Multi-Robot Concurrent Learning in Museum Problem

Zheng LIU[1], Marcelo H. ANG Jr.[2], and Winston Khoon Guan SEAH[3]

[1]Dept. of Electrical & Computer Engineering, National University of Singapore. g0202255@nus.edu.sg. [2]Dept. of Mechanical Engineering, National University of Singapore. mpeangh@nus.edu.sg. [3]Institute for Infocomm Research, Singapore. winston@i2r.a-star.edu.sg.

Abstract:
Multi-robot concurrent learning on how to cooperatively work through the interaction with the environment is one of the ultimate goals in robotics and artificial intelligence research. In this paper, we introduce a distributed multi-robot learning algorithm that integrates reinforcement learning and neural networks (weighting network). By retrieving continuous environment state and implicit feedback (reward), the robots can generate appropriate behaviors without deliberative hard coding. We test the learning algorithm in the "museum" problem, in which robots collaboratively track moving targets. Simulation results demonstrate the efficacy of our learning algorithms.

Key words:
Multi-robot, reinforcement learning, neural networks, concurrent learning, tracking.

1. Introduction

The multi-robot system has been one of the focuses of robotics research in the last two decades. It includes a wide range of research topics such as multi-robot cooperative transportation, exploration and mapping, distributed sensing, robot soccer, etc [1]. The multi-robot system is not simply an extension of the single-robot system by increasing the performance owing to parallel operation; it can accomplish tasks impossible to a single-robot system through "cooperation" [2].

Normally, the cooperation in multi-robot systems is concentrated on the task level [3], whereby the mission is broken down into tasks, and robots choose different tasks (roles) according to the state and behave differently. To achieve mission decomposition, task allocation, and conflict coordination, the designer needs to predict all possible scenarios and preset corresponding actions for each robot to react accordingly. Such development and coding work is undesirable and sometimes ex-

tremely difficult, especially when the mission is very complex and the robot group is heterogeneous. One possible solution is to let the robot learn how to cooperatively work through the interaction with the environment and other robots, hence generating appropriate behaviors without human design or coding.

In this paper, we introduce typical reinforcement learning and its constraints in Section 2, and present our learning algorithms that integrate reinforcement learning and neural networks in Section 3. Following which, we introduce how to implement our learning algorithms for the museum problem in Section 4 and show the simulation and results in Section 5. Finally, Section 6 concludes this paper.

2. Reinforcement Learning

Emergent generation of multi-robot cooperation is one of the ultimate goals of robotics and artificial intelligence research. While there are dozens of basic learning algorithms in machine learning research [4], only reinforcement learning (RL) is extensively studied for behavior based control (cooperation in the task level) in multi-robot systems [5]. An explanation is that compared with other learning algorithms, reinforcement learning has the following advantages [6]:

- Model free – can learn the control policy even if the model of the environment is unknown.
- Not strictly supervised – no need for explicit human training, implicit reward is sufficient.
- Optimal – subject to user defined criteria.
- Practical – simple and real-time.

The basic concept of reinforcement learning is to find the optimal control policy that chooses the appropriate action under any given state; in other words, to find the optimal link/mapping from states to actions. Usually, reinforcement learning can sufficingly solve the control problems that execute in discrete state/action space. For instance, for path planning, if the map is divided into grids, the agent/robot can find the path to approach to the destination by reinforcement learning.

Because of above advantages, reinforcement learning is predominant in both single- and multi-robot system researches. However, the limitation of discrete/finite input (state) and output (action) constrains the application of reinforcement learning. For example, to implement reinforcement learning in robot behavior based control, the designer must define discrete/finite state and action space first (as in [7]). Obviously, this is not realistic because some tasks and missions can hardly be discretized. Furthermore, even if the state and action spaces are discrete, the huge size of the space will badly affect the learning process, which requires clustering (grouping) to reduce the space. Reasonably, one important question arises: can the robot do the state/action discretization and clustering by itself without human in-

tervention, or even more, perform reinforcement learning without discretization or clustering? In this paper, we address this problem by integrating reinforcement learning with neural networks (weighting network).

Besides the limitation of finite and discrete input (state) and output (action), other critical research issues of the reinforcement learning include the Markov Decision Process (MDP) and stationary environment assumptions, reward definition and assignment, state-action link value update, and action selection. However, they are not the main topic of this paper. Related work on these issues can be found in [8-11].

