Strategy Classification in Multi-agent Environment — Applying Reinforcement Learning to Soccer Agents —

Eiji Uchibe, Minoru Asada, and Koh Hosoda
Dept. of Mech. Eng. for Computer-Controlled Machinery
Osaka University, 2-1, Yamadaoka, Suita, Osaka 565, Japan
e-mail:uchibe@robotics.ccm.eng.osaka-u.ac.jp
URL:http://www-robotics.ccm.eng.osaka-u.ac.jp/user/uchibe

Abstract

This paper proposes a method for agent behavior classification which estimates the relations between the learner's behaviors and the other agents in the environment through interactions using the method of system identification. In order to identify the model of each agent, Akaike's Information Criterion(AIC) is applied to the result of Canonical Variate Analysis(CVA). Next, reinforcement learning based on the estimated state vectors is used in order to obtain the optimal behavior. The proposed method is applied to soccer playing robots. Unlike our previous work, the method can cope with a rolling ball. Computer simulations and preliminary experiments are shown and the discussion is given.

Introduction

Building a robot that learns to perform a task 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 (Connel & Mahadevan 1993). In our previous work (Uchibe, Asada, & Hosoda 1996), we proposed a method of modular reinforcement learning which coordinates multiple behaviors taking account of a trade-off between learning time and performance.

In a multi-agent environment, the standard reinforcement learning algorithm does not seem applicable because the environment including the other learning agents seems to change randomly from a viewpoints of the learning agent. We suppose that there are two major reasons why the learning would be difficult in a multi-agent environment.

A The other agent may use a stochastic action selector which could take a different action even if the same sensation occurs to it.

B The other agent may have a perception (sensation) different from the learning agent's. This means that the learning agent would not be able to discriminate different situations which the other agent can do, and vice versa.

Therefore, the learner cannot predict the other agent behaviors correctly even if its policy is fixed unless explicit communication is available. It is important for the learner to understand the strategies of the other agents and to predict their movements in advance to learn the behaviors successfully.

Littman (Littman 1994) proposed the framework of Markov Games in which Q-learning agents 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 strategy is given to the learner (the opponent tries to minimize a single reward function, while it is to be maximized by the learning agent). Sandholm and Crites (Sandholm & Crites 1995) studied the ability of a variety of Q-learning agents to play iterated prisoner's dilemma game against an unknown opponent. They showed that adequate previous moves and sensations are needed in order to learn successfully.

Lin (Lin & Mitchell 1992) 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. However, generally speaking it is still difficult to determine the number of neurons and the structures of network in advance.

As described above, existing methods in multi agent environments need the assumption that the policy of other agent is fixed and known to the learner in order for the learning to converge. Therefore, the classification architecture is required to apply the reinforcement learning. However, what the learning agent can do is to collect all the observed data with motor commands taken during the observation and to estimate the relationship between the observed agents and the learner's behaviors in order to take an adequate behavior although it might not be guaranteed as optimal because of *partial observation* due to the limitation of sensing capability. In this paper, we propose a method which estimates the relations between the learner's behaviors and the other agents through interactions using the method of system identification. Here, we put our em-

phasis on the problem **B**, and we assume that the other agent does not change the strategy. In order to identify the model of each other agent, we apply Akaike's Information Criterion(AIC) (Akaike 1974) to the result of Canonical Variate Analysis(CVA) (Larimore 1990), which is widely used in the field of system identification.

We apply the proposed method to a simple soccer-like game including two active agents. The task of the agent is to discriminate the strategy of the other agents. Here, the other agents consist of the stationary agent (the goal and the line), passive agent (the ball) and active agent (the opponent). After the model identification, we apply reinforcement learning in order to acquire shooting and passing behaviors. In our previous work (Asada *et al.* 1995; Uchibe, Asada, & Hosoda 1996), the changes in size and position of the ball are not considered, therefore the agent could not acquire optimal behavior when the ball is rolling. However, the proposed method can cope with the moving ball because state vector for learning is selected appropriately so as to predict the successive steps. Simulation results and preliminary real experiments are shown and the discussion is given.

Agent Classification

Canonical Variate Analysis(CVA)

In order to succeed in learning, it is necessary for the learner to predict the subsequent situations as mentioned above. In the following, we consider to utilize a method of system identification, regarding motor command and observation results as the input and the output of the system respectively.

