Efficient Behavior Learning based on State Value Estimation of Self and Others

Yasutake Takahashi, Kentaro Noma and Minoru Asada

Dept. of Adaptive Machine Systems, Graduate School of Engineering,

Osaka University, Yamadaoka 2-1, Suita, Osaka, 565-0871, Japan

Email: {yasutake,kentaro.noma,asada}@ams.eng.osaka-u.ac.jp

M. Asada is with JST ERATO Asada Synergistic Intelligence Project

Abstract

The existing reinforcement learning methods have been seriously suffering from the curse of dimension problem especially when they are applied to multiagent dynamic environments. One of the typical examples is a case of RoboCup competitions since other agents and their behavior easily cause state and action space explosion. This paper presents a method of modular learning in a multiagent environment by which the learning agent can acquire cooperative behavior 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 sensor 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 variable in the top layer state space to realize the cooperative/competitive behavior. The method is applied to 4 (defense team) on 5 (offense team) game task, and the learning agent (a passer of the offense team) successfully acquired the teamwork plays (pass and shoot) within much shorter learning time.

keywords: reinforcement learning, cooperative/competitive behavior acquisition, multi-agent system, modular learning system, RoboCup

1 INTRODUCTION

Recently, there have been increasing number of studies on cooperative/competitive behavior acquisition in a multiagent environment by using reinforcement learning methods [1, 2, 3, 4, 5]. In such environments, the state and action spaces for the learning can easily explode since not only objects but also other agents are involved in the state and action spaces, and therefore the sensor and actuator level descriptions easily cause information explosion that disables the learning methods to be applied within practical learning time. Kalyanakrishnan et al. [3] showed that the learning rate 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. [1] achieved the cooperative behavior learning task between two robots in real time 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 behavior in parallel, and various modular architectures have been proposed so far [6, 7, 8, 9]. 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. Layered Approach has been introduced to a soccer robot domain [10, 11]. Lower modules learn basic skills like shooting a ball, approaching a ball, and so on while the higher modules learn high-level behavior or strategies. The state space of the higher modules uses low-level logical sensory information, still. The methodology to abstract the state space is necessary.

Takahashi and Asada [12] proposed to abstract state based on the lower learning modules. They focused on the state and action space abstraction based only on the behavior of self. The prediction of other's behavior is important to realize the cooperative (competitive) behavior with (against) others in general. Takahashi et al. [13] 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. This work indicates that the estimated state value of the other's behavior has valuable information to describe the situation of the other. If this estimation capability is incorporated into the learning system, the learner can efficiently acquire the desired behavior.

This paper presents a method of hierarchical modular learning in a multiagent environment by which the learning agent can acquire cooperative behavior 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 sensor and action spaces for the macro action learning. The state space of the top layer consists of the state values of the action modules of the lower layer and the macro actions construct the action set of the top layer in order to reduce the size of the physical state and action spaces, respectively. Second, the state value of the other, to what extent it is close to its own goal, is estimated by observation and used as a state variable in the top layer state space to realize the cooperative/competitive behavior. The method is applied to 4 (defense team) on 5 (offense team) game task, and the learning agent (a passer of the offense team) successfully acquired the teamwork plays (pass and shoot) within much shorter learning time.

2 MODULAR LEARNING SYSTEM WITH OTHER'S STATE VALUE ESTIMATION MODULES

2.1 Architecture

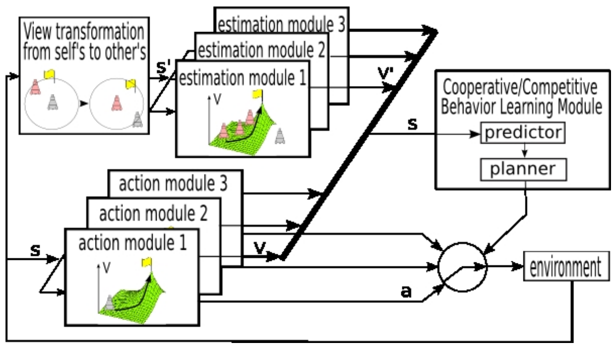


Figure 1: A hierarchical modular learning system for efficient behavior learning based on state value estimation

Figure 1 shows a basic architecture of the proposed system, i.e., a two-layered modular reinforcement learning system. The bottom layer (left side of this figure) consists of two kinds of modules: action modules and estimation ones that estimate the state value of the other's behavior. 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.

