Synthesis, Learning and Abstraction of Skills through Parameterized Smooth Map from Sensors to Behaviors

Y. Nakamura† ‡ T. Yamazaki† N. Mizushima†
Department of Mechano-Informatics
† University of Tokyo ‡ CREST, JST
7-3-1 Hongo, Bunkyo-ku, Tokyo 113, JAPAN
email: nakamura@ynl.t.u-tokyo.ac.jp

Abstract

The integration theory of reactive behaviors is to be discussed in this paper. A linear emerging model is adopted where the motion of a robot is represented as the weighted linear sum of reactive behaviors. The weights are defined as differentiable nonlinear functions of sensor signals and parameters. We proposed approaches toward skill learning and skill abstraction based on the sensor space model, where the parameters are systematically tuned through iteration of trials such that the sensor signals converge to the given teacher signals. The learning algorithm and the abstraction algorithm are experimentally applied to the reactive grasp of a three-fingered robot hand. The experimental results illustrate the effectiveness of the proposed algorithms.

1 Introduction

The synthesis of machine intelligence has been a central issue in robotics research. Many contributions have been made and accumulated by numerous researchers in image understanding, dexterous motion control, motion planning and optimization, and so on. A unique approach of Brooks (1) was based upon the principle that the essence of robotic intelligence comes out of the interaction with the real world through sensors, and proved that behaviors with biological complexity emerge not from the complexity of computation, but from the complexity of environments. More recent works (4; 3) agreed and showed that the networked behaviors and sensors with rather simple hardware and software generate animal-like instinctive and complex behaviors through the interaction with the environments.

The main issues of recent research in the behavioral robotics would be: (1) designing rational networks, (2) providing behaviors with objectivity, (3) structure for learning and self-organizing.

In this paper, we design the behavioral networks using smooth nonlinear functions, and propose to generate behaviors by smoothly blending. We use the radial base functions as nonlinear functions and parameterize the networks with their coefficients. We implemented the idea into the behavioral synthesis of grasping and experimentally discussed the usefulness.

Behavioral approaches to grasping include Speeter (5) where manipulation by a Utah/MIT hand was implemented with more than 50 primitive behaviors. Matsui and Omata (6) designed selectively emerging mechanism adopting multi-agent architecture. On the other hand, Michelman and Allen (7) synthesized complex tasks of a Utah/MIT hand by combining two dimensional primitives of manipulation functions.

2 Behavioral Networks and Nonlinear Functions

2.1 Analytical expression of behaviors

A behavioral primitive $B_j$ ($j = 1, \cdots, n$) can be a functional unit of behavior, or a nonfunctional unit that has no linguistic meaning. We assume in this paper that $B_j$ is represented only by $\theta_j$.

On the other hand, $S_i$ ($i = 1, \cdots, m$) represents a sensor. $S_i$ does not necessarily correspond to a physical sensory device. It is one of sensory information obtained after processing raw sensory signals. For examples, a video image may offer many $S_i$'s. Or one can make $S_i$ using several physical sensors.
We call the normalized sensory information $S_i$ the sensor intensity and represent it by $\alpha_i \ (0 \leq \alpha_i \leq 1)$. The vector form of the sensor intensity is given by

$$\alpha = (\alpha_1 \ \alpha_2 \ \cdots \ \alpha_m)^T \quad (1)$$

$\alpha$ may include the internal states such as joint angles $\theta$. $\alpha$ may also include its history or integration.

Velocity $\dot{\theta}_j$ of primitive behavior $B_j$ can be a constant or a sensory feedback. Therefore, it is generally represented by

$$\dot{\theta}_j = \dot{\theta}_j(\alpha) \quad (2)$$

where $\dot{\theta}_j$ is differential with respect to $\alpha$.

Primitives $\dot{\theta}_j \ (j = 1, \ldots, n)$ are integrated and fused to emerge behavior $\theta$ in the following manner:

$$\dot{\theta} = \sum_{j=1}^{n} \beta_j \dot{\theta}_j \quad (3)$$

where $\beta_j \ (0 \leq \beta_j \leq 1)$ is a scalar named the behavior intensity. It is a function of $\alpha$. Namely,

$$\beta_j = \beta_j(\alpha) \quad (4)$$

$\beta_j$ is assumed differentiable with respect to $\alpha$.

