Modular Learning Systems for Soccer Robot

Yasutake Takahashi and Minoru Asada

Graduate School of Engineering, Osaka University
Osaka, Japan

Abstract. This paper presents a series of the studies of modular learning system for vision-based behavior acquisition of a soccer robot participating in middle size league of RoboCup (Asada, et al. 1999). Reinforcement learning has recently been receiving increased attention as a method for behavior learning with little or no a priori knowledge and higher capability of reactive and adaptive behaviors. However, simple and straightforward application of reinforcement learning methods to real robot tasks is considerably difficult due to its endless exploration of which time is easily scaled up exponentially with the size of the state/action spaces, that seems almost impossible from a practical viewpoint. Further, the existing reinforcement learning approaches have been suffering from the policy alternation of others in multi-agent dynamic environments such as RoboCup competitions since other agent behaviors may cause sudden changes of state transition probabilities of which constancy is necessary for the learning to converge. In order to cope with the above two issues, we introduced a multi-layered modular learning system. To show the validity of the proposed methods, we apply them to simple soccer situations in the context of RoboCup with real robots, and show the experimental results.

Keywords: Reinforcement Learning, Multi-layered learning system, Multi-module learning system, RoboCup

1 Introduction

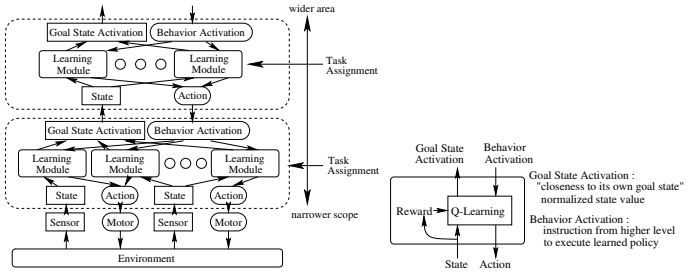
There have been a lot of work on different learning approaches to acquisition of purposive behaviors of robots based on the methods such as reinforcement learning, genetic algorithms, and so on. Especially, reinforcement learning has recently been receiving increased attention as a method for behavior learning with little or no a priori knowledge and higher capability of reactive and adaptive behaviors. However, simple and straightforward application of reinforcement learning methods to real robot tasks is considerably difficult due to its endless exploration of which time is easily scaled up exponentially with the size of the state/action spaces, that seems almost impossible from a practical viewpoint.

One of the potential solutions might be application of so-called "mixture of experts" proposed by Jacobs and Jordan (Jacobs, et al. 1991), in which a set of expert modules learn and one gating system weights the output of the each expert module for the final system output. This idea is very general and has a wide range of applications. However, we have to consider the following two issues to apply it to the real robot tasks:

- Abstraction of state and/or action spaces for scaling up: the original "mixture of experts" consists of experts and and gate for expert selection. Therefore, no more abstraction beyond the gating module. In order to cope with complicated real robot tasks, abstraction of the state and/or action spaces is necessary.
- Adaptation of other agents' policy alternation: the policy alternation of others in multi-agent dynamic environments dynamically changes the state transition probabilities from the viewpoint of the learning agent and a monolithic learning module cannot handle it. A modular learning approach would provide one solution to this problem.

A basic idea to cope with the first issue is that the learning modules generate not only the rational behaviors but also inform the abstracted situations to the higher module; how feasible the module is, how close to its subgoal, and so on. It is reasonable to utilize such information in order to construct state/action spaces of the higher modules from already abstracted situations and behaviors of the lower ones. Thus, the hierarchical structure can be constructed with not only experts and gating module but more layers with multiple homogeneous learning modules. The idea to handle the second issue is that the learning agent could assign one behavior learning module to each situation which is caused by the other agent's policy. The multiple modules are assigned to different situations and learn purposive behaviors for the specified situations which are regarded as the consequence of other agent's behavior under different policies. In this paper, we show a series of studies of multi-layered modular learning system for vision-based behavior acquisition of a soccer robot participating in middle size league of RoboCup (Asada et al. 1999).

