Simultaneous Learning to Acquire Competitive Behaviors in Multi-Agent System based on a Modular Learning System

Yasutake Takahashi, Kazuhiro Edazawa, Kentarou Noma, and Minoru Asada
Dept. of Adaptive Machine Systems,
Graduate School of Engineering, Osaka University
Yamadagaoka 2-1, Suita, Osaka, 565-0871, Japan
{*yasutake, eda, noma, asada*}*@er.ams.eng.osaka-u.ac.jp*

***Abstract*— Existing reinforcement learning approaches have been suffering from the policy alternation of others in multi-agent dynamic environments. A typical example is the case of RoboCup competitions because other agent behaviors may cause sudden changes in state transition probabilities in which constancy is needed for the learning to converge. The keys for simultaneous learning to acquire competitive behaviors in such an environment are**

- **a modular learning system for adaptation to the policy alternation of others; and**
- **an introduction of macro actions for simultaneous learning to reduce the search space.**

This paper presents a method of modular learning in a multi-agent environment in which the learning agents can simultaneously learn their behaviors and adapt themselves to the situations as a consequence of the others' behaviors.

***Index Terms*— reinforcement learning, competitive behaviors acquisition, multi-agent system, modular learning system, simultaneous learning, and RoboCup**

I. Introduction

There have been an increasing number of approaches to robot behavior acquisition based on reinforcement learning methods [1], [2]. The conventional approaches require an assumption that the state transition is caused by an action of a learning agent so that the learning agent can regard the state transition probabilities as constant during its learning period. Therefore, it seems difficult to directly apply the reinforcement learning method to a multi-agent system because a policy alteration of other agents may occur which dynamically changes the state transition probabilities from the viewpoint of the learning agent. RoboCup provides such a typical situation, i.e., a highly dynamic, hostile environment in which agents must obtain purposive behaviors.

There are a number of studies on reinforcement learning systems in a multi-agent environment. Kuhlmann and Stone [3] have applied a reinforcement learning system with a function approximator to the keep-away problem in the situation of the RoboCup simulation league. In their work, only the passer learns his policy is to keep the ball away from the opponents. The other agents (receivers and opponents) follow fixed policies given by the designer beforehand.

Asada et al. [4] proposed a method that sets a global learning schedule in which only one agent is specified as a learner with the rest of the agents having fixed policies. Accordingly, the method cannot handle the alternation of the opponent's policies. Ikenoue et al. [5] showed simultaneous cooperative behavior acquisition by fixing learners' policies for a certain period during the learning process. These studies suggest it is possible to acquire a reasonable behavior in a multi-agent environment if the learner can see the environment, including the other agents, as almost fixed because the others keep their policies for a certain time. In the case of cooperative behavior acquisition, neither agent has any reason to change policies while they continue to acquire positive rewards as a result of their cooperative behavior with each other. The agents tend to update their policies gradually so that the state transition probabilities seem almost fixed from the viewpoint of the other learning agents.

However, in the case of competitive behavior acquisition in a multi-agent environment, it is unlikely the agent will tend to select the action that causes positive rewards for the opponents but a negative reward for itself. Instead the punished agent tends to drastically change its policy by giving a negative reward to its opponents so that it can acquire a positive reward. This policy alternation causes dynamic changes in the state transition probabilities from the viewpoint of the learning agent; therefore, it seems difficult to directly apply the reinforcement learning method to a multi-agent system.

A modular learning approach would provide one solution to this problem. If we can assign multiple learning modules to different situations respectively, each in which the state transition probabilities can be regarded as constant, then the system could demonstrate a reasonable performance. Jacobs and Jordan [6] proposed a mixture of experts, in which a set of the expert modules learn and the gating system weights the output of each expert module for the final system output. This idea is a very general one having a wide range of applications (ex. [7], [8], [9], [10], [11]).