2.2 Action module

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

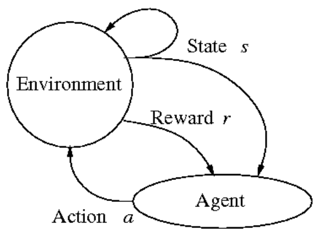


Figure 2: Agent-environment interaction

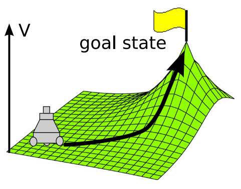


Figure 3: Sketch of a state value function

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:

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

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[14] or a survey of robot learning[15]. 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

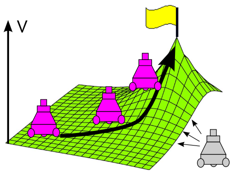


Figure 4: State value estimation of other's behavior

The role of the other's state value estimation module is to estimate the state value that indicates 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 behavior 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.

The top part of Figure 1 shows the estimation procedure of the state values of others. "Estimation module" indicates the other's state value estimation module of which state value function is acquired by the learner beforehand with its own experiences. A component called "View transformation from self's to other's" estimates the sensory information observed by the other. Each "estimation module" receives the estimated sensory information, maps it to the state of the corresponding action module, then estimates the state value of the macro action. Figure 4 shows a sketch of estimation of the other's state value.

2.4 Cooperative/competitive behavior learning module

The gate 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.

(a) **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':

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

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

$$\hat{\mathcal{R}}^a_{ss'} = E\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\} \tag{3}$$

All experiences (sequences of state-action-next state and reward) are simply stored to estimate these models.

(b) **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

$$Q(s, a) = \sum_{s'} \hat{\mathcal{P}}^a_{ss'} \left[\hat{\mathcal{R}}^a_{ss'} + \gamma \max_{a'} Q(s', a') \right] \ , \tag{4}$$

where γ is a discount rate.

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 of the action modules and the estimation ones. 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

3.1 Robots and the environment

Figure 5 shows one game scene that mobile robots wearing purple markers we have designed and built are playing an opponent team. Figure 6 shows the viewer of the 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 Figure 6 shows a situation the agent can encounter while the right images show the simulated ones captured by the normal and omni vision systems, respectively. The mobile platform is an omni-directional vehicle (any translation and rotation on the plane.)

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 robot.

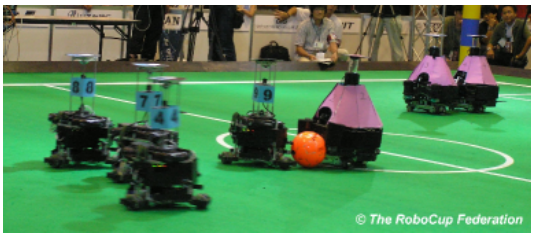


Figure 5: A game scene of RoboCup Middle Size League : Right robots with purple color markers are the ones of our team

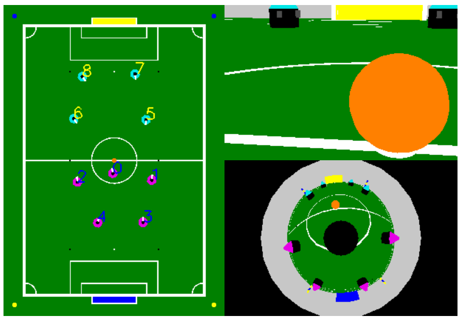


Figure 6: A viewer of the simulator

3.2 Game setting

The game consists of an offense team (five players and one of them can be the passer) and a defense team (four players attempt to intercept the ball). The offense 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 opponent team tries to intercept it (see Figure 7).

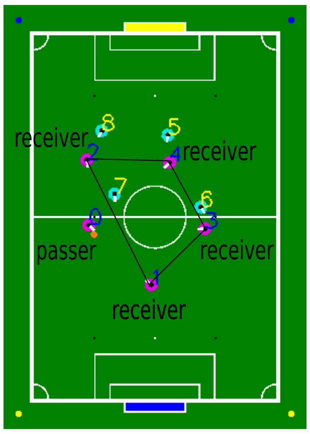


Figure 7: A passer and the defense formation

Only the passer learns its behavior while the receivers and the defense 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 offense (defense) 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.

