

Dynamic Correlation Matrix based Multi-Q Learning for a Multi-Robot System

Hongliang Guo and Yan Meng, *Member, IEEE*

Abstract—Multi-robot reinforcement learning is a very challenging area due to several issues, such as large state spaces, difficulty in reward assignment, nondeterministic action selections, and difficulty in merging learned experiences from other robots. In this paper, we propose a dynamic correlation matrix based multi-Q learning (DCM-MultiQ) method for a distributed multi-robot system. A novel dynamic correlation matrix is proposed, which not only handles each agent's Q value, but also deals with the correlation among agents. Furthermore, a theoretical proof of the convergence of the proposed DCM-MultiQ algorithm is also provided using a feedback matrix control theory. To evaluate the efficiency of the proposed DCM-MultiQ method, several case studies of a multi-robot system in forage tasks have been conducted. The simulation results show the efficiency and convergence of the proposed method.

I. INTRODUCTION

MULTI-AGENT system (MAS) has drawn considerable attentions to entrepreneurs and researchers in the last two decades, since it can be used to fulfill tasks that are quite difficult or even unfeasible to be accomplished by a single robot, especially in the presence of uncertainties, incomplete information, distributed control, and asynchronous computation.

However, there exist many challenging issues in MRSs [1]. These challenges often involve the realization of basic behaviors, such as dynamic task allocation, robot coordination, and team reasoning, etc. Furthermore, MAS is usually bedeviled by the dimensionality when facing the scaling problems. The computation complexity [2] often grows exponentially with the number of agents in MAS. As an emerging field of Artificial Intelligence (AI), MAS aims at providing both principles for the constructions of complex systems involving multiple agents and mechanisms for coordination of independent agent's behaviors [3].

In this paper, we will investigate a multi-robot learning problem. More specifically, we will train a multi-robot system through Reinforcement Learning (RL) [4] until the multi-robot system can smoothly accomplish their job by themselves. For example, a job could be searching and

pushing a box together to the exit of an area. Reinforcement learning is an active area of machine learning research which has received considerable and increasing attention in the last two decades. It is prospective since it assumes that the state space as well as the action space in a given scenario can be divided into discrete ones, and agents in that scenario can learn the optimal policy through reward perceiving. This idea outperforms the traditional control theory which needs to know the model of the environment first.

Reinforcement learning and dynamic programming based methods for optimal control problems have been studied extensively [5, 6, 7, 8]. However, much less work has been conducted on distributed multi-agent reinforcement learning due to the difficulty of formulating and analyzing in theory [9]. Applying single-agent Q-learning methods to a multi-agent system directly may not be an efficient way for MAS since there is no coordination among the agents.

Therefore, to solve the scalability problem in multi-Q learning method, it is desirable to develop a distributed multi-Q learning algorithm, where each agent makes its own decision (instead of a central unit) based on its own Q value as well as its neighbors' Q values, instead of all Q values in centralized systems or only its own Q values in single-agent Q learning. Based on this motivation, in this paper, we present a novel dynamic correlation matrix based multi-Q learning (DCM-MultiQ) approach, which focuses on the cooperation between agents. This DCM-Multi-Q algorithm combines multi-Q learning with dynamic correlation matrix to dynamically represent the Q values of all agents as well as their correlations. A feedback Matrix theory is proposed to update the Q values in the matrix. Based on the Matrix theory, the convergence of the proposed DCM-MultiQ method for a multi-agent system can be proved theoretically. Furthermore, the algorithm itself shows that the computation complexity increases linearly rather than exponentially with the number of agents in MAS.

The paper is organized as follows. Related work is discussed in Section II. The DCM-MultiQ approach is proposed in Section III. To evaluate the proposed method, several case studies of a multi-robot system have been conducted, where multiple robots are searching for a fixed/random box in a grid area. The experimental results of these case studies are shown in Section IV. Conclusion and future work are discussed in Section V.

H. Guo is a graduate student in the Department of Electrical and Computer Engineering, Stevens Institute of Technology, NJ 07030, USA. (e-mail: hguo@stevens.edu).

Y. Meng is with Department of Electrical and Computer Engineering, Stevens Institute of Technology, NJ 07030, USA (Phone: (201) 216-5496, email: yan.meng@stevens.edu).