We adopt the basic idea of the mixture of experts into the architecture of behavior acquisition in the multi-agent environment. Takahashi et al. [12] have shown preliminary ex-

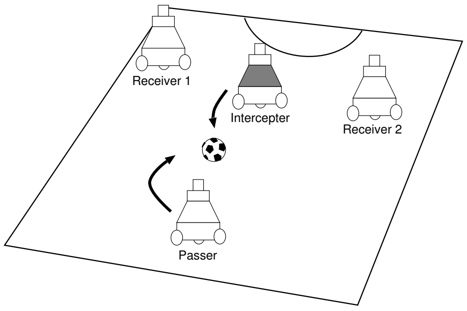


Fig. 1. Task : 3 on 1

perimental results of behavior acquisition in the multi-agent environment; however, the learning modules were assigned by the human designer. In this paper, we first show how it is difficult to directly introduce a multi-module learning system for even single agent learning in a multi-agent environment because of its complexity; instead we introduce a simple learning scheduling which makes it relatively easy to assign modules automatically. Second, we introduce macro actions to realize simultaneous learning in multi-agent environments in which each agent does not need to fix its policy according to some learning schedule. Elfwing et al. [13] introduced macro actions to acquire a cooperative behavior with two real rodent robots. The exploration space with macro actions becomes much smaller than the one with primitive actions; therefore, the macro action increases the possibility of creating cooperative experiences and leads the two agents to find a reasonable solution in a realistic learning time frame. We also show the introduction of macro actions enable the agents to learn competitive behaviors simultaneously. We have applied the proposed multi-module learning system to soccer robots which participate in RoboCup competition and demonstrate the experimental results on computer simulation and real robot implementation.

II. Tasks and Assumption

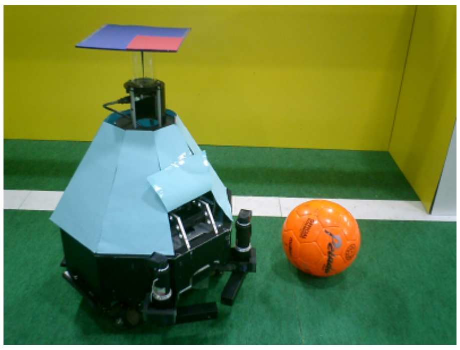


Fig. 2. A real robot

Fig.1 shows a situation the learning agents are supposed to encounter. The game is like a three-on-one involving one

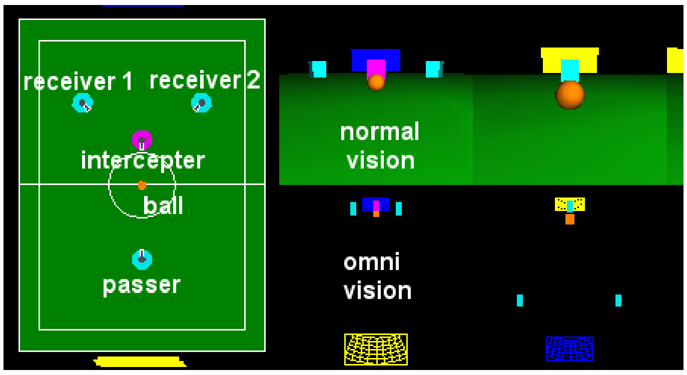


Fig. 3. Viewer of simulator

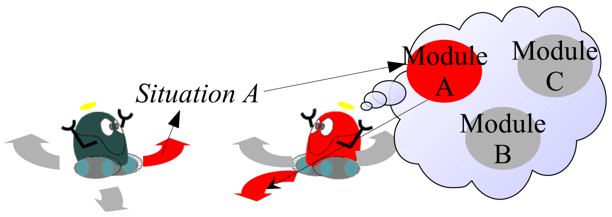


Fig. 4. Adaptive behavior selection based on Multi-module learning system

opponent and three other players. The player nearest to the ball becomes a passer who passes the ball to one of its teammates (receivers) while the opponent tries to intercept it.

Fig. 2 shows a mobile robot we have designed and built. Fig. 3 shows the viewer of our simulator for our robots and the environment. The robot has an omni-directional camera system. A simple color image processing is applied to detect the ball, the interceptor, and the receivers on the image in real-time (every 33ms.) The left of Fig. 3 shows a situation the agent can encounter while the right images show the simulated ones of the normal and omni vision systems. The mobile platform is an omni-directional vehicle (any translation and rotation on the plane.)

III. Multi-Module Learning System

A. Basic Idea

The basic idea is that the learning agent could assign one behavior learning module to one situation which reflects another agent's behavior and the learning module would acquire a purposive behavior under the situation if the agent can distinguish a number of situations, each in which the state transition probabilities are almost constant. We introduce a modular learning approach to realize this idea (Fig. 4). A module consists of both a learning component that models the world and an action planner. The whole system follows these procedures:

- select a model which represents the best estimation among the modules;

- update the model; and
- calculate action values to accomplish a given task based on dynamic programming.

