Advanced Robotic Systems Scientific Book 2005
URL: http:\\www.ars-journal.com
E-mail: publication@ars-journal.com
Vienna University of Technology
Automation and Control Institute
Gusshausstrasse 27-29, A-1040 Wien,
Austria, EU

MULTI-LAYERED LEARNING SYSTEM FOR REAL ROBOT BEHAVIOR ACQUISITION

YASUTAKE TAKAHASHI AND MINORU ASADA
GRADUATE SCHOOL OF ENGINEERING, OSAKA UNIVERSITY

1. Introduction

One of the main issues of autonomous robots is how to implement a system with learning capability to acquire both varieties of knowledge and behaviors through the interaction between the robot and the environment during its lifetime. There have been a lot of different works on learning approaches for robots to acquire behaviors 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, a simple and straightforward application of reinforcement learning methods to real robot tasks is considerably difficult due to its almost endless exploration of which time easily scales up exponentially with the size of the state/action spaces, which 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 & Jordan, 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:

- **Task decomposition:** how to find a set of simple behaviors and assign each of them to a learning module or an expert in order to achieve the given initial task. Usually, human designer carefully decomposes the long time-scale task into a sequence of simple behaviors such that the one short time-scale subtask can be accomplished by one learning module.
- **Abstraction of state and/or action spaces for scaling up:** the original "mixture of experts" consists of experts and a gate for expert selection. Therefore, no more abstraction beyond the gating module. In order to cope with complicated real robot tasks, more abstraction of the state and/or action spaces is necessary.

Connell and Mahadevan (Connell & Mahadevan, 1993) decomposed the whole behavior into sub-behaviors each of which can be independently learned. Morimoto and Doya (Morimoto & Doya 1998) applied a hierarchical reinforcement learning method by which an appropriate sequence of subgoals for the task is learned in the upper level while behaviors to achieve the subgoals are acquired in the lower level. Hasegawa and Fukuda (Hasegawa & Fukuda, 1999, 2001) proposed a hierarchical behavior controller, which consists of three types of modules, behavior coordinator, behavior controller and feedback controller, and applied it to a brachiation robot. Kleiner et al. (Kleiner et al., 2002) proposed a hierarchical learning system in which the modules at lower layer acquires low level skills and the module at higher layer coordinates them. However, in these proposed methods, the designers have done the task decomposition very carefully in advance, or the constructions of the state/action spaces for higher layer modules are independent from the learned behaviors of lower modules. As a result, it seems difficult to abstract situations and behaviors based on the already acquired learning/control modules.

There are a number of works of automatic task decomposition. Digney (Digney, 1996, Digney, 1998) has proposed Nested Q-learning algorithm that generates hierarchical control structures in a learning system. The task decomposition has been done under two criteria; one criterion is based on the received reinforcement signals, and the other is on the frequency of visits to particular state space locations. However, this work has been applied in a simple grid maze world, therefore the state space is fixed and its size is relatively small so that the frequency heuristics can work. In the case of real robots, the size of state space is huge if the state space consists of all sensory information, and it is very rare to visit the same state frequently. Hengst (Hengst, 2000, Hengst 2002) has proposed a method of generating hierarchical structure from state variables based on a heuristics that the almost constant variables represent higher-level states while the frequently changing variables represent lower level states. However, the designer gives these hierarchized variables and usually we cannot expect that real robots have such abstracted variables beforehand.

A basic idea to cope with the above two issues is that any learning module has limited resource constraint, and this constraint of the learning capability leads us to introduce a multi-module and multi-layered learning system. That is, one learning module has a compact state-action space and acquires a simple map from the states to the actions, and a gating system enables the robot to select one of the behavior modules depending on the situation. More generally, the higher module controls the lower modules depending on the situation. The definition of this situation depends on the capability of the lower modules because the gating module selects one of the lower modules based on their acquired behaviors. From the other viewpoint, the lower modules provide not only the rational behaviors but also the abstracted situations for 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 higher modules from already abstracted situations and behaviors of lower ones. Thus, the hierarchical structure can be constructed with not only experts and gating module but also more layers with multiple homogeneous learning modules.

In this paper, we show a series of studies towards the construction of such hierarchical learning structure developmentally. The first one (Takahashi & Asada, 2000) is automatic construction of hierarchical structure with purely homogeneous learning modules. Since the resource (and therefore the capability, too) of one learning module is limited, the initially given task is automatically decomposed into a set of small subtasks each of which corresponds to one of the small learning modules, and also the upper layer is recursively generated to cover the whole task. In this case, the all learning modules in the one layer share the same state and action spaces although some modules need the part of them. Then, the second work (Takahashi & Asada, 2001) and third one (Takahashi et al., 2003a) focused on the state and action space decomposition according to the subtasks to make the learning much more efficient. Further, the forth one

(Takahashi et al, 2003b) realized unsupervised decomposition of a long time-scale task by finding the compact state spaces, which consequently leads the subtask decomposition. We have applied these methods to simple soccer situations in the context of RoboCup (Asada et al., 1998) with real robots, and show the experimental results.

2. Multi-Layered Learning System

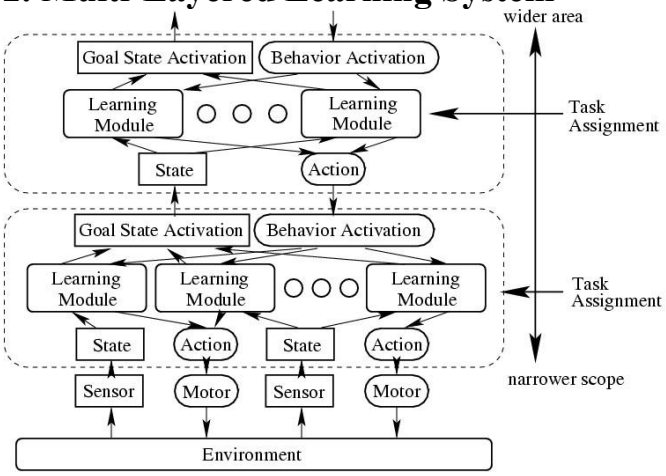

Fig. 1. Hierarchical architecture in multi-layered learning system

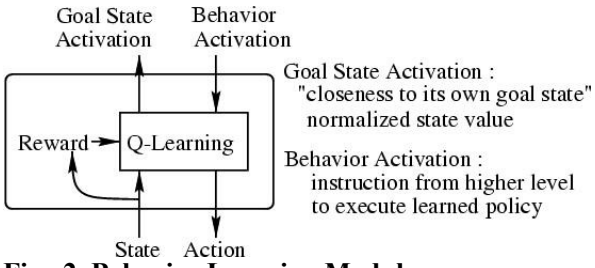

Fig. 2. Behavior Learning Module

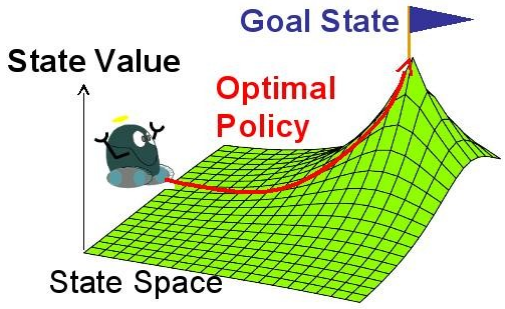

Fig. 3. Sketch of a state value function

Figs. 1 and 2 show the architecture of the multi-layered reinforcement learning system, in which indicate a hierarchical architecture with two levels, and an individual learning module embedded in the layers are indicated. 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, using *Q*-learning method. The state and the action are constructed using sensory information and motor commands, 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 Fig. 2. The goal state activation g is a normalized state value[1], and $g = 1$ when the situation is the goal state. When the module receives the behavior activation b 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, and 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. Fig. 3 shows a sketch of a state value function where a robot receives a positive reward one when it reaches 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.