3.3 Offense 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 Figure 7). The initial positions of the team members are randomly arranged inside their own half of the field.

3.4 Defense team

The defense 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 offense team member and between the offense and its own goal (see Figure 7). The offense team member attempts to catch the ball if it is approaching. In order to avoid the disadvantage of the offense team, the defense 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 own half of the field but outside the center circle.

4 STATE AND ACTION SPACES OF LEARNING MODULES

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 given in advance before the learning of the gate module.

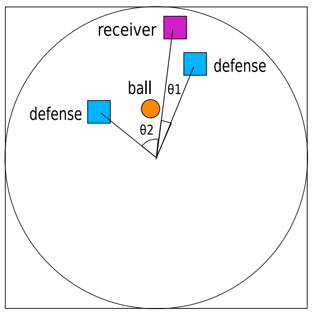
4.1 State and action spaces for lower action modules

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.

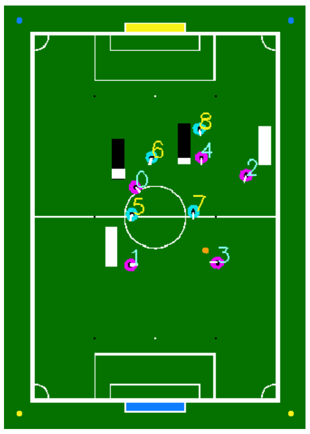
4.1.1 State space for the passing module

The state space of the passing module S is defined on the omni directional camera image as follows (see Figure 8(a)):

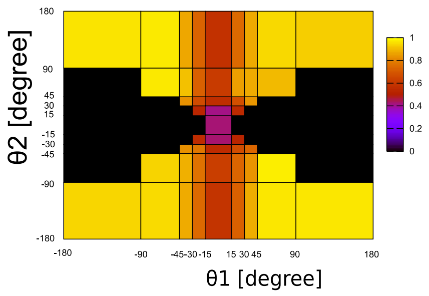
- the smallest angle among angles between the receiver and one the defense players who is nearer to the passer than the receiver (θ_1), and
- the angle between the receiver and one of the defense players who is nearest to the passer (θ_2).



(a)



(b)



(c)

Figure 8: 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 Figure 8(b) where the passer is the robot 3 (hereafter, r3 in short), and the white 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 defense players, their state values are high while the state values for r0 and r4 are low since their pass courses are intercepted by the defense players.

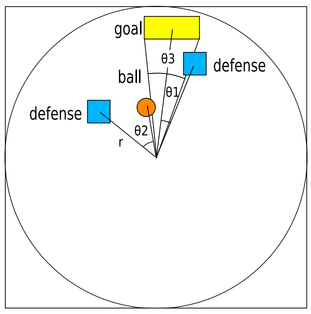
The state value map is shown in Figure 8(c) that indicates the smaller the angle between the receiver and the defense player is, the lower the state value is. The black region (one region is separated in the figure) is inexperienced area.

4.1.2 State space for the dribble-shoot module

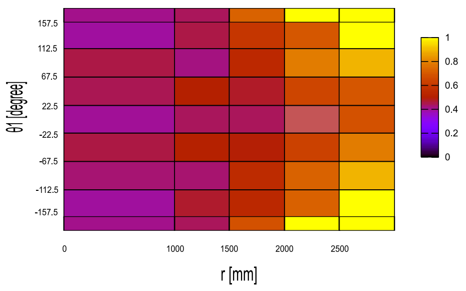
The state space of the dribble-shoot module S is defined on the omni directional camera image as follows (see Figure 9(a)):

- the angle between the opponent goal and one of the defense players who is nearest to the goal (θ_1),
- the angle between the ball and one of the defense players who is nearest to the passer (θ_2),
- the distance to the nearest defense 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.



