Cooperative Behavior Acquisition in Multi Mobile Robots Environment by Reinforcement Learning Based on State Vector Estimation

Eiji Uchibe, Minoru Asada and Koh Hosoda
Dept. of Adaptive Machine Systems, Graduate School of Eng.,
Osaka University, Suita, Osaka 565-0871, Japan
uchibe@er.ams.eng.osaka-u.ac.jp

Abstract

This paper proposes a method that acquires the purposive behaviors based on the estimation of the state vectors. In order to acquire the cooperative behaviors in multi robot environments, each learning robot estimates local predictive model between the learner and the other objects separately. Based on the local predictive models, robots learn the desired behaviors using reinforcement learning. The proposed method is applied to a soccer playing situation, where a rolling ball and other moving robots are well modeled and the learner's behaviors are successfully acquired by the method. Computer simulations and real experiments are shown and a discussion is given.

1 Introduction

Building a robot that learns to perform a task through visual information has been acknowledged as one of the major challenges facing Robotics and AI. Reinforcement learning has recently been receiving increased attention as a method for robot learning with little or no a priori knowledge and higher capability of reactive and adaptive behaviors [1].

In multi-agent environments, the conventional reinforcement learning algorithms do not seem applicable because an environment including other learning robots might change randomly from a viewpoint of an individual learning robot. It is important for the learner to understand the strategies of the other robots and to predict their movements in advance to learn the behaviors successfully.

Littman [5] proposed a framework of Markov Games in which learning robots try to learn a mixed strategy optimal against the worst possible opponent in a zero-sum 2-player game in a grid world. He assumed that the opponent's goal is given to the learner. Lin [4] compared window-Q based on both the current sensation and the N most recent sensations and actions with recurrent-Q based on a recurrent network, and he showed the latter is superior to the former because a recurrent network can cope with historical features appropriately. However, it is still difficult to determine the number of neurons and the structures of network in advance. Furthermore, these methods utilize global information.

Robotic soccer is a good domain for studying multi-agent problems [2]. Stone and Veloso proposed layered learning method which consists of two levels of learned behaviors [6]. The lower is for basic skills (ex. interception of a moving ball) and the higher is one which can make decisions (ex. whether or not to make a pass) based on the decision tree. Uchibe et al. proposed a method of modular reinforcement learning which coordinates multiple behaviors taking account of a tradeoff between learning time and performance [7]. Since these methods utilize the current sensor outputs as states, their methods can not cope with the temporal changes of objects.

As described above, these existing learning methods in multi agent environments need good attributes (state vectors) in order for the learning to converge. Therefore, the modeling architecture is required to enable the reinforcement learning to be applied. In this paper, we propose a method which estimates the relationships between a learner's behaviors and other robots through interactions (observation and action) based on the method of system identification. In order to construct the local predictive model of other robots from the result of Canonical Variate Analysis(CVA) [3], we adopt Akaike's Information Criterion(AIC).

We apply the proposed method to a simplified soccer game. The task of the robot is to shoot a ball which is passed back from the other robot. After the learning robot estimates the local predictive models, the reinforcement learning is applied in order to acquire purposive behaviors.

The rest of this article is structured as follows: at first we show our basic idea, then we give brief explanation of the local predictive model and reinforcement learning. The details of the local predictive model and learning algorithms are described in [8] and [7], respectively. Finally, we show simulation results and real experiments and give a discussion.