3. Behavior Acquisition on Multi-Layered System (Takahashi & Asada 2000)

Fig. 4 Experimental instruments

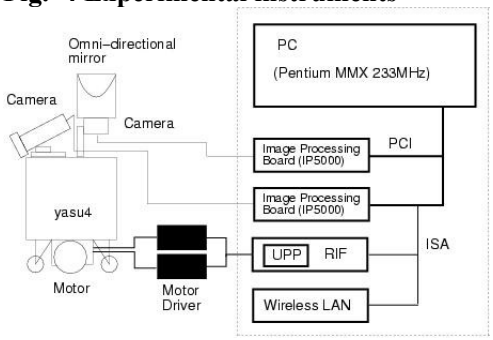

Fig. 5 Overview of the robot system

Fig. 4 shows a picture of a mobile robot that we designed and built, a ball, and a goal, and Fig. shows an overview of the robot system. It has two TV cameras: one has a wide-angle lens, and the other an 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. These parameters of the system are unknown to the robot, and it tries to estimate the mapping from the sensory

[1]The state value function estimates the sum of the discounted reward received over time when the robot takes the optimal policy, and is obtained by Q learning.

information to the appropriate motor commands by the method. The environment consists of the ball, the goal, and the mobile robot.

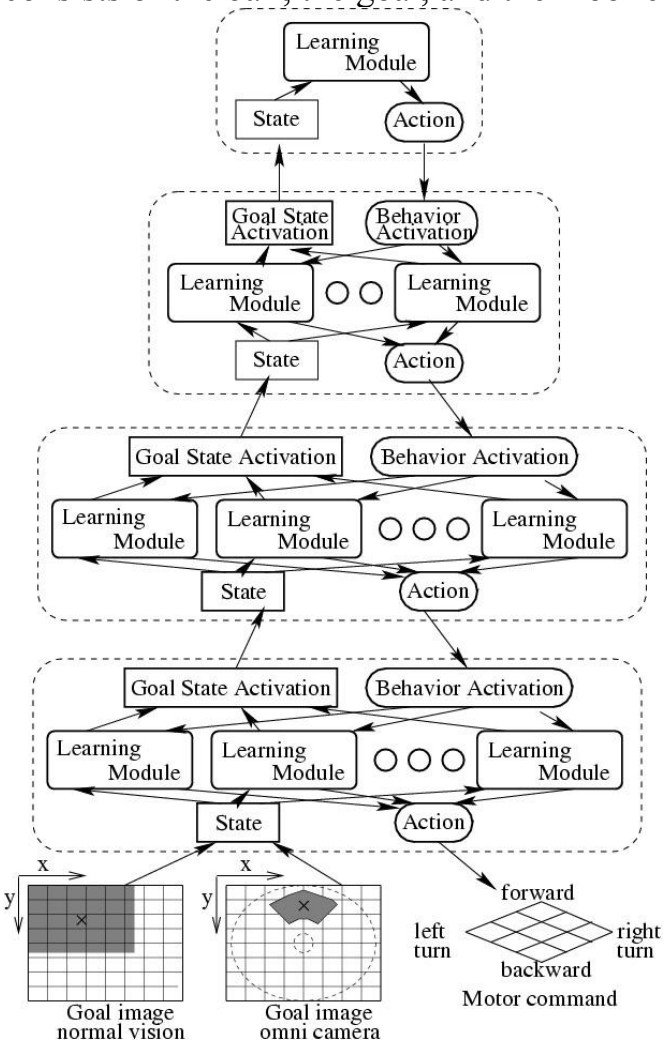

Fig. 6. A hierarchical architecture on a monolithic state space

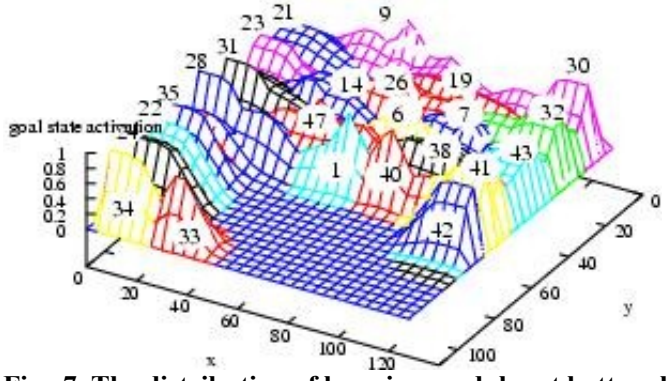

Fig. 7. The distribution of learning modules at bottom layer on the normal camera image

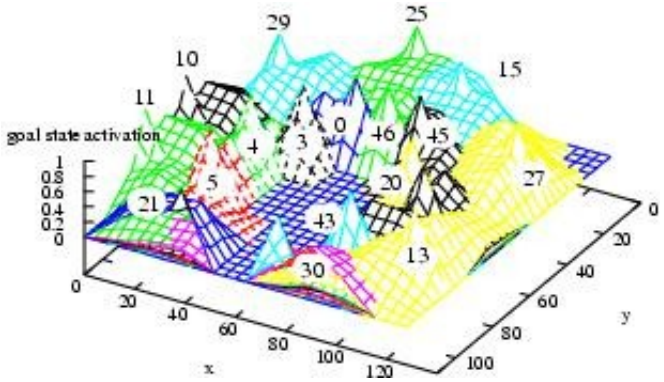

Fig. 8. The distribution of learning modules at bottom layer on the omni-directional camera image

In this experiment, the robot receives the information of only one goal, for the simplicity. The bottom of Fig. 6. show a sketch of the state and action spaces of the bottom layer in the multi-module learning system. The state space is constructed in terms of the centroids of goal images of the two cameras and is tessellated both into 9 by 9 grids each. The action space is constructed in terms of two torque values to be sent to two motors corresponding to two wheels and is tessellated into 3 by 3 grids. Consequently, the numbers of states and actions are 162(9 x 9 x 2) and 9(3 x 3), respectively. The state and action at the upper layer is 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. 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 is assigned at the top layer (Fig. 6). 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. Figs. 7 and 8 show the distributions of goal state activations of learning modules at the bottom layer in the state spaces of wide-angle camera image and omni-directional mirror image, respectively. The x and y axes indicate the centroid of goal region on the images. The numbers in the figures indicate the corresponding learning module numbers. The figures show that each learning module is automatically assigned on the state space uniformly.

Fig. 9 shows a rough sketch of the state transition and the commands to the lower layer on the multi-layer 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.