II. RELATED WORK

Agogino and Tumor [10] proposed a method dealing with multi-agent reinforcement learning, which provides agents with immediate counterfactual rewards. This method maintains the accuracy while increases the efficiency of reward function. Hysteretic Q-learning method proposed in [11] has improved traditional Q-learning through varying learning rates, which aims at forcing agents to converge to an optimal policy. Kaya [12] proposed a modular fuzzy approach which handles the continuous state and action space problem. Liu [13] proposed a strategy that combines reinforcement learning with behavior based control networks. Chalkiadakis [14] examines multi-agent reinforcement learning through a probabilistic point of view. He indicated that the joint actions of multi-robots in a given scenario should be determined sequentially with Bayesian network. By offering a Bayesian model in a multi-agent system, the system can reach the optimal equilibrium within relatively short time.

Littman [15] extends the single agent framework of Markov Decision Process (MDP) into a multi-agent scenario, which is called Markov games or Stochastic games (SG). Furthermore, under SG framework, he proposed a Minimax-Q learning algorithm for zero-sum games in which learning player maximizes its payoffs in the worst situation. Later, Littman [16] combined Minimax and Nash Q learning algorithm, and proposed Friend-or-Foe Q-learning algorithm. Greenwald [17] proposed a Correlated-Q learning algorithm, which aims at finding Nash equilibrium in an efficient way. Hu and Wellman [18, 19] extended the zero-sum framework of Littman [15] to general-sum games and developed a Nash-Q learning algorithm for multi-agent reinforcement learning.

However, extensive computational complexity and difficulty in scalability make the game-based MAS methods hard to be applied to large scale MASs. Later, with more and more researchers working on multi-agent reinforcement learning [23, 24], they inevitably find that MAS is hard to converge, at least lacks theoretical proof of convergence despite of some well-behaved experiments. In this paper, we propose a DCM-MultiQ algorithm which deals with a distributed multi-agent system using a correlated Matrix form, and the explicit theoretical proof of convergence is also provided.

III. THE DCM-MULTI Q APPROACH

A. Standard Q-learning Method

RL research is mostly based on the formalism of Markov Decision Processes (MDPs) [15]. Q-learning method is based on the Dynamic Programming but with the expected immediate reward and the expected maximum action-value of the successor state [20]. It is perhaps one of the most widely used algorithms in Reinforcement learning. Suppose the agent observes a current state s , executes action a , receives

immediate reward r , and then observes a next state s' . The Q-learning algorithm updates the current estimate, $Q_k(s, a)$, using the following equation.

$$Q_{k+1}(s, a) = (1 - \alpha_k)Q_k(s, a) + \alpha_k[r + \gamma \max_{a \in A} Q_k(s', a)] \quad (1)$$

where α_k is the learning rate and γ is the decaying factor.

The values of all the other state-action pairs remain unchanged during this update. If the action values of all admissible state-action pairs are updated infinitely often, and α_k decays with increasing k while obeying the usual stochastic approximation conditions, then $\{Q_k\}$ converges to Q^* with probability of 1.

B. Dynamic Correlation Matrix based Multi-Q Learning

For a typical multi-robot system, cooperative learning between robots is very important for the overall global system performance. To tackle this problem, a dynamic correlation matrix based multi-Q learning (DCM-MultiQ) method is proposed here, which is a decentralized cooperative reinforcement learning method. Basically, the model of DCM-MultiQ method is based on Stochastic Games [15]. Inspired by the distributed value functions proposed in [9], a single robot's Q values is updated with its own rewards as well as some weights of other robots' value function, as shown in the following equations:

$$Q_i(s_i, a_i) = (1 - \alpha)Q_i(s_i, a_i) + \alpha(R_i(s_i, a_i) + \gamma \sum_j f(i, j)V_j(s_j)) \quad (2)$$

$$\text{Where } V_i(s_i) = \max_{a_i \in A_i} Q_i(s_i, a_i) \quad (3)$$

where $Q_i(s_i, a_i)$ is the Q value of robot i at a given state-action pair (s_i, a_i) . $R_i(s_i, a_i)$ is the immediate reward that robot i can get with the state-action pair (s_i, a_i) . α is the learning rate and γ is the decaying factor. $f(i, j)$ is the corresponding weight of each robot's value function. $V_i(s_i)$ is the state value of robot i , which is equal to the largest Q value at state s_i .

