Multi-robot concurrent learning of cooperative behaviours for the tracking of multiple moving targets

Zheng Liu*

Department of Electrical and Computer Engineering,
National University of Singapore,
Singapore 119260
Institute for Infocomm Research, Singapore
E-mail: zhengliu@nus.edu.sg
*Corresponding author

Marcelo H. Ang Jr

Department of Mechanical Engineering,
National University of Singapore,
Singapore 119260
E-mail: mpeangh@nus.edu.sg

Winston Khoon Guan Seah

Network Technology Department, Institute for Infocomm Research,
Agency for Science, Technology and Research,
Singapore 119613
E-mail: winston@i2r.a-star.edu.sg

Abstract: Reinforcement learning has been extensively studied and applied for generating cooperative behaviours in multi-robot systems. However, traditional reinforcement learning algorithms assume discrete state and action spaces with finite number of elements. This limits the learning to discrete behaviours and cannot be applied to most real multi-robot systems that inherently require appropriate combinations of different elementary behaviours. To address this problem, we design a distributed learning controller that integrates reinforcement learning with behaviour-based control networks. This learning controller can enable the robots to generate appropriate control policy without the need for human design or hardcoding. Furthermore, to address the problems in concurrent learning, we propose a distributed learning control algorithm to coordinate the concurrent learning processes. The distributed learning controller and learning control algorithm are applied to multi-robot tracking of multiple moving targets. The efficacy of our proposed scheme is shown through simulations.

Keywords: behaviour-based control; cooperative tracking; multi-robot concurrent learning; reinforcement learning.

Reference to this paper should be made as follows: Liu, Z., Ang, M.H. and Seah, W.K.G. (2006) 'Multi-robot concurrent learning of cooperative behaviours for the tracking of multiple moving targets', *Int. J. Vehicle Autonomous Systems*, Vol. 4, Nos. 2-4, pp.196–215.

Biographical notes: Zheng Liu received his BE degree in the Department of Automation from Tsinghua University, Beijing, China, in 2001. Currently, he is a PhD student in the Department of Electrical and Computer Engineering, National University of Singapore. His research interests are in practical multi-robot surveillance, hybrid robot/sensor networks, robot/sensor localisation in unstructured environment, multi-robot/agent learning, etc.

Marcelo H. Ang, Jr., received his BS degree (Cum Laude) in Mechanical Engineering and Industrial Management Engineering from the De La Salle University (Philippines), his MS degree in Mechanical Engineering from the University of Hawaii (USA) and his MS and PhD degrees in Electrical Engineering from the University of Rochester, New York (USA). His work experience includes heading the Technical Training Division of Intel's Assembly and Test Facility in the Philippines and a faculty position as an Assistant Professor of Electrical Engineering at the University of Rochester, New York. In 1989, he joined the Department of Mechanical Engineering of the National University of Singapore, where he is currently an Associate Professor. In addition to academic and research activities, he is actively involved in the Singapore Robotic Games as its founding chairman. His research interests span the areas of robotics, automation, computer control and artificial intelligence applications.

Winston Khoon Guan Seah received his doctoral degree in engineering from Kyoto University, Kyoto, Japan, in 1997. He is a Senior Scientist in the Network Technology Department of the Institute for Infocomm Research (I^2R). Prior to I^2R, he had been a Principal Member of Technical Staff and the Director of the Internet Technologies programme in the Institute for Communications Research. He is an Adjunct Associate Professor in the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Concurrently, he is an adjunct faculty in the Graduate School for Integrative Science and Engineering and the Department of Computer Science in the National University of Singapore where he lectures in mobile computing. He is actively involved in research and development in the areas of mobile *ad hoc* and sensor networks and co-developed one of the first quality of service models for mobile *ad hoc* networks. His research also includes IPv6, mobile/wireless internet technologies and internet quality of service. He serves on the Steering Committee of the Asia-Pacific IPv6 Task Force and is also an associate of the Singapore Advanced Research and Education Network (SingAREN.) He is a Senior Member of the IEEE.

1 Introduction

In the research on multi-robot systems, one key problem is how to achieve cooperation among robots (Cao et al., 1995), especially in a decentralised (distributed) manner. Normally, the desired cooperation is in task level, as in behaviour-based control (Tangamchit, Dolan and Khosla, 2002), in which the mission is broken down into tasks and robots choose different tasks (roles) according to their current state and behave accordingly and differently. However, to achieve mission decomposition, task allocation and conflict resolution, 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 extremely difficult, especially when the mission is complex and the robots are heterogeneous. Therefore, we are motivated by the need to let the mobile robots learn how to cooperatively work through the interaction among robots and the feedback from the environment, hence generating appropriate behaviours without human design or pre-programming (or coding) of behaviours.

For behaviour-based control, the 'behaviour' is an interpretation of the reactions of the robots. It is a high-level control methodology that does not directly handle low-level input and output. For example, the elementary behaviour 'avoid obstacles' usually means 'when near to an obstacle' (high-level state), 'make a detour along the boundary of the obstacle' (high-level action). Reinforcement learning naturally fits for the behaviour-based control that requires the robot to 'select' optimal actions under any give state (Mataric, 2001). Therefore, in the past two decades, reinforcement learning has been extensively studied for multi-robot concurrent learning of cooperative behaviours.

To apply reinforcement learning for behaviour-based control, the designer needs to define discrete and a very high number of elementary states and actions. But for most real applications, it is hard to give appropriate and accurate definition to these states and actions. Even through the states and actions can be discretised and defined, the behaviours are still discrete. At one time, the robot can perform only one action representing one kind of behaviour. This contradicts the human reasoning or behaviour that usually humans execute several elementary behaviours concurrently to accomplish a task. In addition, the switching of discrete behaviours usually results in the control of the robots becoming unsmooth, which is undesirable in most cases.

Another problem of multi-robot concurrent learning is that the traditional single-agent/robot-based reinforcement learning algorithms may not work appropriately in a multi-robot domain. This is because some basic assumptions in the single robot domain, e.g. Markov decision process and stationary environment, are not valid due to the interference of concurrent learning processes. To address this problem, the distributed learning processes in the robots have to be carefully controlled and coordinated.

In this paper, we propose a learning controller to address the limitation of traditional reinforcement learning. By integrating reinforcement learning with behaviour-based control networks, the learning controller can generate optimal combinations of elementary behaviours to achieve optimal control. In addition, we retrieve ideas from human behaviours for coordinating concurrent learning processes in a distributed manner. The learning architecture and learning control algorithm are applied to multi-robot concurrent tracking of multiple moving targets.