A number of algorithms to identify multi-input multi-output (MIMO) combined deterministic-stochastic systems have been proposed. In contrast to 'classical' algorithms such as PEM (Prediction Error Method), the subspace system identification algorithms (Van Overschee & De Moor 1995) do not suffer from the problems caused by a priori parameterizations. Larimore's Canonical Variate Analysis (CVA) (Larimore 1990) is one of such algorithms, which uses canonical correlation analysis to construct a state estimator.

Let $\boldsymbol{u}(t) \in \Re^m$ and $\boldsymbol{y}(t) \in \Re^q$ be the input and output generated by the unknown system

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

with

$$E\left\{ \begin{bmatrix} \boldsymbol{w}(t) \\ \boldsymbol{v}(t) \end{bmatrix} \begin{bmatrix} \boldsymbol{w}^T(\tau) & \boldsymbol{v}^T(\tau) \end{bmatrix} \right\} = \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{S} \\ \boldsymbol{S}^T & \boldsymbol{R} \end{bmatrix} \delta_{t\tau},$$

and $\boldsymbol{A}, \boldsymbol{Q} \in \Re^{n\times n}$, $\boldsymbol{B} \in \Re^{n\times m}$, $\boldsymbol{C} \in \Re^{q\times n}$, $\boldsymbol{D} \in \Re^{q\times m}$, $\boldsymbol{S} \in \Re^{n\times q}$, $\boldsymbol{R} \in \Re^{q\times q}$. $E\{\cdot\}$ denotes the expected value operator and $\delta_{t\tau}$ the Kronecker delta. $\boldsymbol{v}(t) \in \Re^q$ and $\boldsymbol{w}(t) \in \Re^n$ are unobserved, Gaussian-distributed, zero-mean, white noise vector sequences. Generally speaking, CVA uses a new vector $\boldsymbol{\mu}$ which is a linear combination of the previous input-output sequences since it is difficult to determine the dimension of $\boldsymbol{x}$. Eq.(1) is transformed as follows:

$$\begin{bmatrix} \boldsymbol{\mu}(t+1) \\ \boldsymbol{y}(t) \end{bmatrix} = \boldsymbol{\Theta} \begin{bmatrix} \boldsymbol{\mu}(t) \\ \boldsymbol{u}(t) \end{bmatrix} + \begin{bmatrix} \boldsymbol{T}^{-1}\boldsymbol{w}(t) \\ \boldsymbol{v}(t), \end{bmatrix}, \tag{2}$$

where

$$\hat{\boldsymbol{\Theta}} = \begin{bmatrix} \boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T} & \boldsymbol{T}^{-1}\boldsymbol{B} \\ \boldsymbol{C}\boldsymbol{T} & \boldsymbol{D} \end{bmatrix}, \tag{3}$$

and $\boldsymbol{x}(t) = \boldsymbol{T}\boldsymbol{\mu}(t)$. Therefore, so we can regard $\boldsymbol{\mu}(t)$ as a new state vector, we use $\boldsymbol{\mu}$ as same as $\boldsymbol{x}$ hereafter. CVA can estimate parameter matrix $\boldsymbol{\Theta}$. For more through treatment, see (Larimore 1990). We follow the simple explanation of the CVA method (Larimore 1990).

CVA Algorithms

1. For $\{\boldsymbol{u}(t), \boldsymbol{y}(t)\}$, $t = 1, \cdots N$, construct new vectors

$$\boldsymbol{p}(t) = \begin{bmatrix} \boldsymbol{u}(t-1) \\ \vdots \\ \boldsymbol{u}(t-l) \\ \boldsymbol{y}(t-1) \\ \vdots \\ \boldsymbol{y}(t-l) \end{bmatrix}, \quad \boldsymbol{f}(t) = \begin{bmatrix} \boldsymbol{y}(t) \\ \boldsymbol{y}(t+1) \\ \vdots \\ \boldsymbol{y}(t+k-1) \end{bmatrix},$$