Instead of updating Q values individually by each agent in [9], we use a matrix to represent Q values for each individual robot as well as the correlated Q values of other robots. Apparently, we should have $\sum_j f(i, j) = 1$. We define $f(i, j)$

as $1/n$, where n is the number of robots in a given environment.

First, we define $\Delta Q_i(s_i, a_i)$ as the difference of Q value of robot i between time step i and $i+1$, which can be represented as

$$\Delta Q_i(s_i, a_i) = Q_{i+1}(s_i, a_i) - Q_i(s_i, a_i) \quad (4)$$

Then we apply equation (2) to equation (4), so we have

$$\Delta Q_i(s_i, a_i) = -\alpha Q_i(s_i, a_i) + \alpha(R_i(s_i, a_i) + \gamma \frac{1}{n} \sum_{j=1}^n V_j(s_j)) \quad (5)$$

Then we expand equation (5) to a series of equations for $i = 1, 2, \dots, n$, and rewrite it into a uniform equation as:

$$\Delta \mathbf{Q}(\mathbf{s}, \mathbf{a}) = \mathbf{R}\alpha + \mathbf{W}\mathbf{V}(\mathbf{s}) - \alpha \mathbf{Q}(\mathbf{s}, \mathbf{a}) \quad (6)$$

where

$$\Delta \mathbf{Q}(\mathbf{s}, \mathbf{a}) = [\Delta Q_1(s_1, a_1), \Delta Q_2(s_2, a_2), \dots, \Delta Q_n(s_n, a_n)]^T$$

$$\mathbf{R} = [R_1(s_1, a_1), R_2(s_2, a_2), \dots, R_n(s_n, a_n)]^T$$

$$\mathbf{V}(\mathbf{s}) = [V_1(s_1), V_2(s_2), \dots, V_n(s_n)]^T$$

$$\mathbf{Q}(\mathbf{s}, \mathbf{a}) = [Q_1(s_1, a_1), Q_2(s_2, a_2), \dots, Q_n(s_n, a_n)]^T$$

and a $n \times n$ matrix $\mathbf{W} = \begin{bmatrix} \alpha \frac{\gamma}{n} & \alpha \frac{\gamma}{n} & \dots & \alpha \frac{\gamma}{n} \\ \alpha \frac{\gamma}{n} & \alpha \frac{\gamma}{n} & \dots & \alpha \frac{\gamma}{n} \\ \dots & \dots & \dots & \dots \\ \alpha \frac{\gamma}{n} & \alpha \frac{\gamma}{n} & \dots & \alpha \frac{\gamma}{n} \end{bmatrix}$.

Note that after sufficient iterations, each learning robot will choose the action that holds the largest Q value, which is the same as the state value. From that time on, $\mathbf{V}(\mathbf{s})$ should be equal to $\mathbf{Q}(\mathbf{s}, \mathbf{a})$. Then Equation (6) can be rewritten as follows:

$$\begin{aligned} \Delta \mathbf{Q}(\mathbf{s}, \mathbf{a}) &= \mathbf{R}\alpha + \mathbf{W}\mathbf{Q}(\mathbf{s}, \mathbf{a}) - \alpha \mathbf{Q}(\mathbf{s}, \mathbf{a}) \\ &= \mathbf{R}\alpha + (\mathbf{W} - \alpha \mathbf{I})\mathbf{Q}(\mathbf{s}, \mathbf{a}) \end{aligned} \quad (7)$$

Where \mathbf{I} equals the identity matrix. Under that condition, we can make further combination over equation (7), and generates the following state differential equation:

$$\Delta \mathbf{Q}(\mathbf{s}, \mathbf{a}) = \mathbf{A}\mathbf{Q}(\mathbf{s}, \mathbf{a}) + \mathbf{R}\alpha \quad (8)$$

where $\mathbf{A} = \mathbf{W} - \alpha \mathbf{I}$

where \mathbf{A} is a $n \times n$ square matrix, and α is the learning rate.

