Book Title
Book Editors
IOS Press, 2003

Incremental Purposive Behavior Acquisition based on Modular Learning System

Tomoki Nishi [a], Yasutake Takahashi [a,b,1] and Minoru Asada [a,b]
[a] *Graduate School of Engineering, Osaka University*
[b] *Handai FRC*

Abstract. A simple and straightforward application of reinforcement learning methods to real robot tasks is considerably difficult due to a huge exploration space that easily scales up exponentially since recent robots tend to have many kinds of sensors. One of the potential solutions might be application of so-called "mixture of experts" proposed by Jacobs et al.[1]; it decomposes a whole state space to a number of areas so that each expert module can produce good performance in the assigned small area. This idea is very general and has a wide range of applications, however, we have to consider how to decompose the space to a number of small regions, assign each of them to a learning module or an expert, and define a goal for each of them. In order to cope with the issue, this paper presents a method of self task decomposition for modular learning system based on self-interpretation of instructions given by a coach. Unlike the conventional approaches, the system decomposes a long-term task into short-term subtasks so that one learning module with limited computational resources can acquire a purposive behavior for one of these subtasks. Since instructions are given from a viewpoint of coach who has no idea how the system learns, they are interpreted by the learner to find the candidates for subgoals. Finally, the top layer of the hierarchical reinforcement learning system coordinates the lower learning modules to accomplish the whole task. The method is applied to a simple soccer situation in the context of RoboCup.

Keywords. Reinforcement Learning, Behavior Acquisition, Task decomposition, RoboCup

1. Introduction

One of the most formidable issues of reinforcement learning application to real robot tasks is how to find a compact state space in order to acquire a purposive behavior within reasonable learning time, and this has been much more serious since recent robots tend to have many kinds of sensors like normal and omni-vision systems, touch sensors, infrared range sensors, and so on. They can receive a variety of information from these sensors, especially vision sensors. This fact indicates that the difficulty of applying reinforcement learning to real robot tasks becomes more serious.

[1]Correspondence to: Yasutake Takahashi, YamadaOka 2-1, Suita, Osaka, 565-0871, Japan. Tel.: +81 6 6879 4123; Fax: +81 6 6879 4123; E-mail: yasutake@ams.eng.osaka-u.ac.jp

One of the potential solutions might be application of so-called “mixture of experts” proposed by Jacobs et al.[1], in which a whole state space is decomposed to a number of areas so that each expert module can produce good performance in the assigned area, 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. For example, Doya et al.[2] have proposed MOdular Selection And Identification for Control (MOSAIC), which is a modular reinforcement learning architecture for non-linear, non-stationary control tasks. However, all learning modules share the one state space that consists of a few variables.

In order to make a search space as compact as possible, it is not enough to divide the space to a number of areas and locate an expert to each of them and it is necessary to reduce a number of state variables that construct whole search space. Fortunately, a long time-scale behavior might often be decomposed into a sequence of simple behaviors each of which needs a few state variables in general, and therefore, the search space can be divided into smaller ones. In the existing studies, however, task decomposition and behavior switching procedures are given by the designers (e.g.,[3,4,5]). Others adopt the heuristics or the assumption that are not realistic from the viewpoint of real robot application (ex. [6,7,8,9]).

A basic idea to cope with the above issue is that any learning module has limited resource constraint, and this constraint of the learning capability leads us to introduce a multi-module learning system and specified sub-tasks by itself rather than sub-tasks are defined in advance by others. That is, one learning module has a compact state-action space and acquires a simple map from the states to the actions, and then a sub-task is defined by this module. We have already proposed the basic idea and showed some results with simulation and simple real robot experiments [10,11], however, the proposed algorithm needs a large amount of data in experiences that is hard to acquire with a real robot.

In this paper, we introduce an idea that the capability of a learning module defines the size of sub-tasks. We assume that each module can maintain a few numbers of state variables and this assumption is reasonable for real robot applications. Then, the system decomposes a long-term task into short-term sub-tasks with self-interpretation of coach instructions so that one learning module with limited computational resources can acquire a purposive behavior for one of these sub-tasks. In previous work [10,11], the system had to try all possible actions at all possible states because it used action value Q on each state to estimate availabilities of modules, then, it is almost impossible to apply to real robot tasks. We develop another approach to use state value V instead of the action value. We show experimental results with much more sensors such as normal and omni-vision systems and 8 directions infrared range sensors.