2. Compute estimated covariance matrices $\hat{\boldsymbol{\Sigma}}_{pp}$, $\hat{\boldsymbol{\Sigma}}_{pf}$ and $\hat{\boldsymbol{\Sigma}}_{ff}$, where $\hat{\boldsymbol{\Sigma}}_{pp}$ and $\hat{\boldsymbol{\Sigma}}_{ff}$ are regular matrices.
3. Compute singular value decomposition

$$\begin{aligned} \hat{\boldsymbol{\Sigma}}_{pp}^{-1/2}\hat{\boldsymbol{\Sigma}}_{pf}\hat{\boldsymbol{\Sigma}}_{ff}^{-1/2} &= \boldsymbol{U}_{aux}\boldsymbol{S}_{aux}\boldsymbol{V}_{aux}^T, \\ \boldsymbol{U}_{aux}\boldsymbol{U}_{aux}^T = \boldsymbol{I}_{l(m+q)},\ \boldsymbol{V}_{aux}\boldsymbol{V}_{aux}^T &= \boldsymbol{I}_{kq}, \end{aligned} \tag{4}$$

   and $\boldsymbol{U}$ is defined as:

$$\boldsymbol{U} := \boldsymbol{U}_{aux}^T\hat{\boldsymbol{\Sigma}}_{pp}^{-1/2}.$$

4. The n dimensional new vector $\boldsymbol{\mu}(t)$ is defined as:

$$\boldsymbol{\mu}(t) = [\boldsymbol{I}_n\ 0]\boldsymbol{U}\boldsymbol{p}(t), \tag{5}$$

5. Estimate the parameter matrix $\boldsymbol{\Theta}$ applying least square method to Eq (2).

Classify Agents in an Environment

It is important to decide the dimension n of the state vector $\boldsymbol{x}$ and l (lag operator) when we apply CVA to the classification of agents. Although the estimation is improved if l is larger and larger, much more history information is necessary. However, it is desirable that l is as small as possible with respect to the memory size. For n, we have to take account of the trade off between the number of parameters and the precision of estimation.

In order to determine n, we apply Akaike's Information Criterion (AIC). Let the prediction error be $\boldsymbol{\varepsilon}$ and covariance matrix of error be

$$\hat{\boldsymbol{R}} = \frac{1}{N-k-l+1} \sum_{t=l+1}^{N-k+1} \boldsymbol{\varepsilon}(t)\boldsymbol{\varepsilon}^T(t).$$

Therefore $AIC(n)$ is calculated by

$$AIC(n) = (N-k-l+1)\log|\hat{\boldsymbol{R}}| + 2\lambda(n), \quad (6)$$

where

$$\lambda(n) = n(2p+m) + pm + \frac{1}{2}p(p+1). \quad (7)$$

The optimal dimension n^* is defined as

$$n^* = \arg\min AIC(n),$$

where

$$1 \le n \le \min(l(m+q), kq).$$

However, the parameter l is not under the influence of the $AIC(n)$. Therefore, we utilize $\log|\hat{\boldsymbol{R}}|$ to determine l.

Agent Classification Procedure

1. Memorize the q dimensional vector $\boldsymbol{y}(t)$ about the agent and m dimensional vector $\boldsymbol{u}(t)$ as a motor command.
2. From $l = 1 \cdots$, identify the obtained data.
   (a) If $\log|\hat{\boldsymbol{R}}| < 0$, stop the procedure and determine n based on $AIC(n)$,
   (b) else, increment l until the condition (a) is satisfied or $AIC(n)$ does not decrease.
3. Classify the agent using the $(n/p, l)$ (**Figure 1**).

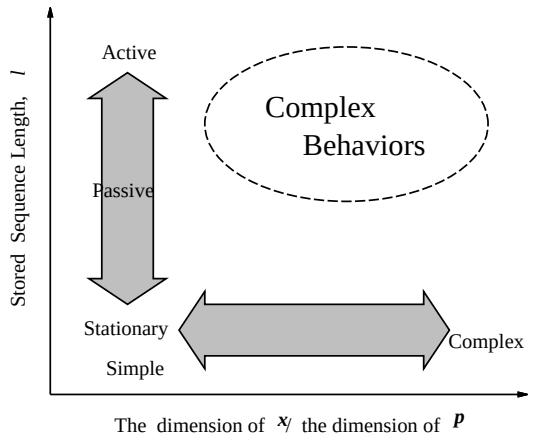


Figure 1: Classification space

We suppose that all the agents are classified into several categories in the classification space (See Figure 1). If the other agent has a complex strategy, l and n/p become large since the large amount of the information are necessary to classify it. On the other hand, it is simple to predict the successive situations in case of the stationary agent such as a goal post.