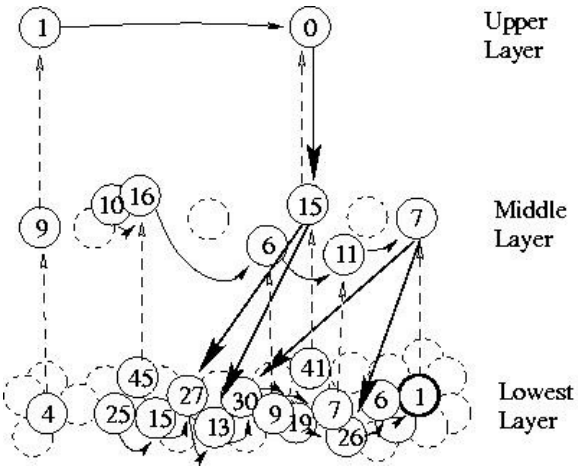

Fig. 9. A rough sketch of the state transition on the multi-layer learning system

4. State Space Decomposition and Integration (Takahashi & Asada, 2001)

The system mentioned in the previous section 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 suppose 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 becomes large. Here, 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 behaviors 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. There might be several layers at one level (see Fig. 10).

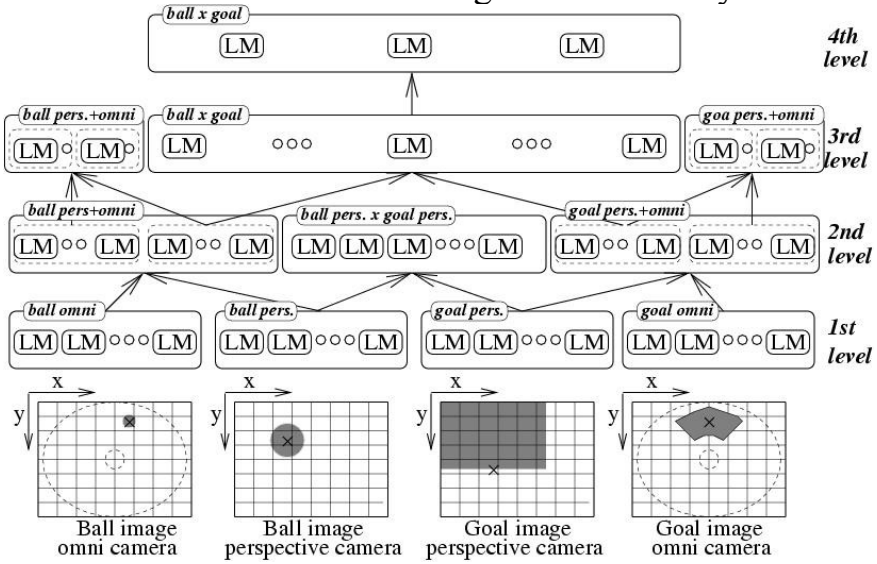

Fig. 10. A hierarchical structure of learning modules

When the higher layer constructs its state-action space based on situations and behaviors acquired by the modules of several lower layers, it should consider that the layers are independent from each other, or there is dependence between them. The layer might be basically independent from each other when the each layer's modules recognize different

objects and learn behaviors for them. For example, in the case of robot in the RoboCup field, one layer's modules could be the experts of ball handling and the other layer's modules the one of navigation on the field. In such a case, the state space is constructed as direct product of module's activations of lower layers. We call this way of state space construction “a multiplicative approach”.

On the other hand, there might be dependence between the layers when modules on both layers recognize the same object in the environment with different logical sensor outputs. For example, our robot recognizes an object with both perspective vision system and omni-directional one. In such a case, the system can recognize the situation complementary using plural layers' outputs even if one layer loses the object on its own state spaces. We call this way of state space construction “a complementary approach”.

Fig. 10 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 three learning layers in which one adopts the multiplicative approach and the two others adopt the complementary approach. The multiplicative approach of the “*ball pers. x 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 in which one adopts the multiplicative approach and the others adopt the complementary approach, again. At the levels higher than third layer, the learning layer is constructed as described in the previous section.

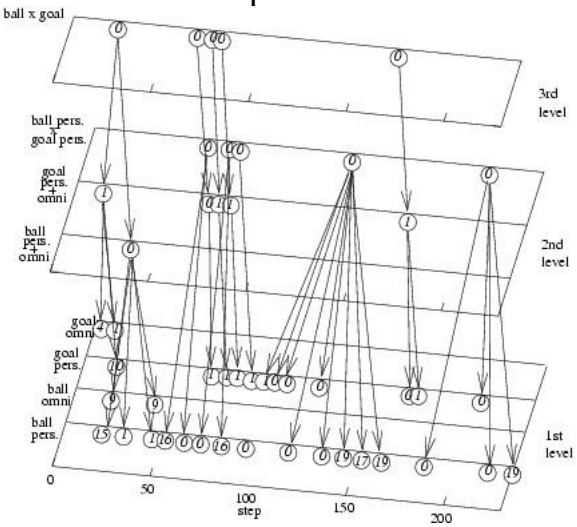

Fig. 11. A sequence of the behavior activation of learning modules and the commands to the lower layer modules

After the learning stage, we let our robot do a couple of tasks. One of them is shooting a ball into the goal using this multi-layer learning structure. 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. Fig. 11 shows the 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.

5. Behavior Segmentation and Coordination

Fig. 12 shows a picture of a soccer robot for middle size league of RoboCup we designed and built, recently. The driving mechanism is PWS, and it equips a pinball like kicking device in front of the body (see Fig. 13). These days, many robots have number of actuators such as navigation devices and object manipulators, and have a capability of execution of many kinds of tasks by coordinating these actuators. If one learning module has to manipulate all actuators simultaneously, the exploration space of action scales up exponentially with the number of the actuators, and it is impractical to apply a reinforcement learning system.

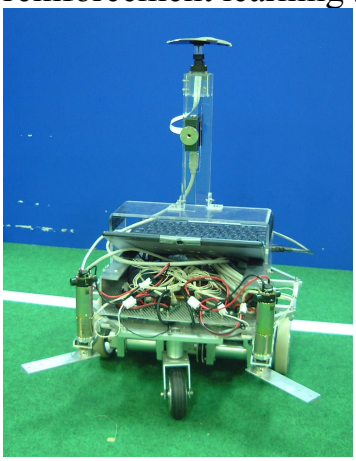

Fig. 12. Robot with kicking devices

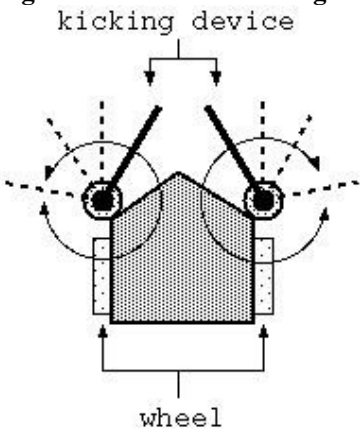

Fig. 13. Configuration of kicking device and wheels

Fortunately, a complicated behavior which needs many kinds of actuators might be often generally decomposed into some simple behaviors each of which needs small number of actuators. The basic idea of this decomposition is that we can classify them based on aspects of the actuators. For example, we may classify the actuators into navigation devices and manipulators, then the some of behaviors depend on the navigation devices

tightly, not on the manipulators, while the others depend on manipulators, not on the navigation. The action space based on only navigation devices seems to be enough for acquisition of the former behaviors, while the action space based on manipulator would be sufficient for the manipulation tasks. If we can assign learning modules to both action spaces and integrate them at higher layer, much smaller computational resources is needed and the learning time can be reduced significantly.