3. Our Learning Algorithms

As introduced before, the aim of our research is to address the problem of discrete and finite input/output space in the reinforcement learning. A reasonable solution is to modify the architecture of reinforcement learning or integrate it with some control algorithms that can deal with continuous and infinite input/output space.

In robotics and artificial intelligence research, the neural network (NN) is a well known control and learning algorithm that has been extensively studied for decades. Neural networks can deal with continuous and infinite input/output spaces; however, the learning in neural networks is normally supervised. As shown in Figure 1, in a typical Back Propagation (BP) neural networks, a "trainer" is needed to generate desired output according to the input, and then some algorithms are used to adjust the parameters/weights inside the neural networks by the error between the desired output and real output of the controller.

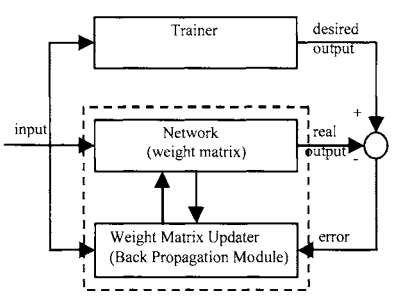

Figure 1. Back Propagation Neural Networks

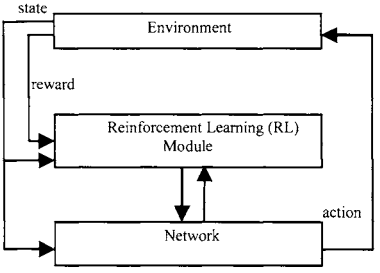

Figure 2. Our Learning Architecture

The design of our learning architecture is shown in Figure 2. In this new architecture, the back propagation module in the BP neural networks is replaced by the reinforcement module, and no trainer is needed to generate the explicit desired output. Instead, the reinforcement learning module adjusts the weight inside the network through the interaction with the environment. By integrating reinforcement learning with neural networks (weighting network), we combine the advan-

tages of both neural networks and reinforcement learning. The neural networks (without the BP module) can retrieve continuous and infinite input, and then generates continuous and infinite output by the weight matrix (weighting network). On the other hand, the reinforcement module which replaces the BP module can find the "best" weight inside the networks through the interaction with the environment. It should be noted that the input/output spaces of the reinforcement learning module are still discrete and finite. However, since the learning module is only used to adjust the weight inside the networks, the input and the output of the overall controller/system is now continuous and infinite.

Because the learning architecture is a framework of a learning methodology, and the learning algorithms are highly coupled with the mission, we need to test our learning algorithms in real applications. For this purpose, museum problem is a good choice. We will introduce the museum problem and show how to implement our learning algorithms in the museum problem in the next section.

4. Learning in Museum Problem

4.1. Museum Problem

Museum problem is the research on multi-robot tracking of multiple moving targets. The assumptions and descriptions of the museum problem are as follows:

- The environment is a large bounded plain area.
- Several targets move in the environment.
- Several mobile robots are in the environment. Each robot has a 360 degree view within a certain range. When an object is inside this circle, the robot can differentiate it as obstacle, target, or robot. The summation of the sensible area of all robots is far less then the size of the environment.
- The targets are mobile and the sensor range of the robot is limited, hence the robot needs to track targets to maintain observation.
- For the robots, the number and motion pattern of the targets are unknown. Localization and intercommunication are unavailable.
- The objective is to maximize the number of targets being observed.

In current research for the museum problem, Artificial Potential Field (APF) based control is mostly used. The concept of APF is very simple: map the targets as attractive force sources and map the robots and obstacles as repulsive force sources; then, let the robot move under the vector sum of the attractive and repulsive forces. Artificial potential field based control can be seen as a kind of competitive neural networks (weighting network) in which the attractive forces compete with the repulsive forces. It is simple and can be used in real-time applications.

The pure potential field based control just adds the repulsive and attractive forces. However, purely summing the repulsive and attractive forces may not achieve desired cooperation in most cases. When two robots find the same target, intuitively, the best cooperation is to let one robot track the target, and the other robot leave to search for other targets, so as to maximize the use of the resource (robots force). This target selection is a kind of high level cooperation. However, pure potential field based control cannot guarantee such cooperation in that case. Neither of the two robots will leave.