Reinforcement Learning

After estimating the state space model represented by Eq.2, the agent begins to learn behaviors using a reinforcement learning method. In the previous section, appropriate dimension n of state vector $\boldsymbol{x}(t)$ is determined, and the successive state is predicted. Therefore, we regard an environment as Markovian.

Q Learning

Q learning (Watkins 1989) is a form of model-free reinforcement learning based on stochastic dynamic programming. We assume that the robot can discriminate the set $\boldsymbol{X}$ of distinct world states, and can take the set $\boldsymbol{A}$ of actions on the world. The world is modeled as a Markov process, making stochastic transitions based on its current state and the action taken by the robot. Let $T(x, a, x')$ be the probability that the world will transit to the next state x' from the current state-action pair (x, a). For each state-action pair (x, a), the reward $r(x, a)$ is defined.

The general reinforcement learning problem is typically stated as finding a policy that maximizes discounted sum of the reward received over time. A policy f is mapping from $\boldsymbol{X}$ to $\boldsymbol{A}$. This sum called the *return* and is defined as:

$$\sum_{n=0}^{\infty} \gamma^n r_{t+n}, \quad (8)$$

where r_t is the reward received at step t given that the agent started in state x and executed policy f. γ is the discounting factor, it controls to what degree rewards in the distant future affect the total value of a policy and is just slightly less than 1.

A simple version of a Q learning algorithm used here is shown in Fig.2.

1. Initialize $Q(x, a)$ to 0 for all state x and action a.
2. Perceives current state x.
3. Choose an action a according to action value function.
4. Carry out action a in the environment. Let the next state be x' and immediate reward be r.
5. Update action value function from x, a, x', and r,

$$\begin{aligned} Q_{t+1}(x,a) &= (1-\alpha_t)Q_t(x,a) \\ &\quad +\alpha_t(r + \gamma \max_{a' \in \boldsymbol{A}} Q_t(x', a')) \quad (9) \end{aligned}$$

   where α_t is a learning rate parameter.
6. Return to 2.

Figure 2: A simple version of the 1-step Q learning algorithm.

Experimental Results

Assumptions

We apply the proposed method to a simple soccer-like game including two agents (**Figure 3**). Each agent has a single color TV camera and does not know the location, the size and the weight of the ball, 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.

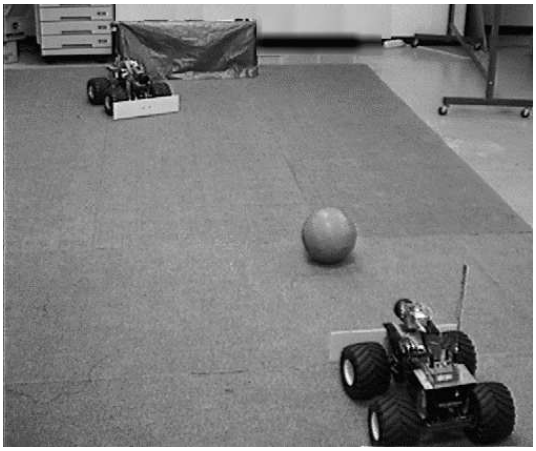


Figure 3: The environment and our mobile robot

Action and State spaces

We construct the action space in terms of two components, where velocity is decomposed to 3 sub-actions, forward, stop and back, and the steering wheel is also decomposed to 3 sub-actions. The input system $\boldsymbol{u}$ to our mobile robot is defined as the 2 dimensional vector.

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

where v and ϕ are the velocity of motor and the angle of steering respectively. However, robot can not move when $\boldsymbol{u}^T = [0, \pm 1]$ (actions that turn the steering wheel and let the velocity be 0). So we remove them from action state space, and as a result, each agent has 7 actions in the action space $\boldsymbol{A}$.

The effects of an action against the environment can be informed to the agent only through the visual information. The output (observed) vectors are shown in **Figure 4**. In case of the ball, the center position of the ball image (x_c, y_c) is used, and the left and right edge (x_l, y_l) and (x_r, y_r) are used for the field line. However, in case of the goal, the x position of the upper-left x_{ul} is approximately equivalent to the bottom-left x_{bl}, then we substitute $x_l = (x_{ul} + x_{bl})/2$ for x_{ul} and x_{bl} (the same procedure is applied to x_r). The two markers are placed on the robot and the center positions of them are used. As a result, the dimension of the observed vector about the ball, the goal, the line, and the agent are 2, 4, 6, 4 respectively.