As an experimental task, we suppose ball passing with the possibility of being intercepted by the opponent (Figs. 1 and 3). The problem for the passer (interceptor) here is to select one model which can most accurately describe the interceptor's (passer's) behavior from the viewpoint of the agent and then to take an action based on the policy which is planned with the estimated model.

B. Architecture

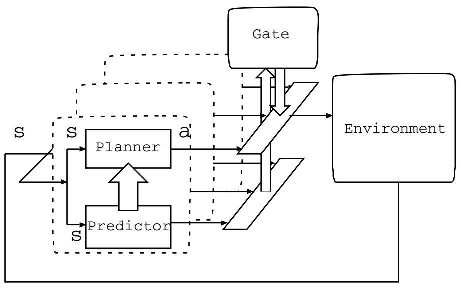


Fig. 5. A multi-module learning system

Fig. 5 shows a basic architecture of the proposed system, i.e., a multi-module reinforcement learning system. Each module has a forward model (predictor) which represents the state transition model and a behavior learner (action planner) which estimates the state-action value function based on the forward model in a reinforcement learning manner. This idea of a combination of a forward model and a reinforcement learning system is similar to the H-DYNA architecture [14] or MOSAIC [11]. The system selects one module which has the best estimation of a state transition sequence by activating a gate signal corresponding to a module while deactivating the gate signals of other modules; the selected module then sends action commands based on its policy.

1) Predictor: Each learning module has its own state transition model. This model estimates the state transition probability $\hat{\mathcal{P}}^a_{ss'}$ for the triplet of state s, action a, and next state s':

$$\hat{\mathcal{P}}^a_{ss'} = Pr\{s_{t+1} = s' | s_t = s, a_t = a\} \tag{1}$$

Each module has a reward model $\hat{\mathcal{R}}^a_{ss'}$, too:

$$\hat{\mathcal{R}}^a_{ss'} = E\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\} \tag{2}$$

We simply store all experiences (sequences of state-action-next state and reward) to estimate these models.

2) Planner: Now we have the estimated state transition probabilities $\hat{\mathcal{P}}^a_{ss'}$ and the expected rewards $\hat{\mathcal{R}}^a_{ss'}$, then, an approximated state-action value function $Q(s,a)$ for a state action pair s and a is given by

$$Q(s,a) = \sum_{s'} \hat{\mathcal{P}}^a_{ss'} \left[\hat{\mathcal{R}}^a_{ss'} + \gamma \max_{a'} Q(s',a') \right] , \tag{3}$$

where γ is a discount rate.

3) Module Selection for Action Selection: The reliability of the module becomes larger if the module does better state transition prediction during a certain period, else it becomes smaller. We assume the module that does the best state transition prediction has the best policy against the current situation because the planner of the module is based on the model which best describes the situation. In the proposed architecture, this reliability is used for gating the action outputs from modules. We calculate an execution-time reliability ${}^{exec}g_i$ of the module i as follows:

$$ {}^{exec}g_i = \prod_{t=-T+1}^{0} e^{\lambda p_i^t} $$

where p_i is an occurrence probability of the state transition from the previous ($t-1$) state to the current (t) one according to the model i, and λ is a scaling factor. The T indicates a period (step) in evaluating the reliability of the module; we set T as 5 in the following experiments. The agent continues to use the module for a certain period, for example 5 step or 1 second, after it changes the module in order to avoid oscillation of the policies.

4) Module Selection for Updating Models: We use an update-time reliability ${}^{update}g_i$ of the module for updating modules. The calculation of this reliability contains the future state transition probabilities:

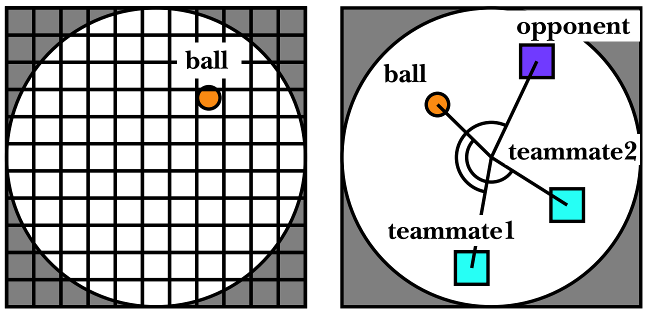
$$ {}^{update}g_i = \prod_{t=t-T}^{t+T} e^{\lambda p_i^t} \quad . $$

IV. Behaviors Acquisition under Scheduling

As we mentioned in I, first, we show how it is difficult to directly introduce the proposed multi-module learning system in the multi-agent system. We introduce a simple learning scheduling to make it relatively easy to assign modules automatically.