Therefore, we can say that if a multi-robot system uses the DCM-MultiQ method as its control method, the control system can be represented as a state differential equation by equation (8):

$$\Delta \mathbf{Q}(\mathbf{s}, \mathbf{a}) = \mathbf{A}\mathbf{Q}(\mathbf{s}, \mathbf{a}) + \mathbf{R}\alpha \quad (9)$$

where $\mathbf{A} = \mathbf{W} - \alpha \mathbf{I}$ is a $n \times n$ state matrix, \mathbf{Q} is the state vector, the learning rate α is the input signal, and reward vector \mathbf{R} is the control signal vector. According to the feedback control theory, we can say the Q values of the robots will converge to a stable vector eventually if we can prove the stability of this state differential equation. In other words, the

robots will find an optimal policy vector $\boldsymbol{\pi} = \{\pi_i\}$ for $i = 1, 2, \dots, n$ eventually to work cooperatively and efficiently for a task over time. Therefore, we have to prove the following theorem for the stability of the system first.

Theorem 1: If we have a system with a state differential equation as follows:

$$\Delta \mathbf{Q}(\mathbf{s}, \mathbf{a}) = \mathbf{A}\mathbf{Q}(\mathbf{s}, \mathbf{a}) + \mathbf{R}\alpha,$$

then, state vector \mathbf{Q} will converge to a stable vector eventually if the real parts of the eigenvalue of matrix \mathbf{A} are less than 0.

Proof. It is reasonable to assume that each iteration of equation (7) is short enough so that it can be rewritten into a continuous version as follows:

$$\dot{\mathbf{Q}}(\mathbf{s}, \mathbf{a}) = \mathbf{A}\mathbf{Q}(\mathbf{s}, \mathbf{a}) + \mathbf{R}\alpha. \quad (10)$$

Equation (10) can be treated as a standard feedback control system. According to the feedback control law, this kind of system will be stable if all the poles of the system's transfer function are strictly located on the left side of the complex plane.

From the control theory point of view, the eigenvalues of matrix \mathbf{A} are equal to the poles of the system. Therefore, the convergence problem can be transferred into a standard eigenvalue and eigenvector problem. In other words, if all of the real parts of the eigenvalues of matrix \mathbf{A} are less than 0, we can say that the system is stable. When the system is stable, it means that state vector \mathbf{Q} will converge to a stable vector according to equation (10). The characteristic equation of the vector differential equation (10) is defined as

$$\det(\lambda \mathbf{I} - \mathbf{A}) = 0 \quad (11)$$

where \mathbf{I} equals the identity matrix. The characteristic equation is a polynomial in λ . If all of the roots of the characteristic equation have negative real parts (i.e., $\text{Re}(\lambda_i) < 0$), then the system is stable.

Put matrix \mathbf{A} into equation (11), we have

$$\det \begin{bmatrix} \lambda - \alpha(\frac{\gamma}{n} - 1) & -\alpha \frac{\gamma}{n} & \dots & -\alpha \frac{\gamma}{n} \\ -\alpha \frac{\gamma}{n} & \lambda - \alpha(\frac{\gamma}{n} - 1) & \dots & -\alpha \frac{\gamma}{n} \\ \dots & \dots & \dots & \dots \\ -\alpha \frac{\gamma}{n} & -\alpha \frac{\gamma}{n} & -\alpha \frac{\gamma}{n} & \lambda - \alpha(\frac{\gamma}{n} - 1) \end{bmatrix} = 0 \quad (12)$$

Then, from the above polynomial equation, we can easily derive it to the following equation:

$$(\lambda + \alpha)^{n-1} (\lambda + \alpha(1 - \gamma)) = 0 \quad (13)$$

Which means

$$\begin{aligned}\lambda_1 = \lambda_2 = \dots = \lambda_{n-1} = -\alpha, \\ \lambda_n = -\alpha(1-\gamma)\end{aligned}\quad (14)$$

Since $0 < \alpha < 1$, we have $-1 < -\alpha < 0$. According to (14), it means that $-1 < \lambda_1 = \lambda_2 = \dots = \lambda_{n-1} < 0$. Similarly, since $0 < \gamma < 1$, we can easily get $-1 < \lambda_n < 0$ from (14). In other words, $\text{Re}(\lambda_i) < 0$ for $i=1, 2, \dots, n$. Thus we can claim that the system with this state differential equation is stable, i.e., \mathbf{Q} will converge to zero eventually, which means that state vector \mathbf{Q} will converge to a stable vector.

