Cooperative/Competitive Behavior Acquisition Based on State Value Estimation of Others

Kentaro Noma[1], Yasutake Takahashi[1], and Minoru Asada[1,2]

[1] Dept. of Adaptive Machine Systems, Graduate School of Engineering
Osaka University
[2] JST ERATO Asada Synergistic Intelligence Project
Yamadaoka 2-1, Suita, Osaka, 565-0871, Japan
{kentaro.noma,yasutake,asada}@ams.eng.osaka-u.ac.jp
http://www.er.ams.eng.osaka-u.ac.jp

Abstract. The existing reinforcement learning approaches have been suffering from the curse of dimension problem when they are applied to multiagent dynamic environments. One of the typical examples is a case of RoboCup competition since other agents and their behaviors easily cause state and action space explosion. This paper presents a method of hierarchical modular learning in a multiagent environment by which the learning agent can acquire cooperative behaviors with its teammates and competitive ones against its opponents. The key ideas to resolve the issue are as follows. First, a two-layer hierarchical system with multi learning modules is adopted to reduce the size of the state and action spaces. The state space of the top layer consists of the state values from the lower level, and the macro actions are used to reduce the size of the action space. Second, the state of the other to what extent it is close to its own goal is estimated by observation and used as a state value in the top layer state space to realize the cooperative/competitive behaviors. The method is applied to 4 (defence team) on 5 (offence team) game task, and the learning agent successfully acquired the teamwork plays (pass and shoot) within much shorter learning time (30 times quicker than the earlier work).

1 Introduction

Recently, there have been increasing number of studies on cooperative/competitive behavior acquisition in a multiagent environment by using reinforcement learning methods [3,4,6,8,10]. In such an environment, the state and action spaces for the learning can be easily exploded since not only objects but also other agents should be involved in the state and action spaces, and therefore the sensor and actuator level descriptions may cause information explosion that disables the learning methods to be applied within practical learning time. Kalyanakrishnan et al. [6] showed that the learning can be accelerated by sharing the learned information in the 4 on 5 game task. However, they need still long learning time since they directly use the sensory information as state variables to decide the situation. Stefan et al. [3]

U. Visser et al. (Eds.): RoboCup 2007, LNAI 5001, pp. 101–112, 2008.

achieved the cooperative behavior learning task between two real robots by introducing the macro action that is abstracted action code predefined by the designer. However, only the macro actions do not seem sufficient to accelerate the learning time in a case that more agents are included in the environment. Therefore, the sensory information should be also abstracted to reduce the size of the state space.

A modular learning system is suitable for observing/learning/executing a number of behaviors in parallel, and various modular architectures have been proposed so far [5,7,12,2]. Each module is responsible for learning to achieve a single goal. One arbiter or a gate module is responsible for merging information from the individual modules in order to derive a single action performed by the robot. The prediction of other's behavior is important to realize the cooperative (competitive) behaviors with (against) others in general. Takahashi et al. [11] proposed a method to infer the other's intention by observation based on the idea that the increase of the state value (the larger the state value, the closer to the goal) means the other intends to achieve the corresponding goal regardless of the differences of viewpoint and/or action to achieve the goal. If this prediction capability is incorporated into the learning system, the learner can efficiently acquire the desired behaviors.

This paper presents a method of hierarchical modular learning in a multiagent environment by which the learning agent can acquire cooperative behaviors with its teammates and competitive ones against its opponents. The key ideas to resolve the issue are as follows. First, a two-layer hierarchical system with multi learning modules is adopted to reduce the size of the state and action spaces. The state space of the top layer consists of the state values from the lower level, and the macro actions are used to reduce the size of the physical action space. Second, the state of the other to what extent it is close to its own goal is estimated by observation and used as a state value in the top layer state space to realize the cooperative/competitive behaviors. The method is applied to 4 (defence team) on 5 (offence team) game task, and the learning agent successfully acquired the teamwork plays (pass and shoot) within much shorter learning time (30 times quicker than the earlier work).

2 Multi Module Learning System with Other's State Value Estimation Modules

2.1 Architecture

Fig.1 shows a basic architecture of the proposed system, i.e., a two-layered multi-module reinforcement learning system. The bottom layer (left side of this figure) consists of two kinds of modules: action modules and estimation ones that infer the other's state value. The top layer (right side of the figure) consists of a single gate module that learns which action module should be selected according to the current state that consists of state values sent from the modules at the bottom layer. The selected module then sends action commands based on its policy.