A. Configuration

The state space is constructed in terms of the centroid of the ball on the image, the angle between the ball and the interceptor, and the angles between the ball and the receivers (see Figs. 12 (a) and (b)). We quantized the ball position space as 11-by-11 as shown in Fig. 12 (a) and each angle into 8. As a result, the number of states become $11^2 \times 8 \times 8 \times 8 = 61952$. The action space is constructed in terms of the desired three velocity values (x_d, y_d, w_d) to be sent to the motor controller (Fig. 7). Each value is quantized into three, so the number of actions is $3^3 = 27$. The robot has a pinball-like



(a) state variables (position) (b) state variables (angle)

Fig. 6. State variables

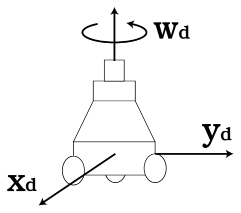


Fig. 7. Action variables

kick device which allows it to automatically kick the ball whenever the ball comes within the region to be kicked. It tries to estimate the mapping from sensory information to appropriate motor commands by the proposed method.

The initial positions of the ball, passer, interceptor, and receivers are shown in Fig. 1. The opponent has two kinds of behaviors: it defends the left side or right side. The passer agent has to estimate which direction the interceptor will defend and go to the position so as to kick the ball in the direction the interceptor does not defend. From the viewpoint of the multi-module learning system, the passer will estimate which situation of the module is going on and select the most appropriate module as its behavior. The passer acquires a positive reward when it approaches the ball and kicks it to one of the receivers.

B. Learning Scheduling

We prepare a learning schedule composed of three stages to show its validity. The opponent fixes its defending policy as a right-side block at the first stage. After 250 trials, the opponent changes the policy to block the left side at the second stage and continues this for another 250 trials. Finally, the opponent changes the defending policy randomly after one trial.

C. Simulation Result

We have applied the method to a learning agent and compared it with only one learning module. We have also compared the performances between the methods with and without the learning scheduling. Fig. 8 shows the success rates of those during the learning process. Success indicates the learning agent successfully kicked the ball without interception by the opponent. The success rate indicates the number of successes in the last 50 trials. The "mono. module" in the figure indicates a "monolithic module" system which tries to acquire a behavior for both policies of the opponent. The multi-module system with scheduling shows a better performance than the one-module system. The monolithic module with scheduling means we applied the learning scheduling mentioned in **IV-B** even though the system has only one learning module. The performance of this system is similar to the multi-module system until the end of the first stage (250 trials); however, it goes down at the second stage because the obtained policy is biased against the experiences at the first stage and cannot follow the policy change of the opponent. Because the opponent uses one of the policies at random in the third stage, the learning agent obtains about 50% of the success rate. The term "without scheduling" means we do not apply learning scheduling and the opponent changes its policy at random from the start. Somehow the performance of the monolithic module system without learning scheduling gets worse after 200 trials. The multi-module system without a learning schedule shows the worst performance in our experiments. This result indicates it is very difficult to recognize the situation at the early stage of the learning process because the modules have too few experiences to evaluate their fitness; thus, the system tends to select the module without any consistency. As a result, the system cannot acquire any valid policies.

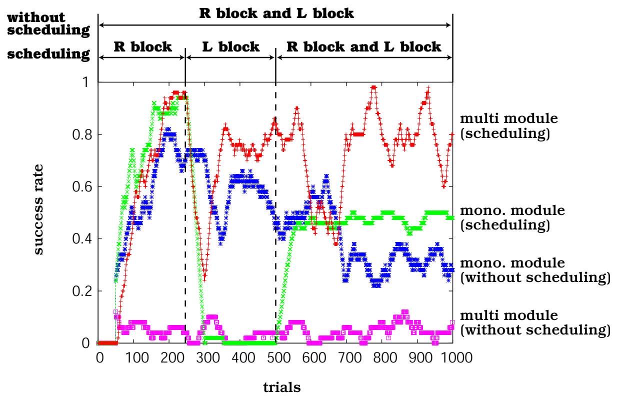


Fig. 8. Success rate during the learning

Figs. 9 and 10 show an example of sequences of the reliabilities while the opponent is blocking the left and right sides. The "LM1" and "LM2" indicate the learning modules that are assigned the left and right block behaviors, respectively. The agent seems to fail to estimate the situation where the opponent is blocking the left or right side during the beginning periods; however, the reliability of the appropriate module improves after a few seconds with the agent

