Reinforcement Learning of Cooperative Behaviors for Multi-Robot Tracking of Multiple Moving Targets

Zheng Liu
Electrical & Computer Engineering
National University of Singapore
10 Kent Ridge Crescent
Singapore 119260
zhengliu@nus.edu.sg

Marcelo H. Ang Jr.
Mechanical Engineering
National University of Singapore
10 Kent Ridge Crescent
Singapore 119260
mpeangh@nus.edu.sg

Winston Khoon Guan Seah
Institute for Infocomm Research
Agency for Science Technology and Research
21 Heng Mui Keng Terrace
Singapore 119613
winston@i2r.a-star.edu.sg

Abstract— **Traditional reinforcement learning algorithms learn based on discrete/finite states and actions, thus limit the learned behaviors to discrete/finite space. To address this problem, this paper introduces a distributed reinforcement learning controller that integrates reinforcement learning with behavior based control networks. This learning controller can enable the robot to generate appropriate control policy which combines different elementary behaviors. In addition, to address the problems in concurrent learning, a distributed learning control algorithm is proposed to coordinate concurrent learning processes. The distributed reinforcement learning controller and learning control algorithm are applied to multi-robot tracking of multiple moving targets. The efficacy is demonstrated by simulations.**

Index Terms— **Reinforcement learning; concurrent learning; behavior based control; multi-robot cooperation.**

I. Introduction

For multi-robot systems, one of the key research problems is to achieve cooperation among robots by decentralized (distributed) control methodology [1]. Normally, the desired cooperation is in the task level [2], which means that the common mission is broken down into tasks, and robots choose different tasks (roles) according to the state and behave differently. However, the design for the task level controller is quite difficult. Therefore, in recent years, machine learning techniques have been proposed and studied for multi-robot systems that aim to enable the robots to learn how to cooperate without the need for human design or coding.

In last two decades, reinforcement learning has been extensively studied for multi-robot concurrent learning of cooperative behaviors. However, traditional reinforcement learning assumes discrete and finite state/action spaces; therefore it can hardly be applied to most real applications that inherently involve with continuous and infinite space. Furthermore, even the states and actions can be discretized and defined; the learned elementary behaviors are still discrete. At one time, the robot can only perform one elementary behavior. This contradicts the human reasoning that usually the optimal behavior is the execution of several elementary behaviors to accomplish a task. In addition, the switching of discrete behaviors usually results in the control of the robots becoming unsmooth, which is undesirable in most cases.

To address this problem, some methods are proposed to enable reinforcement learning in continuous space. For example, function approximation approach [3] and HEDGER [4] can apply a generalizing function approximator to estimate the state-action value instead of using discrete lookup table. However, these approaches usually assume the environment model is known, and have heavy computational burden if the training data set is large.

For multi-robot concurrent learning, the assumptions for single agent/robot learning may not still be valid. Reinforcement learning and most other machine learning algorithms assume the learning process is Markovian and the learning environment is stationary [5]. However, if the robots learn concurrently, the distributed learning processes will interfere with each other. Then, during multi-robot concurrent learning, in the view of an individual robot, the process and environment are neither Markovian nor stationary. This might lead to the undesired learning results as sub-optimal local control policy or the cyclic switching of control policies.

One class of solutions to address this problem is to estimate the influence of other robots, and therefore make the process semi-Markovian and pseudo-stationary for an individual learning robot [6]. Another class of solutions is to coordinate or schedule the distributed learning processes to reduce the interference [7]–[9]. However, the coordination and scheduling of learning processes usually have to be deliberatively designed and require explicit intercommunications among the robots. This degrades the applicability of the learning coordination or scheduling algorithms.

In this paper, a reinforcement learning controller integrating reinforcement learning and behavior based control networks is proposed to address the limitation of traditional reinforcement learning of discrete and finite behaviors. In addition, a distributed learning coordination algorithm is introduced to solve the problems in multi-robot concurrent learning. The learning controller and learning control algorithm are applied to multi-robot concurrent tracking of multiple moving targets.