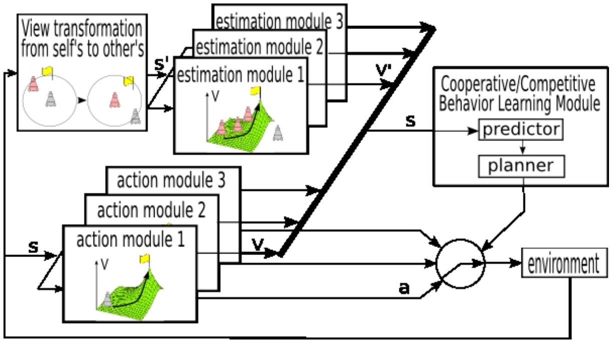
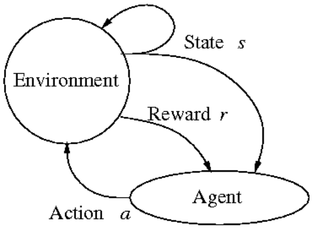


Fig. 1. A multi-module learning system

2.2 Action Module

An action module of the lower layer has a reinforcement learning module which estimates state values for the action.



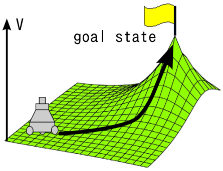


Fig. 2. Agent-environment interaction

Fig. 3. Sketch of a state value function

Figure 2 shows a basic model of reinforcement learning. An agent can discriminate a set S of distinct world states. The world is modeled as a Markov process, making stochastic transitions based on its current state and the action taken by the agent based on a policy π. The agent receives reward r_t at each step t. State value V^π, the discounted sum of the reward received over time under execution of policy π, will be calculated as follows:

$$V^\pi = \sum_{t=0}^{\infty} \gamma^t r_t. \tag{1}$$

In case that the agent receives a positive reward if it reaches a specified goal and zero else, then, the state value increases if the agent follows a good policy π (see Figure 3). The agent updates its policy through the interaction with the environment in order to receive higher positive rewards in future. For further details, please refer to the textbook of Sutton and Barto[9] or a survey of robot learning[1]. Here, we suppose that the state values in each action module have been already acquired before the learning of the gate module.

2.3 Other's State Value Estimation Module

The role of the other's state value estimation module is to estimate the state value that indicate the degree of achievement of the other's task by observation, and to send this value to the state space of the gate module at the top layer. In order to estimate the degree of achievement, the following procedure is taken.

1. The learner acquires the various kinds of behaviors that the other agent may take as macro actions.
2. The learner estimates the sensory information observed by the other through the 3-D reconstruction of its own sensory information.
3. Based on the estimated sensory information of the other, each other's state value estimation module estimates the other's state value by assigning the state value of the corresponding action module of its own.

2.4 Cooperative/Competitive Behavior Learning Module

As shown in Figure 1, the gate module receives state values of lower modules, that is, the action modules and the other's state value estimation ones, and constructs a state space with them. The state space of the gate module is constructed as direct product of the variables of the state values. In order to adopt a discrete state transition model described above, the state space is quantized appropriately. The action set of the gate module is constructed with all action modules of the lower layer as macro actions.

3 Task and Assumptions

The game consists of the offence team (five players and one of them can be the passer) and the defence team (four players attempt to intercept the ball). The offence player nearest to the ball becomes a passer who passes the ball to one of its teammates (receivers) or shoot the ball to the goal if possible while the opposing team tries to intercept it (see Fig. 4).

Only the passer learns its behavior while the receivers and the defence team members take the fixed control policies. The receiver becomes the passer after receiving the ball and the passer becomes the receiver after passing the ball. After one episode, the learned information is circulated among team members through communication channel but no communication during one episode. The action and estimation modules are given a priori.

The offence (defence) team color is magenta (cyan), and the goal color is blue (yellow) in the following figures. The game restarts again if the offense team successfully scores a goal, kicks the ball outside of the field, or the defense team intercepts the ball from the opponent.