We have implemented two kind of hierarchical system to check the basic idea. Each system has been assigned a task. One is placing the ball in the center circle (task 1), and the other is shooting the ball into the goal (task2).

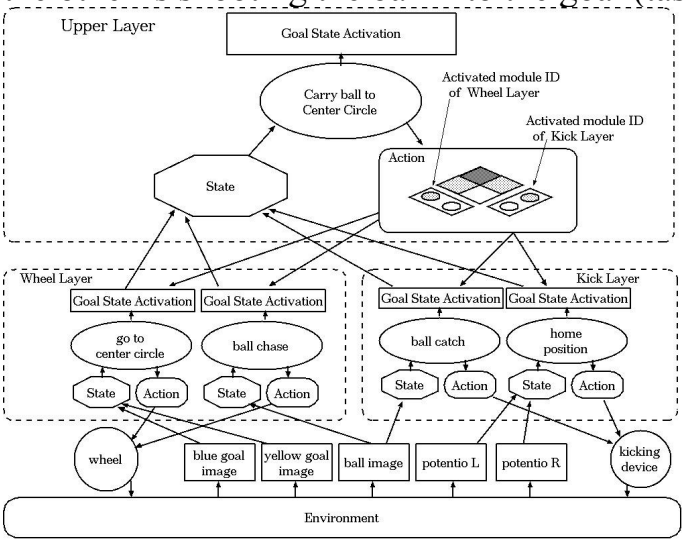

Fig. 14. Hierarchical learning system for task 1

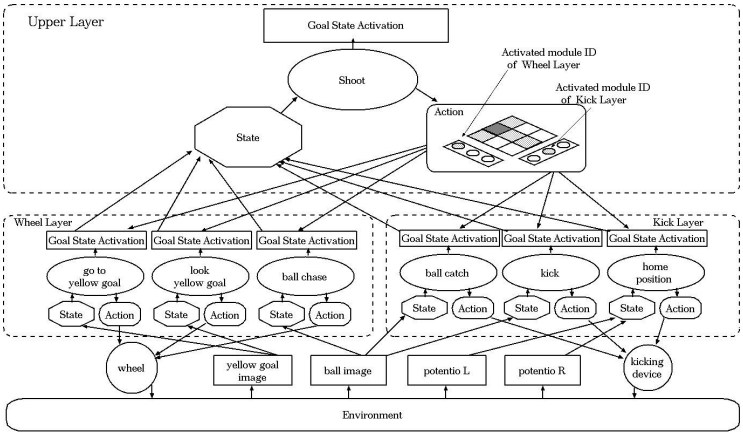

Fig. 15. Hierarchical learning system for task 2

We have prepared the following subtasks for the vehicle: ``*Chasing a ball*'', ``*Looking the goal in front of the body*'', ``*Reaching the center circle*'', and ``*Reaching the goal*''. We have also prepared the following subtasks for the kicking device: ``*Catching the ball*'', ``*Kicking the ball*'', and ``*Setting the kicking device to the home position*''. Then, the upper layer modules integrates these lower ones.

After the learner acquired low level behaviors, it puts new learning modules at higher layer as shown in Figs. 16 and 17, and learn two kinds of behaviors.

Fig. 16 shows the sequence of the goal state activations of lower modules and behavior commands to the lower ones. At the start of this behavior, the robot activates the module of setting home position behavior for the kicking device and ball chasing module for the vehicle at lower layer. The robot reaches the ball, then it activates the module of catching the ball for kicking device and the module of reaching the center circle. Then, it achieves the task of placing a ball to the center circle.

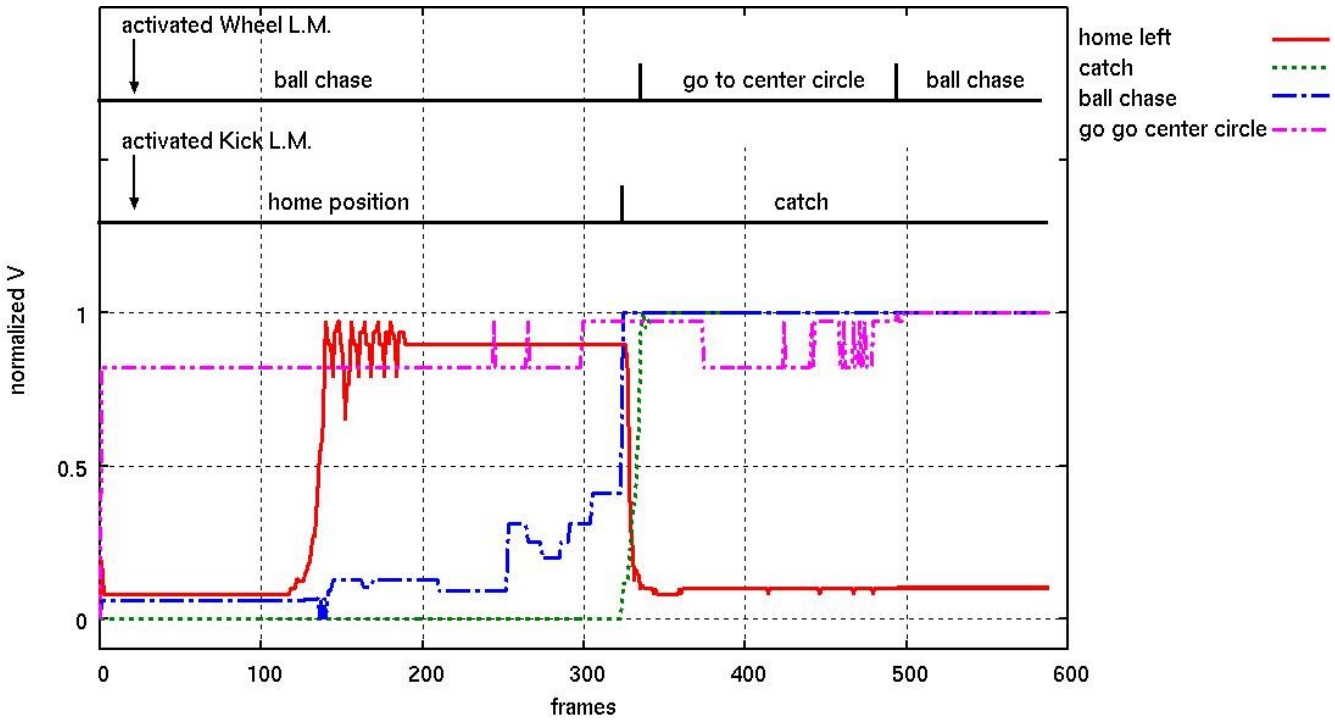

Fig. 16. A sequence of the goal state activations and behavior commands (Task 1)