The vector form of $\beta_j$ is represented by:

$$\beta = (\beta_1 \ \beta_2 \ \cdots \ \beta_n)^T \quad (5)$$

It can be observed from Eq. (3) that $\dot{\theta}_j$ does not appear when $\beta_j = 0$, while $\dot{\theta}$ includes full of $\dot{\theta}_j$ when $\beta_j = 1$. Even if a primitive $\dot{\theta}_j$ is given constant, $\dot{\theta}_j$ may change. $\beta(\alpha)$ is a nonlinear function of $\alpha$.

2.2 Sensor Intensity and Behavior Intensity

Behavior intensity $\beta_j$ represents the blending rate of $\dot{\theta}_j$. It has been common to use binary $\beta_j$ in the selection of behaviors. Namely,

$$\beta_j = \begin{cases} 
1 & \text{if } (\alpha_1 = \alpha_1^0 \ \text{AND} \ \alpha_2 = \alpha_2^0) \\
0 & \text{OR} \ (\alpha_3 = \alpha_3^0) \\
\text{otherwise} & 
\end{cases} \quad (6)$$

An advantage of representing $\beta_j$ by if-then logic like Eq. (6) is the fact that $\beta_j$ can be programmed based upon the semantic understanding of sensory information and primitive behaviors. In this paper, we seek to represent $\beta_j$ by a differential function of $\alpha$.

First of all, $\alpha_i^0 \ (0 \leq \alpha_i^0 \leq 1)$ is defined to be the sensor offset of $B_j$.Namely,

$$\alpha_j = \alpha - \alpha_j^0 \quad (7)$$

$$\alpha_j^0 = (\alpha_1^0 \ \alpha_2^0 \ \cdots \ \alpha_m^0)^T$$

$$\alpha_j = (\alpha_1 \ \alpha_2 \ \cdots \ \alpha_m)^T$$

$\alpha_j^0$ is said to be activable if it activates $B_j$ when $\alpha_j \cong 0$. On the contrary, $\alpha_j$ is inhibitory if it activates $B_j$ when $\alpha_j \cong 0$ is not satisfied. Like logical symbols, activable $\alpha_j$ is represented by $A_i$, while inhibitory $\alpha_j$ is represented by $\bar{A}_i$. We can describe semantic relationship between $\alpha_j$ and primitive $B_j$ in terms of the rules of binary logic. Such a relationship can be expressed by a logic equation. For example,

$$A_j = (A_1 \oplus A_2) + A_3 + (A_4 \oplus A_5) \cdot \bar{A}_6 \quad (8)$$

where $\oplus$ and $\cdot$ represent logical product (AND) and logical sum (OR) respectively.

We define the transformation from a logic equation to a function as follows:

$$\beta_j(\alpha_j) = P(A_j) \quad (9)$$

where $P(\cdot)$ follows the expansion rules:

$$P(A \cdot B) = P(A) \cdot P(B)$$

$$P(A + B) = 1 - (1 - P(A)) \cdot (1 - P(B))$$

$$P(\bar{A}) = 1 - P(A)$$

$$P(E) = 1$$

$$P(O) = 0$$

According to the above equations, $P(A_j)$ is transformed and represented as polynomials of $P(A_i) \ (i = 1, \ldots, m)$. Motivated by $A \cdot A = A$, we now introduce the following reduction rule:

$$P(A) \cdot P(A) = P(A) \quad (10)$$

The reduction rule reduces the order of polynomials. Each term becomes at most first-order of every $P(A_i)$.

We can easily prove that the transformation using the above expansion and reduction rules satisfy the axiom system of the binary Boolean algebra due to Huntington, if we assume $P(\cdot) \in X, X = [0, 1]$.