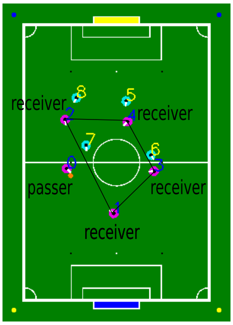


Fig. 4. A passer and the defence formation

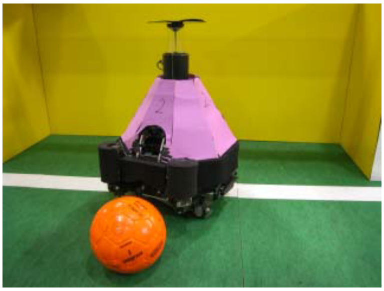


Fig. 5. A real robot

3.1 Offence Team

The passer who is the nearest to the ball learns the team player behavior by passing the ball to one of four receivers or dribbling and shooting the ball to the goal by itself. After its passing, the passer shows a pass-and-go behavior that is a motion to the goal during the fixed period of time automatically. The receivers face to the ball and move to the positions so that they can form a rectangle by taking the distance to the nearest teammates (the passer or other receivers) (see Fig. 4). The initial positions of the team members are randomly arranged inside their territory.

3.2 Defence Team

The defence team member who is nearest to the passer attempts to intercept the ball, and each of other members attempts to "block" the nearest receiver. "Block" means to move to the position near the offence team member and between the offence and its own goal (see Fig. 4). The offence team member attempts to catch the ball if it is approaching. In order to avoid the disadvantage of the offence team, the defence team members are not allowed inside the penalty area during the fixed period of time. The initial positions of the team members are randomly arranged inside their territory but outside the center circle.

3.3 Robots and the Environment

Fig. 5 shows a mobile robot we have designed and built. Fig. 6 shows the viewer of our simulator for our robots and the environment. The robot has an omnidirectional 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. 6 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).

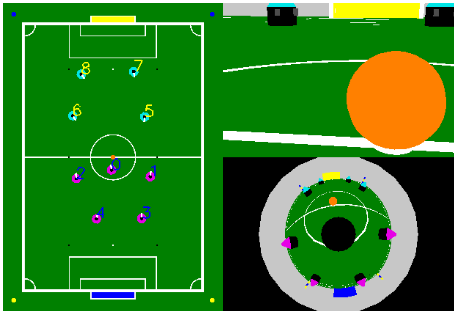


Fig. 6. Viewer of simulator

We suppose that the omni directional vision system provides the robot with 3-D construction of the scene. This assumption is needed for the other's state value estimation module since it is needed to estimate the sensory information observed by other robots.

4 Structure of the State and Action Spaces

4.1 State/Action Spaces for the Gate Module

The passer is only one learner, and the state and action spaces for the lower modules and the gate one are constructed as follows. The action modules are four passing ones for four individual receivers, and one dribble-shoot module. The other's state value estimation modules are the ones to estimate the degree of achievement of ball receiving for four individual receivers, that is how easily the receiver can receive the ball from the passer. These modules are give in advance before the learning of the gate module.

The action spaces of the lower modules adopt the macro actions that the designer specifies in advance to reduce the size of the exploration space without searching at the physical motor level.

The state space S for the gate module consists of the following state values from the lower modules:

- four state values of passing action modules corresponding to four receivers,
- one state value of dribble-shoot action module, and
- four state values of receiver's state value estimation modules corresponding to four receivers.

In order to reduce the size of the whole state space, these values are binarized, therefore its size is 2^4 x 2 x 2^4=512.

The rewards are given as follows:

- 10 when the ball is shot into the goal (one episode is over),
- -1 when the ball is intercepted (one episode is over),

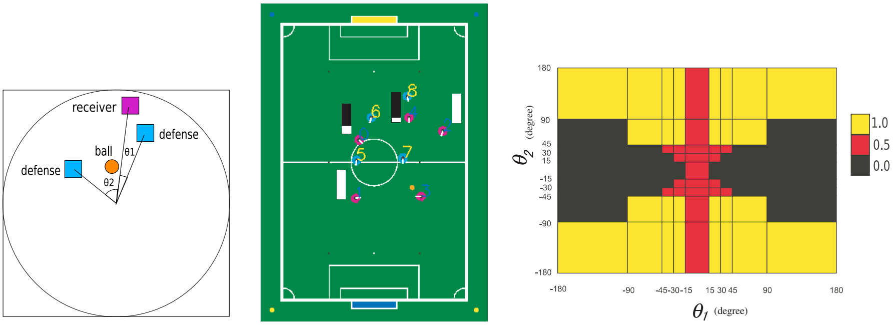
- 0.1 when the ball is successfully passed,
- 0.3 when the ball is dribbled.

When the ball is out of the field or the pre-specified time period elapsed, the game is called “draw” and one episode is over.