This paper is organised as follows. Section 2 introduces the fundamental ideas that serve as background and related work. Section 3 presents the basic idea and the concept of our approach. Then, the details of our approach in multi-robot tracking of multiple moving targets are introduced in Section 4. The simulation and results are discussed in Section 5. Finally, Section 6 concludes this paper and introduces the future work.

2 Related work

2.1 Reinforcement learning of behaviours

Reinforcement learning is a learning algorithm that can learn based on the feedback from the environment (or other robots/agents) to generate optimal control policy subject to user-defined criteria (Sutton and Barto, 1998). The typical Q-learning, a well-known reinforcement learning algorithm, is as introduced in Algorithm 1.

Algorithm 1: The Q-learning algorithm

Step 1. Initialisation: define the state pool $\{s_i\}$ to represent the possible states of the environment; define the action pool $\{a_j\}$ to represent the possible actions to perform; set current time $t = 0$; set the initial state-action values $\{Q(s_i, a_j)\}$. Here $Q(s_i, a_j)$ is the value that represents the expected reward for performing action a_j under state s_i.

Step 2. Detect the state of environment and then choose s_t to represent the real state.

Step 3. Choose an action a_t, such that $a_t = \arg\max (Q(s_t, a) + \text{RandomFactor})$. Here *RandomFactor* is used to avoid local optimality of the selection of actions.

Step 4. Detect the new state s_{t+1} and receive the immediate reward r_t.

Step 5. Update the $Q(s_t, a_t)$ by the Q function (1) in which s_t, a_t, r_t, α, γ, s_{t+1} and a_{t+1} means state, action, reward, learning rate, discount rate, next state and next action, respectively (Sutton and Barto, 1998).

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right] \tag{1}$$

Step 6. Check whether the termination condition is satisfied. If yes, then stop; else, let $t = t + 1$, go to Step 3.

For reinforcement learning, the discrete state and action spaces cannot be too large; otherwise, the learning may take excessively long or never converge to the desired result. This is the problem known as the curse of dimension (Sutton and Barto, 1998). To avoid oversized state and action spaces, the system designer needs to group the low-level control inputs and outputs into some form of high-level abstracts, i.e. primitive or elementary activities. Then, the high-level behaviour stems from these elementary activities. This is the basic concept of the hierarchical reinforcement learning of behaviours (Barto and Mahadevan, 2003), which can enable a robot to learn discrete control behaviours (Farahmand, Ahamadabadi and Araabi, 2004) or sequential actions (Amit and Mataric, 2002). On the other hand, the reinforcement learning algorithms can also be applied to learn the elementary activities (in low-level) while the high-level behaviour is pre-defined by the designer (Chu and Hong, 2000).

Whatever the robot/agent learns, the reinforcement learning algorithm usually results in the discrete control policy, which 'selects' elementary behaviours according to the environmental states. However, in many cases the optimal control policy is to execute several elementary behaviours concurrently. For example, the optimal behaviour of tracking a target might be a mixture of several basic behaviours such as 'approach detected targets', 'search for other targets' and 'avoid obstacle'. In addition, the switching between two elementary elements may make the control unsmooth.

2.2 *Reinforcement learning in continuous space*

As shown in Algorithm 1, traditional reinforcement learning assumes discrete and finite-size state and action space. But for real applications, the input and output space are usually continuous and infinite. To address this problem, discretisation is usually applied to preprocess the inputs and outputs. However, if the discretisation is too coarse, some states may be hidden; therefore, the optimal control policy can not be found; if the discretisation is too fine, the states cannot be generalised and the huge state/action space will badly affect the learning speed and convergence. Furthermore, to generate desired smooth control policy, the learning of discrete and infinite actions may not be acceptable.

To make reinforcement learning work without discretisation, some heuristics of reinforcement learning have been proposed. Function approximation (Boyan and Moore, 1995) and HEDGER (Smart and Kaelbling, 2000) apply the generalising function approximator to estimate the state-action value instead of using discrete lookup table. Autonomous state/action discretisation by Bayesian discrimination method (Yasuda, Ohkura and Taura, 2003) or cognitive learning (Ueno, Takeda and Nishida, 1999) can segment the continuous state and action space without human coding. Doya (1996) and Hagen (2001) propose reinforcement learning to derive optimal feedback control law for linear/nonlinear systems. These approaches have been successfully applied to simple and low-level control applications. However, they usually assume that the environment model is known and it has heavy computational burden. Therefore, they are still not widely used for behaviour learning.

2.3 Multi-robot concurrent learning

Reinforcement learning and most other machine learning algorithms assume that the learning process is Markovian and that the learning environment is stationary (Kaelbling, Littman and Moore, 1996). These two assumptions both require the full/sufficient observation of the environment. However, limited by the sensor ability, robots cannot generate a complete view of the environment. Furthermore, if all 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 states or influences of other robots, such that the input states for learning become more observable and predicable for an individual learning robot (Uchibe, Asada and Hosoda, 1998a; Kawakami, Ohkura and Ueda, 2001). Another class of solutions is to coordinate or schedule the distributed learning processes to reduce the interference. Uchibe, Nakamura and Asada (1998b) and Asada, Uchibe and Hosoda (1999) propose the global scheduling method that synchronises the learning robots. Ikenoue, Asada and Hosoda (2002) propose a distributed learning control algorithm that can enable multiple robots learning concurrently. Bowling and Veloso (2002) introduce the algorithm to change the learning speed to reduce the interference of multi-robot/agent learning. Some approaches even let the multiple robots share the Q value of the state-action pairs to learn cooperatively (Mirfattah and Ahmadabadi, 2002). However, the coordination and scheduling of learning processes and the sharing of information, have to be deliberatively designed and require explicit intercommunications among the robots. This degrades the applicability of the coordination algorithms for multiple robot concurrent learning.

For multi-robot concurrent learning, another important issue is the generation and assignment of the reward. In reinforcement learning, the reward is the key to encourage and reinforce the desired behaviour. To achieve cooperation, greedy learning may not work because the unselfish but cooperative actions also need to be rewarded (Tangamchit, Dolan and Khosla, 2002). In related approaches, the multiple robots/agents either need to broadcast the local reward to the group of robots (Mataric, 1997) or to monitor the progress or efficacy of other robots (Parker, Touzet and Fernandez, 2002). This usually requires explicit inter-robot communications or the ability to detect the progress or even the capability, of other robots.