From Equation (8), it can be seen that the DCM-MultiQ algorithm divides the global state and action space into local ones, and each agent only stores its own local Q values. Meanwhile, the proposed algorithm updates each individual's Q values together with other agents' value of that state to collect group learning experience.

Theorem 1 provides a theoretical foundation that the proposed DCM-MultiQ control method can guarantee a convergent learning curve of the multi-robot system over time. More specifically, an optimal cooperative policy vector for multi robots can be achieved to accomplish the task more efficiently after some iterations of training procedure.

C. Action Selection Strategy

The Boltzmann selection method [21] is applied here as the robot action selection strategy. At any state s , an action a is selected with the following probability:

$$p(a) = \frac{e^{\frac{Q(s,a)}{T}}}{\sum_a e^{\frac{Q(s,a)}{T}}}\quad (15)$$

where T is called temperature and defined to be 1 here.

IV. EXPERIMENTAL RESULTS

To evaluate the efficiency and convergence of the proposed DCM-MultiQ method, several case studies of a multi-robot system are studied. Our simulation environment is a self-developed Java-based simulator.

A. Case Study 1: Multi Robots in a Forage Task with a Static Box

It is assumed that the environment can be divided into grids, as shown in Fig. 1. A static box is randomly located within one grid of the environment, where there are some static walls. The environment is totally unknown to the robots except the total number of grids in the environment. The robots can only know the locations of box, walls and exit when they really hit the wall, see the box, or see the exit physically.

The robots have to learn how to avoid walls to find the box and push the box together to the Exit. Both robots start their movement from the Start grid. The robot can only move up, down, left, or right at each grid. If only one robot find the box while the other is still searching, the one who finds the box

has to wait there for the other one. Once both of them find the box, they start to push the box to Exit, where they don't know the position of Exit until they arrive at the Exit grid.

Since it is a distributed system, each robot makes its own movement decision at any time. During the box-moving phase, if the decisions of movement from two robots are different, for example, robot 1 wants to push up while robot 2 wants to push down, the box stays where it is until both robots make the same decision. The objective of this multi-robot system is to use the proposed DCM-MultiQ method to train the robots so that the robots can find an efficient path to detect the box and push the box cooperatively to Exit. Fig. 1 shows a set of snapshots of the learning procedure in simulation.

When both agents find the box, they start to push it. If they make different decision on the movement of the box, the box cannot be pushed, and both agents would get a penalty of -90. If both robots push to the same direction, and the box doesn't arrive at the exit place, both robots will get the regular reward as -1. When the agents finally push the box to Exit, they would get a reward of 180.

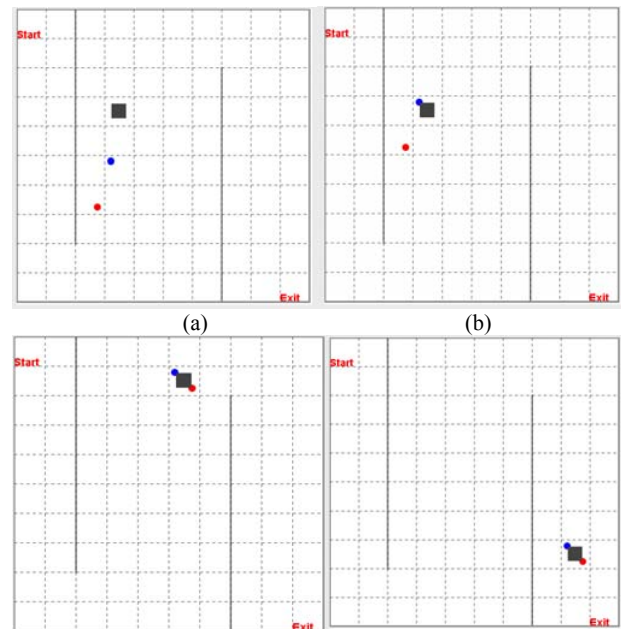


Fig 1. A set of snapshots of multi-robot learning for box searching and pushing in the simulation.