2 Behavior Acquisition based on Multi-Layered Learning System



(a) A whole system

(b) A behavior learning module

Figure 1. A hierarchical learning architecture

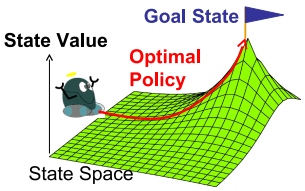


Figure 2. A sketch of a state value function

The architecture of the multi-layered reinforcement learning system is shown in Figure 1, in which (a) and (b) indicate a hierarchical architecture with two levels and an individual learning module embedded in the layers, respectively. Each module has its own goal state in its state space, and it learns the behavior to reach the goal, or maximize the sum of the discounted reward received over time based on Q-learning. The state and action spaces are constructed using sensory information and motor command, respectively at the bottom level. The input and output to/from the higher level are the goal state activation and the behavior activation, respectively, as shown in Figure 1(b). The goal state activation g is a normalized state value, and $g = 1$ when the situation is the goal state. When a module receives a behavior activation from the higher modules, it calculates the optimal policy for its own goal, and sends action commands to the lower module. The action command at the bottom level is translated to an actual motor command, then the robot takes the action in the environment.

One basic idea is to use the goal state activations g of the lower modules as the representation of the situation for the higher modules. Figure 2 shows a sketch of a state value function where a robot receives a positive reward when it reach to a specified goal. The state value function can be regarded as closeness to the goal of the module. The states of the higher modules are constructed using the patterns of the goal state activations of the lower modules. In contrast, the actions of the higher level modules are constructed using the behavior activations to the lower modules.

Figure 3. A mobile robot, a ball and a goal

Figure 3 shows a picture of the environment where a mobile robot we designed and built, a ball, and a goal are included. It has two TV cameras: one has a wide-angle lens, and the other a omni-directional mirror. The driving mechanism is PWS (Powered Wheels Steering) system, and the action space is constructed in terms of two torque values to be sent to two motors that drive two wheels.

In (Takahashi & Asada 2000), the robot receives the information of only one goal for simplicity. The state space at the bottom layer is constructed in terms of the centroids of goal region on the images of the two cameras. The action space is constructed in terms of two torque values to be sent to two motors corresponding to two wheels. The state and action spaces at the upper layer are constructed by the learning modules at the lower layer which are automatically assigned. The experiment is constructed with two stages: the learning stage and the task execution one based on the learned result. First of all, the robot moves at random in the environment for about two hours. The system learns and constructs the four layers and one learning module

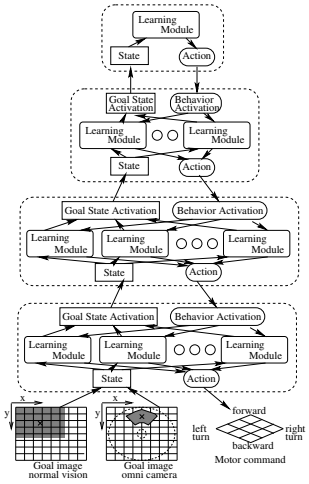


Figure 4. A hierarchical architecture on a monolithic state space

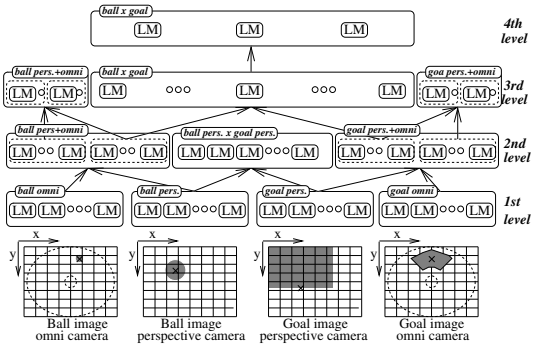


Figure 5. A hierarchy architecture on decomposed state spaces

is assigned at the top layer (Figure 4). We call each layer from the bottom, “bottom”, “middle”, “upper”, and “top” layers. In this experiment, the system assigned 40 learning modules at the bottom layer, 15 modules at the middle layer, and 4 modules at the upper layer. Figure 6 shows a rough sketch of the state transition and the commands to the lower layer on the multi-layered learning system during navigation task. The robot was initially located far from the goal, and faced the opposite direction to it. The target position was just in front of the goal. The circles in the figure indicate the learning modules and their numbers. The empty up arrows (broken lines) indicate that the upper learning module recognizes the state which corresponds to the lower module as the goal state. The small solid arrows indicate the state transition while the robot accomplished the task. The large down arrows indicate that the upper learning module sends the behavior activation to the lower learning module.