3 Concurrent reinforcement learning with behaviour-based control network

As introduced in Sections 1 and 2, there are two problems to be addressed for multi-robot concurrent learning of cooperative behaviours:

- How to generate optimal combination of elementary behaviours for cooperation, based on low-level input states and output actions?
- How to coordinate concurrent learning processes efficiently, especially in distributed manner?

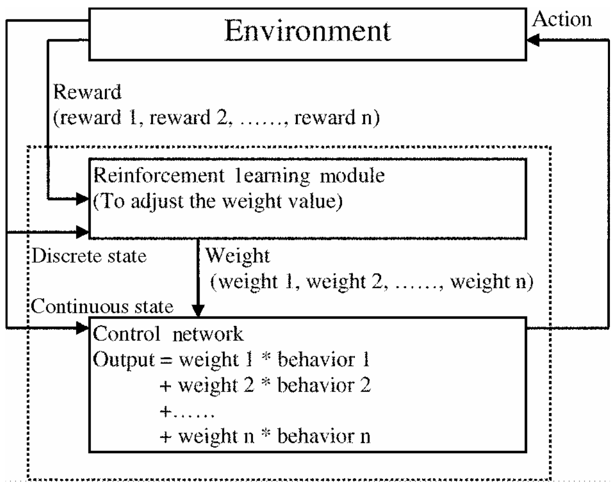
Therefore, the aim of our research is as follows:

- Enable the robots to learn based on low-level input and output without the need for deliberate definition of high-level states and actions.
- Let the robots learn based on a behaviour-based controller created using human knowledge of robot, mission and environment.
- Let the robots learn task-level cooperative behaviours that combine elementary behaviours.
- Coordinate concurrent learning processes to guarantee the generation of optimal control policy or at least, minimise the probability to result in local sub-optimal control policy or other undesired control policies.
- Let the robots generate necessary global reward by local sensing to reinforce the desired cooperative behaviours.
- Minimise the requirement for intercommunications for coordinating concurrent learning processes.

With regard to the problems associated with discrete and finite number of elementary behaviours, we propose the integration of reinforcement learning with behaviour-based control networks. Details of this learning controller are shown in Figure 1 (inside the dotted rectangle). It includes behaviour-based control network module and reinforcement learning module.

The behaviour-based control network module is created according to the human knowledge of the robot, the environment and the mission. In this network, the elementary behaviours are represented by control rules and equations. For example, elementary behaviour ‘obstacle avoidance’ may be represented by a formula while ‘target tracking’ may be a batch of fuzzy rules with corresponding membership functions. Each elementary behaviour can retrieve continuous and infinite number of input signals and generate continuous and infinite number of output commands. Finally, the overall output is the summation of weighted outputs of all elementary behaviours. In this behaviour-based control network, the weight is the key to combining different elementary behaviours: if the weight of one behaviour is large, the robot is more likely to perform this behaviour; otherwise, the robot is less inclined towards this behaviour.

Figure 1 Architecture of proposed learning controller


Environment
Action
Reward
(reward 1, reward 2, ……, reward n)
Reinforcement learning module
(To adjust the weight value)
Discrete state
Weight
(weight 1, weight 2, ……, weight n)
Continuous state
Control network
Output = weight 1 * behavior 1
+ weight 2 * behavior 2
+……
+ weight n * behavior n

In the proposed learning controller, the reinforcement learning module is integrated with the behaviour control network. The aim of this learning module is to adjust the weights inside the control network; thus affecting the combination of elementary behaviours. This is essential for generating optimal combination of behaviours. By retrieving states and rewards, the learning module can gradually find the appropriate weights for each elementary behaviour; therefore, the optimal control policy is learned. It should be noted that the reinforcement learning module needs to retrieve rewards corresponding to each behaviour; otherwise, the learning module cannot estimate the performance or results of acting the behaviour. For example, regarding behaviour ‘avoid obstacles’, if the performance is unsatisfactory, a negative reward (penalty) is given to indicate that the weight of ‘avoid obstacles’ needs to be adjusted.

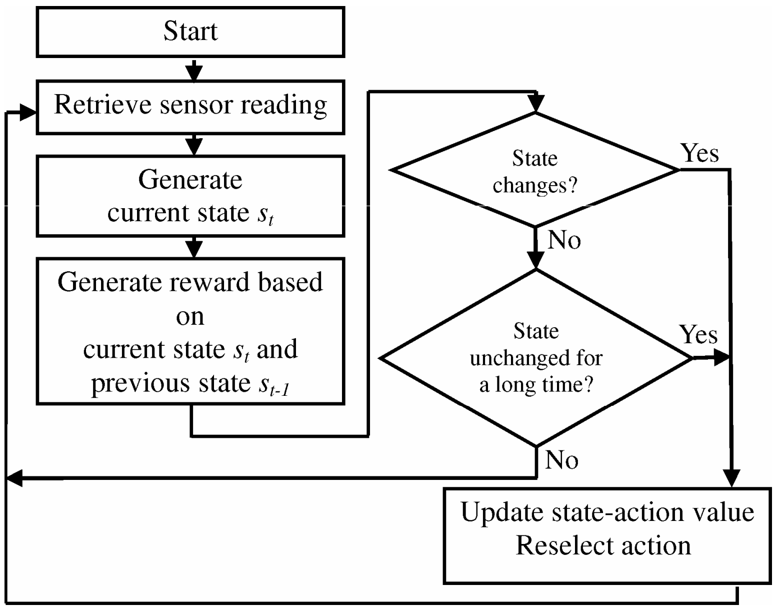
In general, the proposed learning controller has the following properties:

- The behaviour-based control network is designed based on human expertise. The control rules or equations are the representations of elementary behaviours.
- For the control network, the overall output behaviour is the summation of weighted elementary behaviours. The output control command is smooth and continuous.
- The aim of the reinforcement learning module is to adjust the weights in the control network to achieve optimal combination of elementary behaviours. In other words, the output is not the ‘selection’ of exclusive discrete behaviour, but the way to ‘combine’ them to generate continuous and infinite number of behaviours.
- 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 retrieve low-level sensor data and generate appropriate low-level commands representing continuous and infinite number of possible behaviours.