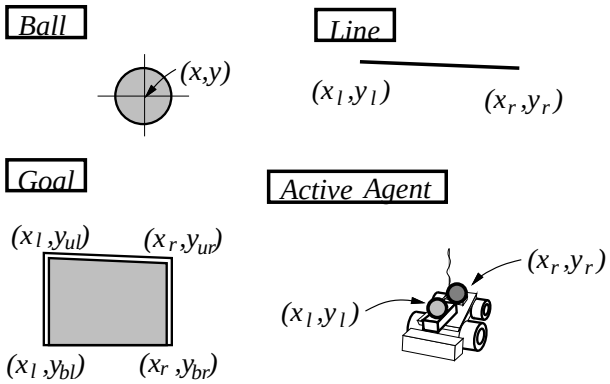


Figure 4: Observed feature points of the ball, goal, line and agent

In order to apply the Q learning, the state space should be quantized. Because the covariance matrix of vector $\boldsymbol{\mu}$ is unit matrix, we segment the each element x_i of state vector $\boldsymbol{x}$ to 5 sub-states as follows:

$$\begin{array}{lll} x_i < -2, & -2 \le x_i < -1 & -1 \le x_i < 1, \\ 1 \le x_i < 2, & 2 \le x_i. & \end{array}$$

However, one action does not always corresponds to one state transition because we segment the state space by hand, . This problem is called *state action deviation problem* (Asada *et al.* 1995). Because of this problem, the robot frequently returns to the same state. This prevents the learning from converging correctly. Therefore, the robot continues to take one action primitive until the current state changes, like Asada's method. This sequence of the action primitive is called action. Once the state has changed, we update the action value function.

Strategy of the other active agent

We assume that the other active agent has some basic behaviors designed by programmer such that 1) keep stopping (stationary), 2) walk randomly (random), 3) turn to the left (forward left), 4) turn to the right (forward right), and that the other agent does not change the strategy.

Simulation Results

Table1 shows the result of identification. In order to predict the successive situations, $l = 1$ is sufficient for the goal and line identification, while the ball needs 2 steps. If the friction of the ball against the field changes, appropriate l also changes. **Figures 5** show the prediction result and prediction error of the ball in terms of $l = 1, 2,$ and 3.

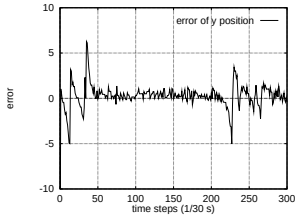
At time steps 40 and 230, the learner kicked the ball, therefore prediction error becomes large regardless of the value of l. However, we can see the larger l reduces the error slightly better than the smaller l. As long as

Table 1: The dimension of stationary agent

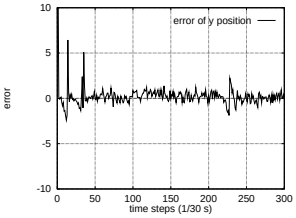
agent	l	n	$\log \lvert \boldsymbol{R} \rvert$	AIC
line	1	1	3.78	1632
		2	0.12	126
		3*	−2.14	−800
		4	−2.93	−1104
goal	1	1	3.37	1859
		2*	−0.001	121
		3	−2.30	−1052
		4	−3.79	−1802
		5	−4.17	−1978
		6	−4.41	−2070

Table 2: The dimension of passive agent(ball)

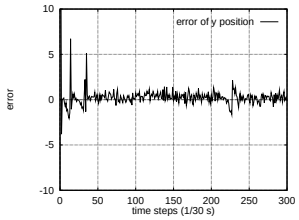
l	n	$\log \lvert R \rvert$	AIC	l	n	$\log \lvert R \rvert$	AIC
1	1	5.82	1964	3	1	5.12	1709
	2	1.91	675		2	0.414	174
2	1	5.13	1725		3	0.187	112
	2	0.518	209		4	5.08×10^{-3}	64
	3	0.342	163		5	3.98×10^{-3}	75
	4*	0.232	138		6	2.00×10^{-3}	87