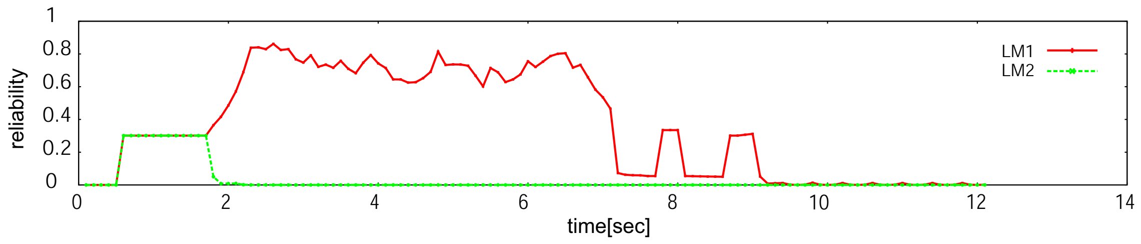


Fig. 9. The sequence of the reliability signals of modules while the opponent is blocking the left side

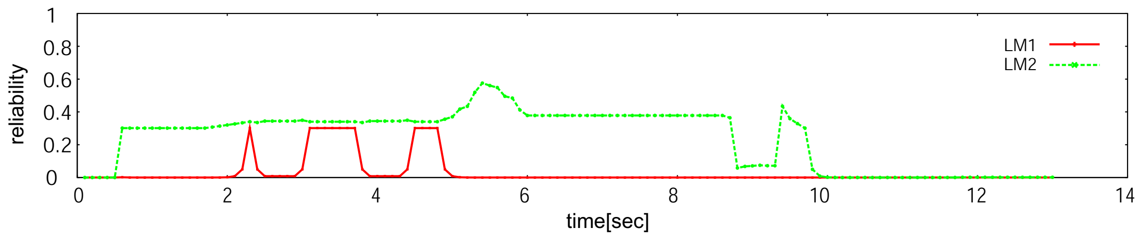


Fig. 10. The sequence of the reliability signals of modules while the opponent is blocking the right side

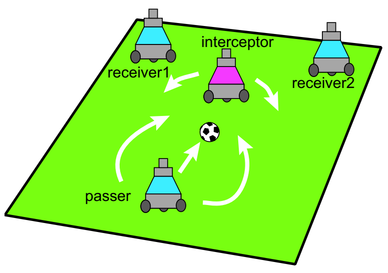


Fig. 11. Macro actions

successfully accomplishing the task.

V. Simultaneous Learning with Macro Actions

We introduce macro actions to realize simultaneous learning in a multi-agent environment in which each agent does not need to fix its policy according to some learning schedule. In this experiment, the passer and the interceptor learn their behaviors simultaneously. The passer learns behaviors for different situations caused by the alternation of the interceptor's policies, i.e., blocking to the left side or the right. The interceptor also learns behaviors for different situations caused by the alternation of the passer's policies, i.e., passing a ball to a left receiver or a right one.

A. Macro Actions and State Spaces

Fig. 11 shows the macro actions of the passer and the interceptor. The macro actions by the interceptor are blocking the pass way to the left receiver and the right one. On the other hand, the macro action by the passer are turning left, turning right around the ball, and approaching the ball to kick it. A ball gazing control is embedded in both learners. The number of the actions is reduced from 27 (see IV-A) primitives to 2 or 3 macro actions. The state space for the

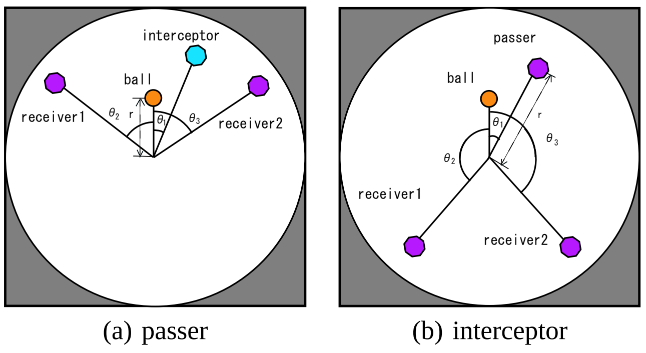


Fig. 12. State variables

passer is constructed in terms of the y position of the ball on the normal image, the angle between the ball and the centers of interceptor, and the angles between the balls and the two receivers on the image of omni-directional vision. The number of the states is reduced from 61952 (see IV-A) to 3773 because the set of macro actions enable us to select a smaller number of state variables and coarser quantization. The state space for the interceptor is constructed in terms of the y position of the passer on the image of normal vision system, the angle between the ball and the passer, and the angles between the ball and the two receivers on the image of omni-directional vision. The number of the states is 2695.