In addition to solving the discrete behaviours problem, another research issue is coordinating concurrent learning processes. To address this problem, we propose a solution inspired by natural human behaviours. Assuming two persons are approaching each other in a corridor and they want to avoid the collision, what will they do? If they are both trying, they may ‘struggle’ several rounds to find the best solution. So, in real life, usually one (say *A*) will fix his/her policy first, e.g. keeping left, then the other one (say *B*) can choose another side. In this encounter case, the optimal cooperation is that the two people choose opposite sides. Whatever *A* chooses initially, if *B* can finally learn to choose another side, 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/agents can eventually find appropriate control policy to respond to the former one. In other words, for some cooperative multi-robot systems, the ‘optimisation’ lies in the relationship among robots.

Our distributed learning coordination algorithm is proposed based on the above considerations. As shown in Algorithm 2, 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 in 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. This learning coordination algorithm is embedded into the proposed learning process (Figure 2). Before the robot updates its state-action link value, it will run this local learning coordination algorithm to decide whether to change the state-action value or just keep the value unchanged. This learning control algorithm is entirely distributed and does not need intercommunications among robots. This algorithm should be effective for the kind of corridor encounter applications.

Figure 2 Flow chart of learning process


Start
Retrieve sensor reading
Generate current state s_t
Generate reward based on current state s_t and previous state s_{t-1}
State changes?
Yes
No
State unchanged for a long time?
Yes
No
Update state-action value Reselect action

Algorithm 2: Learning coordination

Step 1. For each state s_i in the state space, set flag(s_i) = 0.

Step 2. Check the state-action value for each state s_i: $Q(s_i, a_j)$, where a_j is the jth possible action. If there exists an j, such that $Q(s_i, a_j) > (1/m) \times \sum_{n=1}^{m} Q(s_i, a_n) +$ Threshold, then set flag(s_i) = 1. Here m is the number of possible actions for state s_i, Threshold is a positive value that indicates how much larger the best action's value should be to stop learning.

Step 3. If flag(s_i) = 0, the $Q(s_i, a_j)$ can be updated; otherwise $Q(s_i, a_j)$ will not be updated.

To test the efficacy of the proposed learning controller, we apply the learning architecture and algorithm to the task of multi-robot tracking of multiple moving targets. The details of the implementation are introduced in Section 4.

4 Learning in multi-robot tracking of multiple moving targets

4.1 Museum problem: multi-robot tracking of multiple moving targets

In robotics research, the problem of multi-robot tracking of multiple moving targets is also referred to as the 'museum problem' or 'art gallery problem'. The assumptions and descriptions of the 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° view within a certain range. When an object is inside this range, the robot can detect the distance and angle towards this object. Besides, the robot can differentiate the observed object as obstacle, target or other robot.
- For the robots, the number, distribution and the motion pattern of targets are unknown.
- For the robots, the size and map of the environment are unknown. The robots cannot localise themselves in the environment.
- There are no explicit intercommunications (e.g. wireless communications) available among robots.
- The summation of the sensible area of all robots is far less then the size of the environment. Since the targets are mobile and the robot sensor range is limited, the robot needs to track (move together with) the targets to maintain observation.
- For one target, only one robot is needed to track it.
- The objective is to maximise the number of targets being simultaneously observed (detected within the robot's sensing range) and minimise the number of robots that are needed to track.

4.2 Research issues in the museum problem

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 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. However, purely summing the attractive and repulsive forces may not achieve the 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 maximise the number of targets which are observed.

A solution to avoid the triangular pattern of pure APF is giving a weight $\left(w_{R_i}^{T_j}\right)$ to the attractive force for each robot as shown in Equation (2). In this equation, $\vec{F}_{R_i}$ is the summation of the attractive and repulsive forces for robot R_i; $\vec{A}_{R_i,T_j}$ is the virtual attractive force from robot R_i to target T_j. The orientation of the force is the angle to T_j and the magnitude is a fixed value, e.g. 1.0 or a value depending on the distance between R_i and T_j; $w_{R_i}^{T_j}$ is the weight to change the attractive force from robot R_i to target T_j; $\vec{R}_{R_i,R_l}$ is the virtual repulsive force from robot R_l to robot R_i. The orientation of the force is the angle to R_i and the magnitude is a fixed value, e.g. 1.0 or a value depending on the distance between R_l and R_i (Liu, Ang and Seah, 2004); d*t* and d*r* are the sets of neighbour targets and robots within the sensor range, respectively.

$$\vec{F}_{R_i} = \sum_{T_j \in \mathrm{d}t} w_{R_i}^{T_j} \times \vec{A}_{R_i,T_j} + \sum_{R_l \in \mathrm{d}r} \vec{R}_{R_i,R_l} \tag{2}$$

Examining Equation (2), we can find if the weight of attractive force is zero; the robot will only have one behaviour, i.e. avoiding neighbouring robots; if the weight of attractive force is infinity, the robot will only have one behaviour, i.e. tracking targets. Changing the value of the weight means changing the preference to two behaviours 'avoiding neighbouring robots' and 'tracking targets'. In previous research, two classes of algorithms are proposed to adjust the weight. One is the all-adjust heuristic (Parker, 2002) that lets the robot decrease the weight when it finds that another robot is also tracking the same target (Algorithm 3). The other solution is the selective-adjust heuristic (Liu, Ang and Seah, 2004), whereby only the further robot(s) decreases the weight (Algorithm 4).

Algorithm 3: All-adjust heuristic of potential field-based control (for Robot R_i)

Step 1. Set initial force to move as $\vec{F}$. The orientation of $\vec{F}$ is a random value uniformly distributed between $[0, 2\pi]$; the magnitude of $\vec{F}$ is a fixed value, e.g. 1.0.

Step 2. Scan the surrounding environment; find the sets of detected targets and neighbour robots as d*t* and d*r*.

Step 3. If d$t \neq \phi$, let $\vec{F} = 0$. For all $T_j \in$ dt if another robot is found tracking it (e.g. robot $R_k \in$ dr and distance between R_k and T_j is less than the sensor range of R_k), let $\vec{F} = \vec{F} + \vec{F} + w_j \vec{A}_{R_i,T_j}$; else, let $\vec{F} = \vec{F} + \vec{A}_{R_i,T_j}$. Here, the w_j is the All Weight Decrease Ratio (AWDR) between [0, 1].