It implies that the transformation is closed under the basic theorems of the binary Boolean algebra, such as the associative law, the absorption law, the idempotency law, the identity element, and the de Morgan theorem. In other words, we can transform an arbitrary logic equation between
\( \alpha_i \) and \( \beta_i \) like Eq. (8) into a polynomial equation of \( P(A_{ij}) \) with complete logical consistency.

Concretely, we choose \( P(A_{ij}) \) as follows:

\[
P(A_{ij}) = \exp[-w_{ij} \alpha_i^2] \in X
\]

(11)

For examples, \( P^\text{A.AND}_j \), \( P^\text{A.OR}_j \), \( P^\text{I.AND}_j \) and \( P^\text{I.OR}_j \) are Activation (AND), Activation (OR), Inhibition (AND) and Inhibition (OR) and defined by

\[
P^\text{A.AND}_j \triangleq \begin{cases} 
P(A_{ij} \cdot A_{2j}) \\
= \prod_{i=1}^{2} \exp[-w_{ij} \alpha_i^2]
\end{cases}
\]

(12)

\[
P^\text{A.OR}_j \triangleq \begin{cases} 
P(A_{ij} + A_{2j}) \\
= 1 - \prod_{i=1}^{2} \left(1 - \exp[-w_{ij} \alpha_i^2]\right)
\end{cases}
\]

(13)

\[
P^\text{I.AND}_j \triangleq \begin{cases} 
P(\overline{A_{ij} + A_{2j}}) = 1 - P(A_{ij} \cdot A_{2j}) \\
= 1 - \prod_{i=1}^{2} \exp[-w_{ij} \alpha_i^2]
\end{cases}
\]

(14)

\[
P^\text{I.OR}_j \triangleq \begin{cases} 
P(\overline{A_{ij} + A_{2j}}) = 1 - P(A_{ij} + A_{2j}) \\
= \prod_{i=1}^{2} \left(1 - \exp[-w_{ij} \alpha_i^2]\right)
\end{cases}
\]

(15)

The mappings of functions are shown in Fig. 1.

Note that \( \beta_j \) in Eq.(9) is differentiable in terms of not only \( \alpha \), but also sensor offset \( \alpha_j^0 \) and sensor weight \( w_{ij} \). Namely,

\[
\beta_j = \beta_j(\alpha, \alpha_j^0, W_j)
\]

(12)

where

\[
W_j = \text{diag}\{w_{ij}\} \in \mathbb{R}^{m \times m}
\]

In this paper, we choose \( w_{ij} \) and \( \alpha_j^0 \) as parameters for learning and self-organizing.

### 2.3 Sensory space model

\( w_{ij} \) in Eq.(11) yields the strength of influence from \( S_i \) to \( B_j \). In order to describe the relationship between the sensors and the primitive behaviors, we use an Euclidean model of sensory space.

The sensory space is an Euclidean space with as many dimensions as the total number \( m \) of sensors \( S_i \) \((i = 1, \ldots, m)\). We assume that there exist point-masses as many as the total number \( n \) of primitives \( B_j \) \((j = 1, \ldots, n)\). Note that the "point masses" do not have physical dimension of mass. We introduced them to design the structure of mechanics.

A point mass is paired with a primitive behavior and called behavioral point-mass \( B_j \). The position of the point-mass in the sensory space is represented by \( b_j \). Namely,

\[
b_j = (b_{ij} \ b_{2j} \ \cdots \ b_{nj})^T \quad (j = 1, \ldots, n)
\]

(14)

Suppose sensor \( S_i \) activates when \( b_{ij} \geq 0 \), while it inhibits primitive \( B_j \) when \( b_{ij} \leq 0 \). In the scope of this paper, we also assume that \( b_{ij} \) does not change its sign. \( w_{ij} \) in Eq.(11) is given by

\[
w_{ij} = b_{ij}^2
\]

(15)

If behavioral point-mass \( B_j \) is far from the origin along the \( i \)-th coordinate, sensor \( S_i \) clearly affects primitive \( B_j \). If \( B_j \) is near the origin, it becomes insensitive to sensor \( S_i \). Likewise, by continuously changing the configuration of the behavioral point-masses, we can continuously change the mapping from sensor \( S_i \) \((i = 1, \ldots, m)\) to behavior \( B_j \) \((j = 1, \ldots, n)\).