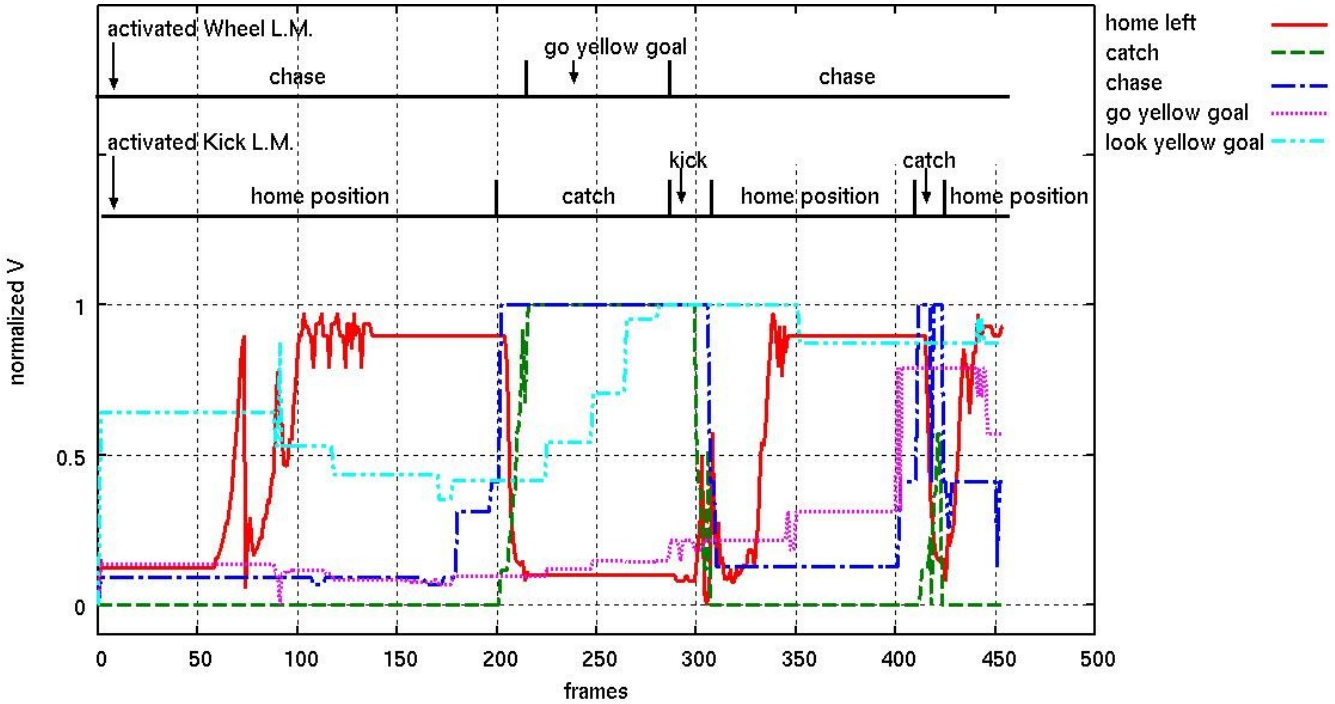

Fig. 17. A sequence of the goal state activation and behavior activation (Task 2)

Figs. 17 and 18 shows the sequence of the goal state activations of lower modules and behavior commands to the lower ones and the scene sequence of a real robot experiment while the robot shoots a ball into a goal. At the start of this behavior (Fig.18-1), the robot activates the module of setting home position behavior for the kicking device and ball chasing module for the vehicle at lower layer (Fig.18-2,3). The robot reaches the ball (Fig.18-4,5), then it activates the module of catching the ball for kicking device and the module of reaching the goal for the vehicle (Fig.18-6). When the robot captures the goal in front of the body and gets near to the goal (Fig.18-7), it activates the module of kicking the ball, then successfully shoots the ball into the goal (Fig.18-7).

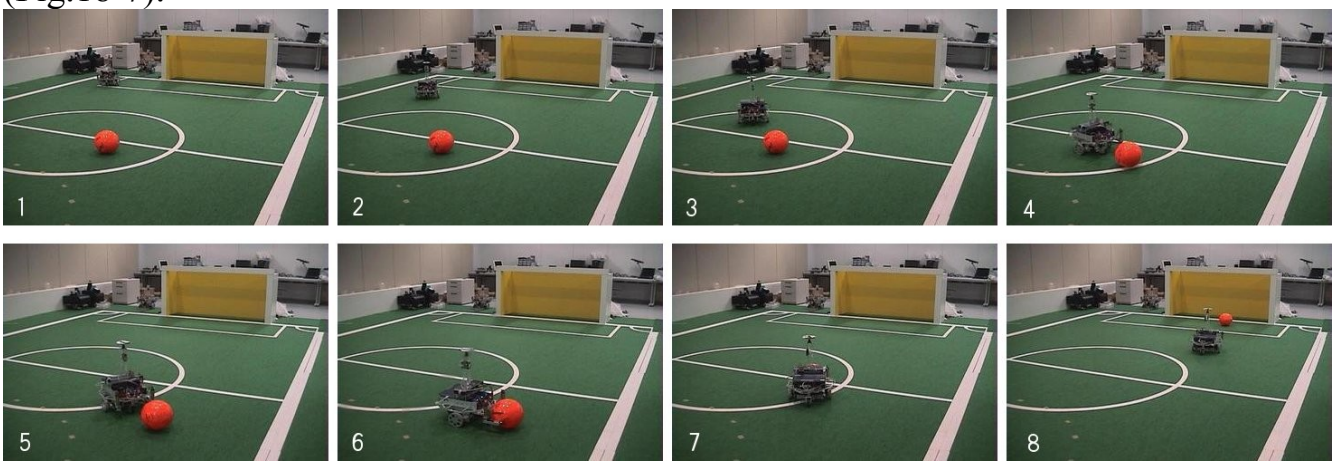

Fig. 18. ***A sequence of an acquired behavior (Shooting)***

6. Task Decomposition based on Self-interpretation of Instruction by Coach (Takahashi & Asada 2003)

When we develop a real robot which learns various behaviors in its life, it seems reasonable that a human instructs or shows some example behaviors to the robot in order to accelerate the learning before it starts to learn. We proposed a behavior acquisition method based on hierarchical multi-module leaning system with self-interpretation of coach instructions. The proposed method enables a robot to

1. decompose a long term task into a set of short term subtasks,
2. select sensory information needed to accomplish the current subtask,
3. acquire a basic behavior to each subtask,
4. and integrate the learned behaviors to a sequence of the behaviors to accomplish the given long term task.

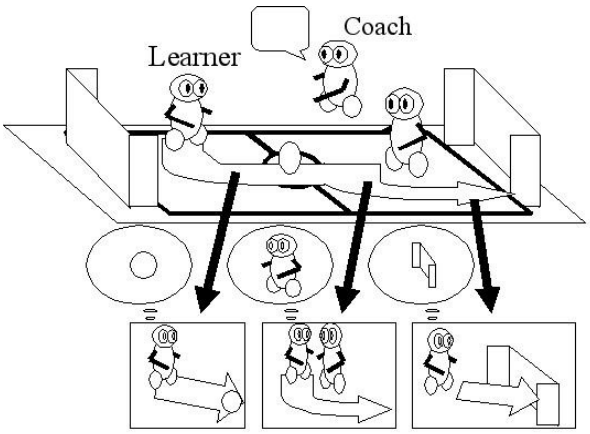

Fig. 19. Basic concept: A coach gives instructions to a learner. The learner follows the instruction and finds basic behaviors by itself.

Fig.19 shows a rough sketch of the basic idea. There are a learner, an opponent, and a coach in a simple soccer situation. The coach has *a priori* knowledge of tasks to be played by the learner. The learner does not have any knowledge on tasks but just follows the instructions. In Fig. 19, the coach shows a instruction of shooting a ball into a goal without collision to an opponent. After some instructions, the learner segments the whole task into a sequence of subtasks, acquires a behavior for each subtask, finds the purpose of the instructed task, and acquire a sequence of the behaviors to accomplish the task by itself. When the coach gives new instructions, the learner reuses the learning modules for familiar subtasks, generates new learning modules for unfamiliar subtasks at lower level. The system generates a new module for a sequence of behaviors of the whole instructed task at the upper level.

