Multi-method Learning and Assimilation

Shinya Takamuku [a] and Ronald C. Arkin [b]

[a] *Department of Adaptive Machine Systems*
Graduate School of Engineering, Osaka university
2-1 Yamadaoka, Suita, Osaka, 565-0871, Japan
takamuku@er.ams.eng.osaka-u.ac.jp

[b] *Mobile Robot Laboratory and GVU Center*
College of Computing, Georgia Institute of Technology
Atlanta, GA, USA
arkin@cc.gatech.edu

Abstract

Considering the wide range of possible behaviors to be acquired for domestic robots, applying a single learning method is clearly insufficient. In this paper, we propose a new strategy for behavior acquisition for domestic robots where the behaviors are acquired using multiple differing learning methods that are subsequently incorporated into a common behavior selection system, enabling them to be performed in appropriate situations. An example implementation of this strategy applied to the entertainment humanoid robot QRIO is introduced and the results are discussed.

Key words: humanoid robot, multi-method learning, social learning

1 Introduction

The field of domestic robots, intended to live with human users in everyday life, has been explored rapidly in recent years [1][2][3][4]. Many of these robots have shown impressive demonstrations such as playing music, dancing, and singing. Despite these successes, researchers have noticed a difficulty in creating robots that can entertain and attract the users for *extended* periods of time. For long-term human-robot interaction, choreographed motion replay behaviors often are generally inadequate. In order to realize a rich human-robot relationship, robots need to obtain various interactive behaviors which offer end-users the ability to explore and affect the robot's reactions. Although most behaviors implemented in existing commercial robots are hand-coded, the difficulty of designing behaviors will only grow as robots try to obtain increasingly complex behaviors. The complexity of the behavioral response can increase in complex environments, by introducing various sensors and actuators, and by dealing with people, who are resistant to accurate modeling. A humanoid robot is one extreme example where interactive behaviors are difficult to design. Users expect human-like natural behaviors which do not contradict the humanoid's appearance.

In order to obtain such behaviors, various learning methods such as reinforcement learning [5], imitation [6], and other learning mechanisms utilizing internal reward systems [7] have been proposed. These learning methods have shown the ability to produce complex behaviors including the application to humanoid robots. Any single learning method, however, is inadequate for acquiring the wide repertoire of behaviors required for domestic robots. The No-Free-Lunch theorem also indicates that no single learning method is capable of learning all behaviors at the highest efficiency [8]. This all argues for the design of an architecture capable of utilizing multiple learning strategies, incorporating their outputs into a single coherent behavioral end-product.

In this article, we develop a new strategy for acquiring various interactive behaviors for domestic robots, which is referred to as *Multi-method Learning and Assimilation*. Instead of searching for a single almighty learning methodology, we develop a system that can incorporate various behaviors acquired using different learning methods in a consistent natural manner. An example implementation of this strategy applied to the humanoid entertainment robot QRIO [9] is shown and the results discussed.

2 Multi-method learning and Assimilation

From the persuasive arguments provided by Wolpert and Macready [8], we infer that a single universal learning method that could learn any arbitrary behavior in the most effective way does not exist. It is often the case that learning methods are carefully chosen considering the characteristics of the specific task to be learned. In the case of domestic robots where a wide range of behaviors is expected to be acquired, utilizing a single learning method is clearly inefficient. Therefore, we adopt the approach of learning new behaviors by utilizing multiple learning methods and then incorporating these acquired behaviors into a common behavior selection architecture that enables them to run in a consistent and natural manner (Fig. 1). This incorporation process is called *assimilation* in the Piagetian sense [10][11].

The process of assimilation is realized by incorporating newly learned behaviors as a state-action mapping:

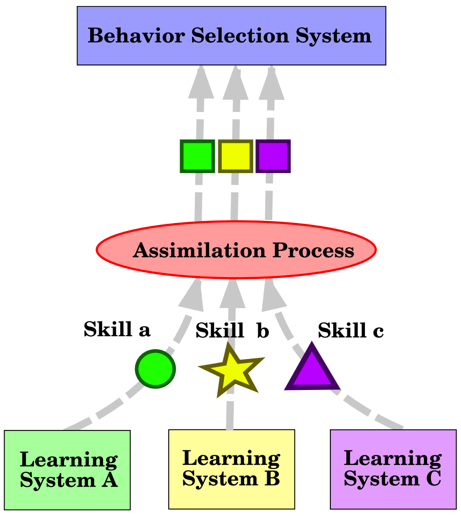


Fig. 1. Notional View of Multi-method Learning and Assimilation.

$$\boldsymbol{a} = f(\boldsymbol{s}) \quad \boldsymbol{a} : action, \;\; \boldsymbol{s} : state \tag{1}$$

and is associated with other necessary information involved in behavior selection such as the motor resources required for the performance of those behaviors. States are variables calculated from sensory data, which are expected to provide sufficient information for selecting appropriate actions for certain tasks. One difficulty for assimilation lies in the fact that learned behaviors involve different state space, action space, and representation of mappings depending on the specific learning process. State space and action space are usually carefully selected for each behavior to be learned, often reducing the search space for learning and sometimes establishing a simple relationship between them. Representation of state-action mappings also vary where, for example, some adopt tile codings and others adopt neural networks. One important role of the assimilation process is to convert these diverse specific state-action mappings into a more general mapping. The assimilated mapping adopts a common state-action mapping representation where state-action spaces are selected from a common set suitable for behaving in real world environments.