### 3 Skill Learning

#### 3.1 Ideal sensor intensity

We assume that the sensory intensity \( \alpha_d(t) \) \((0 \leq t \leq T)\) of the ideal behavior is known a priori, though we do not know what configuration of the behavioral point-masses generate the ideal mapping. We also assume the change of environments and their initial setting is precisely repeatable while learning. We determine the configuration of behavioral point-masses for the \( k + 1 \)-th learning from the sensor intensity \( \alpha \) obtained after the \( k \)-th learning.

#### 3.2 Learning

We prepare \( b \) from the configuration of the behavioral point-masses \( b_j \) \((j = 1, \ldots, n)\) as follows:

\[
b = (b_1^T \ b_2^T \ \cdots \ b_n^T)^T \in \mathbb{R}^{m \times n}
\]

(16)

The performance index of learning is defined as follows, where \( Q \) is a positive definite matrix:

\[
I(b) = \int_0^T (\alpha(t) - \alpha_d(t))^T Q (\alpha(t) - \alpha_d(t)) dt
\]

(17)

The index becomes zero only when the sensory signal completely agrees with the desired one. The change of the index \( \Delta I \) is computed in response to a small perturbation of the behavioral point-masses from \( b \) to \( b + \Delta b \):

\[
\Delta I(b) = \frac{\partial I(b)}{\partial b} \Delta b
\]

(18)

Therefore, if we choose \( \Delta b \) as follows:
then, we have the following result:

$$\Delta I(b) = -k \left( \frac{\partial I}{\partial b} \right)^T \leq 0$$  \hspace{1cm} (20)

Note that $\Delta I(b)$ is negative semi-definite, not negative definite. Therefore, it is anticipated that the learning will converge to a local minimum.

3.3 Computation of $\frac{\partial I}{\partial b}$

The essential difficulty of learning is the computation of $\frac{\partial I}{\partial b}$. As we discuss later more, the exact computation $\frac{\partial I}{\partial b}$ requires the model of environments. We propose to compute $\frac{\partial I}{\partial b}$ by approximation.

$\frac{\partial I}{\partial b}$ is represented as follows:

$$\frac{\partial I}{\partial b} = \int_0^T 2(\alpha - \alpha_0)^T Q \frac{\partial \alpha}{\partial b} dt$$  \hspace{1cm} (21)

Therefore, we need to compute $\frac{\partial \alpha}{\partial \theta}$ and $\frac{\partial \alpha}{\partial \theta}(t)$ ($0 \leq t \leq T$) implies the rate of change of the sensor intensity in response to the configuration change of the behavioral point-mass.

Since $\alpha$ changes its value by the motion of robot and the temporal change of environments it, $\alpha$ is a function of these quantities. Accordingly,

$$\alpha = \alpha(\theta, \theta, \theta, t)$$  \hspace{1cm} (22)

We now have

$$\frac{\partial \alpha}{\partial b} = \frac{\partial \alpha}{\partial \theta} \frac{\partial \theta}{\partial b} + \frac{\partial \alpha}{\partial \theta} \frac{\partial \theta}{\partial b} + \frac{\partial \alpha}{\partial \theta} \frac{\partial \theta}{\partial b}$$  \hspace{1cm} (23)

- The 1st term

Strictly speaking, $\alpha$ is a function of $\tilde{\theta}$ since the sensory information from the force sensor include the inertia force due to $\tilde{\theta}$. However, we formulate the behavioral networks by Eq. (3) neglecting such dynamic quantities. The 1st term of Eq.(23) is neglected.

$$\frac{\partial \alpha}{\partial \theta} \approx 0$$  \hspace{1cm} (24)

- The 2nd term

$\frac{\partial \alpha}{\partial \theta}$ implies the effect of the angular velocity of finger joints to all the sensors. Strict computation required the knowledge of shape and compliance. We cannot assume this, since we do not use the environmental model. We approximate as follows:

$$\frac{\partial \alpha}{\partial \theta}$$

is a matrix whose row and column numbers are equal to those of the generalized coordinates and the total number of sensors. The column number is larger than that of the row number. $\theta$ will widely change its value.