2 Our Approach

2.1 Architecture

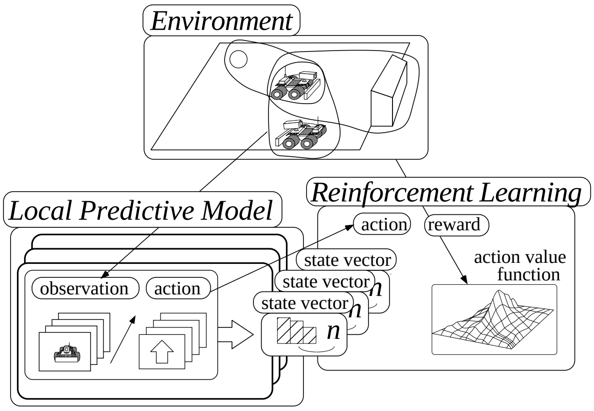


Figure 1: Proposed architecture

Figure 1 shows a learning architecture for each robot. At first, the learning robot constructs the local predictive models from the sequences of not only sensor outputs but also its own action since it needs to acquire the state vectors which can predict future states in dynamic environments. Next, it learns the cooperative behaviors based on the estimated state vectors from the local predictive models. The reason why two phases learning is as follows. Strictly speaking, all the robots do in fact interact with each other. Therefore, the learning robots should construct the local predictive model taking these interactions into account. However, it is intractable to collect the adequate input-output sequences and estimate the proper model because the dimension of state vector increases drastically. Therefore, the learning (observing) robot first estimates the local predictive models to individual (observed) robots or objects in an environment separately and it obtains the higher interactions among robots through the post reinforcement learning process.

2.2 Learning schema

In order to acquire the cooperative behaviors in multi robot environments, we schedule for multi robots reinforcement learning. The actual learning process can be categorized into three ways.

1. Learning the policy in a real environment:
except an easy task in a simple environment, it seems difficult to implement.

2. Learning the policy in computer simulation and policy transfer to a real environment:
since there are still a gap between the simulation environment and the real one, we need some modification in the real experiment.

3. Combination of computer simulation and real experiments:
based on the simulation results, learning in a real environment is scheduled.

We adopt the third one and make a learning schedule (see Figure 2).

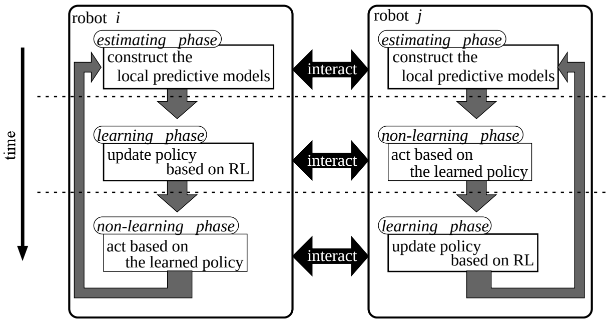

robot *i*
robot *j*
time
estimating phase
construct the local predictive models
interact
learning phase
update policy based on RL
non-learning phase
act based on the learned policy
interact
non-learning phase
act based on the learned policy
learning phase
update policy based on RL
interact

Figure 2: Schedule for learning in multi robots environments

If the multiple robots learn the behaviors simultaneously, the learning process may be unstable, especially in the early stage of learning. Therefore, we plan for the robots to learn the behaviors in turn to make the learning process stable. At first, each robot constructs the local predictive models (*estimating phase*) in the computer simulation. The robots move randomly because they do not have any policies due to initialized zeros in their action value functions. Next, we select one robot to make learn, and fix the action strategy of other robots. The robot executes random actions with a fixed probability or the optimal actions (*learning phase*). Other robots execute actions based on the strategies which are acquired previously (*no-learning phase*). Therefore, the robots except learning robot is stationary in the first period of behavior learning. After the selected robot finishes learning, we select the other robot to learn. We repeat this for robots to acquire the purposive behaviors. Finally, we transfer the result of the computer simulation to the real robots. The robots construct the local predictive models and learns the behavior in the same way. As a result, enormous learning time can be reduced.

3 Local predictive models in the multi agent environment

A number of algorithms to identify multi-input multi-output (MIMO) combined deterministic-stochastic systems have been proposed. Among them, Larimore's Canonical Variate Analysis (CVA) [3] is typical one, which uses canonical correlation analysis to construct a state estimator. We utilize CVA to realize the local predictive model. For more through treatment of CVA, see [3]. Here, we give a brief explanation of CVA method.