Step 4. If d$r \neq \phi$, let $\vec{F} = \vec{F} + \sum_{R_l \in \mathrm{d}r} \vec{R}_{R_i,R_l}$.

Step 5. Let robot R_i move under the virtual force $\vec{F}$.

Step 6. Go to Step 1.

Algorithm 4: Selective-adjust heuristic of potential field-based control (for Robot R$_i$)

Step 1. Set initial force to move as $\vec{F}$. The orientation of $\vec{F}$ is a random value uniformly distributed between [0, 2π]; the magnitude of $\vec{F}$ is a fixed value, e.g. 1.0.

Step 2. Scan the surrounding environment; find the sets of detected targets and neighbour robots as dt and dr.

Step 3. If d$t \neq \phi$, let $\vec{F} = 0$. For all $T_j \in$ dt, if another robot is found tracking T_j and is nearer to it than R_i (e.g. robot $R_k \in$ dr and distance between R_k and T_j is less than the distance between R_i and T_j), let $\vec{F} = \vec{F} + w_j \vec{A}_{R_i,T_j}$; else, let $\vec{F} = \vec{F} + \vec{A}_{R_i,T_j}$. Here, the w_j is the Selective Weight Decrease Ratio (SWDR) between [0, 1].

Step 4. If d$r \neq \phi$, let $\vec{F} = \vec{F} + \sum_{R_l \in \mathrm{d}r} \vec{R}_{R_i,R_l}$.

Step 5. Let robot R_i move under the virtual force $\vec{F}$.

Step 6. Go to Step 1.

These two heuristics of pure potential field-based control are shown to be effective in experiments; 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.

4.3 Applying our learning controller in museum problem

To implement the proposed learning controller to the museum problem, the key is to learn the optimal weights for potential field-based control; therefore, the robots can track targets cooperatively and efficiently.

Figure 2 shows the flowchart of the learning process in the learning module of the learning controller. The main research issues 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 neighbour are detected, the state is (2, 1). The output of the learning module (action) is the weight of the attractive force.

For reinforcement learning, one important issue is the generation of rewards. This is because reward represents the objective of the system designer and can directly affect the learning results. Since task level cooperation is desired, following behaviours should be encouraged: track target; leave the target being tracked by other robots. For this purpose, we define four kinds of rewards:

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

For each individual robot, these rewards are generated by its local sensing. For example, if both robot *A* and robot *B* are tracking the same target, in the view point from robot *A*, there will be *Reward_WT* if *B* is detected.

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 Q function [Equation (1)] as introduced in Algorithm 1.

Every time the state changes, the robot will reselect the action (weight). Furthermore, if the state is unchanged for a long period of time (N simulation steps), the robot also reselects the 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. It should be noted that this exploration factor is just for action selection; it will not affect the state-action value.

Regarding the coordination of the distributed learning processes, the robots do not need to communicate to share any information. 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 may less likely generate local sub-optimal control policy or the cyclic switching of control policies.

5 Simulation and discussion

5.1 Simulation methodology

The aim of our research is to let the mobile robots learn how to cooperatively work through the interaction with environment and other robots, hence generating appropriate behaviours without human design or coding. This research aim includes two main aspects:

- The learning approach can generate cooperative behaviours.
- The performance of the learning system should be comparable to other approaches that have been deliberatively hardcoded and tuned.

To test the efficacy of our approach, we simulate four control modes as follows:

- Pure APF-based control.
- All-adjust heuristics to pure APF.
- Selective-adjust heuristics to pure APF.
- Robot learning controller: with and without coordination.

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

5.2 Simulation settings

The parameters and settings of the environment are as follows:

- The simulations are run on Webots, a differential-wheel robot simulator.
- Museum: 4 × 4 to 6 × 6 m^2 plain area with no obstacles inside. The simulated robot and target are less then 0.1 m in diameter.
- For each control mode, run about ten episodes to get the average of the simulation results. Each episode is 15,000 simulation step long. One simulation step is about 0.1 sec long in real time.

For the all-adjust heuristics of pure potential field-based control, if two or more robots find the same target and they find each other, they will all decrease the weight of the attractive force to target. In the simulation, we test five different All Weight Decrease Ratio (AWDR): 0.1, 0.3, 0.5, 0.7 and 0.9.

For the selective-adjust heuristics of pure potential field-based control, if two or more robots find the same target and they find each other, the further robot(s) will decrease the weight of the attractive force to the target. In the simulation, we test five different SWDR: 0.1, 0.3, 0.5, 0.7 and 0.9.

The settings of the learning controller are as follows:

- The initial Q values of all state-action pairs are set as 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.
- Possible weights to learn are 0.1, 0.3, 0.5, 0.7 and 0.9.

- For learning ‘with coordination’, under one state, if one action’s value is 25% above the average, stop learning in this state.
- If the state is unchanged for $N = 100$ simulation steps, the robot reselects the action.
- When selecting action, a random number uniformly distributed in [–1, 1] is added to the real state-action value as the exploration factor. It should be noted that the exploration factor is just for action selection, but not for updating the state-action value.

5.3 Simulation results and discussion

5.3.1 Tracking performance

Since the research aim of museum problem is to maximise the number of observed targets and minimise the number of robots needed to track targets, we use the following two metrics to evaluate the performance of the multi-robot systems:

- Average number of tracked targets – the higher the better.
- Average number of robots needed to tracked one target – the less the better.

The simulation results are shown in Figures 3–7. In each of these figures, the leftmost column is the result of pure potential field-based control; the second to the left is the result of learning with coordination (the learning without coordination is discussed later); the third and fourth columns are the results of controllers with all-adjust heuristic and selective-adjust heuristic. In all simulation runs, the settings of the potential field-based controller and the learning controller are unchanged, i.e. the weight value (for potential field-based control) and the learning parameters (for proposed learning controller) are constant. Therefore, the performance of these two controllers shown in the figures is a single value that is the average of all simulation runs. On the other hand, as introduced in Section 5.2, in the simulation, different parameter settings for the all-adjust heuristic and selective-adjust heuristic controllers are tested, i.e. the AWDR (all-adjust heuristic) and SWDR (selective-adjust heuristic) are changed in different simulation runs. With regard to each parameter setting, we get one performance value. The values shown in the figure are the highest and lowest ones and the average of the all five values. For example, in the leftmost figure (for average number of tracked targets) in Figure 4, we show three performance values for the all-adjust heuristic controller: the lowest is 2.634 (when AWDR equals 0.1), highest is 2.765 (when AWDR equals 0.7) and the average value is 2.692.