3 Assimilation utilizing the "Seed Schema"

After the conversion of the state-action mapping, the problem of behavior selection, i.e., when to run the new behavior, arises. In our approach, we adopt the use of a Situated Behavior Layer (hereafter referred to as SBL) [12], a behavior selection method developed for application to the humanoid robot QRIO, to address this problem. In this section we first explain the SBL architecture and subsequently introduce the concept of "Seed Schema" which enables the running of newly learned behaviors within the SBL framework.

SBL is a behavior selection architecture aimed to select and execute behaviors in a consistent and natural manner. The architecture autonomously selects behaviors based on a "homeostasis regulation rule" [13]. Namely, behaviors are selected to maintain predefined internal states within proper ranges. In addition to this, the architecture provides the following functions required for humanoid robot control.

(1) Integration of behaviors.
(2) Parallel execution of behaviors.
(3) Interruption and resumption of behavior execution.

In SBL, the robot's overall behavior consists of behavioral units called *schemas*. Each schema is able to execute a simple behavior independent of other schemas, and complex behaviors can be formed by simultaneously or sequentially executing multiple schemas toward a common goal. Combination of schemas are realized by having a parent schema that could execute multiple child schemas sharing common target information, and these parent schemas can also have parent schemas that form even more complex behaviors. Thus, the robot's overall behavior is realized as a tree-structured set of schemas, an example of which is shown in Fig. 2.

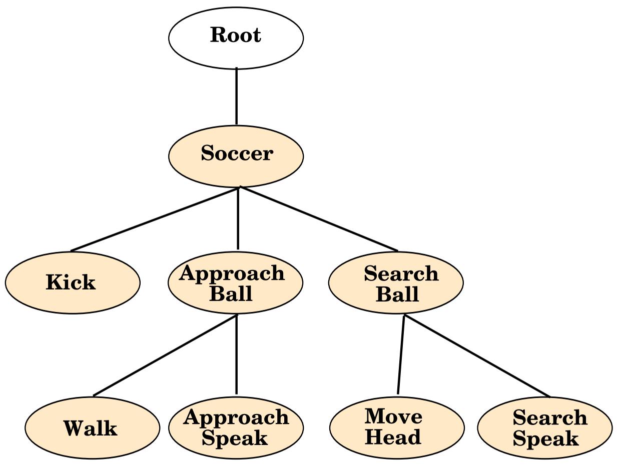


Fig. 2. Example of tree structure of schema.

In the behavior selection phase, each schema computes its suitability and outputs the motor resources required for its execution. The computed suitability of the schema, hereafter referred to as the activation level, is calculated based on the external stimuli, internal states of the robot, and intentional signals (Fig. 3). Intentional signals are used for integrating deliberative behavior control. Then, according to the activation level and motor resource information, parallel execution of schemas is realized by the following steps:

(1) Schema competition is conducted between schemas that are compatible with the available resources. The schema with the highest activation level is selected.
(2) The resources required for the selected schema is removed from the list of available resources.
(3) If there remain other schemas that could run with the remaining resources, return to step (1). If no other schema can run given the remaining resources, the schema selection phase is complete.

For behavior execution, each schema is represented by a state machine as shown in Fig. 4. Schemas include a *sleep* state in order to resume interrupted behaviors.

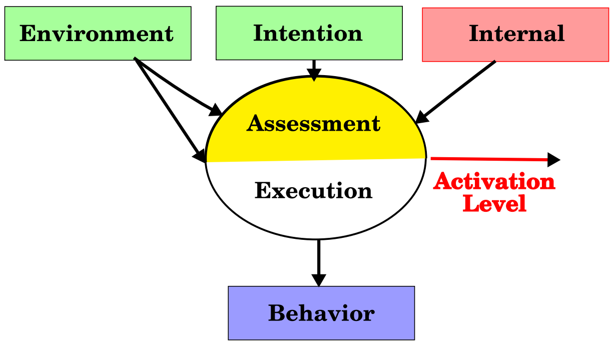


Fig. 3. Activation level calculation of schema.

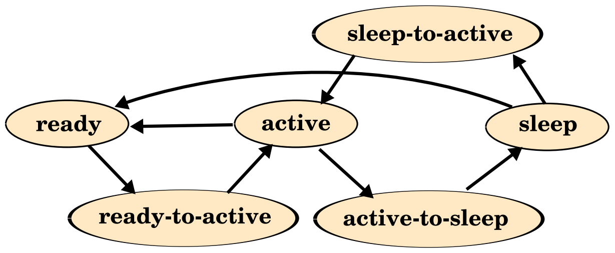


Fig. 4. State transition of schema.

3.2 Seed Schema

A *Seed Schema* is a template schema which loads and runs newly acquired behaviors as a state-action mapping. The data to be loaded into the Seed Schema are generated from the learned data from each learning system by the following process:

(1) A general state space and action space are selected for each of the learned behaviors which are to be assimilated.
(2) Considering the change resulting from this generalization of the state and action spaces, the previously learned state-action mapping must also be converted. This conversion is performed by generating state-action pairs

from the previous mapping, converting these state-action pairs into new state-action pairs, and then applying a regression function algorithm to learn the new mapping.

In order to run the learned behaviors under the control of the SBL, the Seed Schema also loads the additional information listed below:

(1) Motor resources required for executing the behavior.
(2) An Activation function $A_l(T)$ for the learned behavior.

(1) is usually determined from the action space that the learned behavior adopts, so the process of setting this value can be automated without difficulty. On the other hand, (2) is not always considered during the behavior learning phase. Nonetheless, there is some information that can be used to specify the activation function. One example is the range of the state value set S that enables the execution of the behavior. The activation level is set to 0 when the state value s is out of range.

$$if\ \boldsymbol{s} \notin \boldsymbol{S}\ then\ A_l(T) = 0 \tag{2}$$

The activation level when the state value is in the proper range is currently determined by the designer. The activation function, however, can be learned by executing the schema and exploring the effect they have on the external and internal state of the robot. One approach of automatically adjusting the activation function is proposed by Sawada et al [14], and a similar approach should also be applicable to Seed Schemas.