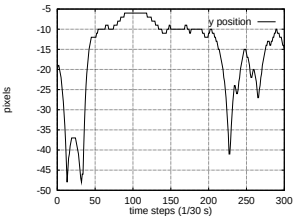
(a) $l = 1$, $|\hat{R}| = 6.78$



(b) $l = 2$, $|\hat{R}| = 1.26$



(c) $l = 3$, $|\hat{R}| = 1.01$



(d) real trajectory of y position of the ball

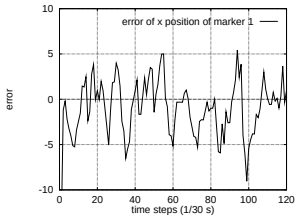
Figure 5: Prediction errors in cases of $l = 1, 2, 3$

Table 3: The dimension of agent(other agent)

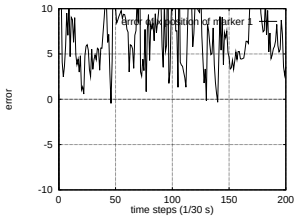
strategy	l	n
stationary	1	3
random walk	3	6
forward left	3	6
forward right	3	6

the ball is stationary, the value of $l = 1$ is sufficient, but it seem difficult to model the ball when it is kicked by the learner because of non-linearity.

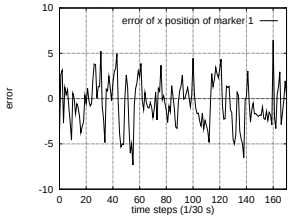
Next, the results of identification of the other agent are shown in **Table 3** and **Figure 6**. The observing agent can not predict the random walk agent as a matter of course. The forward left agent can be identified by the same dimension of the random agent, but the prediction error is much smaller than that of random walk.



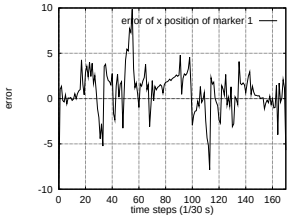
(a) stationary agent



(b) random agent



(c) forward left agent



(d) forward right agent

Figure 6: Prediction errors for other agents (x position of the marker 1)

Shooting behavior **Figure 7** shows a sequence of shooting a slowly moving ball into the goal using CVA method. During the learning, the ball is rolled from the goal to the learning robot slowly. **Table 4** shows a comparison about the success rate of shooting. We assign a reward value 1 when the ball was kicked into the goal or 0 otherwise and the environment consists of the ball, the goal and the line. The two lines emerged from the agent show the visual angle.

In (Asada *et al.* 1995; Uchibe, Asada, & Hosoda 1996), the learning agent uses the only current information about the ball and the goal, therefore the leaning agent can not acquire the optimal behavior when the ball is rolling. In other words, the action value function does not become stable because the state and action spaces are not consistent with each other.

Passing behavior Passing a ball to the other agent is regarded as shooting (kicking) a ball into the moving goal. The result of the shooting behavior mentioned above is transfered to the strategy of the other robot, and the other robot moves based on the action value of the shooting behavior. Therefore, the other robot does **not** detect the learning robot, only the ball, goal, and line.

We assign a reward value 1 when the ball was kicked into the other agent, -0.8 when the learner makes a collision with the other agent, or 0 otherwise and the environment consists of the ball and the other agent.

Table 4: Comparison between the proposed method and previous work (Asada *et al.* 1995; Uchibe, Asada, & Hosoda 1996)

state vector	success of shooting (%)	success of passing (%)
current position	10.2	9.8
using CVA	78.5	53.2

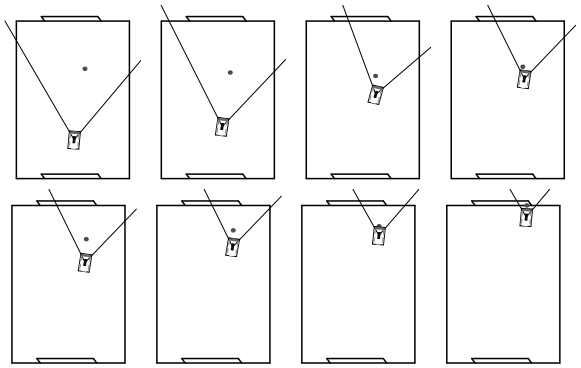


Figure 7: The robot succeeded in shooting a moving ball into a goal

Real System

Figure 9 shows a configuration of the real mobile robot system(Uchibe, Asada, & Hosoda 1996). The image taken by a TV camera mounted on the robot is transmitted to a UHF receiver and processed by Datacube MaxVideo 200, a real-time pipeline video image processor. In order to simplify and speed up the image processing time, we painted the ball, the goal, and the