The system mentioned above dealt with a whole state space from the lower layer to the higher one. Therefore, it cannot handle the change of the state variables because the system supposes that all tasks can be defined on the state space at the bottom level. Further, it is easily caught by a curse of dimension if number of the state variables is large. In (Takahashi & Asada 2001), we introduce an idea that the system constructs a whole state space with several decomposed state spaces. At the bottom level, there are several decomposed state spaces in which modules are assigned to acquire the low level behavior in the small state spaces. The modules at the higher level manage the lower modules assigned to different state spaces. In this paper, we define the term “layer” as a group of modules sharing the same state space, and the term “level” as a class in the hierarchical structure. Figure 5 shows an example hierarchical structure. At the lowest level, there are four learning layers, and each of them deals with its own logical sensory space (ball positions on the perspective camera image and omni one, and goal position on both images). At the second level, there are four learning layers. The “*ball pers.*$\times$*goal pers.*” layer deals with lower modules of “*ball pers.*” and “*goal pers.*” layers. The arrows in the figure indicate the flows from the goal state activations to the state vectors. The arrows from the action vectors to behavior activations are eliminated. At the third level, the system has three learning layers, again.

After the learning stage, we let our robot do a couple of behaviors, for example, chasing a ball, moving in front of the goal, and shooting a ball into the goal, using this multi-layer learning structure. When the robot behaves chasing a ball, the system uses “*ball pers.*” and “*ball omni*” layers at first level, “*ball pers.+omni*” at second level, and “*ball pers.+omni*” at third level. Also when the robot shoots a ball into a goal, the system uses all four layers at the first level, all three layers at second level, “*ball x goal*” layer at the third level, and the layer at the fourth level. All layers at the first level and “*ball pers.+omni*” and “*goal pers.+omni*” layers are reused by these three behaviors. In the case of the shooting behavior, the target situation is given by reading the sensor information when the robot pushes the ball into the goal; the robot captures the ball and goal at center bottom in the perspective camera image. As an initial position, the robot is located far from the goal, faced opposite direction to it. The ball was located between the robot and the goal. Figure 7 shows a sequence of the behavior activation of learning modules and the commands to the lower layer modules. The down arrows indicate that the higher learning modules fire the behavior activations of the lower learning modules.

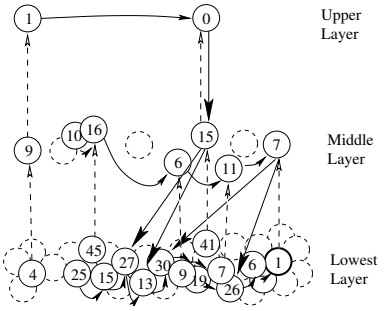


Figure 6. A rough sketch of the state transition on the multi-layer learning system

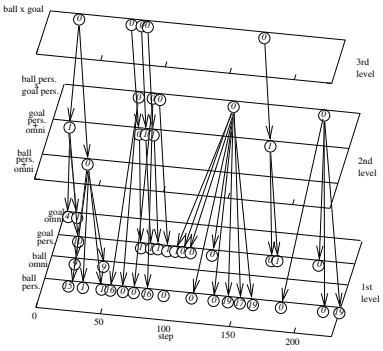


Figure 7. A sequence of the behavior activation of learning modules and the commands to the lower layer modules

3 Modular Learning System for Multi-Agent System

The basic idea to cope with other agents' policy alternation is that the learning agent could assign one behavior learning module to each situation which is caused by the other agents and the learning module would acquire a purposive behavior under the situation if the learning agent can distinguish a number of situations in which the state transition probabilities are constant. We introduce a modular learning approach to realize this idea. A module consists of a learning component that models the world and an execution-time planning component. The whole system performs the following procedures simultaneously:

- find 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 prepare a case of ball passing behavior without interception by the opponent player (Figures 10,12). In the environment there are a learning agent (passer), a ball, an opponent, and two teammates (receivers). The problem here is to find the model which can most accurately describe the opponent's behavior from the viewpoint of the learning agent and to execute the policy which is calculated under the estimated model.