There are two learning phases in the simulation: box detection phase and box pushing phase. During the box detection phase, each robot makes its own decision to detect the box. During the box pushing phase, the robots have to work together to push the box into the same direction. After enough iteration, the robots finally can find the box efficiently and push the box through a shortest path to the Exit.

To show this learning procedure and the convergence of the proposed DCM-MultiQ learning method, the mean values

of $\Delta Q_i(s_i, a)$ for each iteration have been collected. The learning rate is setup as $\alpha = 0.9$ and decline factor is setup as $\gamma = 0.9$. Fig. 2 to Fig. 3 show the simulation results of these mean values. Theoretically, after sufficient rounds of learning episode, the mean value of $\Delta Q_i(s_i, a)$ should decrease to zero, which means that the learning become converge eventually over time.

In Fig. 2, when the robots hit the wall, they would get a penalty of -90, and when they find the box, they would get a reward of 180. So after enough iteration, robots learn how to avoid hitting the walls and the difference of Q value will gradually decrease to zero. Fig. 3 demonstrates the learning curve of robots when they try to push the box. Apparently, robots get a penalty of -90 much more often than the box searching phase. That's because it holds relatively more probability that they get penalty due to making non-cooperative decisions. From Fig. 2 to Fig. 3, it can be seen that after certain times of iteration, the mean value of $\Delta Q_i(s_i, a)$ will decrease to zero gradually, which demonstrate the convergence of the proposed DCM-MultiQ learning algorithm.

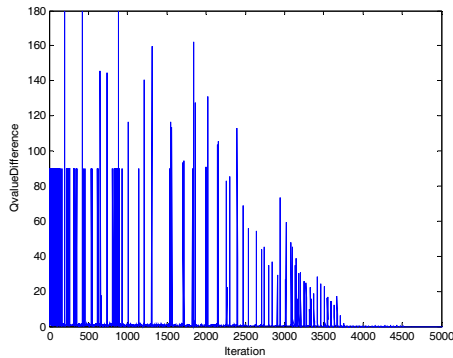


Fig.2. Mean values of $\Delta Q_i(s_i, a_i)$ for two robots during the box-searching phase

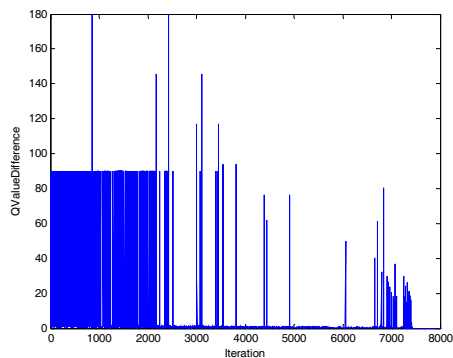


Fig.3. Mean values of $\Delta Q_i(s_i, a_i)$ for two robots during the box-pushing phase

To further evaluate the learning performance, extensive simulations have been conducted to collect the statistic data. In Fig.4, we can see that robots can only finish several tasks

within 1000 steps at the beginning phase of the learning. Then after some learning iteration, the robots can finish much more tasks within 1000 steps. And eventually the curve converges to some constant value.

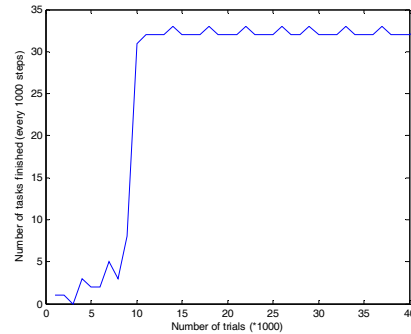


Fig. 4. Number of tasks finished for every 1000 steps.

B. Case Study 2: Multi Robots in a Forage Task with a Randomly Moving Target

To further testify our DCM-MultiQ algorithm, we applied it into a more challenging scenario in a forage task with a randomly moving target.