By adding newly learned behaviors to the SBL utilizing the Seed Schema, we can incrementally and developmentally enrich the interactive behavior of the robot.

3.3 Reorganization of tree structure

A difficult problem not yet discussed is where and how to position the Seed Schema of the newly learned behavior within the SBL tree. Suppose the robot learned a ball catching behavior. The robot may need to learn that the behavior can run under the basketball subtree, but should not be executed under the soccer subtree. The robot may also need to determine that this behavior is likely to run more smoothly when combined with a ball tracking behavior. In order to explore the appropriate place and combination of the Seed Schema, the ability to reconstruct the tree is essential. Our implemented approach for relocating the newly assimilated Seed Schema is described in Section 4.4.

4 Implementation

The assimilation process of newly learned behaviors to the SBL utilizing the Seed Schema was implemented on the humanoid robot QRIO (Fig. 5). The overall architecture is based on OPEN-R [15] and sketched in Fig. 6. Sensory information is collected in Short Term Memory (STM) which calculates the state information of the environment considering the identity of both objects and people. The SBL then calculates the activation level of each schema from the current values of STM and the Internal State Model (ISM) in order to select the appropriate schemas for execution. Finally, the selected schemas output their actions to the Resource Manager which distributes them to the appropriate controllers for execution.

In this section, we introduce the implemented design of the Seed Schema, and show two different examples of new behavior assimilation from two quite different learning systems. The first example involves the assimilation of a bell-ringing behavior (Fig. 13) that was acquired using MINDY (Model of INtelligence DYnamics) [16], a learning system utilizing internal reward. The second example demonstrates the assimilation of a walk-hand-in-hand behavior (Fig. 17) which was acquired through social learning using the actor-critic learning method [17]. The bell ringing behavior was selected as a simple example of a single agent behavior and the walk-hand-in-hand behavior was selected as a typical social behavior suitable for domestic robots like QRIO. Since the characteristics of the two behaviors are quite different, any single learning method is unlikely to learn both of them effectively.

Fig. 5. SDR-4XII (QRIO) - Sony Entertainment Humanoid Robot.

4.1 Design of Seed Schema

We now overview the implemented design for the Seed Schema concept for QRIO.

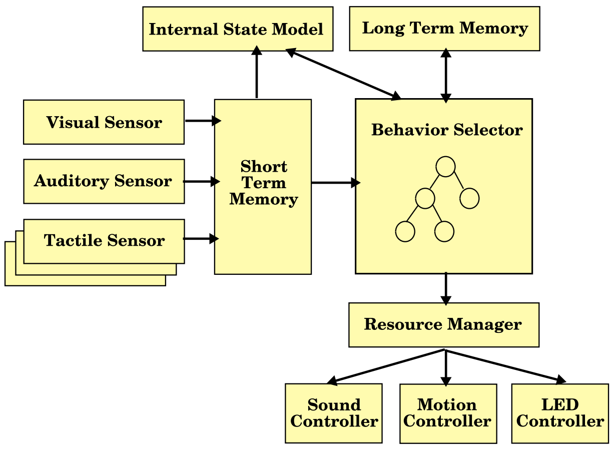


Fig. 6. Overview of Humanoid robot SDR-4XII (QRIO) architecture.

4.1.1 Information Contained in the Seed Schema

The Seed Schema loads the following information to run the actual acquired behavior:

(1) **Definition of state information**: This information defines the general state information to be used for selecting the action. In the current implementation, state information is selected from those available in the STM. The range of state values where the behavior can be executed is also included.

(2) **Definition of action parameter**: This information defines the action parameters used for the behavior execution. The range of action parameters is also included.

(3) **State-action mapping**: A Support Vector Regression (SVR) style representation [18] is adopted for the state-action mapping, aiming to output a smooth action sequence even when the learned behavior is initially given in a discrete form.

(4) **Required resources for behavior execution**: This information is usually determined from the action parameters.

(5) **Preparatory action (Initialization)**: Since some learned behaviors need to start from a special posture or state, the ability to run a preparatory action before performing the actual behavior is provided here. This function guarantees the independence of the Seed Schema and eases the design of the tree.

(6) **Activation Function**: The activation function of the schema is determined by setting the following parameters.
- Initial condition to run the schema (Optional).
- End condition to stop the schema (Optional).
- Activation level when the schema is running.

In the case of a one-shot behavior, the activation level of the schema returns to 0 when the behavior ends.

In the current implementation, only the external state information is considered when calculating the activation level.

4.1.2 Limitations of the current Seed Schema Implementation

Although the Seed Schema is conceptually designed to be general, the current implementation has some limitations on the behaviors that can be assimilated. The limiting preconditions for a new skill to be assimilated successfully are as follows:

(1) The information to specify the state for the new behavior must be obtainable from STM. The state information currently calculated in STM includes the position of the effectors, users, balls, landmarks, and other predefined color regions. Instantaneous information such as direction and length of sound, voice, and motions are also provided.
(2) The degrees of freedom for the action of the new behavior must already be provided by the controllers within the QRIO architecture. All the joint angles except the legs and the feet can be set to arbitrary values. The leg and foot movements were limited to predefined types of behavior, such as walking, standing up, and sitting down. The interface is open to add other types of behaviors, but the function to set arbitrary joint angles was not provided in order to prevent QRIO from falling down inadvertently. The walking action can take parameters such as the position of the next step, the speed of the step, and a change of direction. In addition to motion control, the color of the robot's LEDs can also take 2^8 level of RGB values, and arbitrary spoken sentences can be output from the robot's speaker.
(3) The new behavior must be expressed as a single state-action mapping. In the behavior execution phase, the same action should always be chosen when the same states are given. Behaviors which require randomness or memory of past states are examples that cannot be currently assimilated.