(a)



(b)

Figure 9: State variables (a) and state values (b) for the dribble and shoot module

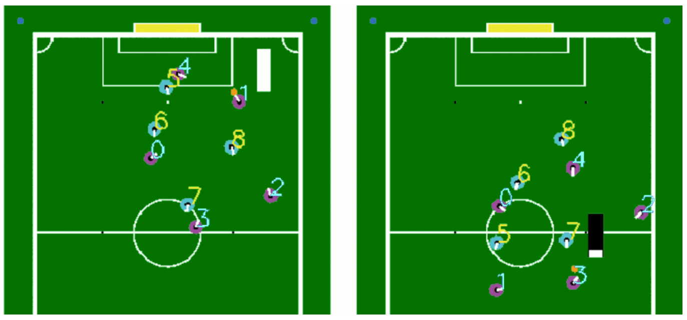


Figure 10: Two examples of the state values: high (left) and low (right)

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 Figure 9(b) that indicates the nearer the defense 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 Figure 10 where the white 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 defense players around in Figure 10 (left), the state value is high while the state value of the passer (r3) in Figure 10 (right) is low since it is located far from the goal and the defense players are around it.

4.1.3 State Space For the Receiver's State Value Estimation Module

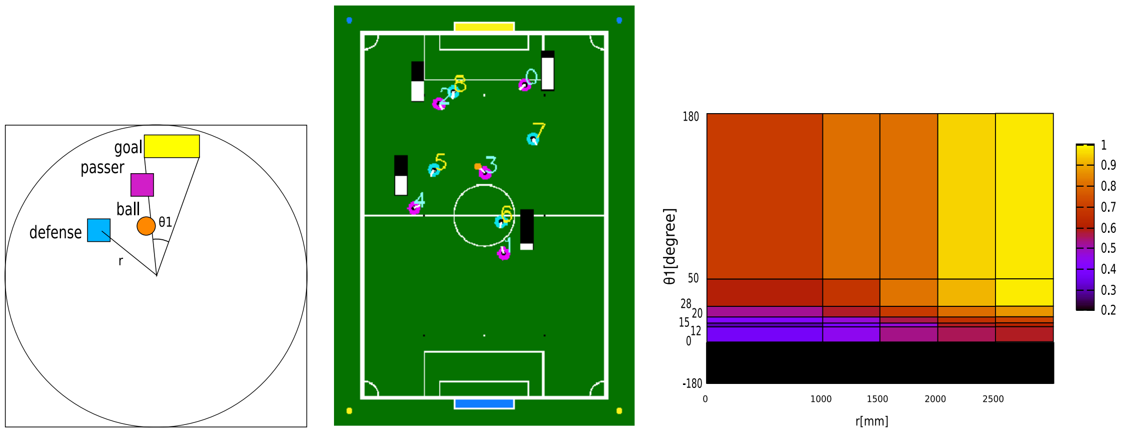
The passer infers each receiver's state that indicates how easily the receiver can shoot the passed ball to the goal by reconstructing its TV camera view of the scene from the passer's omnidirectional view. Since we suppose that the passer has already learned the shooting behavior, the passer can estimate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

- The angle between the both side edged of the opponent goal (θ_1) that represents the distance to the goal (see Figure (11(a)).).

The values of r and θ_1 are quantized into five and seven levels, respectively, therefore the number of states are 5 x 7 = 35.



(a) (b) (c)

Figure 11: State variables (a), examples of state values (b), and state value map (c) of the receiver module

An example of the state values of the four receiver's state value estimation modules is shown in Figure 11(b) where the white 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 defense players around, the state value is high while the state values of other receivers (r1, r2, and r4) are low since they are located far from the goal and/or the opponents are around them.

The state value map of the receiver's state value estimation module in terms of θ_1 and r is shown in Figure 11(c) that indicates the nearer (further) the defense player is and the further (nearer) the goal is, the smaller (larger) the state value is. The black region is inexperienced area.

4.2 State/action spaces for the cooperative/competitive behavior learning module

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, that is, the value smaller than 0.5 is 0 and else 1. Therefore its size is 2^4 x 2 x 2^4=512. The action set A for the gate module consists of all action modules, that is, one shooting behavior and behavior of passing to 4 different teammates.

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.