This paper is organized as follows. Section 2 presents the concept of our approach. Then, Sections 3 introduces the implementation of the proposed learning in the multi-robot tracking of multiple moving targets. Section 4 shows the simulation results and discussion. Finally, Section 5 concludes this paper and introduces the future work.

II. REINFORCEMENT LEARNING OF CONTINUOUS BEHAVIORS WITH COORDINATION

In this paper, we aim to address two problems of multi-robot concurrent learning of cooperative behaviors: 1) to generate optimal combination of elementary behaviors for cooperation based on low level input states and output actions; and 2) to coordinate concurrent learning process by distributed methodologies. Specifically, the aims are:

- Enable the robot to learn based on low level input and output without the need for deliberate definition of high level states and actions.
- Let the robot learn based on existing behavior based controller - utilizing human knowledge of robot, mission, and environment.
- Let the robot learn cooperative behaviors that combine elementary behaviors.
- Coordinate concurrent learning processes to generate optimal control policy. In other words, increase the probability that the robots learn cooperative behaviors.
- Minimize the requirement for inter-robot communications to coordinate concurrent learning processes.

With regards to discrete and finite space limitation, we propose the integration of reinforcement learning with behavior based control networks. The architecture of this learning controller is shown in Figure 1. It has two main parts, the behavior based control network module and reinforcement learning module.

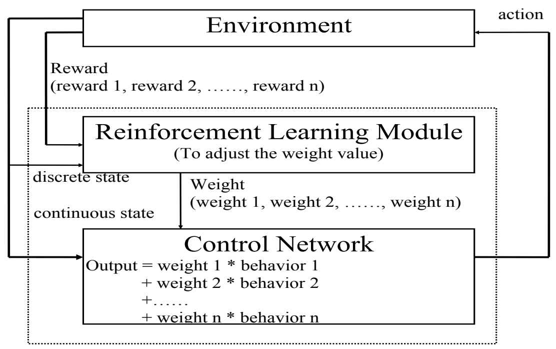
Environment
action
Reward
(reward 1, reward 2, ……, reward n)
Reinforcement Learning Module
(To adjust the weight value)
discrete state
continuous state
Weight
(weight 1, weight 2, ……, weight n)
Control Network
Output = weight 1 * behavior 1
+ weight 2 * behavior 2
+……
+ weight n * behavior n

Fig. 1. Architecture of Proposed Learning Controller

The behavior based control network is created according to human knowledge of the robot, environment, and the mission. In this network, the elementary behaviors are represented by control rules and equations. Each elementary behavior can retrieve input signal and generate corresponding output command both in continuous and infinite space. The overall output of the controller is the summation of weighted outputs of all elementary behaviors. In this behavior based control network, the weight is the key to combine different elementary behaviors: if the weight of one behavior is large, the robot is more likely to perform this behavior; otherwise the robot is reluctant to this behavior.

The reinforcement learning module is the key of the controller. It is integrated with the control networks, the aim is to adjust the weight inside the control network; therefore affect the combination of elementary behaviors. This is the key to generate optimal combination of behaviors. By retrieving states and rewards, the learning module can gradually find the appropriate weight value for each elementary behavior. It should be noted that the reinforcement learning module needs to retrieve rewards corresponding to each behavior; otherwise the learning module cannot estimate the performance or results of taking the behavior. For example, regarding behavior “avoid obstacles”, if the performance is unsatisfactory, a negative reward should be given to indicate that the weight of “avoid obstacles” needs to be adjusted. Regarding the learning, the main research issues include state/action definition, reward generation, state-action value update, and action selection. Since these issues are related to the robot, mission, and environment, we will elaborate them in the next section when introducing the implementation of this learning controller in multi-robot tracking of multiple moving targets.

