, and the final cumulative average return was obtained, as shown in Fig.5.

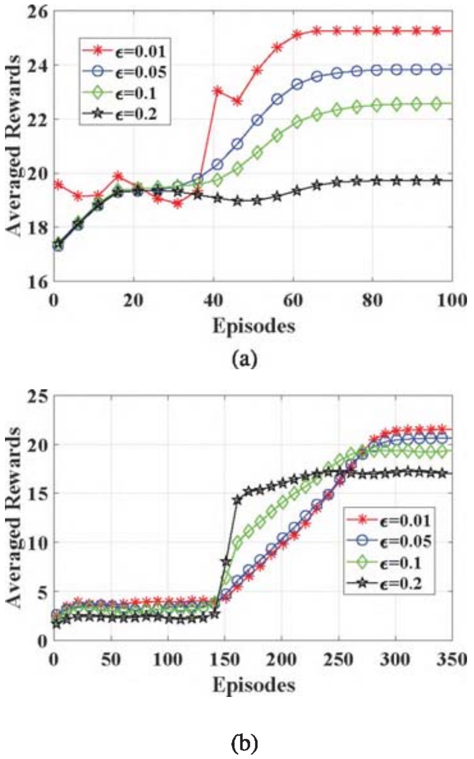
The validity of the shallow-trial training in accelerating Q-learning iteration speed is verified in map (a). Setting parameters such as table I, applying rule 1 and rule 2, and comparing Q-learning algorithm with the same parameters, the cumulative average return is shown in Fig.5(a), and the convergence speed of RSRL is faster than that of the traditional Q-learning algorithm. When the map is complex and difficult to formulate better rules, only rule 1 and rule 2 are applied to verify the effectiveness of increasing shallow-trial episodes appropriately to speed up Q-learning iteration. Setting parameters such as table I, and comparing with Q-learning algorithm with the same parameters, as shown in Fig.5(b), the results show that rule-based shallow-trial can be increased in



(a)

(b)

Fig. 2: Contrast of learning rate.



(a)

(b)

Fig. 3: Contrast of exploration rate.

the case of complex maps. with a faster Q-learning iteration speed.

TABLE I: Algorithmic parameters

parameters	Map (a)	Map (b)
α	0.6	0.12
ε	0.01	0.01
γ	0.95	0.95
Ep_{rule}	40	150
R_{goal}	1000	1000
$R_{rule-step}$	1	0.1
$R_{learning-step}$	-1	-1

Through simulation experiments, it has been found that the behavior rule-based shallow-trial training path is of faster iteration speed than those without rules, and after rule-episodes, fewer episodes are needed to reach the target position from the starting position. The behavior rule updating strategy reduces a lot of invalid explorations in the whole training process, and it improves the efficiency of the algorithm.

V. Conclusions

In this paper, we proposed a method of rule-based shallow-trial reinforcement learning on robots path planning. Firstly, we discussed the RL algorithm and the rule-based shallow-trial strategy. Then, according to the characteristics of the environment, we formulate some rules for agent, and combine Q-learning algorithm to achieve RSRL algorithm. It has been shown how to make use of prior knowledge to increase planning efficiency in the early stage of learning exploration, as well as the detailed process and algorithmic details of rule-based reinforcement learning on robots path planning.

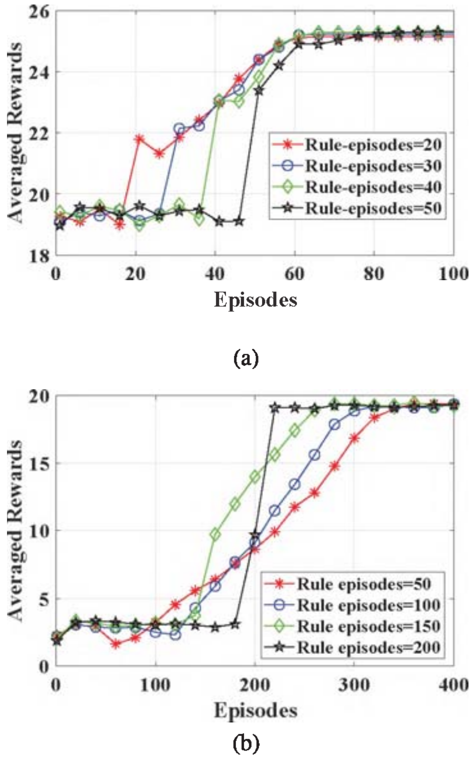
Finally, we carried out the experiment and obtained good results. It is found that the path planning with rule update is faster than that of the algorithm without rules, and after rule-episodes, the number of steps required to reach the target position from the starting position is reduced, and the average return is higher and faster. Based on the rule update strategy, robots greatly reduced the invalid exploration throughout the training process, and the efficiency of the algorithm has been improved.

Acknowledgment

This work was supported by the National Key Research and Development Program of China (No.2016YFD0702100) and the National Natural Science Foundation of China (Nos.61432008, 71732003, 71671086 and 61773208).