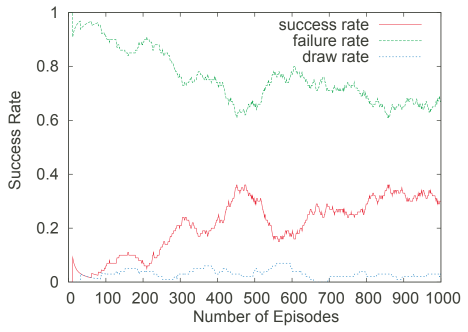
5 EXPERIMENTAL RESULTS

It took about 300 episodes for each to learn the state value function of the low-level module. After the state value functions were acquired, the cooperative/competitive behavior at the gate module started. The success rate during the learning of the cooperative/competitive behavior in the mini game is shown in Figure 12(a) where the action selection is 80% greedy and 20% random to cope with new situations. Around the 900th episode, the learning seems to have converged at 30% success, 70% failure, and 10% draw.

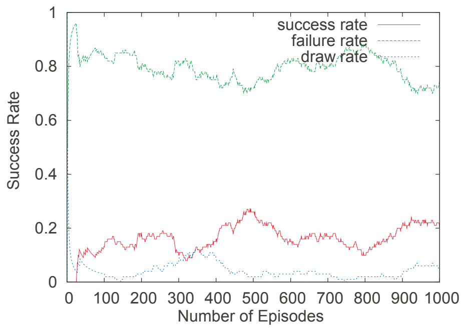
Kalyanakrishnan et al. [3] had experiments on the RoboCup Soccer robot simulator. The experimental setup is similar to ours. The number of robots for each team are exactly same with ours, that is, 4 robots belong to the defense team and 5 robots to the offense team. They also use omni-vision system and the robots receive low-level information to estimate the positions of players and the ball. They use the state using a set of variables involving distances and angles between players. The number of state variables is 17. In our experiments, the gate module uses the state using estimated state values of lower modules of self and others and the number of state variables is 9. The lower learning modules use low-level logical sensory information to construct state space described in the last section. The action set of [3] is almost equivalent to the one of ours; the actions in [3] are passing to a teammate (one of 4 teammates), dribbling, and shooting, while our robots has the macro actions of the passing to a teammate and the shooting.

Compared to the results of [3] that has around 30% success rate with 30,000 trials, the learning time is drastically improved (30 times quicker). Even if the learning time for low-level behavior is taken into account, the total learning time of our method is around 1,800 trials and still much shorter than [3]. Figure 12(c) indicates the number of passes where it decreases after the 350 trials that means the number of useless passes decreased.

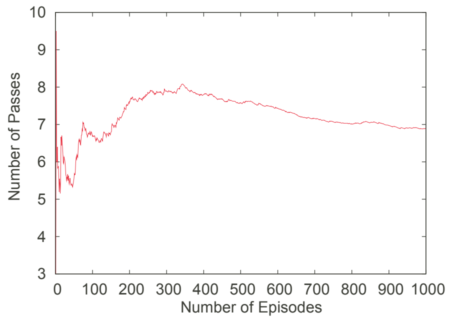
In cases of the success, failure, and draw rates when 100% greedy and 100% random are 55%, 35%, 10%, and 2%, 97%, 1%, respectively. The reason why the success rate in case of 100% greedy is better



(a) Success rate with the receiver's state value estimation modules



(b) Success rate without the receiver's state value estimation modules



(c) Number of passes

Figure 12: Success rates and number of passes during learning

than in case of 80% greedy seems that the control policies of the receivers and the defense players are fixed, therefore not so new situations happened.

An example of acquired behavior is shown in Figure 13 where a sequence of twelve top views indicates a successful pass and shoot scene. In each view, the following behavior happened.

1. All players take the random initial positions in own territories, and the offense player r3 nearest to the ball becomes the passer and it kicked off.
2. r3 dribbles the ball and attempts to pass the ball to the receiver r1.
3. r3 passes the ball to r1 and runs to the goal (pass and go behavior), and r1 becomes the passer and attempts to pass the ball to r0.
4. r1 passes the ball to r0 and runs to the goal (pass and go behavior), and r0 becomes the passer.
5. r0 dribbles the ball.
6. r0 attempts to pass the ball to r2.
7. r0 runs to the goal (pass and go behavior), and r2 becomes the passer.
8. r2 dribbled the ball.
9. r2 attempts to pass the ball to r3.