A solution to achieve better target selection is to modify the pure potential field based control by adding a weight to the attractive forces. This weight of the attractive forces represents the preference that a robot tracks targets. If the weight value is high, the robot is likely to keep tracking detected targets; if the weight value is low, the robot is likely to leave detected targets if other robots are already around it. In previous research [12-14], some algorithms have been designed to adjust this weight. However, it is difficult to find the best weight value for each robot, especially when the scenario is very complex and the robot team is heterogeneous. A natural thought is to let the robot get the best weight value through learning. Hence the museum problem is well suited to the implementation and testing of our learning algorithms.

4.2. Implementation of Learning in Museum Problem

To implement reinforcement learning, we need to design the functions and set the parameters for state-action definition, reward generation and allocation, state-action link value update and action selection.

Firstly, we need to define the states and actions for the reinforcement learning module. To make the learning simple, yet not lose its generality, we define the input state of the learning as the number of targets and robots detected.

Secondly, we need to define the rewards for reinforcement learning. Since the objective of museum problem is to maximize the observation of moving targets, the reward should be given to the robot that tracks target or cooperates with other robots. For this purpose, we define three kinds of rewards:

- *Reward_TT*: track target reward (positive) – if target(s) is within the sensor range.
- *Reward_NR*: near robot reward (negative) – if other robot(s) is nearby.
- *Reward_SC*: state change reward (positive/negative) – if the new state has less neighbor or more targets, the reward is positive, else negative.

Because the learning process is distributed and there is no intercommunications among the robots. For each individual robot, these three kinds of rewards are all generated by its local sensing.

Thirdly, in reinforcement learning, the learning process needs to update the value of the state-action links based on the reward received. In our learning algorithms, the state-action link value is updated in every simulation step. The new value of the state-action link is the summation of the previous value and the rewards.

Finally, the robots need to select the action (weight) according to state and the value of the state-action links. Every time the state changes, the robot will reselect the action (weight). However, if the state is unchanged for a long period of time (N simulation step), we also let the robot reselect the action (weight) because we want to accelerate the learning speed. In reinforcement learning, for action selection, the learning process needs to both explore and exploit the action space. Therefore, when selecting action, we add an exploration factor to the real state-action link value, and then choose the action that has the highest resultant value.

5. Simulation and Results

5.1. Simulation Scenario

We set a scenario where two robots track one target. If pure potential field based control is used, the two robots will both track the target. Obviously, this is a waste of robot force and we expect that through learning, one of the two robots will learn to neglect the target tracked by other robots. To test our learning algorithms, we simulate three kinds of control mode in both homogeneous and heterogeneous robot groups:

- Pure Artificial Potential Field based controller.
- All-adjust heuristics of APF controller: if a robot detects a target and finds that some other robots are near to that target, it will decrease the weight of the attractive force to the target. [12, 13]
- Robot concurrent learning controller by our learning algorithms.

For above three kinds of controllers, we aim to find the following results:

- Waste time length – represent the cooperation level.
- Learning results – the difference in learned weight for tracking targets.

5.2. Simulation Parameters

The parameters and settings of the simulation are as follows:

- The simulations are run on Webots, a differential-wheel robot simulator.
- Museum: 4m * 4m square plain area with no obstacles inside.
- Each learning episode is 20000 simulation step long. For each scenario, 100 episodes is run to get the average of the simulation results.

- In the heterogeneous robot group, one robot is 30% faster than the other(s).
- For the all-adjust heuristics of pure potential field based control mode, the All-adjust Weight Decrease Ratio (AWDR) is 0.95.
- For the learning mode, the input state of the learning is *(robot number, target number)*, e.g., state *(1, 1)* means there are one neighbor robot and one target detected. The output action (weight) space is {0.5, 1.0, 1.5}.
- For the learning mode, the initial value of all state-action links is 10.
- For the learning mode, *Reward_TT* = 0.005, *Reward_NR* = - 0.01, *Reward_SC* = (m-a)*0.5 – (n-b)*2.0 (m/n is the current target/robot number; a/b is the previous target/robot number).
- For the learning mode, if the state changes or if the state has been unchanged for N = 100 simulation steps, the robot will reselect the action (weight). When reselecting the action, an exploration factor (uniformly distributed in [-1, 1]) is added to the real state-action link value, then the action having the highest resultant value will be chosen.