In general, the proposed learning controller has following properties:

- The behavior-based control network is designed based on human experience. The control rules or equations are the representation of elementary behaviors.
- For the control network, the overall output behavior is the summation of weighted elementary behaviors. The output control command is in continuous and infinite space.
- The aim of the reinforcement learning module is to adjust the weight in the control network to achieve optimal combination of elementary behaviors. In other words, the output is not the “selection” of exclusive discrete behavior, but the way to “combine” them to generate optimal behaviors.
- While the reinforcement learning module still works in discrete and finite space. In the macro view, the learning controller works in continuous space in that it can learn based on low level inputs and outputs (continuous and infinite).

In addition to the continuous behavior space problem, another research issue is coordinating concurrent learning processes. To address this problem, we propose a solution inspired by natural human behaviors. Assuming two humans are approaching the same corridor and they want to avoid collision, if both of them are trying, they may “struggle” several rounds to pass. So, in real life, usually one of them (say A) fixes the policy first, e.g., keeping left, then the other one (say B) can choose another side. In this case, the cooperation is the behavior that the two people choose opposite sides. Whatever A chooses initially, if B can learn to correspond to A's action, the resultant control policy is optimal. Many real world applications have the same property: even if the learning process of one robot/agent stops very early, the resultant control policy of the whole system can still be optimal because other learning robots can eventually find appropriate control policy to respond to the former one. Our distributed learning coordination algorithm is proposed based on this consideration. For a robot, if in one state, the best action's value is much larger than other actions, the robot will stop learning in this state and after that it will always choose this best action for this state. In other words, a robot will fix its control policy when it feels that it has learned enough; and the future improvement of the group performance is left to other learning robots.

III. LEARNING IN MULTI-ROBOT TRACKING OF MULTIPLE MOVING TARGETS

A. Multi-Robot Tracking of Multiple Moving Targets

In robotics research, multi-robot tracking of multiple moving targets is also referred to as the "museum problem" or "art gallery problem", which aims to find the best control solution of a group of robots to maximize the number of targets being simultaneously observed [10], [11]. In our work,

- The number, distribution and motion pattern of the moving targets are unknown.
- The size and map of the environment are unknown.
- The robots cannot localize themselves in the environment. The robots have no intercommunications (e.g., no wireless communications).

In current research for the museum problem, Artificial Potential Field (APF) control is mostly used. The concept of APF is simple: map the targets as sources of attractive force, and map the robots and obstacles as sources of repulsive force. Then, let the robot move under the vector sum of the attractive and repulsive forces. However, pure APF (purely summing the attractive and repulsive forces) may not achieve desired cooperation in most cases. For example, if two robots detect a same target, both of them will track this target and therefore they will form a triangular pattern. This is not the optimal cooperation; the robot force is wasted because one of the robots can leave and search for other targets to maximize the number of observed targets.

A solution to avoid the triangular pattern of pure APF is giving a weight $W_{R_i}^{Target}$ to the attractive force for each robot as shown in (1). In this equation, $\vec{F}_{R_i}$ means the summation of the attractive and repulsive forces for R_i; $\vec{T}_{R_i,T_j}$ means the attractive force to target T_j; $\vec{R}_{R_i,R_l}$ means the repulsive force from neighbor robot R_l; dt means the set of detected targets; dr means the set of the detected neighbor robots.

$$\vec{F}_{R_i} = \sum_{j \in dt} w_{R_i}^{Target} \cdot \vec{T}_{R_i,T_j} + \sum_{l \in dr} \vec{R}_{R_i,R_l} \qquad (1)$$

Examining (1), it should be noted that if the weight of attractive force is zero, the robot will only have one behavior to avoid neighboring robots; if the weight of attractive force is infinity, the robot will only have one behavior to track targets. Changing the value of the weight means changing the preference to the two behaviors "tracking target" and "leaving neighboring robots". In previous research, two classes of algorithms are proposed to adjust the weight. One is the all-adjust heuristic [10] that lets one robot decrease the weight when it finds another robot is also tracking the same target. The other solution is selective-adjust heuristic [11], whereby only the further robot(s) decreases the weight. These two heuristics of pure potential field based control are proved effective; however, to make them work, the designer needs to carefully select appropriate weight decrease ratio for each robot. This is extremely difficult when the scenario is complex and the robot team is heterogeneous. A natural modification is to find optimal weight value through learning. Hence the museum problem is well suited to the implementation of our learning controller.