We use a discrete 10×10 grid environment in which multiple robots have to learn to catch a single randomly moving target. Three groups of simulations have been conducted, where two, four, and eight robots are applied separately in each group. Fig. 5 and Fig. 6 show the snapshots of two-robot case and four-robot cases, respectively.

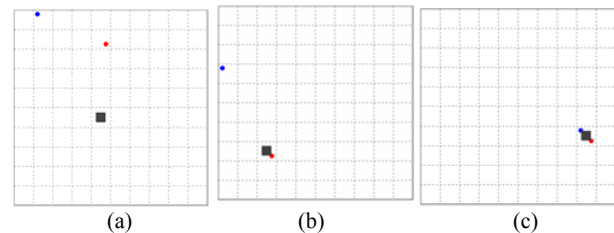


Fig.5. Snapshots of simulations of two robots catching a randomly moving target. Circles represent the robots and black rectangle represents the moving target. (a) Both robots are searching for the target; (b) One robot grabs the target and the other is still searching; (c) Both two robots grab the target (task finished!)

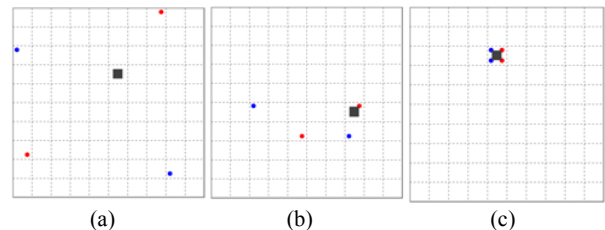


Fig. 6. Snapshots of simulations of four robots catching a randomly moving target. (a) Four robots are searching for the target; (b) One of the robots grabs the target and the other three are still searching; (c) All of the four robots grab the target (task finished!).

The learning rate is setup as $\alpha = 0.9$ and decline factor is

setup as $\gamma = 0.9$. By applying the proposed DCM-MultiQ learning algorithm with the Boltzmann action selection method, the simulation results are shown in Fig. 7.

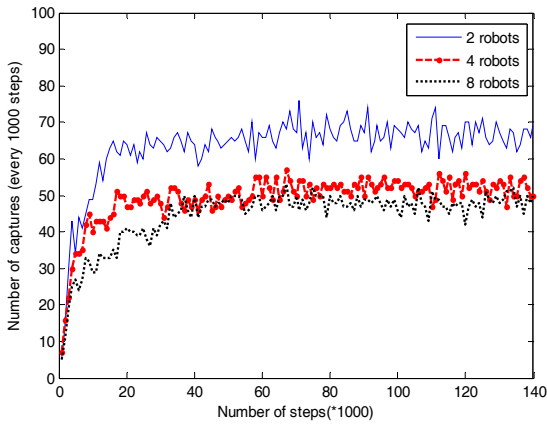


Fig. 7. Number of captures for every 1000 steps using two, four, and eight robots.

From Fig. 7, we can see that the numbers of captures per 1000 trials for all three cases are very low at the beginning of the learning phase. They are increased significantly after some learning time (around 20k steps), then are converged to some stable values with small oscillations. The main reason for the small oscillations on the convergence is due to the random movement of the target.

V. CONCLUSION AND FUTURE WORKS

In this paper, we have presented a novel multi-robot reinforcement learning approach for a distributed cooperative multi-robot system. Since the DCM-MultiQ method is a distributed learning algorithm, it can provide a multi-robot system more robustness and scalability. The computational cost only increases linearly rather than exponentially with the number of robots. It has been demonstrated empirically and theoretically that the proposed DCM-MultiQ algorithm can achieve successful convergence of the learning system, as well as the efficient cooperation among multiple robots.

In the future, we will investigate a more efficient definition of Matrix A. Apparently, a well-defined matrix A can definitely make multi-robot cooperate better. Meanwhile, we will put more weight on the agent's own value function rather than uniform distribution. Furthermore, we will vary the learning rate α and the declining factor γ to reach rapid convergence in our cycle of reinforcement learning as presented in [22].

REFERENCES

[1] E. Yang and D. Gu, "Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey". *Technical Report CSM-404*, Department of Computer Science, University of Essex, 2004

[2] E. Klavins, "Communication Complexity of Multi-Robot Systems" *Springer Tracts in Advanced Robotics*, Volume 7/2003 page 275-292.