Observing these figures, we may find that in most cases, the pure potential field-based controller performs the worst. The two heuristics can improve the performance; however, the improvement is not consistent. Sometimes they perform the best, but sometimes they are even worse than the pure potential field-based control, e.g. Figure 5. This is because the hardcoded controllers are sensitive to the parameter settings and for different scenarios the optimal parameter setting varies. For example, as shown in Figure 8, for the all-adjust heuristic of potential field-based control, the different parameter (AWDR) settings have quite different results in different scenarios. In scenario ‘three targets and six robots’ and homogeneous robot team, the AWDR = 0.7 achieves the best, while in scenario ‘six targets and three robots’ the AWDR = 0.1 achieves the best.

Figure 3 One target and two robots – 1T2R (left two for homogeneous, the other two are for heterogeneous) (P: pure potential field-based control; L: learning; A: all-adjust heuristic; S: selective-adjust heuristic)

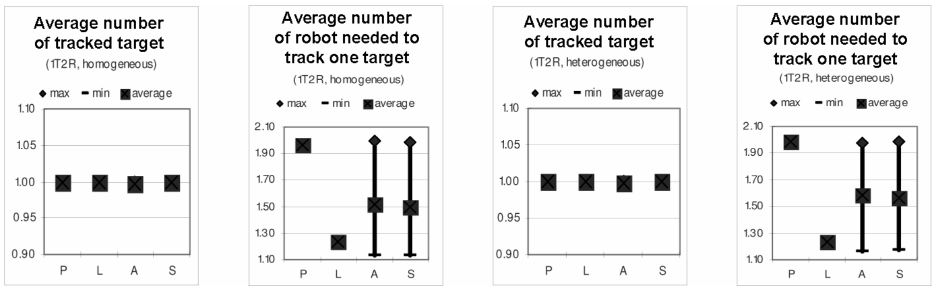


Figure 4 Three targets and three robots – 3T3R (left two for homogeneous, the other two are for heterogeneous) (P: pure potential field-based control; L: learning; A: all-adjust heuristic; S: selective-adjust heuristic)

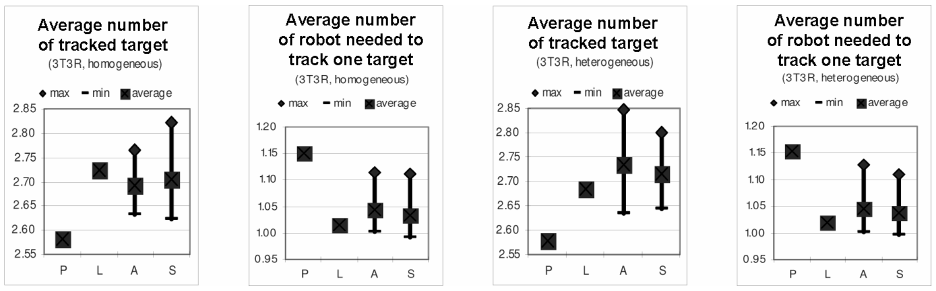


Figure 5 Three targets and six robots – 3T6R (left two for homogeneous, the other two are for heterogeneous) (P: pure potential field-based control; L: learning; A: all-adjust heuristic; S: selective-adjust heuristic)

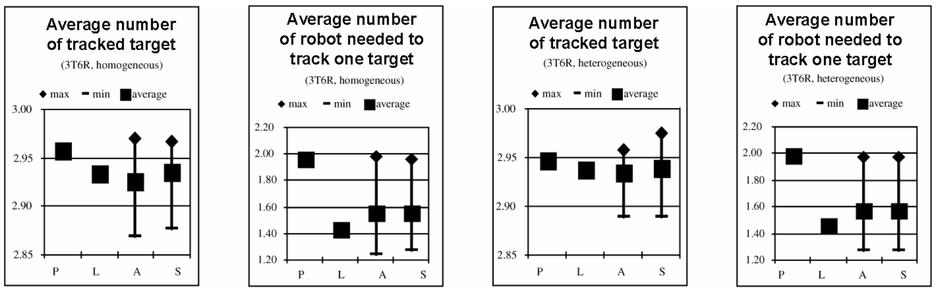


Figure 6 Six targets and three robots – 6T3R (left two for homogeneous, the other two are for heterogeneous) (P: pure potential field-based control; L: learning; A: all-adjust heuristic; S: selective-adjust heuristic)

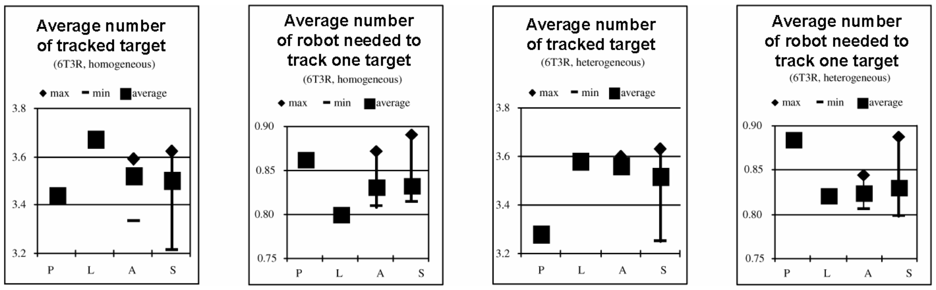


Figure 7 Six targets and six robots – 6T6R (left two for homogeneous, the other two are for heterogeneous) (P: pure potential field-based control; L: learning; A: all-adjust heuristic; S: selective-adjust heuristic)