4.2 Assimilation of Bell-ringing behavior

4.2.1 MINDY

MINDY [16] is an open-ended learning system which autonomously develops a behavior controller through interaction with the environment. By adopting both extrinsic and intrinsic motivations, the system forms a hierarchical network of behavior control modules, each composed of a predictor and controller (Fig. 7). The generation of the hierarchical network is realised by the following processes. In the model, the agent seeks a causal correlation between observation variables (sensor variables, and internal states, etc.) and action variables (variables which the agent can directly change). If there exists a causal correlation, both a predictor and a controller for the observation variable to control can be learned. The predictor can be used for offline learning of the controller and facilitates the learning process. Once the controller is learned successfully for a particular observation variable, the variable can now be viewed as a new action variable, opening the possibility of forming more complex state variable control.

Real robot experiments were carried out using the MINDY architecture. The designer predefined the network structure in this experiment as shown in Fig. 8. The module to control the hand position was firstly learned, and was later utilized to form two other modules: one to ring a bell and the other to roll a ball. These latter modules were formed to control the bell's sound strength and the velocity of the object respectively.

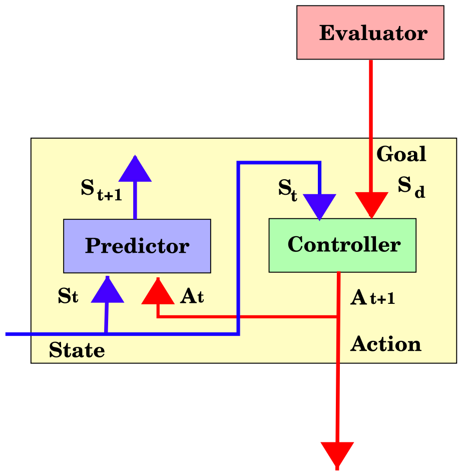


Fig. 7. Design of basic module in MINDY.

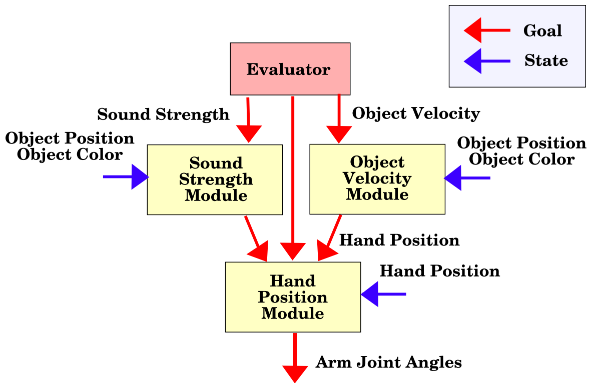


Fig. 8. Bell ringing and ball rolling behavior derived control network.

4.2.2 Assimilation with Seed Schema

In our research, we extracted the bell ringing behavior as a bell position - hand position mapping from the controller of the sound strength module by setting a ringing bell sound to the goal value. The actual component of the Seed Schema data appears as follows:

(1) **Definition of the state information**: The state space was remapped to the color region position and the color of the object. The state information range was set to the values where QRIO could reach the bell. This range was learned within the MINDY learning system.
(2) **Definition of action parameter**: A motion controller to move the hand effector to a specified position was used for action execution. The action space was the cartesian X, Y, Z position of the hand's goal position. The range of the action space was set to the area which QRIO could reach. This range was also specified within MINDY.
(3) **State-action mapping**: The state-action mapping was obtained by setting a constant bell sound to the goal value of the controller of the sound strength module. The SVR mapping was obtained by learning from a randomly selected set of 1000 teaching data points.
(4) **Required resources for behavior execution**: The robot's right arm and palm were set as the required resources for the execution of the bell ringing behavior.
(5) **Preparatory action (Initialization)**: The motion of raising QRIO's right hand was chosen as the preparation action.
(6) **Activation Function**: The activation function was set to be positive whenever the bell is in the proper range. No initial condition or end condition was set. The schema's activation level returns to 0 when the arm motion ends.

4.3 Assimilation of Walk hand-in-hand behavior

4.3.1 Social Learning with Actor-Critic Method

As pointed out recently by Lindblom and Ziemke [19], development of individual intelligence requires not only physical situatedness but also social situatedness. *Social Learning*, interactive learning where another individual plays an essential role, is also a necessity for humanoid robots. The requirements for a social learning method are as follows:

(1) Learning should reach a sufficient level of performance within a realistic time.
(2) The learning method should be tolerant to uncertainty produced by the other agent.

The Actor-Critic reinforcement learning method (Fig. 11) was selected as the learning algorithm. The method has proved to be effective for short duration learning and is also suitable for uncertainty management [20]. It is also well-suited for learning in a continuous action space which makes the method effective for learning humanoid robot behaviors [5].