[3] P. Stone, M. Veloso "Multiagent Systems: A Survey from a Machine Learning Perspective." *Autonomous Robots*, 8(3):345-383, July 2000.

[4] S. Sutton and G. Barto, "Reinforcement Learning: An Introduction" *The MIT Press Cambridge, Massachusetts London, England*

[5] X. Meng, R. Babuska, L. Busoniu, Y. Chen and W. Tan "An Improved Multiagent Reinforcement Learning Algorithm" pp. 337-343 *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology - Volume 00* Compiegne University of Technology, France, September 19-22, 2005.

[6] N. Suematsu, A. Hayashi "A Multiagent Reinforcement Learning Algorithm using Extended Optimal Response". *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*. Bologna, Italy, July 2002.

[7] S. Kapetanakis, D. Kudenko "Reinforcement learning of coordination in heterogeneous cooperative multi-agent systems," *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*.

[8] D. P. Bertsekas. "Dynamic Programming and Optimal Control". *Athena Scientific, 2001*.

[9] J. Schneider, W. K. Wong, A. Moore and M. Riedmiller "Distributed Value Functions" *Proc. 16th International Conf. on Machine Learning*. Bled, Slovenia, June 27-30, 1999.

[10] A. K. Agogino and K. Tumer, "QUICR-learning for Multi-Agent Coordination," *In Proceedings of the 21st National Conference on Artificial Intelligence*, Boston, MA, July 2006.

[11] L. Matignon, G. J. Laurent and N. L. Fort-Piat, "Hysteretic Q-Learning: An Algorithm for decentralized reinforcement learning in cooperative multi-agent teams", *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Diego, CA, 2007.

[12] M. Kaya, R. Alhajj "Reinforcement Learning in Multiagent Systems: A Modular Fuzzy Approach with Internal Model Capabilities", *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'02)*. Washington, DC, USA, 2002.

[13] Z. Liu, M. H. Ang Jr and W. K. G. Seah "Reinforcement Learning of Cooperative Behaviors for Multi-Robot Tracking of Multiple Moving Targets" *International Conference on Robots and Systems 2005(IEEE/RSJ05)*. Edmonton, Alberta, Canada August 2-6, 2005.

[14] G. Chalkiadakis, C. Boutilier "Coordination in Multiagent Reinforcement Learning: A Bayesian Approach" *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems.(AAMAS'03)* July 14-18, 2003, Melbourne , Australia.

[15] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning" *Proceedings of the Eleventh International Conference on Machine Learning*, pp.157-163

[16] M. L. Littman, "Friend-or-foe Q-learning in general-sum games," *in Proceedings of The 18th International Conference on Machine Learning*, 2001, pp. 322-328.

[17] A. Greenwald and K. Hall "Correlated-Q learning" *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)*,

[18] J. Hu and M.P. Wellman, "Multiagent reinforcement learning in stochastic games", *Proceedings of the Fifteenth International Conference on Machine Learning*, 1998.

[19] J. Hu and M.P. Wellman, "Nash Q-learning for general-sum stochastic games," *Journal of Machine Learning Research*, vol. 4, pp. 287-308, 2000.

[20] A. G. Barto, S. Mahadevan "Recent Advances in Hierarchical Reinforcement Learning", *Discrete Event Dynamic Systems: Theory and Applications*, 13,41-77, 2003

[21] M. Coggan, "Exploration and Exploitation in Reinforcement Learning" *Fourth International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'01)*. Shonan International Village Yokosuka City, Japan Oct 30th - Nov 1st 2001

[22] K.Cousin and G.L.Peterson, "Cooperative Reinforcement Learning using an Expert-Measuring Weighted Strategy with WoLF", *Artificial Intelligence and Soft Computing ~ASC 2005*, Benidorm, Spain, Sept. 12-14, 2005.

[23] M. Lauer, M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems", *Proc. 17th International Conf. on Machine Learning*, Morgan Kaufmann, San Francisco, CA, 2000, pp. 535-542.

[24] M. Bowling, M. Veloso, "Multiagent learning using a variable learning rate", *Artificial Intelligence*, 136, 2002, 215-250.