References

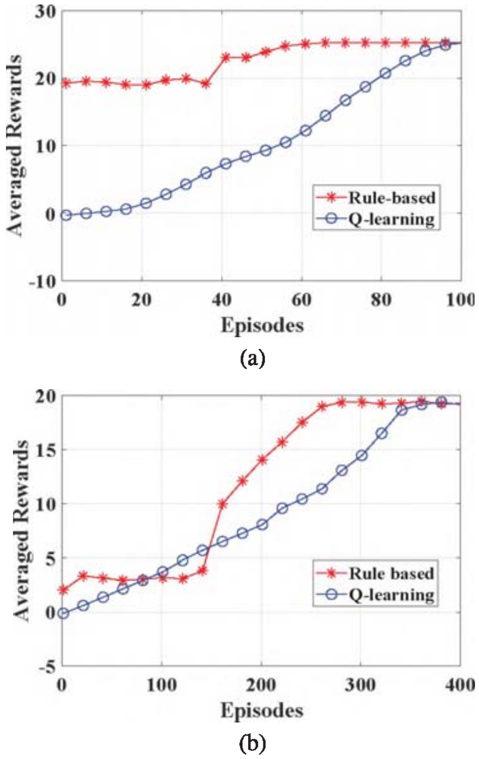
[1] Z. Zhi, H. Zhao, R. Swanson, and et al. Design, development, and evaluation of a noninvasive autonomous robot-mediated joint attention intervention system for young children with asd. *IEEE Transactions on Human-Machine Systems*, 48(2):125–135, 2018.



(a)

(b)

Fig. 4: The effect of rule episodes on convergence rate.



(a)

(b)

Fig. 5: The contrast of RSRL and Q-learning.

[2] E. Masehian and D. Sedighizadeh. Classic and heuristic approaches in robot motion planning-a chronological review. *Proceedings of World Academy of Science, Engineering and Technology*, 29(1):101C106, 2007.

[3] S. Ge, X. C. Lai, and A. A. Mamun. Sensor-based path planning for nonholonomic mobile robots subject to dynamic constraints. In *ROBOTICS AND AUTONOMOUS SYSTEMS.*, volume 55, pages 513–526, 2007.

[4] D. Tamilselvi, S. M. Shalinie, A. F. Thasneem, and et al. Optimal path selection for mobile robot navigation using genetic algorithm in an indoor environment. In *Advanced Computing, Networking and Security*, 2012.

[5] C. L. Chen, H. X. Li, and D. Y. Dong. Hybrid control for robot navigation - a hierarchical q-learning algorithm. In *IEEE Robotics and Automation Magazine*, volume 15, pages 37–47, 2008.

[6] L. W. Zhou, P. Yang, C. L. Chen, and Y. Gao. Robust learning control design for quantum unitary transformations. *IEEE Transactions on Cybernetics*, 47(5):1238–1250, 2017.

[7] J. Kober and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[8] R. Akrour, A. Abdolmaleki, H. Abdulsamad, and et al. Model-free trajectory optimization for reinforcement learning. 2016.

[9] C. Xie, S. Patil, T. Moldovan, and et al. Model-based reinforcement learning with parametrized physical models and optimism-driven exploration. *IEEE International Conference on Robotics and Automation.*

[10] K. Mehdi, V. George, T. Theodore, and et al. Robot fast adaptation to changes in human engagement during simulated dynamic social interaction with active exploration in parameterized reinforcement learning. In *IEEE Transactions on Cognitive and Developmental Systems*, pages 1–1, 2018.

[11] V. G. Santucci, G. Baldassarre, and M. Mirolli. Grail: a goal-discovering robotic architecturefor intrinsically-motivated learning. *IEEE Transactions on Cognitive and Developmental Systems*, pages 214–231, 2016.

[12] C. Z. Wu, B. Qi, C. L. Chen, and et al. Robust learning control design for quantum unitary transformations. *IEEE Transactions on Cybernetics*, 47(12):4405–4417, 2017.

[13] Z. P. Ren, D. Y. Dong, H. X. Li, and C. L. Chen. Self-paced prioritized curriculum learning with coverage penalty in deep reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems.*

[14] V. C. Meola, D. Caligiore, V. Sperati, and et al. Interplay of rhythmic and discrete manipulation movements during development: A policy-search reinforcement-learning robot model. *IEEE Transactions on Cognitive and Developmental Systems*, 8(3):152–170, 2016.

[15] B. Kim and J. Pineau. Socially adaptive path planning in human environments using inverse reinforcement learning. *International Journal of Social Robotics*, 8(1):51–66, 2016.

[16] M. Roozegar, M. J. Mahjoob, M. J. Esfandyari, and et al. Xcs-based reinforcement learning algorithm for motion planning of a spherical mobile robot. *Applied Intelligence*, 45(3):1–11, 2016.

[17] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. In *IEEE Robotics and Automation Magazine*, volume 4, pages 23–33, 1997.

[18] M. Duguleana and G. Mogan. Neural networks based reinforcement learning for mobile robots obstacle avoidance. *Expert Systems with Applications*, 62:104–115, 2016.

[19] J. Hemminghaus and S. Kopp. Towards adaptive social behavior generation for assistive robots using reinforcement learning. *Acm/ieee International Conference on Human-robot Interaction*, 2017.

[20] C. H. Kim, Y. Kon, R. Navarro, and et al. Effective reward function in discernment behavior reinforcement learning based on categorization progress. *IEEE-RAS International Conference on Humanoid Robots*, 2017.

[21] Y. Yu, S. Y. Chen, Q. Da, and et al. Reusable reinforcement learning via shallow trails. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–12, 2018.