The walk-hand-in-hand behavior is learned as a state-action mapping where the state values are the hand position in a horizontal plane (Fig. 9) obtained at high frequency from an OPEN-R object managing the joint angle information of the robot's arms and shoulder, and the action space is the distance and the direction (yaw angle) of the robot's next footstep (Fig. 10). The state space was quantized into 7×7 values, and action was expressed as normalized distributions whose mean values and standard deviations are modified to increase the probability of actions with positive TD-errors. A negative reward (punishment) $r = -1.0$ was provided when QRIO fell down, and a positive reward $r = 0.3$ was given when walking hand-in-hand was successful. In order to let the users change the arm angle of QRIO freely, an OPEN-R object to free the arm gain when the hand touch sensor is pushed was added to QRIO's control system. Height and speed of the next step was changed according to the chosen distance and direction of the next step to smooth the walking motion.

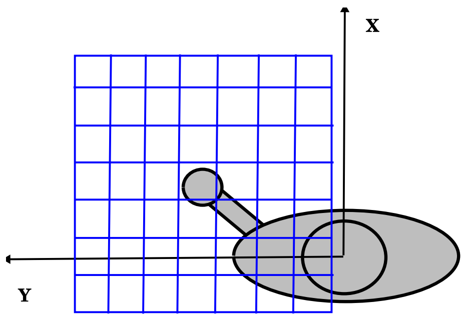


Fig. 9. State space of walk hand-in-hand behavior learning.

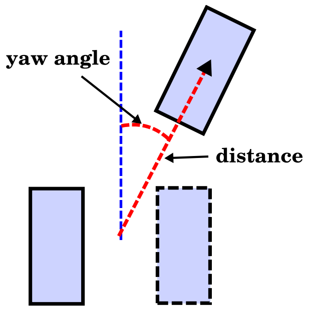


Fig. 10. Action space of walk hand-in-hand behavior learning.

4.3.2 Assimilation with Seed Schema

The walk hand-in-hand behavior was realized with SBL using the Seed Schema method by setting the schema parameters as follows:

(1) **Definition of the state information**: State information was mapped from the position of the left hand, which is obtained from the STM. The

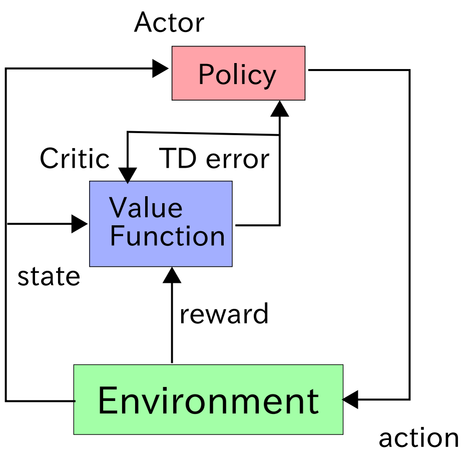


Fig. 11. Actor-Critic Method.

range of acceptable hand positions is set to the area within which QRIO can move its hands.

(2) **Definition of action parameters**: Walking motion using the parameters of distance, direction, and speed of step was used for action execution.
(3) **State-action mapping**: The SVR-style state-action mapping was obtained utilizing the AOSVR method from teaching data produced from each quantized state value.
(4) **Required resources for skill execution**: The robot's legs and left arm were set as the required resources for the execution of the walk-hand-in-hand behavior.
(5) **Preparatory action (Initialization)**: Standing up behavior was set as a preparation behavior. The motion of standing up was set as the preparatory action.
(6) **Activation Function**: The schema was set to be active when QRIO's hand was grabbed by the user. This was realized by setting the initial condition as an event when the user grabs QRIO's hand, and the end condition as an event when the user presses QRIO's head button.

The OPEN-R object to release the arm gain was also utilized with the Seed Schema.

4.4 Reorganization Interface

An interface to dynamically reconstruct the SBL tree was implemented to explore the correct combination and positioning of newly acquired schemas. The implemented interface enables the following management of the SBL tree in real-time without it being restarted:

(1) Generation of the predefined class schema including the Seed Schema.
(2) Connection/disconnection of schema connections within the SBL tree.

Note that the reconstruction of the tree is realized by connection/disconnection of the schema connection instead of addition/removal of schemas. This enables adding/moving/removing behaviors as a unit of a subtree. Once these modifications are made, the change to the tree structure can be saved and loaded later. The implemented SBL reconstruction interface (the *SBL Editor*) is shown with the Viewer (the *SBL Viewer*) that depicts the current SBL structure (Fig. 12).

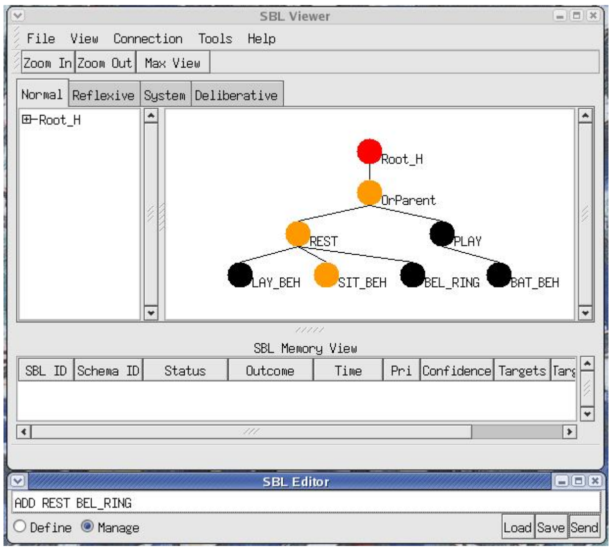

SBL Viewer
File View Connection Tools Help
Zoom In Zoom Out Max View
Normal Reflexive System Deliberative
Root_H
Root_H
OrParent
REST
PLAY
LAY_BEH
SIT_BEH
BEL_RING
BAT_BEH
SBL Memory View
SBL ID Schema ID Status Outcome Time Pri Confidence Targets Targ
SBL Editor
Define Manage
Load Save Send
ADD REST BEL_RING

Fig. 12. SBL Editor - SBL reconstruction interface with SBL Viewer.