B. Experimental Results

We have checked how the simultaneous learning of the passer and interceptor works on our computer simulation.

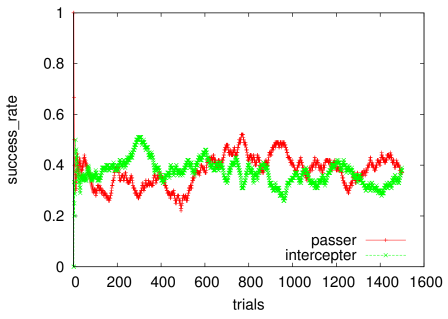


Fig. 13. Sequence of success rates during simultaneous learning

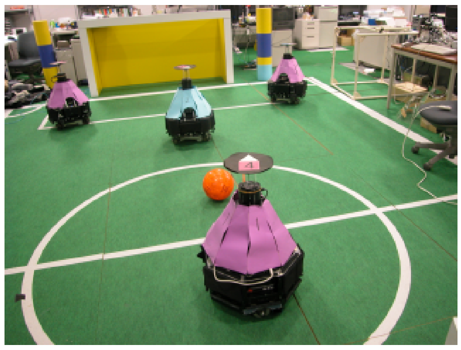


Fig. 14. A scene of the real robot experiments

Both agents start to learn their behaviors from scratch and have 1500 trials without any scheduling. Fig. 13 show the success rates during the simultaneous learning of the passer and the interceptor. This figure shows the interceptor has a higher success rate at the beginning of the learning process while the passer is getting to acquire the appropriate behaviors corresponding to the interceptor's behaviors, then both agents have an almost equal success rate at the end of the learning stage. The sum of both success rates is not 1 because both players sometimes simultaneously failed to pass or intercept.

To check whether both learners acquired appropriate behaviors as against the opponent's behaviors, we fixed one agent's policy and checked to see if the other could select an appropriate behavior, then determined its success rate. Table I shows these results.

Both players have two modules and were assigned to appropriate situations by themselves. LM and the digit number right after the LM indicate respectively the Learning Module and index number of the module. For example, if the passer uses both LM0 and LM1 and the interceptor uses only LM0, then the passer's success rate, interceptor's success rate, and draw rate are 59.0 %, 23.0%, and 18.0%, respectively. Apparently, the player switching multi-modules achieves a higher success rate than the opponent using only one module. These results demonstrate the multi-module learning system works well for both.

We have applied the same architecture to the real robots. Fig. 14 shows the top view of the experimental environment. Fig. 15 shows one example of behaviors by real robots. First, the interceptor tried to block the left side, then the passer approached the ball with the intention of passing it to the right receiver. The interceptor found it was trying to block the wrong side and changed to block the other (right) side, but it was too late to intercept the ball and the passer successfully passed the ball to the right receiver.

VI. Conclusion

In this paper, we proposed a method by which multiple modules are assigned to different situations which are caused by the alternation of the other agent's policy so that an agent may learn purposive behaviors for the specified situations as consequences of the other agent's behaviors.

We introduced macro actions to realize simultaneous learning of competitive behaviors in a multi-agent system. We have shown results of a soccer situation and the importance of the learning scheduling in case of none-simultaneous learning without macro actions, as well as the validity of the macro actions in case of simultaneous learning in the multi-agent system.

References

[1] M. Asada, S. Noda, S. Tawaratumida, and K. Hosoda, "Purposive behavior acquisition for a real robot by vision-based reinforcement learning," *Machine Learning*, vol. 23, pp. 279–303, 1996.

[2] J. H. Connell and S. Mahadevan, *ROBOT LEARNING*. Kluwer Academic Publishers, 1993.

[3] G. Kuhlmann and P. Stone, "Progress in learning 3 vs. 2 keepaway," in *RoboCup-2003: Robot Soccer World Cup VII*, D. Polani, B. Browning, A. Bonarini, and K. Yoshida, Eds. Berlin: Springer Verlag, 2004.