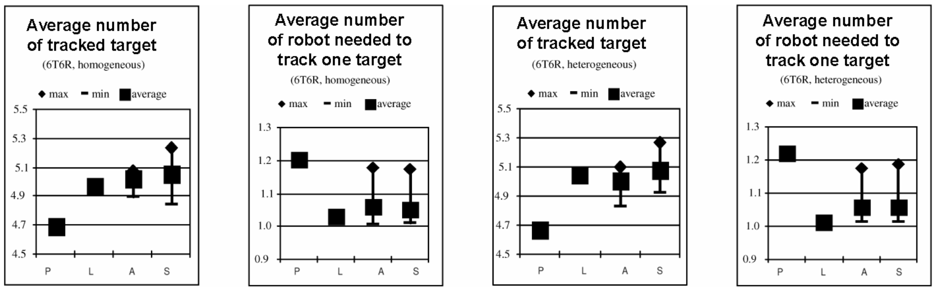
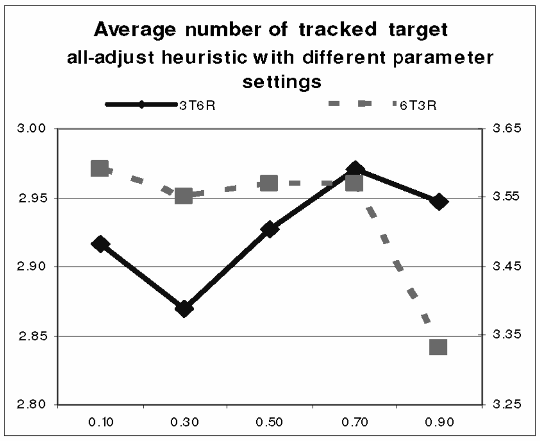


Figure 8 Performance with different parameter settings



In contrast to the hardcoded controllers that require tuning to achieve optimal control, the learning controller does not need to select important parameters, but the system can maintain high performance in all cases as shown in Figures 3–7. The simulations show that the proposed learning controller has the capability to adapt to different scenarios. Furthermore, in both homogeneous and heterogeneous robot teams, the performance of the learning robots is satisfactory and consistent, thus also showing the adaptation of the learning controller.

Observing the simulation results, a question may arise that why the performance of the learning controller is sometimes worse than the selective-adjust weight heuristic of the pure potential field-based control. A reasonable explanations is that the selective-adjust weight heuristic (Liu, Ang and Seah, 2004) also considers the distance between robots and targets (as shown in Algorithm 3), but the proposed learning controller does not utilise this information because we want to avoid the increase in the input state space and to make the learning converge within satisfactory time.

5.3.2 Analysis of concurrent learning processes

For the museum problem, the research aim is to maximise the number of observed targets and minimise the number of robots needed to track targets. To achieve this, the robot force needs to be fully utilised such that the robots should track detected targets and also try to find undetected targets. For example, when two robots find the same target, they should behave differently such that one keeps tracking and the other leaves to search for other targets. In our learning controller, to achieve this kind of cooperation, the two robots should learn different weights when they detect the same target: the robot with higher weight will keep tracking and the robot with lower weight will leave and search for other targets.

Figure 9 shows the learning results of the homogeneous and heterogeneous robot teams, respectively. In the figures, the *x*-axis represents different scenarios; the *y*-axis represents the normalised successful rate that the robots learn to cooperate. The higher rate means better system performance. These figures show that in most cases (more than 90%), the proposed learning coordination algorithm can help the robots learn the cooperative behaviours, while without this coordination the robots are less likely to learn the desired cooperation.

Figure 9 Learning success rate

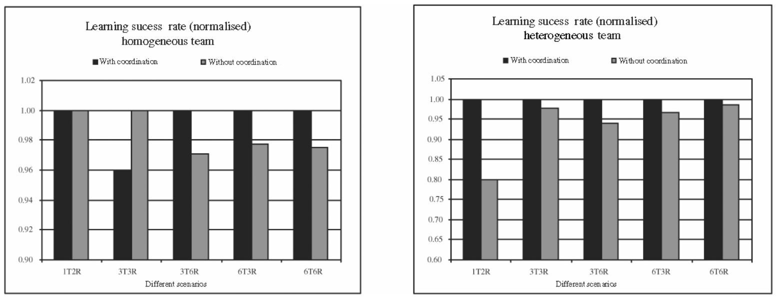
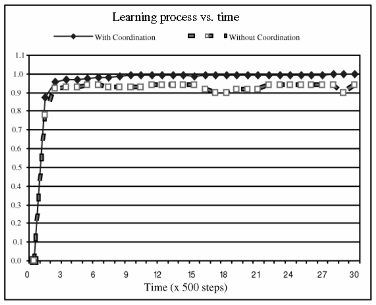


Figure 10 shows the normalised learning progress vs. time in scenario 'three targets and six robots' (3T6R), heterogeneous robot team. The left one is the overview of the learning progress in the whole simulation period and the right one is the progress in the last 7000 steps. The figures show that the learning with coordination can achieve better learning results and it is faster and more stable than the learning without coordination.

Because the learning coordination algorithm can improve the performance of learning to let the robots generate desired cooperative behaviour, the tracking performance of the learning controller with coordination is better than the learning controller without coordination. Figure 11 compares the average number of tracked target (normalised) between these two controllers in homogeneous and heterogeneous robot teams, respectively. The results show that the proposed learning coordination algorithm can enable the robots to learn to track targets more efficiently.

Figure 10 Learning progress vs. time (scenario: 3T6R, heterogeneous); left: progress in the whole simulation; right: progress in the last 7,000 simulation steps



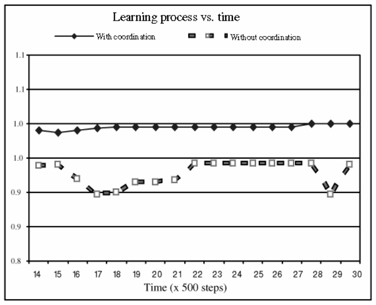
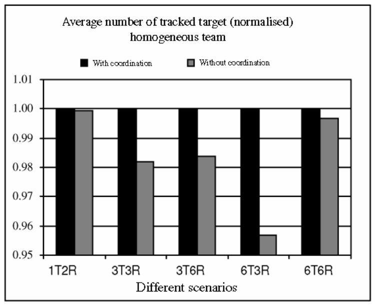
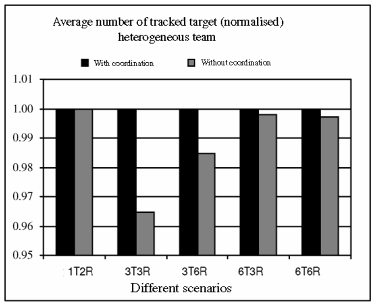


Figure 11 Average number of tracked target (normalised)





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. In this paper, we propose a distributed learning controller that integrates reinforcement learning with behaviour-based control networks. This controller can enable the robots to generate cooperative behaviours in continuous space. In addition, we propose a natural inspired distributed learning control algorithm to coordinate the concurrent learning processes. This algorithm can help avoid the generation of local sub-optimal control policy or the cyclic switching of control policies without the need for explicit intercommunications among the robots. Our approach is tested in multi-robot tracking of multiple moving targets and the efficacy is shown by simulation. The learning controller can achieve the performance as good as the controllers deliberatively designed.