5.3. Simulation Results and Discussion

Figure 3 shows the waste time length. When two robots are simultaneously tracking a target, it is a waste of resource since one of the robots can leave and search for other targets. Obviously, short waste time length means high level cooperation. The results demonstrate the efficacy of our learning algorithms that the waste time length is greatly shortened, even better than the deliberative coded control mode (all-adjust heuristics of pure potential field based control). It should be noted that the performance of the heuristic controller is highly dependent to the value of weight decrease ratio. If an optimal value is selected, the performance may be better. However, considerable human effort is needed to find this optimal value; while the learning controller can learn the best weight value without such work.

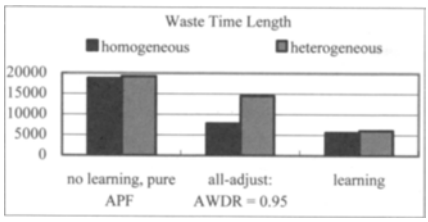

Figure 3. The Waste Time Length in Different Scenario

Since the objective of the learning is to generate cooperative behaviors between the two robots when they meet the same one target, the most meaningful learning results are for the state *(1, 1)* (one neighbor robot and one target detected). Figures 4 and 5 show the learning results (the x-axis represents the difference of the learned weights between two robots; the y-axis represents the probability of getting such result). In the end of the simulation, each robot will find a preferred

weight for this state *(1, 1)* as low, mid, or high. Here we show the "difference in learned weight" instead of the "value of learned weight". This is because the difference in the weights between two robots is the key for cooperation. The robot with high weight will keep tracking and the robot with low weight will leave.

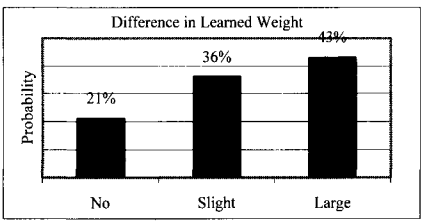

Figure 4. Learning Results of Homogeneous Robot Group

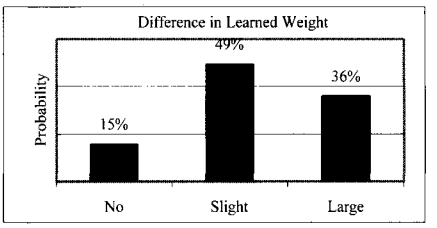

Figure 5. Learning Results of Heterogeneous Robot Group

As shown in Figures 4 and 5, for both homogeneous and heterogeneous robot groups, in most cases, the two robots can learn different weights. This is the key for the cooperation. However, in some cases (homogeneous 24%, heterogeneous 15%) the two robots may learn the same weight in the end. This draw case is undesired because we require the robots to have different weights for cooperation. This problem may be explained by the following reasons:

- In the learning environment, there is no localization and intercommunication available for the robots. The only input of the robots is the local sensed data. Since the sensor ability is quite limited, the partial observation may badly affects the Markov Decision Process (MDP) and stationary environment assumptions.
- Furthermore, since the two robots are learning concurrently, their learning process may interfere with each other, therefore fall in local minima or change control policy cyclically. This problem further affects the two assumptions.

For the homogeneous and heterogeneous robot groups, the learning results are different. The heterogeneous group has less draw cases. This may be due to the fact that the difference in functionality catalyzes the role differentiation between the robots. Besides, the homogeneous robot group prefers highly different weights; while the heterogeneous robot group prefers slightly different weights. An explanation is that in the heterogeneous robot group, the two robots are already quite different that one robot is 30% faster than the other. Therefore a slight weight difference is enough for them to generate cooperative behaviors.