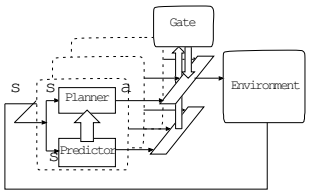


Figure 8. A multi-module learning system

Figure 8 shows a basic architecture of the proposed system, that is, a multi-module reinforcement learning system. Each module has a forward model (predictor) which represents the state transition model, and a behavior learner (policy planner) which estimates the state-action value function based on the forward model in a reinforcement learning manner. This idea of combination of a forward model and a reinforcement learning system is similar to the H-DYNA architecture (Singh 1992) or MOSAIC (Doya, et al. 2000). 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, and the selected module sends action commands based on its policy.

Each learning module has its own state transition model and a reward model. An approximated state-action value function for a state action pair is calculated based on these models. The gating signal of the module becomes larger if the module does better state transition prediction during a certain period, else it becomes smaller. We assume that the module which 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 describes the situation

best. In our proposed architecture, the gating signal is used for gating the action outputs from modules. If all modules show wrong prediction of state transition, that means all gating signals of the modules become small, the system adds one learning module and feeds data of sensory-motor sequence to this modules for a while.

Figure 9 shows a mobile robot we have designed and built. Figure 10 shows the simulator of our robots and the environment. The robot has an omni-directional camera system. A simple color image processing is applied to the detection of the ball area and opponent ones in the image in real-time (every 33ms). The left of Figure 10 shows a situation in which the agent can encounter and the bottom right shows the simulated image of the camera with the omni-directional mirror mounted on the robot. The robot consists of an omni-directional vehicle of which motion (any translation and rotation on the plane) can be controlled.

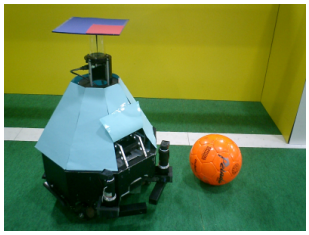


Figure 9. A real robot

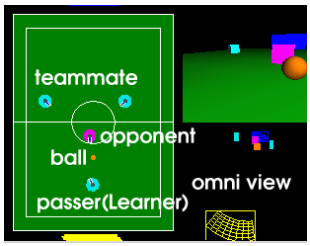
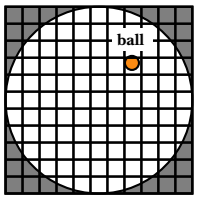


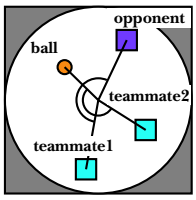
Figure 10. A simulation environment

The state space is constructed in terms of the centroid of the ball on the image, the angle between the ball and the opponent, and the angles between the ball and the teammates (see Figures 11 (a) and (b)). The action space is constructed in terms of desired three velocity values (x_d, y_d, w_d) to be sent to the motor controller (Figure 11 (c)). The robot has a pinball like kick device, and it automatically kicks the ball whenever the ball comes to the region to be kicked. It tries to estimate the mapping from sensory information to appropriate motor commands by the proposed method.

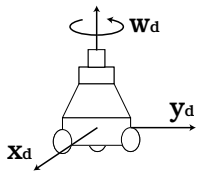
The initial positions of the ball, the passer, the opponent, and teammates are shown in Figure 12. The opponent has two kinds of behaviors; it defend the left side, or right side. The passer agent has to estimate which direction the opponent will defend and go to the position in order to kick the ball to the direction the opponent does not defend. From a viewpoint of the multi-module learning system, the passer agent will estimate which situation of the module is going on, select the most appropriate module to do. The passer agent acquires a positive reward when it approach to the ball and kicks it to one of the teammate dodging the opponent.