2. Basic Idea

There are a learner and a coach in a simple soccer situation (Figure 1). The coach has *a priori* knowledge of a task to be played by the learner while s/he does not have any idea about the system of the learner. On the other hand, the learner just follows the instructions without any knowledge of the task. After some instructions given by a coach, the learner decomposes the whole task into a sequence of subtasks, acquires a behavior for each

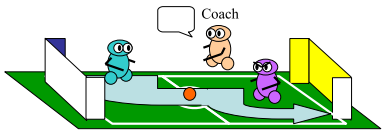

Figure 1. A coach gives instructions to a learner

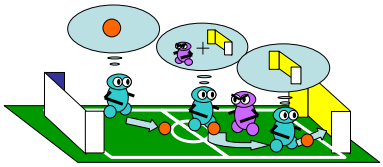

Figure 2. The learner follows the instructions and finds basic behaviors by itself

subtask, and coordinates these behaviors to accomplish the task by itself. In Figure 1, the coach instructs a shooting a ball into a yellow goal with obstacle avoidance. Figure 2 shows an example that the system decomposes this task into three subtasks and assigns them to three modules that maintain state spaces consist of ball variables, opponent and goal ones, and goal ones, respectively.

3. Robot, Tasks, and assumption

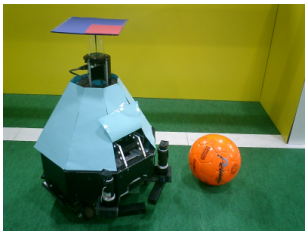

Figure 3. A real robot

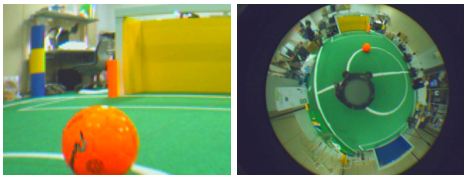

Figure 4. Captured camera images

Figure 3 shows a mobile robot we have designed and built. The robot has a normal camera in front of body, an omni-directional camera on the top, and infrared distance sensors around the body. Figure 4 show the images of both cameras. A simple color image processing is applied to detect the ball, the goal, and an opponent in the image in real-time (every 33ms). The robot has also 8 directions infrared range sensors. The robot has totally 39 candidates of state variables. The details of the candidates are eliminated because of space limitations. The mobile platform is an omni-directional vehicle (any translation and rotation on the plane). The tasks for this robot are chasing a ball, navigating on the field, shooting a ball into the goal, and so on. We assume that the given task has some absorbing goals, that is, the tasks are accomplished if the robot reaches to certain areas in state spaces that consist of a few state variables. We demonstrated only shooting behaviors for the task decomposition and the coordination. Figure 5 shows four examples of the behaviors instructed by the coach. The total number of instruction is 4 for this experiment.

4. Task Decomposition Procedure

Figure 6 show a rough sketch of the idea of the task decomposition procedure. The top of the Figure 6 shows a monolithic state space that consists of all state variables

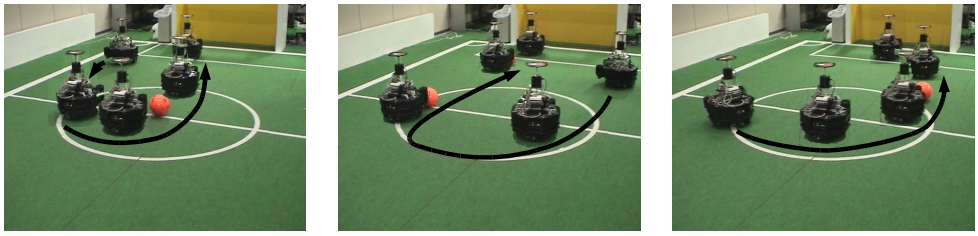

Figure 5. Examples of Instructed behaviors

$(x_1, x_2, \cdots, x_n)$. The red lines indicate sequences of state value during the given instructions. As we assume beforehand, the system cannot have such a huge state space, then, decomposes the state space into subspaces that consist of a few state variables. The system regards that the ends of the instructions represent goal states of the given task. It checks all subspaces and selects one in which the most ends of the instruction reach a certain area (G_{task} in Figure 6). The system regards this area as the subgoal state of a subtask that is a part of the given long-term task. The steps of the procedure are as follows:

1. find module unavailable areas in the instructions and regard them as unknown subtask.
2. assign a new learning module.
   (a) list up subgoal candidates for the unknown subtasks on the state space based on all state variables.
   (b) decompose the state space into subspaces that consist of a few state variables.
   (c) check all subspaces and select one in which the subgoal candidates reach a certain area best (G_{sub} in Figure 6).
   (d) generate another learning module with the selected subspace as a state space and the certain area as the goal state.
3. check the areas where the assigned modules are available.
4. exit if the generated modules cover all segments of instructed behaviors. Else goto 1.