B. Applying Our Learning Controller in Museum Problem

As introduced previously, the main research issues for the reinforcement learning module include state/action definition, reward generation, state-action value update, and action selection.

For museum problem, one robot may meet many situations. To make the learning simple, yet not lose generality, we define the input state as the number of targets and robots detected. For example, if two targets and one robot neighbor are detected, the state is (*2, 1*). The output of the learning module (action) is the weight of the attractive force. In this approach, we have three kinds of weight: small, mid, and large.

For reinforcement learning, one important issue is the generation of rewards. To enable the robots to learn cooperative behaviors, the following behaviors should be encouraged: 1) track target; 2) leave the target being tracked by other robots. For this purpose, four kinds of rewards are defined as follows (rewards are generated by the robot's local sensing):

- *Reward_TT*: track target reward (positive) - if the robot tracks targets.
- *Reward_NR*: near robot reward (negative) - if the robots detect other robots.
- *Reward_SC*: state change reward (positive or negative) - if the new state has less neighbor and more targets, the reward is positive, otherwise negative.
- *Reward_WT*: waste time reward (negative) - if the robot tracks a target being tracked by others.

For reinforcement learning, the learning process needs to update the state-action value $Q(s,a)$ based on the reward received. In our approach, this value is updated by the following Q-function (2), in which s, a, r, α, γ, s', and a' means state, action, reward, learning rate, discount rate, next state, and next action, respectively [12].

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma max_{a'} Q(s',a') - Q(s,a)) \qquad (2)$$

Every time state changes, the robot will reselect the action (weight). Furthermore, if the state is unchanged for a long period of time, i.e., N simulation steps, the robot also reselects action (weight) to accelerate learning speed. When selecting an action, the robot both explores and exploits the action space: an exploration factor is added to the real state-action value, and then the action having highest resultant value will be chosen. This exploration factor is just for action selection, and will not affect the state-action value.

Regarding the coordination of distributed learning processes, we propose a distributed algorithm to coordinate learning processes: for a robot, if for one state, the best action's value is much larger then other actions, the robot will stop learning for this state and it will always choose this action in future. By this method, the concurrent learning is more likely to generate cooperative behaviors. It should be noted that this distributed learning coordination algorithm does not require the robots to communicate to share any information.

As a summary, the learning module in the proposed controller has three main difference to the traditional reinforcement learning process [12]:

- The reinforcement learning is integrated with control networks.
- During the learning process, the update of state-action value may take place if the state does not change for a long time
- During the learning process, the robot may stop learning when it "feels" it has learned enough for a given state.

IV. SIMULATION AND DISCUSSION

A. Simulation Methodology

In this paper, we aim to let the mobile robots learn through the interaction with environment and other robots, hence generating appropriate behaviors to cooperate. This aim has three aspects:

- The learning approach can generate cooperative behaviors.
- The performance of the learning system should be comparable to other approaches that have been deliberatively hardcoded and tuned.
- The learning controller with the proposed coordination algorithm should achieve better performance than the controller without coordination.

To justify the efficacy of our approach, we simulate four control modes as follows: 1) Pure Artificial Potential Field (APF) based control; 2) All-adjust heuristics to pure APF; 3) Selective-adjust heuristics to pure APF; and 4) Robot learning controller: with and without coordination.

These control modes are tested in both homogeneous and heterogeneous robot group. "Homogeneous" means the robots are identical; "heterogeneous" means one of the robots is 30% faster than the other. Regarding the learning controller, "coordination" means the distributed learning coordination algorithm proposed by us.

B. Simulation Settings

The parameters and settings of the environment are as follows:

- Simulator: Webots.
- Environment: 4m x 4m square plain area; robots/targets: 0.1m in diameter.
- For each control mode, run about 30 episodes to get the average. Each episode has 20000 simulation steps. One simulation step is about 0.1s long in real time.