4.2 State Space for the Passing Module

The state space of the passing module S is defined on the omni directional camera image as follows (see Fig. 7(a)):

- the smallest angle among angles between the receiver and one the defence players who is nearer to the passer than the receiver (θ_1), and
- the angle between the receiver and one of the defence players who is nearest to the passer (θ_2).



(a) (b) (c)

Fig. 7. State variable (a), examples of state values (b), and state value map of the pass module (c)

The both angles are quantized into ten levels including an invisible case, therefore the total number of states is 100. An example of the state values of four receivers is shown in Fig. 7(b) where the passer is the robot 3 (hereafter, r3 in short), and the color bars near four robots (r0, r1, r2, and r4) indicate the state values of the pass modules for four receivers, respectively. The higher the bar is, the higher the state value is. Since the pass courses for r1 and r2 are not intercepted by the defence players, their state values are high while the state values for r0 and r4 are low since their pass courses are intercepted by the defence players.

The state value map is shown in Fig. 7(c) that indicates the smaller the angle between the receiver and the defence player is, the lower the state value is. The black region (one region is separated in the figure) is inexperience area.

4.3 State Space for the Dribble-Shoot Module

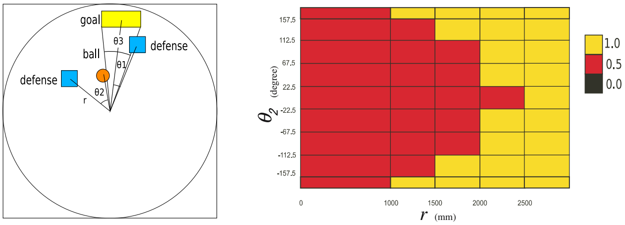
The state space of the dribble-shoot module S is defined on the omni directional camera image as follows (see Fig. 8(a)):

- the angle between the opponent goal and one of the defence players who is nearest to the passer (θ_1),
- the angle between the ball and one of the defence players who is nearest to the passer (θ_2),
- the distance to the nearest defence player (r), and
- the angle between the both edges of the opponent goal (θ_3) that represents the distance to the goal.

These state values are quantized into eight, five, eight, and seven, respectively. The total number of states is 8 x 8 x 5 x 7 = 2240.

The state value map of the dribble-shoot module in terms of θ_1 and r with fixed values of θ_2 and θ_3 is shown in Fig. 8(b) that indicates the nearer the defence player is, the smaller the state value is.

Two examples of the state values of the passer expected to take a role of a shooter is shown in Fig. 9 where the color bars near the passer indicate the state values of the dribble-shoot modules. The higher the bar is, the higher the state value is. Since the passer (r1) is near the goal and no defence players around in Fig. 9 (left), the state value is high while the state value of the passer (r3) in Fig. 9 (right) is low since it is located far from the goal and the defence players are around it.



(a) (b)

Fig. 8. State variables (a) and state values (b) for the dribble and shoot module

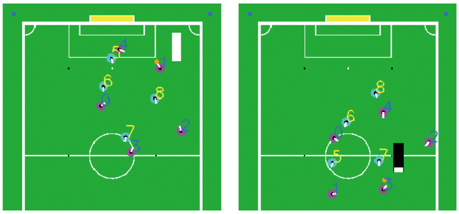
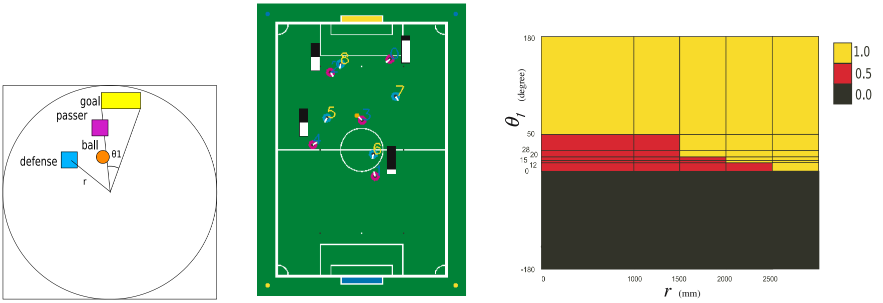


Fig. 9. Two examples of the state values: high (left) and low (right)



(a) (b) (c)

Fig. 10. State variables (a), examples of state values (b), and state value map (c) of the receiver module