5. Availability Evaluation and New Learning Module Assignment

The learner needs to check the availability of learned behaviors that help to accomplish the task by itself because the coach neither knows what kind of behaviors the learner has already acquired nor shows perfect example behaviors from the learner's viewpoint. The learner should suppose a module as valid if it accomplishes the subtask even if the greedy policy seems different from the example behavior. In order to judge the module is valid for executing the instructed behavior, a sequence of state value will be available. Figure 7 shows a sketch of state value function in which the state value is the biggest at the goal state and it distributes like a mountain. When an agent follows an optimal policy, it goes up the state value function. Then, if it goes up the state value mountain, it means the module seems valid for explaining the executing behavior.

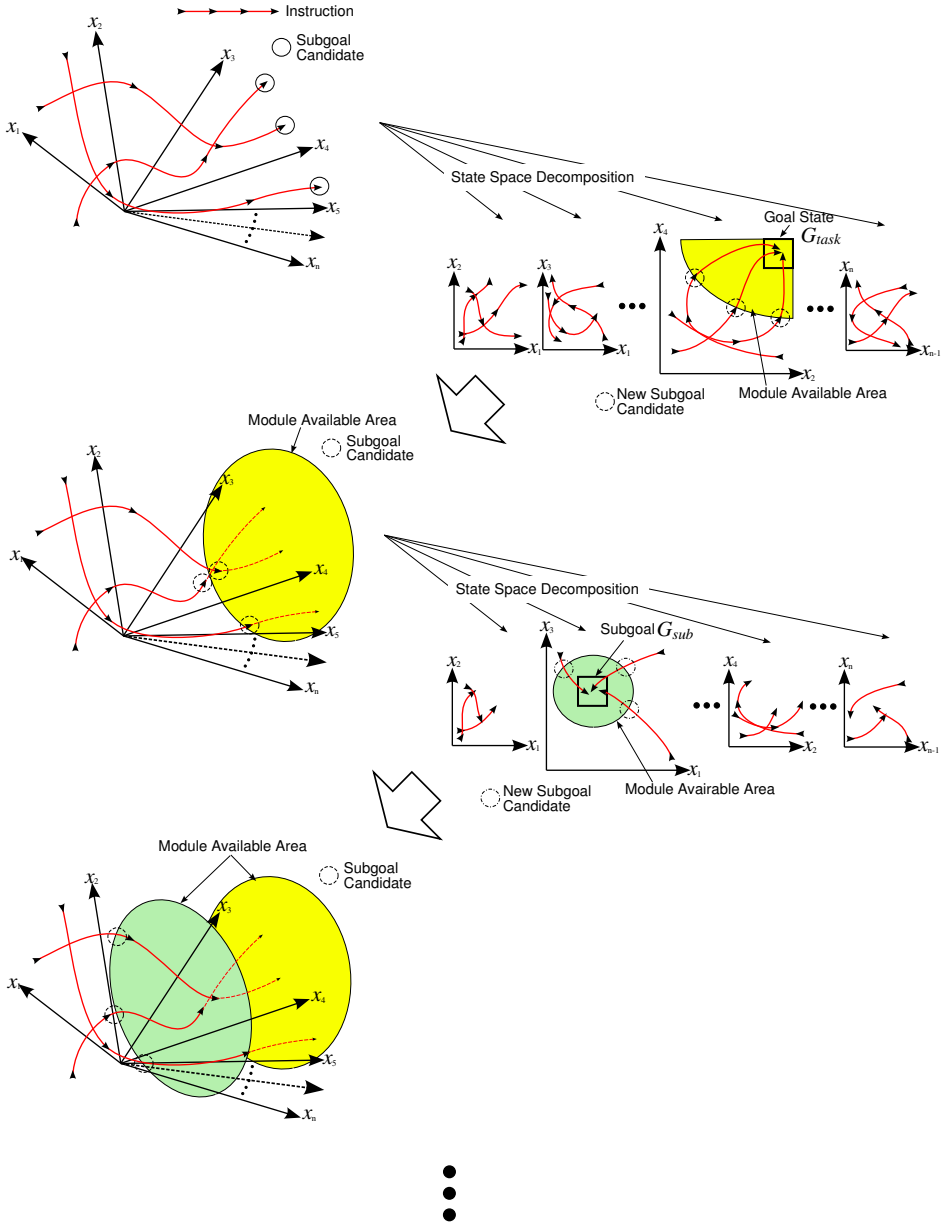

Figure 6. Rough sketch of the idea of task decomposition procedure

Now, we introduce AE in order to evaluate how suitable the module's policy is to the subtask:

$$AE(s) = \gamma \frac{V(s')}{V(s)} \ , \tag{1}$$

where γ, s, s', and $V(s)$ indicate discount factor, current state, next state, and state value