For the all-adjust heuristics of pure potential field based control, in the simulation, we test two all weight decrease ratio ($AWDR$): 0.95 and 0.80. For the selective-adjust heuristics, we test two selective weight decrease ratio ($SWDR$): 0.1 and 0.5.

The settings of the learning controller are as follows:

- The initial value of all state-action is 10.
- *Reward_TT* = 0.005 * track target time.
- *Reward_NR* = - 0.01 * near robot time.
- *Reward_SC* = (m-a)*0.5 - (n-b) * 2.0 (m, n are the current target/robot number; a, b are the previous target/robot number).
- *Reward_WT* = - 0.1 * waste time.
- For learning "with coordination", under one state, if one action's value is 25% above the average, stop learning in this state. "Without coordination" means never stop learning for any state.
- During learning, if the state is unchanged for N = 100 simulation steps, the robot reselects the action.
- When selecting action, a number uniformly distributed in [-1, 1] is added to the real state-action value as the exploration factor.

C. Simulation Results and Discussion

1) Simplest Scenario: One target and two robots.

We evaluate the performance of the controller by following three metrics:

- Learned weight difference - how much is the difference in learned weight between two robots.
- Track target length - percentage of the time that the target is tracked.
- Waste time length - percentage of the time that 2 robots both track 1 target.

For the museum problem, to maximize the number of observed targets, the robot force needs to be fully utilized that the robots should both track detected targets and try to search for targets. In our learning controller, to achieve this kind of cooperation, the two robots should learn different weights for tracking target when they detect one target and one robot neighbor, i.e., in state (*1, 1*).

The simulation results show that the concurrent learning processes with and without coordination will generate quite different learning results. Figure 2 shows the "learned weight difference" for homogeneous and heterogeneous robot groups in state (*1, 1*) without (left) and with (right) the proposed learning coordination algorithm. The x-axis represents the difference level, and the y-axis represents the frequency (% of trials) that the two robots result in such difference. The results show that without coordination, the robots sometimes learn the same weights, while with coordination, none of the robot pair results in the same weights. This difference in weight is the key to generate desired cooperation: the robot with large weight will keep tracking the target; the robot with small weight will tend to leave. Obviously, the learning with coordination performs better than the learning without coordination. In the following parts of this section, all the results related to learning are for learning with coordination.

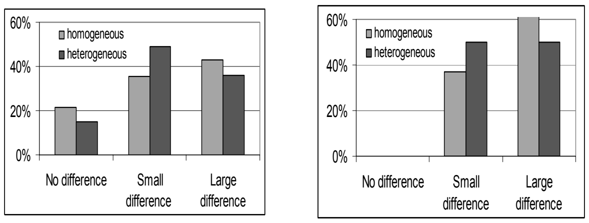

Fig. 2. Learned Weight Difference - Without Coordination (left), With Coordination (right)

Observing Figure 2, we also find that in both homogeneous and heterogeneous groups, the learning results in various levels of weight difference; but not a fixed one. This may be due to the fact that there is no exact "optimal" solution for this kind of mission and environment, e.g., different start points or motion patterns of the targets may have different optimal tracking policies. Another possible reason is that the partial observation of the environment and the interference among concurrent learning robots make the environment non-Markovian and dynamic for the learning robot. Even the coordination of learning can "encourage" the robots to cooperate; however, sometimes the robots cannot avoid the selfish behavior because the "coordination" of learning process is only by the local sensing of the robots.

Another notable result is that the learned weight difference in homogeneous and heterogeneous robot groups is different. This reflects the adaptation capability of our learning controller: in homogeneous robot team, all robots are exactly the same, therefore to generate cooperative behaviors (one robot keeps tracking and the other robot leaves), a large weight difference is required; while in heterogeneous team, one of the robots is 30% faster than others, therefore not as large weight of attractive force (compared to the homogeneous robot case) can enable the faster robot to keep tracking the target and force the slower robot to leave the tracked target. Also, simulation shows that in heterogeneous robot group, the faster robot is more likely to become the tracker who will keep tracking (60% chance); while in homogeneous group, the two robots have almost the same chance to become the tracker (46.7%). The reason may be that the faster robot is more competitive so that it has more chance to win in the struggle for tracking and thus it is reinforced to be the tracker.