CVA uses a discrete time, linear, state space model as follows: Let be the input and output generated by the unknown system

$$\begin{aligned} \boldsymbol{x}(t+1) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t), \end{aligned} \qquad (1)$$

where $\boldsymbol{x}(t)$, $\boldsymbol{u}(t) \in \Re^m$ and $\boldsymbol{y}(t) \in \Re^q$ denote state vector, action code vector, and observation vector respectively. $\boldsymbol{A} \in \Re^{n\times n}$, $\boldsymbol{B} \in \Re^{n\times m}$, $\boldsymbol{C} \in \Re^{q\times n}$, and $\boldsymbol{D} \in \Re^{q\times m}$ represent matrices. CVA estimates a state vector $\boldsymbol{x}$ which is a linear combination of the previous observation and action sequences as follows:

$$\boldsymbol{x}(t) = [\boldsymbol{I}_n \ \ \boldsymbol{0}]\boldsymbol{U}\boldsymbol{p}(t), \qquad (2)$$

where

$$\boldsymbol{p}(t) = [\boldsymbol{u}(t-1) \cdots \boldsymbol{u}(t-l)\, \boldsymbol{y}(t-1) \cdots \boldsymbol{y}(t-l)]^T ,$$

and $\boldsymbol{U} \in \Re^{l(m+q)\times l(m+q)}$ is a matrix which is calculated by CVA.

Figure 3 shows an overview of the local predictive model. The local predictive model estimates the state vector $\boldsymbol{x}$ from the sequences of input $\boldsymbol{u}$ and output $\boldsymbol{y}$. If the model can not obtain the adequate precision, it increases the historical length l to improve the model. Next, It reduces the order of the estimated state vector n based on the Akaike's Information Criterion (AIC) to make the size of the state space tractable. By approximating the relationships between the learner's action and the resultant observation, the local predictive model gives the learning agent not only the successive state of the agent but also the priority of the state vectors, which means that the validity of the state vector with respect to the prediction.

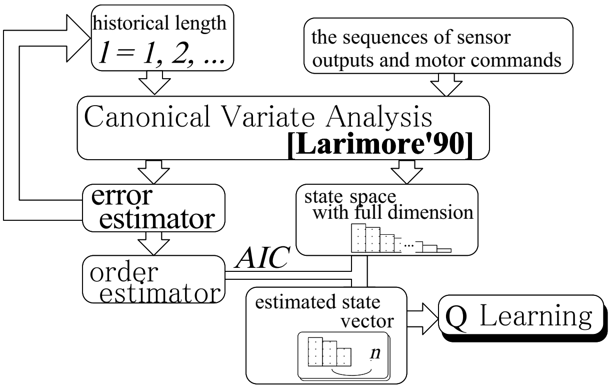


Figure 3: Local predictive model

4 Reinforcement learning based on the local predictive models

Since the local predictive model merely represents the local interaction between the learner and one of other objects separately, the learning robot has to estimate the global interaction among models and decide to take actions to accomplish given tasks.

In the following, we give a brief explanation of Q learning and modular reinforcement learning to accelerate the learning time with multiple goals.

4.1 Q learning

A Q learning method provides robots with the capability of learning to act optimally in a Markovian environment. A simple version of Q learning algorithm is shown as follows:

1. Initialize $Q(x,u)$ to 0s for all combination of $\boldsymbol{X}$ and $\boldsymbol{U}$.

2. Perceive current state x.

3. Choose an action u according to the action value function.

4. Execute an action u in the environment. Let the next state be x' and immediate reward be r.

5. Update the action value function from x, u, x', and r,

$$\begin{aligned} Q_{t+1}(x,u) &= (1-\alpha_t)Q_t(x,u) \\ &+ \alpha_t(r + \gamma \max_{u' \in \boldsymbol{U}} Q_t(x',u')) \end{aligned} \quad (3)$$

   where α_t is a learning rate parameter and γ is a fixed discounting factor between 0 and 1.