[4] M. Asada, E. Uchibe, and K. Hosoda, "Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development," *Artificial Intelligence*, vol. 110, pp. 275–292, 1999.

[5] S. I. M. Asada and K. Hosoda, "Cooperative behavior acquisition by asynchronous policy renewal that enables simultaneous learning in multiagent environment," in *Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2002, pp. 2728–2734.

[6] R. Jacobs, M. Jordan, N. S, and G. Hinton, "Adaptive mixture of local experts," *Neural Computation*, no. 3, pp. 79–87, 1991.

[7] S. P. Singh, "Transfer of learning by composing solutions of elemental sequential tasks." *Machine Learning*, vol. 8, pp. 323–339, 1992. [Online]. Available: citeseer.nj.nec.com/singh92transfer.html

[8] ——, "The effeicient learnig of multiple task sequences," in *Neural Information Processing Systems 4*, 1992, pp. 251–258.

[9] J. Tani and S. Nolfi, "Self-organization of modules and their hierarchy in robot learning problems: A dynamical systems approach," Technical Report: SCSL-TR-97-008, Tech. Rep., 1997.

[10] ——, "Self-organization of modules and their hierarchy in robot learning problems: A dynamical systems approach," Sony CSL Technical Report, SCSL-TR-97-008, Tech. Rep., 1997.

[11] K. Doya, K. Samejima, K. ichi Katagiri, and M. Kawato, "Multiple model-based reinforcement learning," Kawato Dynamic Brain Project Technical Report, KDB-TR-08, Japan Science and Technology Corporatio, Tech. Rep., June 2000.

TABLE I

SUCCESS RATES FOR PASSER AND RECEIVER IN DIFFERENT CASES

passer	interceptor	passer's success rate[%]	interceptor's success rate [%]	draw rate[%]
LM0,LM1	LM0	59.0	23.0	18.0
LM0,LM1	LM1	52.7	34.3	13.0
LM0	LM0,LM1	25.6	55.0	19.4
LM1	LM0,LM1	26.0	59.3	14.7
LM0,LM1	LM0,LM1	37.6	37.3	25.1

[12] Y. Takahashi, K. Edazawa, and M. Asada, "Multi-module learning system for behavior acquisition in multi-agent environment," in *Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2002, pp. CD–ROM 927–931.

[13] S. Elfwing, E. Uchibe, K. Doya, and H. I. Christensen1, "Multi-agent reinforcement learning: Using macro actions to learn a mating task," in *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep 2004, pp. CD–ROM.

[14] S. P. Singh, "Reinforcement learning with a hierarchy of abstract models," in *National Conference on Artificial Intelligence*, 1992, pp. 202–207. [Online]. Available: citeseer.nj.nec.com/singh92reinforcement.html

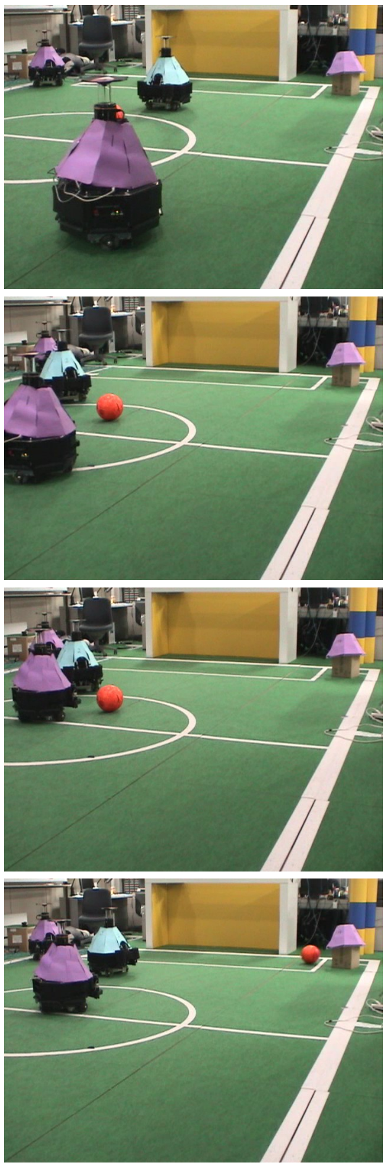


Fig. 15. A sequence of a behavior of passing a ball to the right receiver while the interceptor blocks the left side