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 and infinite space learning algorithm or at least, let the robot do state/action discretisation by itself through learning. This is an important research issue to be studied.

Another problem of the learning controller is that the behaviour-based control network coded by us is specific for the tracking task. If other task is selected, e.g. cooperative table carrying, we have to design other specific behaviour-based control network accordingly. If the network is inappropriately designed and does not fit the task, the reinforcement learning may not work properly, e.g. generate fatal error of local sub-optimal control policy. Therefore, it will be much better if the behaviour-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 (less than 10% as shown in the simulation) even though we have proposed a distributed learning coordination algorithm. How to perfectly coordinate concurrent learning processes by minimal intercommunications is still a critical research topic for both robotics and artificial intelligence research.

References

Amit, R. and Mataric, M.J. (2002) 'Learning movement sequences from demonstration', Paper presented at the *International Conference on Development and Learning (ICDL2002)*, pp.203–208, MIT, Cambridge, Massachusetts, June 12–15. In proceedings.

Asada, M., Uchibe, E. and Hosoda, K. (1999) 'Cooperative behaviour acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development', *Artificial Intelligence*, Vol. 110, pp.275–292.

Barto, A.G. and Mahadevan, S. (2003) 'Recent advances in hierarchical reinforcement learning', In *Discrete Event Dynamic Systems*, Vol. 13, pp.341–379.

Bowling, M. and Veloso, M. (2002) 'Multiagent learning using a variable learning rate', *Artificial Intelligence*, Vol. 136, pp.215–250.

Boyan, J.A. and Moore, A.W. (1995) 'Generalization in reinforcement learning: safely approximating the value function', In G. Tesauro, D.S. Touretzky and T.K. Lee (Eds), *Advances in Neural Information Processing Systems 7*, (pp.369–376). Cambridge, MA: MIT Press.

Cao, Y.U., Fukunaga, A.S., Kahng, A.B. and Meng, F. (1995) 'Cooperative mobile robotics: antecedents and directions', Paper presented at the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1, pp.226–234. In proceedings.

Chu, H.T. and Hong, B.R. (2000) 'Cooperative behaviour acquisition in multi robots environment by reinforcement learning based on action selection level', Paper presented at the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, pp.1397–1402. In proceedings.

Doya, K. (1996) 'Temporal difference learning in continuous time and space', In *Advance in Neural Information Processing Systmes*, Vol. 8, (pp.1073–1079). Cambridge, MA. MIT Press.

Farahmand, A.M., Ahmadabadi, M.N. and Araabi, B.N. (2004) 'Behaviour Hierarchy Learning in a behaviour-based system using reinforcement learning', Paper presented at the *International Conference on Intelligent Robots and Systems (IROS2004)*, Sendai, Japan. In proceedings.

ten Hagen, S.H.G. (2001) 'Continuous state space Q-learning for control of nonlinear systems', PhD thesis, Computer Science Institute, University of Amsterdam.

Ikenoue, S., Asada, M. and Hosoda, K. (2002) 'Cooperative behaviour acquisition by asynchronous policy renewal that enables simultaneous learning in multiagent environment', Paper presented at the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.2728–2734. In proceedings.

Kaelbling, L.P., Littman, M.L. and Moore, A.W. (1996) 'Reinforcement learning: a survey', *Artificial Intelligence Research*, Vol. 4, pp.237–285.

Kawakami, K.I., Ohkura, K. and Ueda, K. (2001) 'Reinforcement learning approach to cooperation problem in a homogeneous robot group', Paper presented at the 2*001 IEEE International Symposium on Industrial Electronics (ISIE2001)*, Vol. 1, pp.423–428. In proceedings.

Liu, Z., Ang, M.H. and Seah, W.K.G. (2004) 'Searching and tracking for multi-robot observation of moving targets', Paper presented at the *8th Conference on Intelligent Autonomous Systems*. In proceedings.

Mataric, M.J. (1997) 'Learning social behaviour', *Robotics and Autonomous Systems*, Vol. 20, pp.191–204.

Mataric, M.J. (2001) 'Learning in behaviour-based multi-robot systems: policies, models and other agents', In R. Sun (Ed.), *Cognitive Systems Research, special issue on Multi-Disciplinary Studies of Multi-Agent Learning*, Vol. 2, pp.81–93.

Mirfattah, S.M.R. and Ahmadabadi, M.N. (2002) 'Cooperative Q-learning with heterogeneity in actions', Paper presented at the *2002 IEEE International Conference on Systems, Man and Cybernetics*. In proceedings.

Parker, L.E. (2002) 'Distributed algorithm for multi-robot observation of multiple moving targets', *Autonomous Robots*, Vol. 12, pp.231–255.

Parker, L.E., Touzet, C. and Fernandez, F. (2002) 'Techniques for learning in multi-robot teams', In T. Balch and L.E. Parker (Eds), *Robot Teams: From Diversity to Polymorphism*.Wellesley, MA: A.K. Peters.

Smart, W.D. and Kaelbling, L.P. (2000) 'Practical reinforcement learning in continuous spaces', Paper presented at the *7th International Conference on Machine Learning*. In proceedings.

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

Tangamchit, P., Dolan, J.M. and Khosla, P.K. (2002) 'The necessity of average rewards in cooperative multirobot learning', Paper presented at the *IEEE International Conference on Robotics and Automation*. In proceedings.

Uchibe, E., Asada, M.M. and Hosoda, K. (1998) 'Cooperative behaviour acquisition in multi mobile robots environment by reinforcement learning based on state vector estimation', Paper presented at the *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 1998. In proceedings.

Uchibe, E., Nakamura, M. and Asada, M. (1998) 'Co-evolution for cooperative behaviour acquisition in a multiple mobile robot environment', Paper presented at the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Victoria, B.C., Canada. In proceedings.

Ueno, A. Takeda, H. and Nishida, T. (1999) 'Cooperation of cognitive learning and behaviour learning', Paper presented at the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.387–392. In proceedings.

Yasuda, T., Ohkura, K. and Taura, T. (2003) 'Cooperative behaviour acquisition mechanism for a multi-robot system based on reinforcement learning in continuous space', Paper presented at the *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, July 16–20, Kobe, Japan. In proceedings.