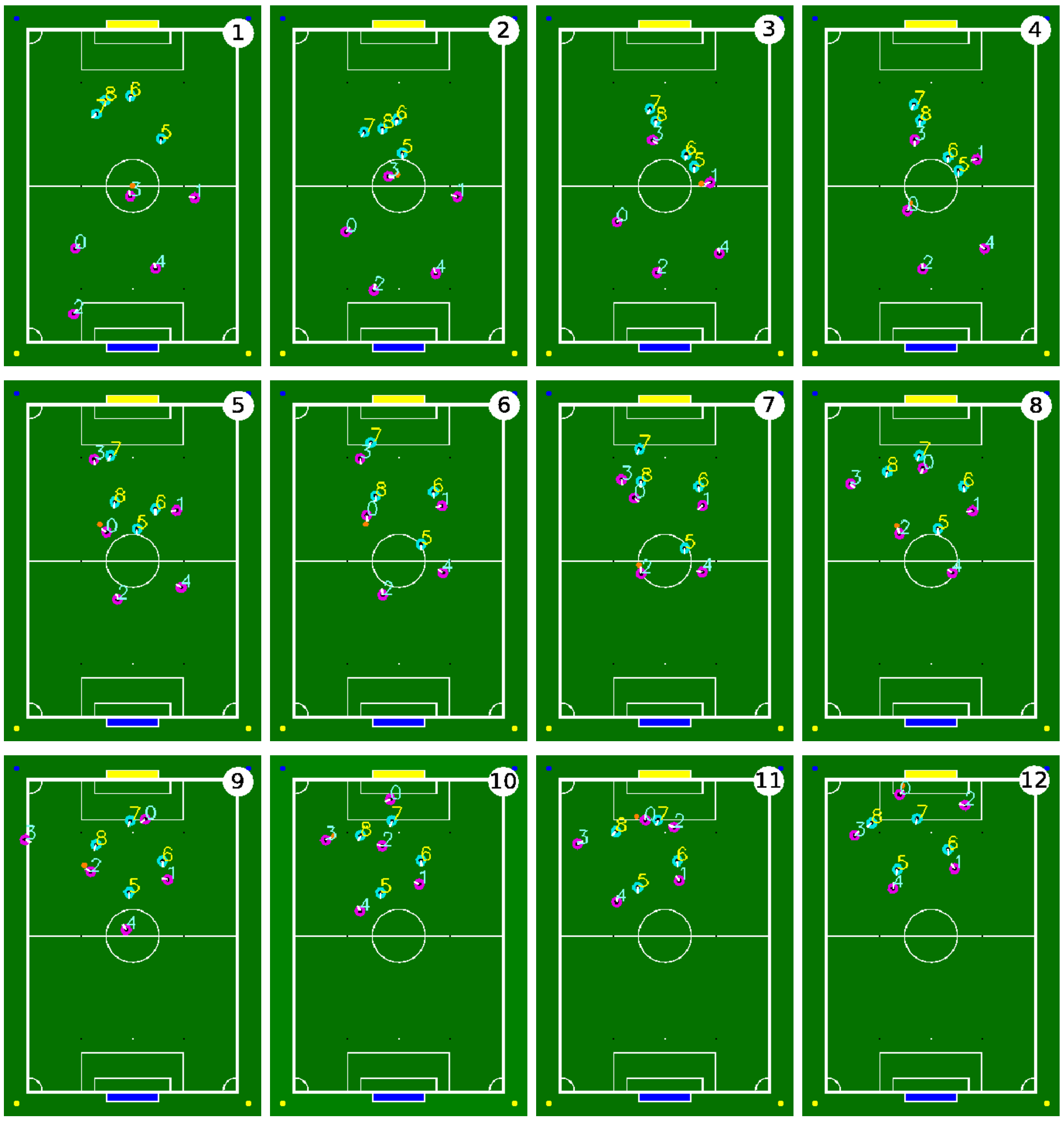


Figure 13: An example of the acquired behavior

10. r2 runs to the goal (pass and go behavior), and r3 becomes the passer.