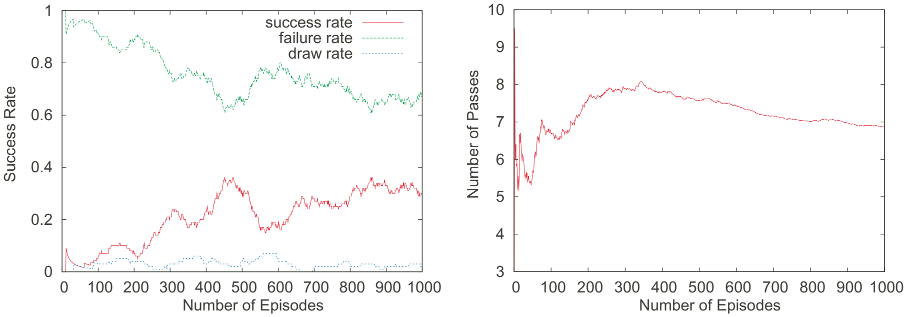
4.4 State Space for the Receiver's State Value Estimation Module

The passer infers each receiver's state that indicates how easily the receiver can shoot the passed ball to the goal by reconstructing its TV camera view of the scene from the passer's omnidirectional view. Since we suppose that the passer has already learned the shooting behavior, the passer can estimate the receiver's state value by assigning its own experienced state of the shooting behavior.

The state space S for the receiver's state value estimation module consists of:

- The distance to the nearest defence player (r)
- The angle between the both side edged of the opponent goal (θ_1) that represents the distance to the goal (see Fig. (10(a)).).

The both are quantized into five and seven levels, therefore the number of states are 5 x 7 = 35.