To further validate the efficacy of our learning algorithms, we extend the learning to three targets and three robots scenario using the same parameters as before. The simulation results of this "three plus three" scenario is consistent to the previous "one plus two" scenario. For the pure potential field based controller, all adjust heuristic controller, and the learning controller, the average number of targets being tracked during the simulation are almost the same (2.66, 2.68 and 2.59). How-

ever, if the robots can cooperate, fewer robots are needed to track the targets because the situation that several robots track one same target is avoided. Therefore, large average free robot number means high level cooperation. In the learning mode, 0.26 robots are free (no need for tracking) and they can search targets in the environment; while in the all adjust heuristic control mode and the pure potential field based mode, only 0.15 and 0.10 robots are free. (The above numbers are the average of homogeneous and heterogeneous robot groups.) The results of the extended scenario also show that for our learning environment, the cooperation is mostly happened in the situation that two robots meet one same target. This is possibly due to the limitation of the robot sensor ability that it can only cover a small region around the robot.

Simulation videos can be found in http://guppy.mpe.nus.edu.sg/~mpeangh/kevin.

6. Conclusion and Future Work

Multi-robot concurrent learning on how to cooperatively work is one of the ultimate goals of robotics and artificial intelligence research. Reinforcement learning has achieved great success for this purpose. However, typical reinforcement learning cannot deal with continuous and infinite inputs and outputs. In this paper, we address these problems by integrating reinforcement learning with neural networks (weighting network). The efficacy of our learning algorithms is proved by simulation results.

For multi-robot concurrent learning, the Markov Decision Process and stationary environment assumptions, reward generation and allocation, action selection, and state-action link value update, are critical research issues that may greatly affect the learning process and results. By carefully designing and choosing the functions and parameters of our algorithms for the museum problem, the learning results are satisfactory.

Integrating reinforcement learning with neural networks (weighting network) is a good solution to solve the discrete and finite input/output problem. However, in our learning architecture, the reinforcement module still needs to retrieve discrete input state and perform discrete actions (weights). A more challenging work is to design a totally continuous and infinite learning algorithm, or at least, let the robot do state/action definition or discretization by itself through learning. The answer is to be found in the future research.

Reference

[1] Balch, T., and Parker, L. E. (2002), Robot Teams: From Diversity to Polymorphism, Natick, Massachusetts, A K Peters Ltd.

[2] Cao, Y. U., Fukunaga, A. S., Kahng, A. B., and Meng, F. (1995), Cooperative Mobile Robotics: Antecedents and Directions, in proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol:1, pp:226-234.

[3] Tangamchit, P., Dolan, J. M., and Khosla, P. K. (2002), The Necessity of Average Rewards in Cooperative Multirobot Learning, in proceedings of IEEE International Conference on Robotics and Automation.

[4] Mitchell, T. M. (1997), Machine Learning, McGraw Hill.

[5] Mataric, M. J. (1997), Reinforcement Learning in the Multi-Robot Domain, in Autonomous Robots, Vol:4(1), pp:73 - 83.

[6] Sutton, R. S., and Barto, A. G. (1998), Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA.

[7] Chu, H. T., and Hong, B. R. (2000), Cooperative Behavior Acquisition in Multi Robots Environment by Reinforcement Learning Based on Action Selection Level, in proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol:2, pp:1397-1402.

[8] Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996), Reinforcement Learning: A Survey, in Artificial Intelligence Research, Vol: 4, pp237-285.

[9] Kawakami, K. I., Ohkura, K., and Ueda, K. (1999), 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.

[10] Uchibe, E., Asada, M., and Hosoda, K. (1998), Cooperative Behavior Acquisition in Multi Mobile Robots Environment by Reinforcement Learning Based on State Vector Estimation, in proceedings of IEEE International Conference on Robotics and Automation, Leuven, Belgium, Vol:1, pp:425 - 430.

[11] Michael Bowling, and Manuela Veloso (2003), Simultaneous Adversarial Multi-Robot Learning, in proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico.

[12] Parker, L. E. (2002), Distributed Algorithm for Multi-Robot Observation of Multiple Moving Targets, Autonomous Robots, vol. 12(3), May.

[13] Parker, L. E., and Emmons, B. A. (1997), Cooperative Multi-robot Observation of Multiple Moving Targets, in proceedings of IEEE International Conference on Robotics and Automation, vol. 3, pp. 2082-2089, Albuquerque, New Mexico, USA.

[14] Liu, Z., Ang, M. H., and Seah, W. K. G. (2003), A Searching and Tracking Framework for Multi-Robot Observation of Multiple Moving Targets, in International Journal of Advanced Computational Intelligence & Intelligent Informatics (JACI3), 8-1, 2004.