Now we compare our learning approach with other approaches. Simulation results show that for all four control modes, the track target duration is almost the full duration of the simulation episode: whatever the control mode is, the track target duration is around 98% of the total time. This is because there is only one target; once this target is tracked, it will be continuously tracked, by one or both robots. In this case, the waste time duration becomes an important metric for evaluating the performance of the control algorithms. The shorter the waste time duration is, the better the cooperation between robots is. Figure 3 shows the waste time duration under different control modes. We may find that the pure potential field based control is unacceptable. The all-adjust and selective-adjust heuristics can greatly shorten the waste time duration. However, different weight decrease ratios ($AWDR$ and $SWDR$) results in different performance. If the parameter is appropriate, the outcome is satisfactory; otherwise the outcome may be unacceptable. However, by learning, we do not need to decide such parameters. Instead, the robots can automatically generate desired cooperation by learning. Simulation results show that the outcome of our learning approach is as good as the two heuristics; and even slightly better in the heterogeneous robot group.

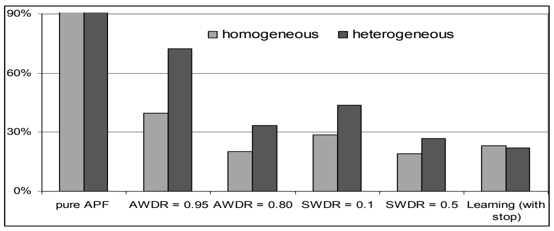

Fig. 3. Waste Time Duration (percentage of total time)

2) Extended Scenario: More robots and targets.

The settings in the extended scenario are the same as the simplest scenario except for the following:

- Environment: 6m x 6m.
- For the all-adjust heuristic, $AWDR$ = 0.8; For the selective-adjust heuristic, $SWDR$ = 0.5.
- For the learning controller, only the "with coordination" mode is tested.

In the extended simulation scenario, we evaluate and compare the performance of the controllers by the following two metrics:

- Average number of tracked targets.
- Average number of busy robots - in average, how many robots are used for tracking targets. The lower this number is, the better the performance.

We keep the target number as three, and then choose different simulation scenario including three, six, nine, and twelve robots, respectively. Simulation results are shown in Figures 4 and 5.

Since the number of robots is no less than the number of targets (3:3, 6:3, 9:3, 12:3), in the simulation, almost all the targets are being tracked and therefore the average number

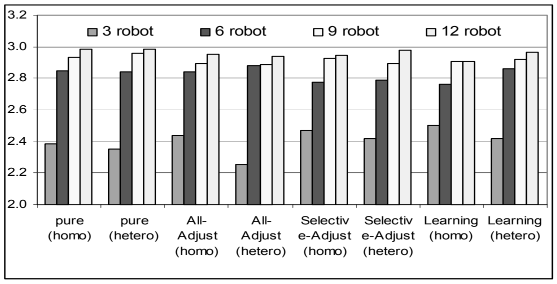

Fig. 4. Average Number of Tracked Target

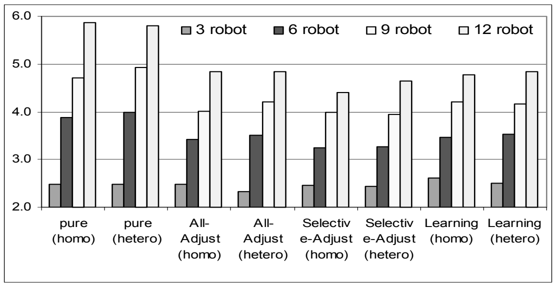

Fig. 5. Average Number of Busy Robot

of tracked target is quite similar in the four control modes (Figure 4). In this case, the average number of busy robot becomes an important metric to evaluate the performance. As shown in Figure 5, pure potential field based control requires more robots than the two heuristics and the learning controller. The outcome of our learning controller is as good as the all-adjust heuristic and selective-adjust heuristic.