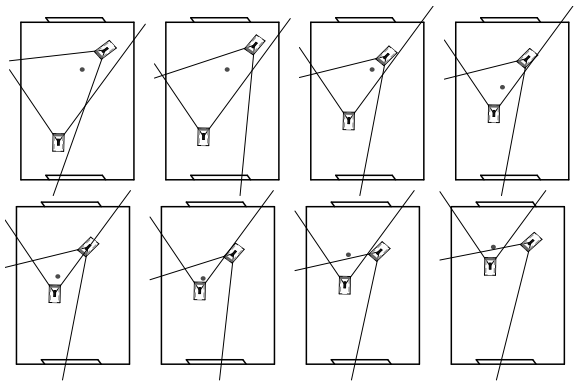


Figure 8: The robot succeeded in passing a ball to the other agent

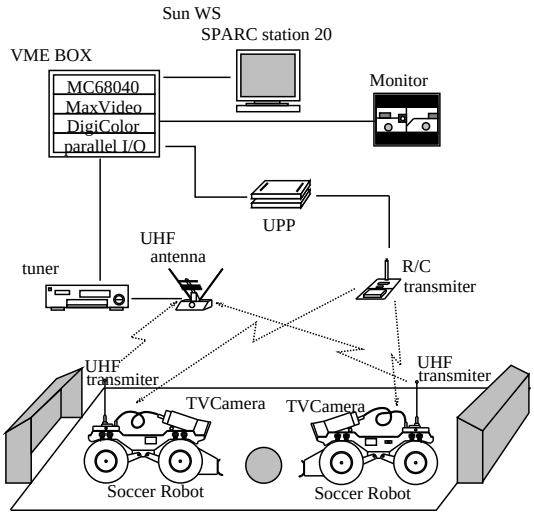


Figure 9: A configuration of the real system.

enemy in red, blue, and yellow, respectively. We have constructed the radio control system of the robot, following the remote-brain project by Inaba et al. (Inaba 1993). The tilt angle is about -26 [deg] so that robot can see the environment effectively. The horizontal and vertical visual angles are about 67 [deg] and 60 [deg], respectively (for more details, see (Uchibe, Asada, & Hosoda 1996)).

Preliminary Experiments

Unfortunately, since we can not perceive the positions of markers equipped the robot, the experiments of classification of the other agent has not be carried out yet. **Figure10** and 11 show that one example of the sequence of data. We show the preliminary experimental results of classification in **Table 5** and **Figure 12**.

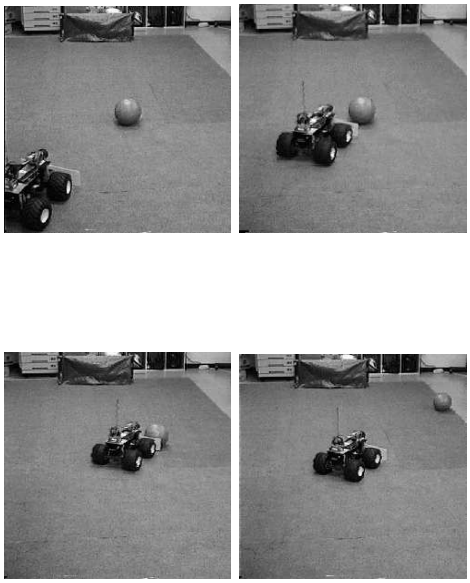


Figure 10: one sequence of sampled data (kick the ball, and move back)

The value of l for the ball is bigger that of computer simulation, because the ball sometimes moves towards unpredictable direction (complex movement) due to its eccentricity of the centroid. In case of the goal, the value of n is bigger than that of simulation because of the noise of the image processing.