Fig. 20 shows a rough sketch of the idea of the task decomposition procedure. The top of the Fig. 20 shows a monolithic state space that consists of all state variables (x_1, x_2, ..., 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 Fig. 20). The system regards this area as the subgoal state of a subtask which 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 whole state space.

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 Fig. 3).

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.

The details are described in (Takahashi & Asada, 2003).

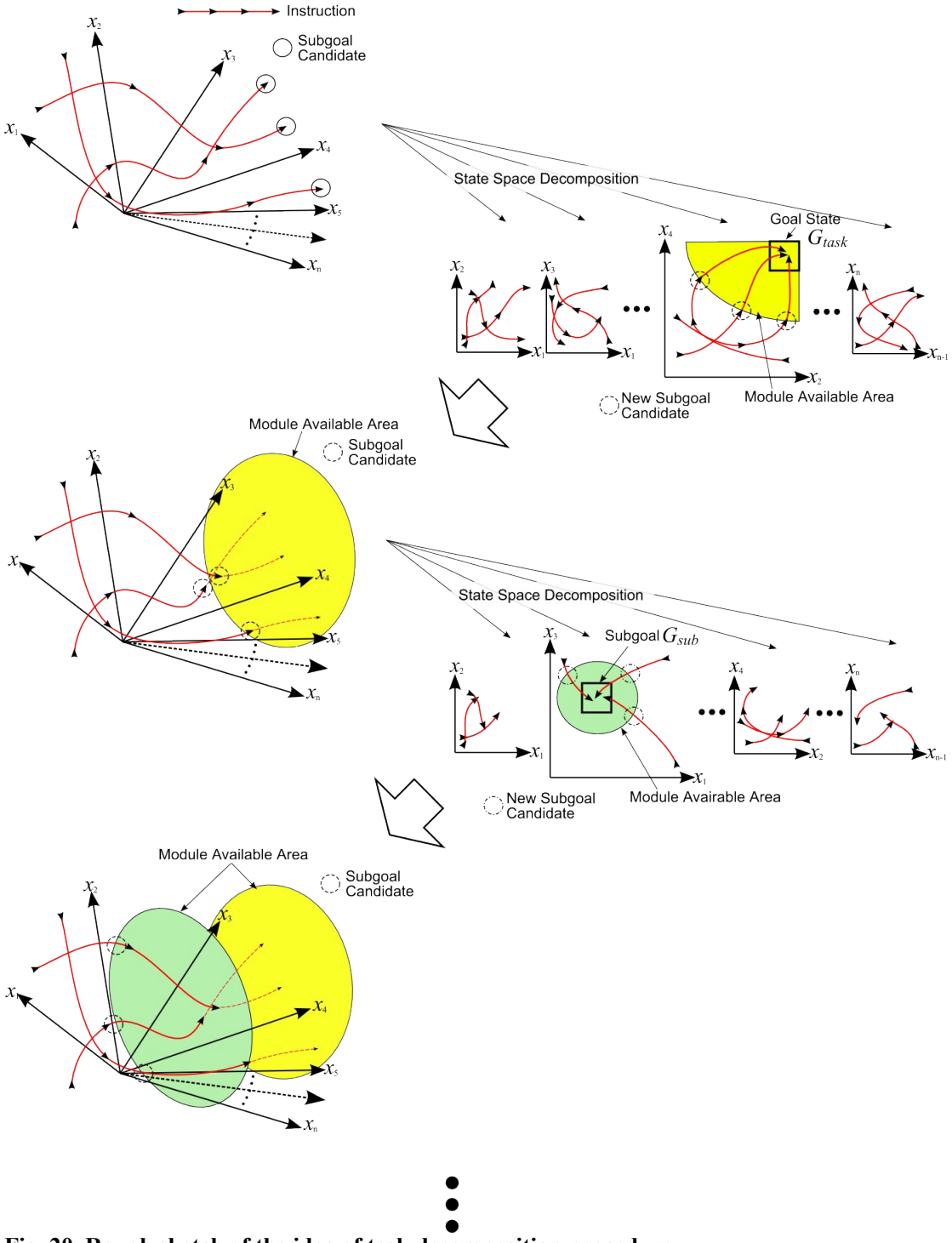

Fig. 20. Rough sketch of the idea of task decomposition procedure

Fig. 21 shows the mobile robot and a situation with which the learning agent can encounter. The robot has an omni-directional camera system. A simple color image

processing is applied to detect the ball area and an opponent one in the image in real-time (every 33ms).

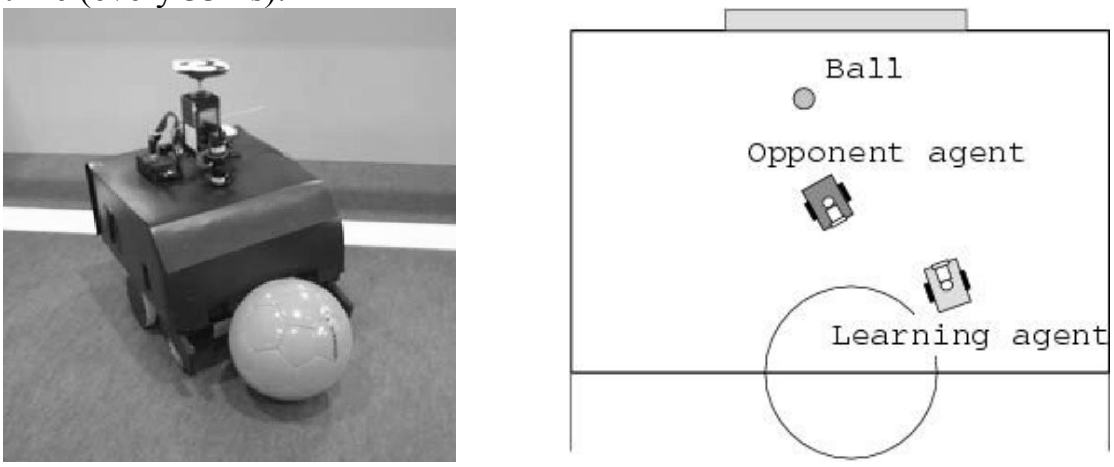

Fig. 21. A real robot and a ball (left), and a top view of the simulated environment (right)

The robot receives instructions for the tasks in the order as follows:

Task 1: chasing a ball

Task 2: shooting a ball into a goal without obstacles

Task 3: shooting a ball into a goal with an obstacle

Figs. 22, 23, and 24 show the ones of the example behaviors for task 1, 2, and 3, respectively. Figs. 25, 26, and 27 show the constructed systems after the learning of the tasks. First of all, the coach gives some instructions for the ball chasing task (task 1). The system produce one module which acquired the behavior of ball chasing (Fig.25). At the second stage, the coach gives some instructions for the shooting task (task 2). The learner produces another module which has a policy of going around the ball until the directions to the ball and the goal become same (Fig.26). At the last stage, the coach gives some instructions for the shooting task with obstacle avoidance (task 3). The learner produces another module which acquired the behavior of going to the intersection between the opponent and the goal avoiding the collision (Fig.27). Fig.28 shows a sequence of a acquired behavior of the real robot for task 3.