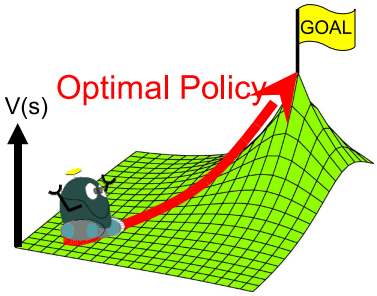

Figure 7. Sketch of state value function and optimal policy

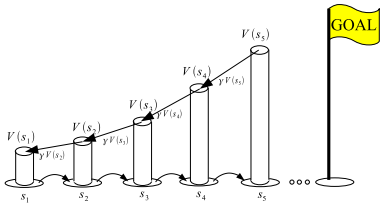

Figure 8. Sketch of propagation of state value

of state s, respectively. AE becomes larger if the instructed action leads to the goal state of the module while it becomes smaller if it leaves from the goal state. Figure 8 show a rough image of state value propagation in a simple case: States are chained straight one by one and an agent can move to only the next state in deterministic way. When the agent moves from, for example, state s_3 to state s_4 successfully, the state value $V(s_3)$ is propagated as $\gamma V(s_4)$, and $AE(s_3)$ will be $AE(s_3) = \gamma V(s_4)/V(s_3) = 1$. On the other hand, if the agent moves to s_2, the $AE(s_3)$ will be smaller. In order to decide whether the module is valid or not, we prepare a threshold AE_{th}, and the learner evaluates the module as valid for a period if $AE > AE_{th}$. If there are modules whose AE exceeds the threshold AE_{th}, the learner selects the module which keeps $AE > AE_{th}$ for longest period among the modules (see Figure 9). In Figure 6, "Module Available Area" indicates the one in which $AE > AE_{th}$.

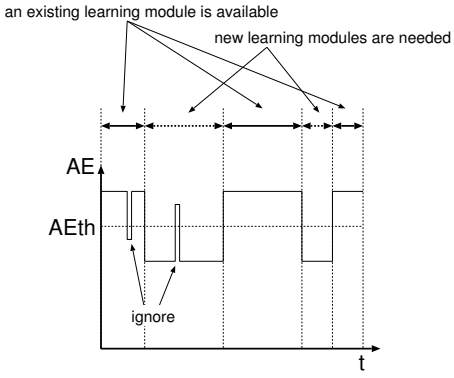

Figure 9. Availability identification during the given sample behavior

If there is no module which has $AE > AE_{th}$ for a period, the learner creates a new module which will be assigned to the subtask (see procedure 2 in **??**). To assign a new module to such a subtask, the learner identifies the state space and the goal state. The system follow the two steps to select an appropriate state space and the goal state for the subtask:

- selection of one state variable that specifies the goal state, and

- construction of a state space including the selected state variable.

In order to find one state variable that specifies the goal state best, the system lines up the candidates for a goal region in term of state variables. On the other hand, in order to select another state variable, the system evaluates performance of Q value estimation. The details of the procedure are eliminated because of space limitations.

6. Experiments

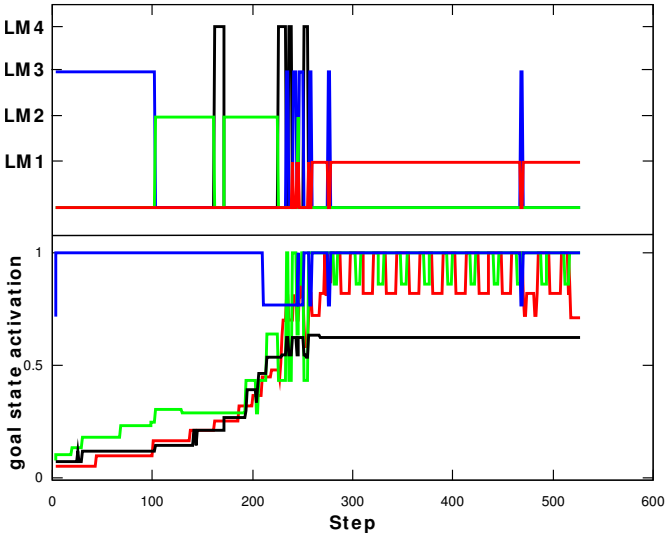

Figure 10. Sequences of the selected module, availability evaluations and goal state activations of modules through an instruction