5 Results and Discussions

The results of the assimilation process for the two separately learned skills are now described and discussed. The possibility of assimilating other learned behaviors, an issue regarding the generality of the Seed Schema method, is discussed in the last part of this section.

5.1 Assimilation of bell ringing behavior

The MINDY architecture enabled QRIO to acquire the bell ringing behavior within approximately 10 trials of direct teaching by the user. Quantitative results on this learning process are described by Sabe et al. [16] and will not be discussed further in this paper.

This newly acquired bell ringing behavior was successfully assimilated into the SBL utilizing the Seed Schema approach. We observed the Seed Schema of the bell ringing behavior turning active only when the bell was put inside the reach of QRIO, and once the bell was removed, another appropriate schemas performed their actions without any interference.

Although the assimilated behavior worked fine in general, some problems were also noticed. First, QRIO often failed to hit the bell properly when the bell was moved quickly. This was due to the delay of state information calculated in the STM, which was not used in the standalone MINDY learning. The delay of state information tends to increase in the assimilated process since the general state information requires more time for calculation compared with specific state information used for the behavior learning phase. In the current imprementation, the general state information provided by the STM is calculated considering the identity of each object, and is able to handle objects out of the field of view by considering the actions the robot undertakes. Robust state information is obtained in exchange for a delay in state information update. Robust state information is needed to execute multiple schemas simultaneously, and is useful in environments with multiple objects of interest. This robustness, however, is not always needed during the separate standalone behavior learning phase.

Another issue that we noticed during this experiment involves integrating supportive behaviors for learned skills. In the case of the bell ringing behavior, running the color region tracking behavior simultaneously would increase the accuracy of the bell ringing. It is often the case that such supportive behaviors are implemented to run during the learning phase, in order to make the learning task easier. We also note that some of those supportive behaviors such as tracking are not easy to learn by themselves. In order to successfully assimilate new skills learned with supporting behaviors, the assimilation process should also provide these supportive behaviors to run in tandem with the newly learned skill assimilated into SBL. This can be realized by an addition to the Seed Schema that enables the running of supportive behaviors simultaneously.

5.2 Assimilation of walk hand-in-hand behavior

First, the results of the learning process itself for this behavior are discussed, followed by the results of the assimilation process into the SBL. The evolution of the number of successful steps QRIO was able to take until it fell down during the learning process is shown in Fig. 14. The figure's horizontal axis shows the number of learning trials from the run's beginning to where each trial ends when QRIO falls. After about 50 trials (approximately one hour), QRIO was able to smoothly walk hand-in-hand with the user. The learning proceeded as follows. First, the critic converged after 10 trials. The state value function after 10 trials is shown in Fig. 15. Since it takes time for QRIO to change its

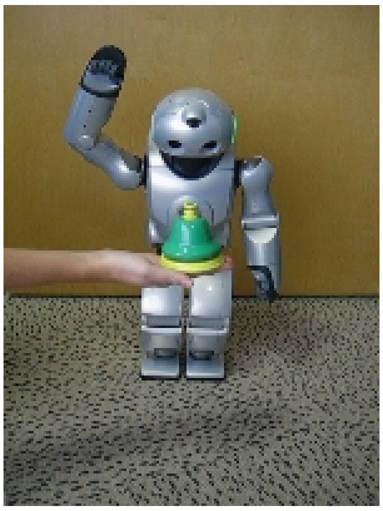


Fig. 13. Bell ringing behavior.

walking direction, it is reasonable that the stable state for QRIO is to have the hand in front. Transition of the actor's greedy action output is shown in Fig. 16 with the associated standard deviation of the output. The vectors show the distance and direction of the next step for each state. The middle vector is the greedy action output, and the ones on the sides are drawn as mean value ± standard deviation vectors. QRIO is learning the actions necessary to keep the grabbed hand at the front left part of its body, and the decreasing standard deviation indicates that the learning is converging. Although the user changed his walking direction on his own during the learning process, the learning process was stable and acquired the behavior within a realistic learning time. The actor-critic method's ability to learn social behaviors was affirmed.

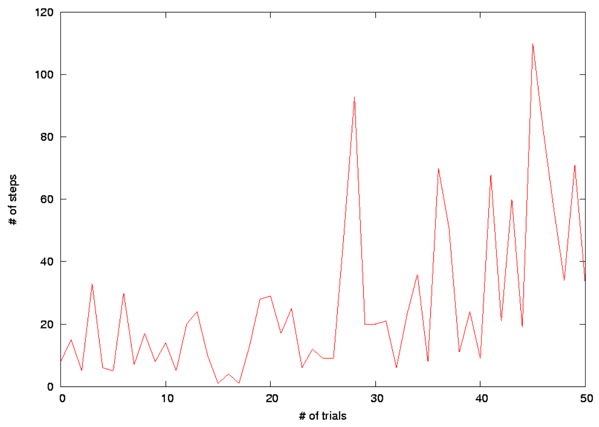


Fig. 14. The number of successful steps taken by QRIO until falling down for each learning trial.