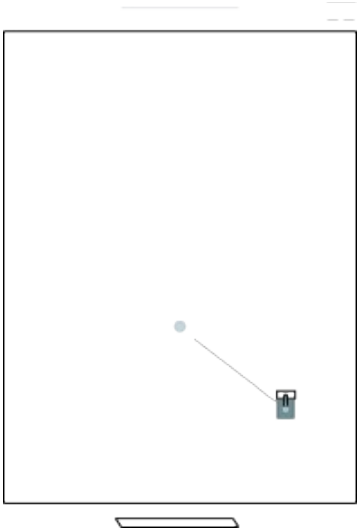

Fig. 22. One of the example behaviors for task 1

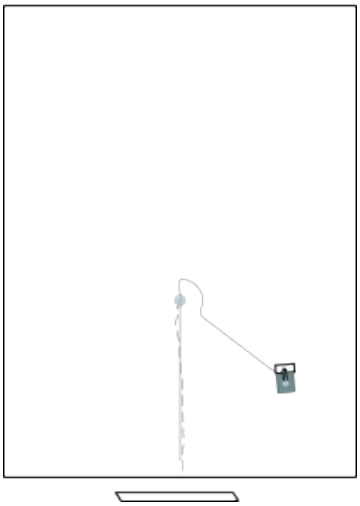

Fig. 23. One of the example behaviors for task 2

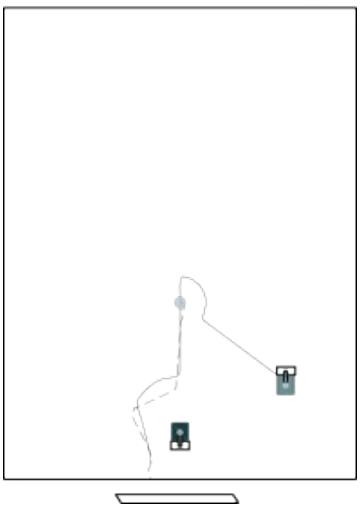

Fig. 24. One of the example behaviors for task 3

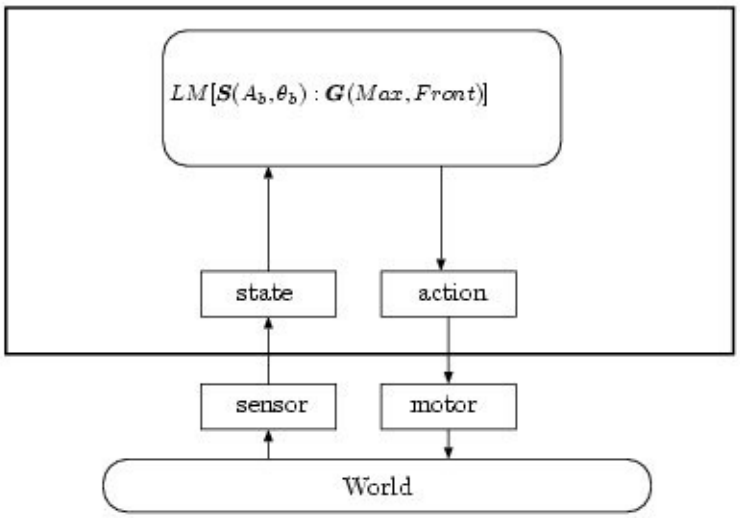

Fig. 25. Acquired learning module for task 1

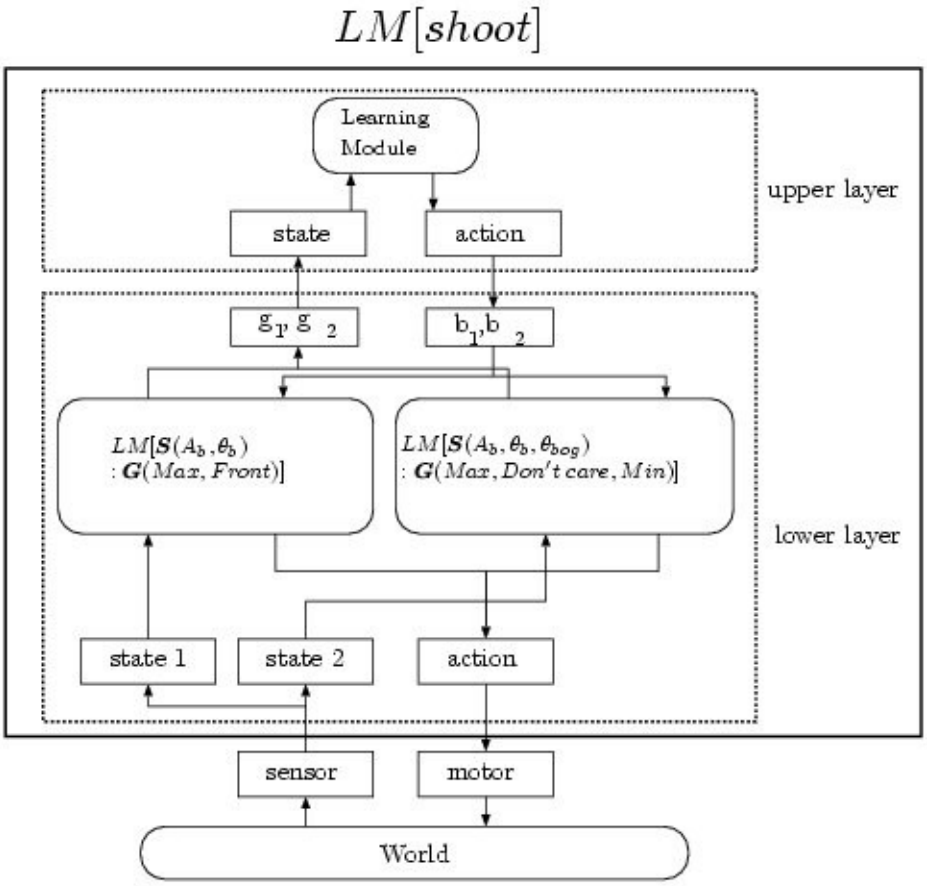

Fig. 26. Acquired hierarchical structure for task 2

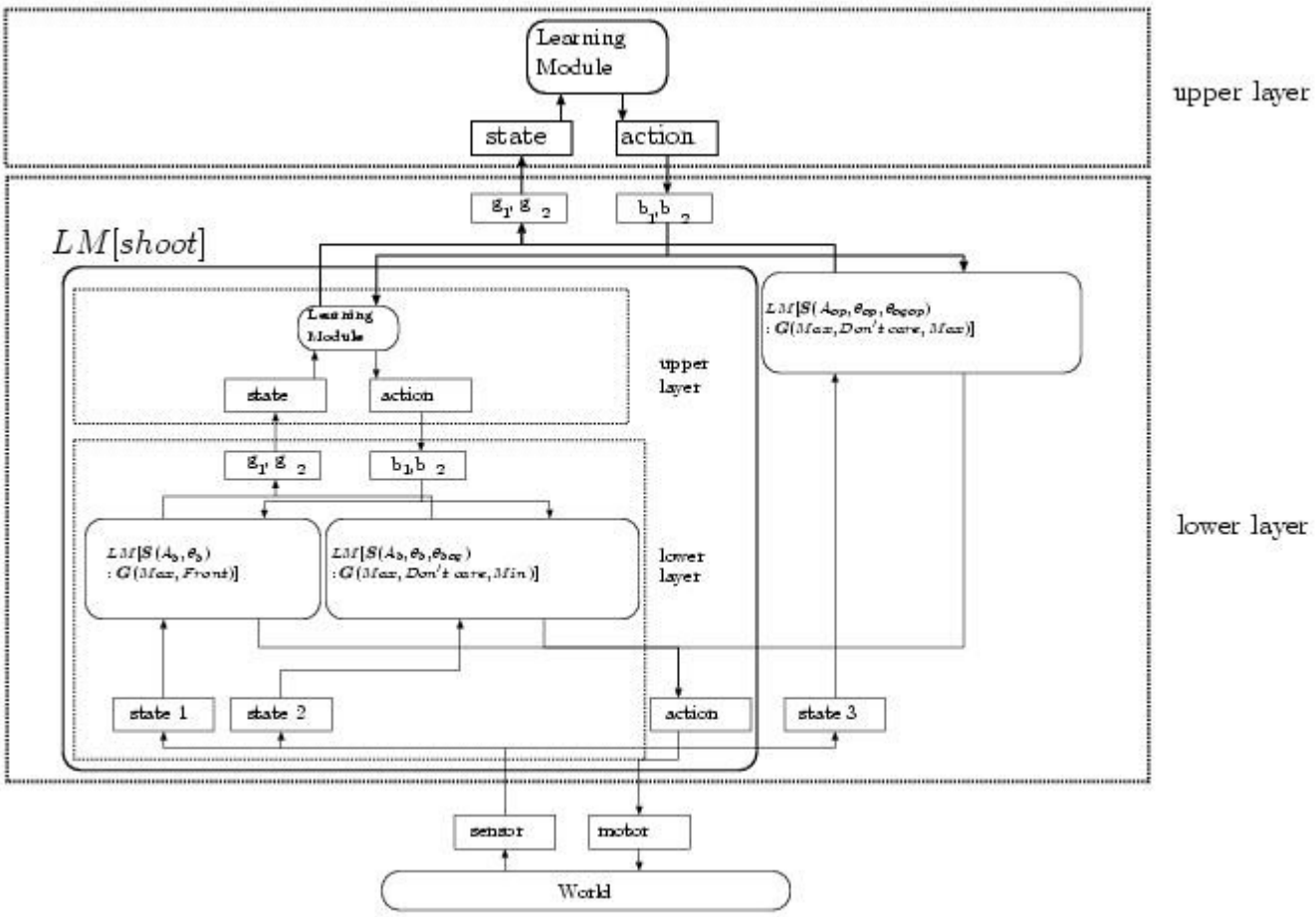

Fig. 27. Acquried heirarchical structure for task 3

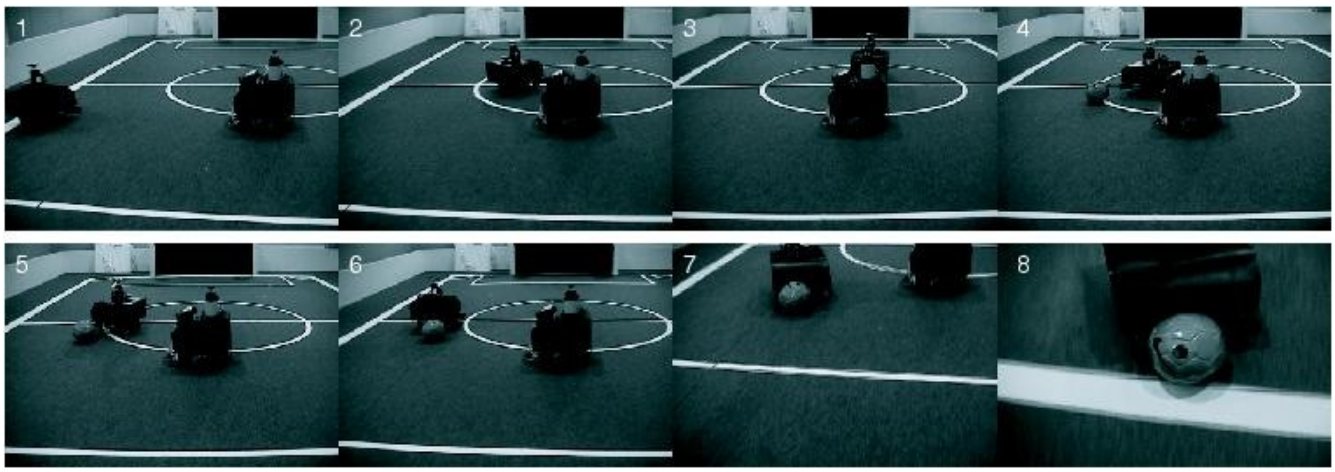

Fig. 28. A sequence of real robot behavior : shooting a ball into a goal with an obstacle (task3)

7. Discussion

We showed a series of approaches to the problem of decomposing the large state action space at the bottom level into several subspaces and merging those subspaces at the higher level. As future works, there are a number of issues to extend our current methods.

Interference between modules

One module behavior might have inference 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 the paper (Takahashi & Asada, 2000), the system distributes modules on the state space uniformly, however, it is not so efficient. In the paper (Takahashi & Asada, 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.

8. References

Asada, M.; Kitano, H.; Noda, I. & Veloso, M. (1999). RoboCup: Today and tomorrow – what we have learned. *Artificial Intelligence,* 193–214.

Connell, J. H. & Mahadevan, S. 1993. *ROBOT LEARNING*. Kluwer Academic Publishers. chapter "RAPID TASK LEARNING FOR REAL ROBOTS".

Digney, B. L., "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.