Discussion

In order to obtain the model of the dynamics of a certain phenomenon, the strategy of the robot needs to be random. In other words, the inputs vector (motor command) applied to the system (environment) should

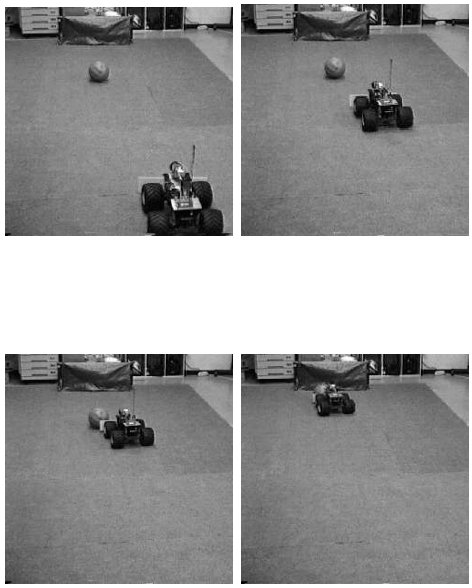
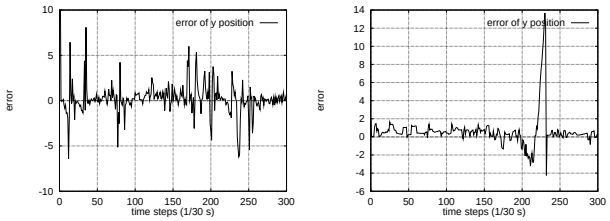


Figure 11: one sequence of sampled data (kick the ball, and move to the goal)

Table 5: The dimension of the ball and the goal

agent	l	n	$\log \lvert \boldsymbol{R} \rvert$	AIC
ball	4	4	1.88	284
goal	1	3	-1.73	-817



(a) y position of the ball

(b) y position of the left-upper of the goal

Figure 12: Prediction error of the ball and the goal in real environment

be *persistently exciting*. However, it is almost impossible to sample the data because the time is limited and undesired inputs causes the destruction of the robots. Therefore, if the sampled data are biased, the matrix $\boldsymbol{\Theta}$ of the Eqn.3 changes. For example, the robot seldom kicks the ball, the model of the ball will be regarded as a stationary one.

Generally, for the less biased data, the more data and longer time are necessary. An effective method for data sampling should be developed, but there is a trade-off between the effectiveness and *a priori* knowledge on the environment and the robot.

We used the state space designed by a human programmer in order to apply the Q learning algorithm. However, there is no guarantee that such a state space is always appropriate for the robot. We have to develop the method which segments the state space suitable for method. Asada et al. proposed the method of action-based state space construction for vision-based mobile robots (Asada, Noda, & Hosoda 1996). Takahashi et al. proposed the method of incrementally segmenting the sensor space based on the experiences of the robot (Takahashi, Asada, & Hosoda 1996). These method might be promising.

Concluding Remarks

This paper presents a method of strategy classification in order to apply reinforcement learning to the environment including other agents. Our method takes account of the trade-off among the precision of prediction, the dimension of state vector, and the length of steps to identify. A soccer robot can shoot a ball and pass a ball even if a ball is rolling.

As future work we hope to challenge topics as follows: 1) segmentation procedure of state vector and 2) strategy learning (when to shoot or pass).

Acknowledgment

The authors thank Mr. Yasutake Takahashi and Masateru Nakamura for their efforts in implementing a visual processing.

References

Akaike, H. 1974. A new look on the statistical model identification. *IEEE Trans. AC-19* 716–723.

Asada, M.; Noda, S.; Tawaratsumida, S.; and Hosoda, K. 1995. Vision-based reinforcement learning for purposive behavior acquisition. In *Proc. of IEEE International Conference on Robotics and Automation*, 146–153.

Asada, M.; Noda, S.; and Hosoda, K. 1996. Action-based sensor space categorization for robot learning. In *Proc. of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

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

Inaba, M. 1993. Remote-brained robotics : Interfacing ai with real world behaviors. In *Preprints of ISRR'93*.

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

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

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

Sandholm, T. W., and Crites, R. H. 1995. On multiagent q-learning in a semi-competitive domain. In *Workshop Notes of Adaptation and Learning in Multiagent Systems Workshop, IJCAI-95*.

Takahashi, Y.; Asada, M.; and Hosoda, K. 1996. Reasonable performance in less learning time by real robot based on incremental state space segmentationaction-based sensor space categorization. In *Proc. of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

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

Van Overschee, P., and De Moor, B. 1995. A unifying theorem for three subspace system identification algorithms. *Automatica* 31(12):1853–1864.

Watkins, C. J. C. H. 1989. *Learning from Delayed Rewards*. Ph.D. Dissertation, King's College, University of Cambridge.