6. Return to 2.

4.2 Modular reinforcement learning

Since the time needed to acquire an optimal behavior mainly depends on the size of the state space, it seems difficult to apply the normal Q learning to multiple tasks. Therefore, we use the modular reinforcement learning method [7].

Figure 4 shows the basic idea of the modular reinforcement learning, where the number of the tasks n is two for the sake of reader's understanding. In order to reduce the learning time, the whole state space $\boldsymbol{X}$ is classified into two categories based on the maximum action values separately obtained by Q learning: the area where one of the learned behaviors is directly applicable (*no more learning area*), and the area where learning is necessary due to the competition of multiple behaviors (*re-learning area*). Eventually the whole state space $\boldsymbol{X}$ is classified into the no more learning area $\boldsymbol{X}_i$, $i = 1 \cdots n$ and the re-learning area $\boldsymbol{X}_{rl}$. These areas are exclusive.

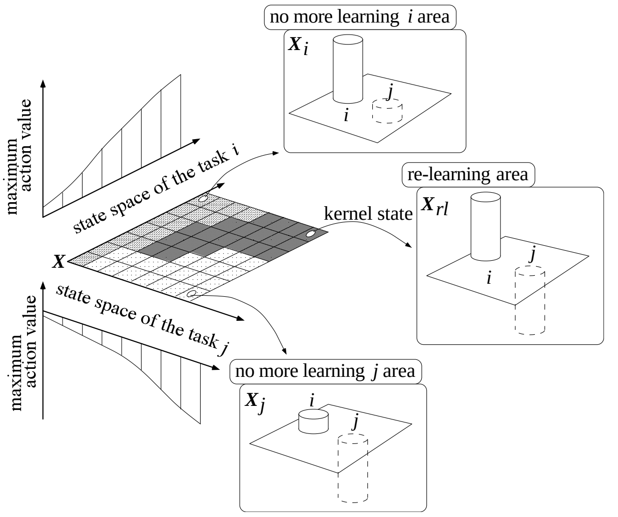


Figure 4: Basic idea of the modular reinforcement learning

In the case of states belonging to the no more learning area, the learning robot uses the action value functions which are acquired previously since it does not need to update action value function any more. If the learning robot is in the re-learning area, the robot estimates the discounted value γ to learn the action value function appropriately. As a result, the modular reinforcement learning can take account of a tradeoff between the learning time and performance when the robot coordinates multiple behaviors.

5 Experiments

5.1 Task and assumptions

We apply the proposed method to a simplified soccer game including two mobile robots (Figure 5). Each robot has a single color TV camera and does not know the locations, the sizes and the weights of the ball and the other agent, any camera parameters such as focal length and tilt angle, or kinematics/dynamics of itself. They move around using a 4-wheel steering system. The effects of an action against the environment can be informed to the agent only through the visual information. As motor commands, each agent has 7 actions such as go straight, turn right, turn left, stop, and go backward. Then, the input $\boldsymbol{u}$ is defined as the 2 dimensional vector as

$$\boldsymbol{u}^T = [v \;\; \phi], \quad v, \phi \in \{-1, 0, 1\},$$

where v and ϕ are the velocity of motor and the angle of steering respectively and both of which are quantized.

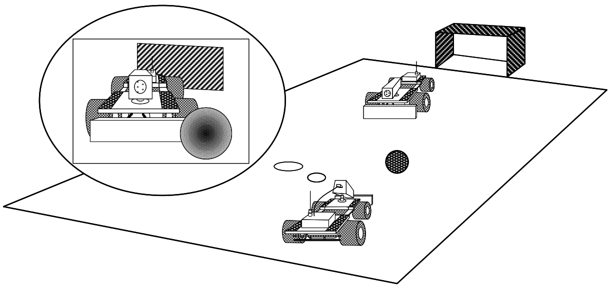
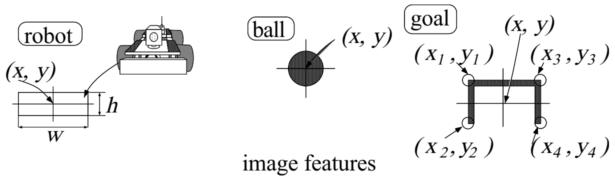