11. r3 passed the ball to r0.

12. r0 becomes the passer and shoots the ball into the goal.

6 DISCUSSION

We have used the state values and macro actions instead of the sensor values and motor commands in the physical real world, respectively, 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. We discuss the learning results compared to that of the method by Kalyanakrishnan et al. [3] since their experimental setup is very close to ours.

The results of their method were 32% success with communication at around the 30,000th trial when the learning seems to have converged. We have much improved the learning time to achieve the almost same success rate (see Figures 12(a)) owing to adopting the state values instead of the physical sensor values by them.

Their result of the success rate without communication was 23% that seems corresponding to our result, around 21%, without the receiver's state value estimation modules (see Figure 12(b)). This suggests that the receiver's state value estimation modules can contribute to almost the same role of the communication without the explicit information exchange (communication) in their method. That is, our method can share more abstracted information with the others while their method exchanges sensory level information among players.

The main contribution of the state abstraction based on low-level behavior is to keep the size of state small and enables fast learning. This paper shows 4 on 5 game and the proposed abstraction methodology will work on, for example, 5 on 5 or 5 on 6 team matches. Taylor et al. [16] shows that transferring knowledge learned in case of 3 on 2 game to learning behavior in case of 4 on 3 game improved learning time performance. The transfer learning will work as well in our method of the state abstraction.

We may conclude that the state and action space abstraction (the use of state values of behavior of self and others and macro actions) contributed to the reduction of the learning time while the use of the receiver's state value estimation modules contributed to the improvement of the teamwork performance. Real robot implementation is our future work.

REFERENCES

[1] S. Elfwing, E. Uchibe, K. Doya, and H. I. Chirstensen, "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. 13, Sendai, Japan, 2004, pp. 3164–2220.

[2] S. Ikenoue, 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*, Lausanne, Switzerland, 2002, pp. 2728–2734.

[3] S. Kalyanakrishnan, Y. Liu, and P. Stone, "Half field offense in robocup soccer: A multiagent reinforcement learning case study," in *RoboCup 2006 Symposium papers and team description papers*, G. Lakemeyer, E. Sklar, D. Sorrenti, and T. Takahashi, Eds., Bremen, Germany, Jun 2006, pp. CD–ROM.

[4] P. Stone, R. S.Sutton, and G. Kuhlmann, "Scaling reinforcement learning toward robocup soccer," *Journal of Machine Learing Research*, vol. 13, pp. 2201–2220, 2003.

[5] 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*, Lausanne, Switzerland, October 2002, pp. CD–ROM 927–931.

[6] R. Jacobs, M. Jordan, N. S, and G. Hinton, "Adaptive mixture of local expoerts," *Neural Computation*, vol. 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] S. Whitehead, J. Karlsson, and J. Tenenberg, "Learning multiple goal behavior via task decomposition and dynamic policy merging," in *ROBOT LEARNING*, J. H. Connell and S. Mahadevan, Eds. Kluwer Academic Publishers, 1993, ch. 3, pp. 45–78.

[9] 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 Corporation, Tech. Rep., June 2000.

[10] P. Stone and M. Veloso, "Layered approach to learning client behaviors in the robocup soccer server," *Applied Artificial Intelligence*, vol. 12, no. 2-3, 1998.

[11] A. Kleiner, M. Dietl, and B. Nebel, "Towards a life-long learning soccer agent," in *The 2002 International RoboCup Symposium Pre-Proceedings*, G. A. Kaminka, P. U. Lima, and R. Rojas, Eds., Fukuoka, Japan, June 2002, pp. CD–ROM.

[12] Y. Takahashi and M. Asada, "Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, Takamatsu, Japan, 2000, pp. 395–402.

[13] Y. Takahashi, T. Kawamata, and M. Asada, "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*, Beijing, China, 2006, pp. 25–31.

[14] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction.* Cambridge, MA: MIT Press, 1998. [Online]. Available: citeseer.ist.psu.edu/sutton98reinforcement.html

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

[16] M. E. Taylor, P. Stone, and Y. Liu, "Transfer learning via inter-task mappings for temporal difference learning," *Journal of Machine Learning Research*, vol. 8, no. 1, pp. 2125–2167, 2007.