The assimilation of the walk-hand-in-hand behavior into the SBL was also successful. When the hand was grabbed, QRIO performed the walk-hand-in-hand behavior and when it was lifted and its head button was pressed for a

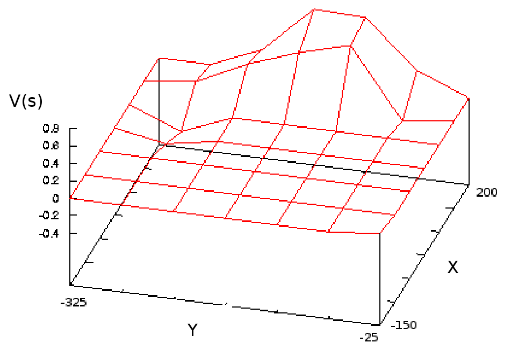
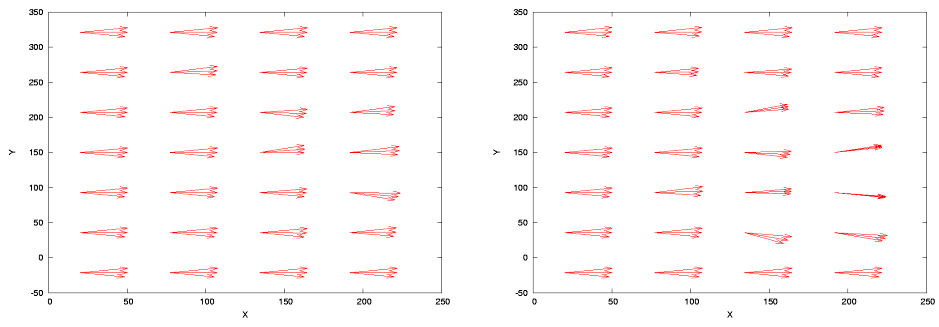


Fig. 15. State values acquired after 10 trials.



(a) After 10 trials

(b) After 50 trials

Fig. 16. Transition of actor parameters (mean value and standard deviation for each action parameter) for each state. The vector shows the distance and direction of the next step and the direction that QRIO is facing is to the right. Three vectors are shown for each state where the middle vector is the greedy output, and the ones on the sides are calculated as mean value ± standard deviation. A state with closed vectors has converged action parameters.

short time, the robot stopped the walk-hand-in-hand behavior and returned to its usual behaviors.

Two issues were noticed during this assimilation experiment. The first involves the delays in sensory information which we discussed previously. The delay of sensor information seemed more critical in a feedback control behavior as in this case. We observed some oscillation in the action due to this delay, but the assimilated behavior was still performed at a sufficient level. Another issue is related to change in the environment. When we try to let QRIO walk on different floor conditions, the assimilated walk-hand-in-hand behavior is likely to fail. This phenomenon reminds us that learning is not only for acquiring behaviors, but also for modifying the behavior parameters to adapt to the change in body structure, motors, sensors, or the environment. Our approach of freezing the learning process at a fixed skill level and then assimilating the learned behavior can be criticized in that it loses the ability of further modifying the behavior parameters. The learning methods, however, required for acquiring the behaviors and those used for adapting the assimilated behavior's parameters are not necessarily the same. Dividing the two processes and adopting different learning methods for each may turn out to be an effective approach.

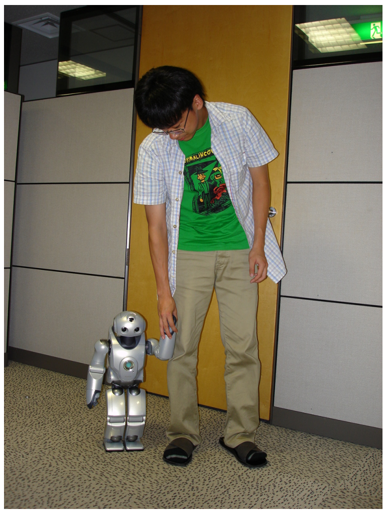


Fig. 17. Walk hand-in-hand behavior.

5.3 Assimilation of other behaviors

Theoretical discussions on the possiblity of assimilating other behaviors were described earlier in Section 4.1.2. If the learned behaviors can be expressed as mappings of the states and actions provided in the QRIO architecture, the behaviors can be assimilated independently of how they are acquired. For behaviors, however, that are not best represented in this manner, difficulties may arise. For example, experimental results indicate that some learned behaviors might not assimilate properly if they are required to use other than state-action representations. Dynamic behaviors that require high frequency state update information might fail to be assimilated properly. One way to avoid this problem is to assimilate the behavior as a sensorimotor mapping instead of a state-action mapping. Behaviors that depend heavily on change in the environment are also not well-suited for the current assimilation methods. Applying parameter tuning methods such as gradient descent methods should help in addressing this problem.

6 Related Work

There is a large body of research conducted on the general topic of machine learning for humanoid robots, with a few attacking the problem in a manner similar to this research. For example, Fitzpatrick and Metta [21] studied learning object affordances as causal relations, while Ogata et al. [22] focussed on learning a walking hand-in-hand behavior. However, most researches involved with machine learning of humanoids thus far have focused on a single task category, adopting only a single learning method. The idea of this research in this article is to utilize multiple learning methods to acquire the wide repertoire of behaviors required for humanoid robots.

The idea of utilizing multiple different learning methods for robot control is not commonplace, but there is significant research conducted under the term of *multi-strategy learning* [23]. Gordon and Subramanian [24] developed a multistrategy system that combines two learning methods: operationalization of high-level advice provided by a human with incremental, refinement using a genetic algorithm. The approach provided two advantages: an initial boost of learning utilizing high level knowledge, and robustness and improvement with the genetic algorithm. Ram and Santamaria [25] examined the effect of combining case-based reasoning and reinforcement learning for an autonomous robotic navigation task. The system performed online adaptation resulting in improvement of reactive control, as well as using an online case based learning algorithm that resulted in a library of cases that capture environmental regularities necessary for online adaptation. The difference between these research examples when compared to our approach is that they consider combining learning methods that work at different levels, while in our research, we strive to form a general interface for incorporating skills acquired under different learning methods but working at the same behavioral level.