Figure 5: The environment and our mobile robot



robot	ball	goal
area	area	area
center position	center position	center position
height	radius	4 corners
width		

Figure 6: Image features of the ball, goal, and agent

Table 1: The estimated dimension

observer	target	l	n	$\log \lvert \boldsymbol{R} \rvert$	AIC
computer simulation					
shooter	ball	2	4	0.23	138
	goal	1	2	-0.01	121
	passer	3	6	1.22	210
passer	ball	2	4	0.78	142
	shooter	3	5	0.85	198
real experiments					
shooter	ball	4	4	1.88	284
	goal	1	3	-1.73	-817
	passer	5	4	3.43	329
passer	ball	4	4	1.36	173
	shooter	5	4	2.17	284

Table 2: Performance result in real experiments

	before learning	after learning
success of shooting	57/100	32/50
success of passing	30/100	22/50
number of collisions	25/100	6/50
average steps	563	483

The output (observed) vectors are shown in Figure 6. As a result, the dimensions of the observed vector about the ball, the goal, and the other robot are 4, 11, and 5 respectively.

5.2 Computer simulation and real experiments

At first, the shooter and the passer construct the local predictive models for the ball, the goal, and the other robot in computer simulation. Next, the passer begins to learn the behaviors under the condition that the shooter is stationary. After the passer has finished its learning, we fix the policy of the passer. Then, the shooter starts to learn shooting behaviors. We assign a reward value 1 when the shooter shoots a ball into the goal and the passer passes the ball toward the shooter. Further, a negative reward value -0.3 is given to the robots when a collision between two robots is happened. In these processes, the modular reinforcement learning is applied for shooter (passer) to learn shooting (passing) behaviors and avoiding collisions.

Next, we transfer the result of computer simulation to the real environments. In order to construct the local predictive models in the real environment, the robot selects actions using the probability based on the semi uniform undirected exploration. In other words, the robot executes random actions with a fixed probability (20 %) and the optimal actions learned in computer simulation (80 %). We performed 100 trials in real experiments. After the local predictive models are updated, the robots improve the action value function again based on the obtained real data. If the local predictive model in the real environment increases the estimated order of the state vector, the action value functions are initialized based on the action value functions in computer simulation in order to accelerate the learning. Finally, we performed 50 trials to check the result of learning in the real environment.

Table 1 shows the result of the estimated state vectors in computer simulation and real experiments, where $\log |\boldsymbol{R}|$ and AIC denote the logarithm of covariance matrix of error of the local predictive model and Akaike's information criterion, respectively. In order to predict the successive situation, $l = 1$ is sufficient for the goal, while the ball needs 2 steps. We suppose the reasons why the estimated orders of state vectors are different between computer simulation and real experiments are :

- because of noise, the prediction error of real experiments is much larger than that of computer simulation, and
- in order to collect the sequences of observation and action, the robots do not select the random action but move based on the result of computer simulation. Therefore, the experiences of passer and shooter are quite different from each other.