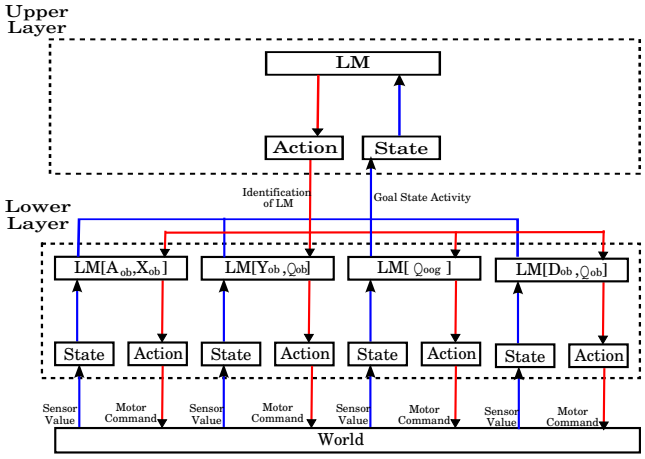

Figure 11. Acquired hierarchy for the shooting behavior

According to the learning procedure, the system produced four modules for the instructed behaviors. The modules are $LM_1(A_{pb}, X_{pb})$, $LM_2(\theta_{og})$, $LM_3(Y_{ob}, X_{ob})$, and $LM_4(A_{ob}, \theta_{ob})$. For example $LM_1(A_{pb}, X_{pb})$ indicates that the modules has a state space that consists of the area of ball on the normal camera image (A_ob) and the x position of the ball on the normal camera image (X_{pb}). Figure 10 shows sequences of the selected module, availability evaluations and goal state activations of modules through an instruction.

7. Conclusion

We proposed a hierarchical multi-module learning system based on self-interpretation of instructions given by a coach. We applied the proposed method to our robot and showed results for a simple soccer situation in the context of RoboCup.

References

[1] R. Jacobs, M. Jordan, N. S, and G. Hinton, "Adaptive mixture of local experts," *Neural Computation*, vol. 3, pp. 79–87, 1991.

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

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

[4] 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.

[5] P. Stone and M. Veloso, "Team-partitioned, opaque-transition reinforcement learning," in *RoboCup-98: Robo Soccer World Cup II* (M. Asada and H. Kitano, eds.), pp. 261–272, Springer Verlag, Berlin, 1999.

[6] B. L. Digney, "Emergent hierarchical control structures: Learning reactive/hierarchical relationships in reinforcement environments," in *From animals to animats 4: Proceedings of The fourth conference on the Simulation of Adaptive Behavior: SAB 96* (P. Maes, M. J. Mataric, J.-A. Meyer, J. Pollack, and S. W. Wilson, eds.), pp. 363–372, The MIT Press, 1996.

[7] B. L. Digney, "Learning hierarchical control structures for multiple tasks and changing environments," in *From animals to animats 5: Proceedings of The fifth conference on the Simulation of Adaptive Behavior: SAB 98* (R. Pfeifer, B. Blumberg, J.-A. Meyer, and S. W. Wilson, eds.), pp. 321–330, The MIT Press, 1998.

[8] B. Hengst, "Generating hierarchical structure in reinforcement learning from state variables," in *6th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2000)* (R. Mizoguchi and J. K. Slaney, eds.), vol. 1886 of *Lecture Notes in Computer Science*, Springer, 2000.

[9] B. Hengst, "Discovering hierarchy in reinforcement learning with HEXQ," in *Proceedings of the Nineteenth International Conference on Machine Learning (ICML02)*, pp. 243–250, 2002.

[10] Y. Takahashi and M. Asada, "Multi-layered learning systems for vision-based behavior acquisition of a real mobile robot," in *Proceedings of SICE Annual Conference 2003 in Fukui*, vol. CD-ROM, pp. 2937–2942, Aug 2003.

[11] Y. Takahashi, T. Nishi, and M. Asada, "Self task decomp osition for mo dular learning system through interpretation of instruction by coach," in *RoboCup 2005 Symposium papers and team description papers*, pp. CD–ROM, Jul 2005.