(a) state variables (position)



(b) state variables (angle)



(c) action variables

Figure 11. A state-action space

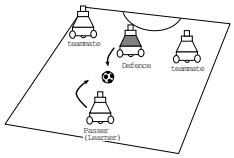
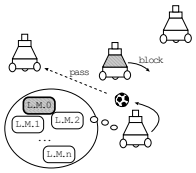
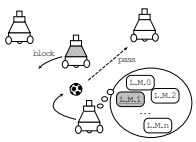


Figure 12. Task : 3 on 1



(a) right block



(b) left block

Figure 13. Module switching

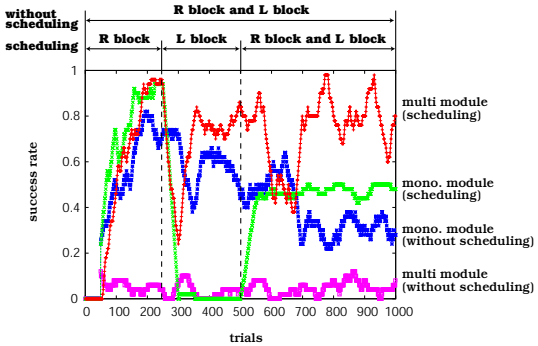


Figure 14. Success rate during the learning

We prepare a learning schedule composed of three stage to show its validity. The opponent fixes its defending policy as 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. Then, the opponent changes the defending policy randomly every trial.

We have applied the method to a learning agent and compared it with one module learning system. We have also compared the performances between the methods with and without the learning schedule. Figure 14 shows the success rates of these methods during the learning. The success means that the learning agent successfully pass the ball without interception by the opponent and the success rate indicates the rate of the number of successes in the last 50 trials. The multi-module system with schedule shows better performance than the one-module system. The monolithic module system ("mono. module" in the figure) tries to acquire a single behavior for both policies of the opponent (left and right defending) with only one learning module. The "without scheduling" means that we do not applied learning schedule and the opponent changes its policy at random from the beginning. The multi-module system without learning schedule shows the worst performance in our experiments. This result indicates that it is very difficult to recognize the situation at the early stage of the learning because the modules has too few experiences to evaluate their fitness, then the system tends to select the module without any consistency. As a result, the system cannot acquires any valid policies at all.

4 Future Work

As future work, there are a number of issues to extend our current methods.

Interference between modules One module behavior might have interference to another one which has different actuators. For example, the action of a navigation module will disturb the state transition from the view point of the kicking device module; the catching behavior will be success if the vehicle stays while it will fail if the vehicle moves.

Self-assignment of modules It is still an important issue to find a purposive behavior for each learning module automatically. In (Takahashi & Asada 2000), the system distributes modules on the state space uniformly, however, it is not so efficient. In (Takahashi, et al. 2003), the system decomposes the task by itself, however, the method uses many heuristics and needs instruction from a coach. In many cases, the designers have to define the goal of each module by hand based on their own experiences and insights.

Self-construction of hierarchy Another missing point in the current method is that it does not have the mechanism that constructs the learning layer by itself.

Self-segmentation of situation (Edazawa, et al. 2004) shows that the straightforward application of a modular learning system to the multi-agent system has not shown the enough performance of self-segmentation of situation.

References

M. Asada, et al. (1999). 'RoboCup: Today and tomorrow – What we have learned'. *Artificial Intelligence* pp. 193–214.

K. Doya, et al. (2000). 'Multiple Model-based Reinforcement Learning'. Tech. rep., Kawato Dynamic Brain Project Technical Report, KDB-TR-08, Japan Science and Technology Corporatio.

K. Edazawa, et al. (2004). 'Modular Learning System and Scheduling for Behavior Acquisition in Multi-Agent Environment'. In *RoboCup 2004 Symposium papers and team description papers*, pp. CD–ROM.

R. Jacobs, et al. (1991). 'Adaptive mixture of local expoerts'. *Neural Computation* **3**:79–87.

S. P. Singh (1992). 'Reinforcement Learning with a Hierarchy of Abstract Models'. In *National Conference on Artificial Intelligence*, pp. 202–207.

Y. Takahashi & M. Asada (2000). '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, pp. 395–402.

Y. Takahashi & M. Asada (2001). 'Multi-Controller Fusion in Multi-Layered Reinforcement Learning'. In *International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI2001)*, pp. 7–12.

Y. Takahashi, et al. (2003). 'Incremental Purposive Behavior Acquisition based on Self-Interpretation of Instructions by Coach'. In *Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. CD–ROM.