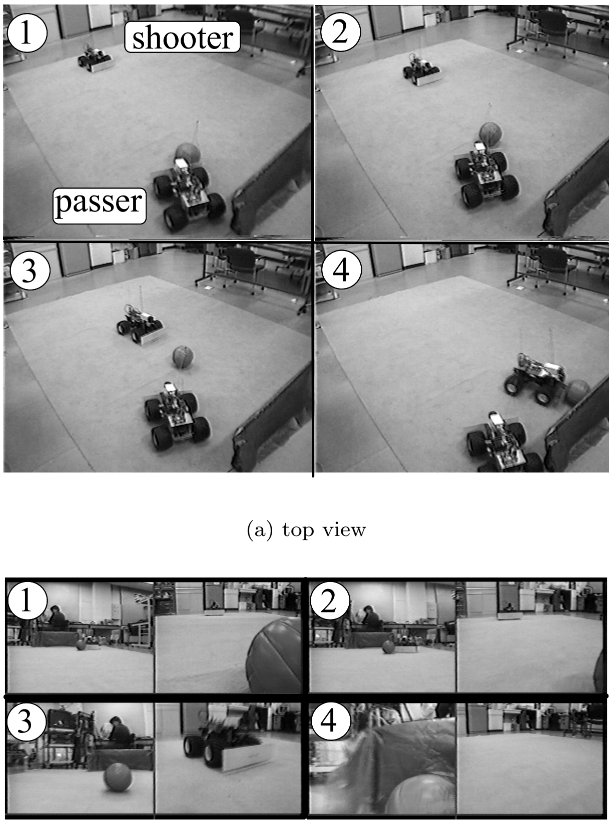
As a result, the historical length l of the real experiments is larger than that of the computer simulation. On the other hand, the estimated order of state vector n for the other robot of real experiments is smaller than that of computer simulation since the components for higher and more complicated interactions can not be discriminated from noise in the real environments.

Table 2 shows the comparison of performance between the simple transfer of the result of computer simulation and the result of re-learning in real environments. We checked what happened if we replace the local predictive models between the passer and the shooter. Eventually, large prediction errors of both sides were observed. Therefore the local predictive models can not be replaced between physical agents. Figure 7 shows a sequence of images where the shooter shoots a ball which is kicked by the passer.

6 Concluding remarks

This paper proposes a method of behavior acquisition so as to apply the reinforcement learning to the multi robot environments. Our method takes account of the tradeoff among the precision of prediction, the dimension of state vector, and the length of steps to predict. The local predictive model can also be applied to controlling the enviromental complexity so as to learn the policy efficiently [9].

In the current system, we consider just two robots, and regard that the current system can cope with global interactions. However, more robots in the field we have, more complicated and higher interactions occur. As future works, we challenge to extend our method when more than two robots learn cooperative and competitive behaviors.



(a) top view

(b) obtained images (left:shooter, right:passer)

Figure 7: Acquired behavior

References

[1] J. H. Connel and S. Mahadevan. *Robot Learning*. Kluwer Academic Publishers, 1993.

[2] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. Robocup a challenge problem for ai. *AI Magazine*, 18(1):73–85, 1997.

[3] W. E. Larimore. Canonical variate analysis in identification, filtering, and adaptive control. In *Proc. 29th IEEE Conference on Decision and Control*, pp. 596–604, Honolulu, Hawaii, December 1990.

[4] L.-J. Lin and T. M. Mitchell. Reinforcement learning with hidden states. In *Proc. of the 2nd International Conference on Simulation of Adaptive Behavior: From Animals to Animats 2.*, pp. 271–280, 1992.

[5] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proc. of the 11th International Conference on Machine Learning*, pp. 157–163, 1994.

[6] P. Stone and M. Veloso. Using machine learning in the soccer server. In *Proc. of IROS-96 Workshop on Robocup*, 1996.

[7] E. Uchibe, M. Asada, and K. Hosoda. Behavior coordination for a mobile robot using modular reinforcement learning. In *Proc. of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1329–1336, 1996.

[8] E. Uchibe, M. Asada, and K. Hosoda. State space construction for behavior acquisition in multi agent environments with vision and action. In *Proc. of International Conference on Computer Vision*, pp. 870–875, 1998.

[9] E. Uchibe, M. Asada, and K. Hosoda. Environmental complexity control for vision-based learning mobile robot. In *Proc. of IEEE International Conference on Robotics and Automation*, 1998.