(a) Success rate (b) Number of passes

Fig. 11. Success rate and the number of passes

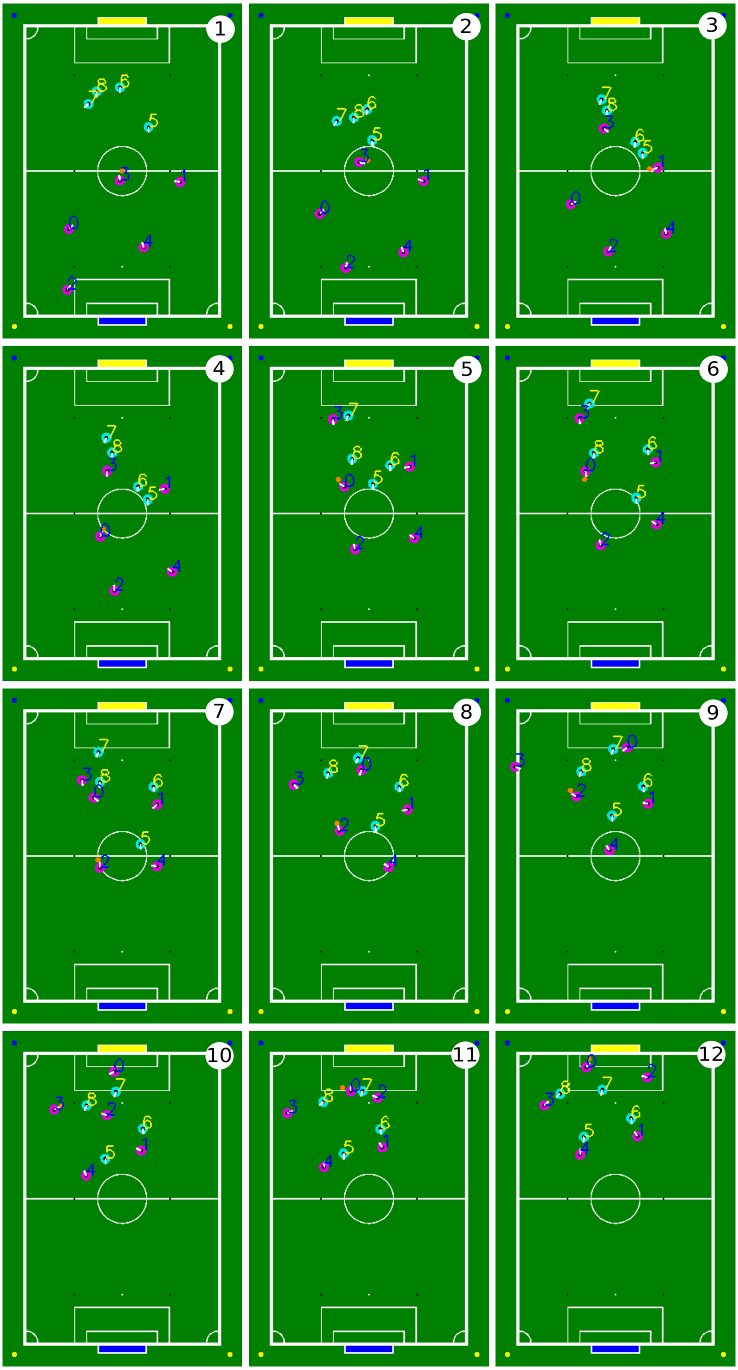


Fig. 12. An example of the acquired behavior

An examples of the state values of the receiver's state value estimation modules is shown in Fig. 10(b) where the color bars near the four receivers indicate their state values. The higher the bar is, the higher the state value is. Since the receiver (r0) is near the goal and no defence players around, the state value is high while the state values of other receivers (r1, r2, and r4) are low since it is located far from the goal and/or the defence players are around it.

The state value map of the receiver's state value estimation module in terms of θ_1 and r is shown in Fig. 10(c) that indicates the nearer (further) the defence player is and the further (nearer) the goal is, the smaller (larger) the state value is. The black region is inexperienced area.

5 Experimental Results

The success rate is shown in Fig. 11(a) where the action selection is 80% greedy and 20% random to cope with new situations. Around the 900th trial, the learning seems to have converged at 30% success, 70% failure, and 10% draw. Compared to the results of [6] that has around 30% success rate with 30,000 trials, the learning time is drastically improved (30 times quicker). Fig. 11(b) indicates the number of passes where it decreases after the 350 trials that means the number of useless passes decreased.

In cases of the success, failure, and draw rates when 100% greedy and 100% random are 55%, 35%, 10%, and 2%, 97%, 1%, respectively. The reason why the success rate in case of 100% greedy is better than in case of 80% greedy seems that the control policies of the receivers and the defence players are fixed, therefore not so new situations happened.

An example of acquired behavior is shown in Fig. 12 where a sequence of twelve top views indicates a successful pass and shoot scene.

6 Conclusion

We have used the state values instead of the physical sensor values and macro actions instead of the physical motor commands, and adopted the receiver's state value estimation modules that infer how easy for each receiver to receive the ball in order to accelerate the learning. As a result, we have much improved the learning time (30 times quicker!) compared to the result of the existing method [6] that has 32% success with communication and 23% without communication at around the 30,000th trial when the learning seems to have converged.

References

1. Connell, J.H., Mahadevan, S.: ROBOT LEARNING. Kluwer Academic Publishers, Dordrecht (1993)
2. Doya, K., Samejima, K., Katagiri, K.i., Kawato, M.: Multiple model-based reinforcement learning. Technical report, Kawato Dynamic Brain Project Technical Report, KDB-TR-08, Japan Science and Technology Corporation (June 2000)

3. Elfwing, S., Uchibe, E., Doya, K., Chirstensen, H.I.: 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, vol. 4, pp. 3164–3169 (2004)
4. Ikenoue, S., Asada, M., Hosoda, K.: 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, pp. 2728–2734 (2002)
5. Jacobs, R., Jordan, M., Nowlan, S., Hinton, G.: Adaptive mixture of local experts. Neural Computation 3, 79–87 (1991)
6. Kalyanakrishnan, S., Liu, Y., Stone, P.: Half field offense in robocup soccer: A multiagent reinforcement learning case study. In: Proceedings CD RoboCup (2006)
7. Singh, S.P.: Transfer of learning by composing solutions of elemental sequential tasks. Machine Learning 8, 323–339 (1992)
8. Stone, P., Sutton, R.S., Kuhlmann, G.: Scaling reinforcement learning toward robocup soccer. Journal of Machine Learing Research 13, 2201–2220 (2003)
9. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
10. Takahashi, Y., Edazawa, K., Asada, M.: Multi-module learning system for behavior acquisition in multi-agent environment. In: Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. CD–ROM 927–931 (October 2002)
11. Takahashi, Y., Kawamata, T., Asada, M.: Learning utility for behavior acquisition and intention inference of other agent. In: Proceedings of the 2006 IEEE/RSJ IROS 2006 Workshop on Multi-objective Robotics, pp. 25–31 (2006)
12. Whitehead, S., Karlsson, J., Tenenberg, J.: Learning multiple goal behavior via task decomposition and dynamic policy merging. In: Connell, J.H., Mahadevan, S. (eds.) ROBOT LEARNING, ch.3, pp. 45–78. Kluwer Academic Publishers (1993)