Finally, the work by Caruana [26] is suggestive. In our approach, the learning processes for the different behaviors are fully independent from each other. However, Caruana's work indicates that some tasks are correlated and thus may be learned more effectively by using complementary learning methods.

7 Conclusions and Future work

A new strategy for behavior acquisition in domestic robots, *Multiple-method learning and Assimilation* was described and an example implementation of the approach to the entertainment robot QRIO was introduced. We showed that skills acquired under different learning methods can be incorporated into a common behavior selection architecture, enabling them to be performed at appropriate situations.

In the current implementation, most of the assimilation process is carried out manually by the designer. One important issue remaining as future work is the automation of this process.

Acknowledgment

The authors would like to thank Toshitada Doi for giving us the opportunity to study in Sony Intelligence Dynamics Laboratory, Masahiro Fujita and Hideki Shimomura for the considerate support and discussions. The authors would also like to thank Kuniaki Noda, Yukiko Hoshino, Kenichi Hidai, and Kazumi Aoyama for the extensive support on the implementations and their valuable comments.

References

[1] M. Fujita, H. Kitano, Development of a quadruped robot for robot entertainment, Autonomous Robots 5 (1998) 7–18.

[2] S. Ohnaka, T. Ando, T. Iwasawa, The introduction of the personal robot papero, IPSJ SIG Notes 37 (7) (2001) 37–42.

[3] H. Ishiguro, T. Ono, M. Imai, T. Maeda, T. Kanda, R. Nakatsu, Robovie: an interactive humanoid robot, Industrial Robot 28 (6) (2001) 498–503.

[4] T. Shibata, T. Mitsui, K. Wada, K. Tanie, Subjective evaluation of seal robot: Paro -tabulation and analysis of questionnaire results, Journal of Robotics and Mechatronics 14 (1) (2002) 13–19.

[5] J. Peters, S. Vijayakumar, S. Schaal, Reinforcement learning for humanoid robotics, in: Third IEEE-RAS International Conference on Humanoid Robots, 2003.

[6] S. Schaal, Is imitation learning the route to humanoid robots?, Trends in Cognitive Sciences 3 (6) (2002) 233–242.

[7] F. Kaplan, P.-Y. Oudeyer, Maximizing learning progress: an internal reward system for development, in: F. Iida, R. Pfeifer, L. Steels, Y. Kuniyoshi (Eds.), Embodied Artificial Intelligence, Lecture Notes in Computer Science 3139, Springer-Verlag, 2004, pp. 259–270.

[8] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1997) 67–82.

[9] M. Fujita, Y. Kuroki, T. Ishida, T. Doi, Autonomous behavior rcontrol architecture of entertainment humanoid robot sdr-4x, in: Proceedings of International Conference on Intelligent Robots and Systems, 2003, pp. 960–967.

[10] J. H. Flavell, The Developmental Psychology of Jean Piaget, Van Nostrand Reinhold, 1963.

[11] R. C. Arkin, Behavior-Based Robotics, MIT Press, 1998.

[12] Y. Hoshino, T. Takagi, U. D. Profio, M. Fujita, Behavior description and control using behavior module for personal robot, in: Proceedings of IEEE International Conference on Robotics and Automation, Vol. 4, 2004, pp. 4165–4171.

[13] R. C. Arkin, M. Fujita, T. Takagi, R. Hasegawa, Ethological modeling and architecture for an entertainment robot, in: Proceedings of IEEE International Conference on Robotics and Automation, Vol. 1, 2001, pp. 453–458.

[14] T. Sawada, T. Takagi, Y. Hoshino, M. Fujita, Learning behavior selection through interaction based on emotionally grounded symbol concept, in: Proceedings of IEEE International Conference on Humanoid Robots, 2004.

[15] M. Fujita, K. Kageyama, An open architecture for robot entertainment (1997) 435–442.

[16] K. Sabe, K. Hidai, K. Kawamoto, H. Suzuki, A proposal of intelligence model, mindy for open-ended learning system, in: Proceedings of IEEE International Conference on Humanoid Robots Workshop on Intelligence Dynamics, 2005.

[17] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.

[18] A. Smola, B. Schoelkopf, A tutorial on support vector regression, Tech. rep., NeuroCOLT Technical Report NC-TR-98-030, Royal Hlloway College (1998).

[19] J. Lindblom, T. Ziemke, Social situatedness: Vygotskiy and beyond, in: Proceedings of Second International Workshop on Epigenetics Robotics, 2002, pp. 71–78.

[20] R. Tedrake, T. W. Zhang, H. S. Seung, Learning to walk in 20 minutes, in: In Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems, 2005.

[21] P. Fitzpatrick, G. Metta, Early integration of vision and manipulation, Adaptive Behavior 11 (2) (2003) 109–128.

[22] T. Ogata, N. Masago, S. Sugano, J. Tani, Interactive learning in human-robot collaboration, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, 2003, pp. 162–167.

[23] R. Michalski, G. Tecuci, Machine Learning: A Multistrategy Approach Volume IV, Elsevier, 1993.

[24] D. Gordon, D. Subramanian, A multistrategy learning scheme for agent knowledge acquisition, Informatica 17 (4) (1993) 331–346.

[25] A. Ram, J. Carlos, Multistrategy learning in a reactive control systems for autonomous robotic navigation, Informatica 17 (4) (1993) 347–369.

[26] R. Caruana, Multitask learning: A knowledge-based source of inductive bias, Machine Learning 28 (1) (1997) 41–75.