Observing the simulation results, a question may arise that why the performance of the learning control mode is sometimes worse than the selective-adjust weight heuristic of the pure potential field based control. This happens in both the simplest scenario and the extend scenario. A reasonable explanations is that the selective-adjust weight heuristic [11] is theoretically more optimal than the learning controller because it also concerns the distance between robots and targets.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a distributed learning controller that integrates reinforcement learning with behavior based control networks. This controller can enable the robot to generate cooperative behaviors in continuous space. We also propose a distributed learning control algorithm to coordinate the concurrent learning processes. This algorithm can help eliminate the interference among the learning robots without explicit intercommunications. This learning approach is tested in multi-robot tracking of multiple moving targets. The efficacy is proved by simulation results.

However, in our learning controller, the reinforcement learning module still needs to retrieve discrete input state (target/robot number) and perform discrete actions (weights). A more challenging work is to design a totally continuous learning algorithm, or at least, let the robot do state/action discretization by itself through learning. This is an important research issue to be studied.

Another problem of the learning controller is that the behavior based control network coded by us is specific for the tracking task. It will be much better if the behavior based control network in our learning controller can be generic and effective for all kinds of control problem. This is another important research issue to be studied.

In addition, due to the interference among the concurrent learning robots, the distributed learning controller sometimes generates unsatisfying results even though we have applied a distributed learning coordination algorithm. How to perfectly coordinate concurrent learning processes by minimal intercommunications is still a critical research topic for future research.

REFERENCES

[1] Y. U. Cao, A. S. Fukunaga, A. B. Kahng, and F. Meng. “Cooperative mobile robotics: antecedents and directions”, in proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol:1, pp:226-234. 1995.
[2] P. Tangamchit, J. M. Dolan, and P. K. Khosla. “The necessity of average rewards in cooperative multirobot learning”, in proceedings of IEEE International Conference on Robotics and Automation. 2002.
[3] J. A. Boyan, and A. W. Moore. “Generalization in reinforcement learning: safely approximating the value function”, in Advances in Neural Information Processing Systems 7. 1995.
[4] W. D. Smart, and L. P. Kaelbling. “Practical reinforcement learning in continuous spaces”, in proceedings of the 7th International Conference on Machine Learning. 2000.
[5] L. P. Kaelbling, M. L. Littman, and A. W. Moore. “Reinforcement learning: a survey”, in Artificial Intelligence Research, Vol: 4, pp237-285. 1996.
[6] K. I. Kawakami, K. Ohkura, and K. Ueda. “Adaptive role development in a homogeneous connected robot group”, in proceedings of IEEE International Conference on Systems, Man, and Cybernetics, Vol:3, pp:254-256. 1999.
[7] E. Uchibe, M. Nakamura, and M. Asada. “Co-evolution for cooperative behavior acquisition in a multiple mobile robot environment”, in proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Victoria, B. C., Canada. 1998.
[8] M. Asada, E. Uchibe, and K. Hosoda. “Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development”, Artificial Intelligence, Vol.110, pp.275-292. 1999.
[9] S. Ikenoue, M. Asada, and K. Hosoda. “Cooperative behavior acquisition by asynchronous policy renewal that enables simultaneous learning in multiagent environment”, in proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.2728-2734. 2002.
[10] L. E. Parker. “Distributed algorithm for multi-robot observation of multiple moving targets”, Autonomous Robots, vol. 12(3). 2002.
[11] Z. Liu, M. H. Ang, and W. K. G. Seah. “Searching and tracking for multi-robot observation of moving targets”, in proceedings of the 8th Conference on Intelligent Autonomous Systems. 2004.
[12] R. S. Sutton, and A. G. Barto. Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA. 1998.