Digney, B. L., "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.

Hasegawa, Y. & Fukuda, T. (1999). Learning method for hierarchical behavior controller. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, 2799–2804.

Hasegawa, Y.; Tanahashi, H. & Fukuda, T. (2001). Behavior coordination of brachation robot based on behavior phase shift. In *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume CD-ROM, 526–531.

Hengst, B, "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.

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

Jacobs, R.; Jordan, M.; S, N. & Hinton, G. (1991). Adaptive mixture of local experts. *Neural Computation* 3:79–87.

Kleiner, A.; Dietl, M. & Nebel, B. (2002). Towards a life-long learning soccer agent. In Kaminka, G. A.; Lima, P. U.; and Rojas, R., eds., *The 2002 International RoboCup Symposium Pre-Proceedings*, CD–ROM.

Morimoto, J., and Doya, K. (1998). Hierarchical reinforcement learning of low-dimensional subgoals and highdimensional trajectories. In *The 5th International Conference on Neural Information Processing*, volume 2, 850–853.

Takahashi, Y. & Asada, M. (2000). Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 395–402.

Takahashi, Y. & Asada, M. (2001). Multi-controller fusion in multi-layered reinforcement learning. In *International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI2001)*, 7–12.

Takahashi, Y. & Asada, M. 2003. Multi-layered learning systems for vision-based behavior acquisition of a real mobile robot. In *Proceedings of SICE Annual Conference 2003 in Fukui*, volume CD-ROM, 2937–2942.

Takahashi, Y.; Hikita, K. & Asada, M. 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*, CD–ROM.

< end of manuscript>