Therefore, we can assume $\frac{\partial \alpha}{\partial \theta}$ is row-full rank. Accordingly,

$$\frac{\partial \alpha}{\partial \theta} \frac{\partial \theta}{\partial b} = E$$  \hspace{1cm} (25)

From Eq.(25) we can approximate $\frac{\partial \alpha}{\partial \theta}$ by the following least square solution:

$$\frac{\partial \alpha}{\partial \theta} \approx \left( \frac{\partial \theta}{\partial \alpha} \right)^\#$$  \hspace{1cm} (26)
where \((\ast)\) is the pseudoinverse. \(\frac{\partial \dot{\theta}}{\partial \alpha}\) can be analytically computed from Eqs. (3) and (9) as follows:

\[
\frac{\partial \dot{\theta}}{\partial \alpha} = \sum_{j=1}^{n} \frac{\partial \beta_j}{\partial \alpha} \dot{\theta}_j + \beta_j \frac{\partial \dot{\theta}_j}{\partial \alpha}
\]

(27)

On the other hand, \(\frac{\partial \dot{\theta}}{\partial b}\) is also analytically obtained.

\[
\frac{\partial \dot{\theta}}{\partial b} = \sum_{j=1}^{n} \frac{\partial \beta_j}{\partial b} \dot{\theta}_j
\]

(28)

To summarize, the 2nd term is computed by Eq.(26), (27) and (28).

- The 3rd term \(\frac{\partial \alpha}{\partial \theta}\) represents the effect of the change of generalized coordinates on all the sensors. \(\alpha\) includes elements that is determined by robot itself. We reserve and consider those determined by robot itself in \(\alpha_\theta \in \mathcal{R}^m\) and set the other elements zero. Namely,

\[
\frac{\partial \alpha}{\partial \theta} \approx \frac{\partial \alpha_\theta}{\partial \theta}
\]

(29)

We can compute \(\frac{\partial \dot{\theta}}{\partial b}\) as follows:

\[
\frac{\partial \dot{\theta}}{\partial b}(t) = \frac{\partial}{\partial b} \left[ \dot{\theta}(0) + \int_0^t \dot{\theta} dt \right]
\]

(30)

where \(\frac{\partial \dot{\theta}}{\partial b}\) is given by Eq.(28). To summarize, the 3rd term is computed using Eq.(29) and (30).

In this section, we developed the computation of \(\frac{\partial L}{\partial b}\) by adopting least-square solution and neglecting higher order terms. The differentiable analytical model such as Eqs.(3), (9), (15) enabled us these analytic computations.

4 Skill Abstraction

The idea of skill learning in the previous section was to find a parameter vector, \(b \in \mathcal{R}^{m,n}\), that reproduce the temporal pattern of sensory information provided as the teacher. Although it would be difficult to discuss the existence of solutions, we optimistically assumed that having sufficient number of sensory information and primitive behaviors would guarantee this.

On the other hand, having many of them and, therefore, a large dimension of \(b\), it is natural to have plural solutions. Namely, the same skill can be attained not by the unique solution of \(b\), but by a set of parameter vectors, \(b\). Furthermore, if we assume differentiability for the environments, the sensors, and the computation algorithms of sensory information and primitive behaviors in addition to natural differentiability of dynamics and kinematics of mechanisms, the set of parameter vectors would be represented as a manifold in \(\mathcal{R}^{m,n}\). From this viewpoint, we represented the process of skill learning as seen in Fig.2.

Figure 2: The process of skill learning

Rigorously speaking, the differentiability assumption is not satisfied in general. However, we would like to point out that we also tend to estimate the world and the future and make our decisions assuming continuity or even differentiability of the world and ourselves.

The manifold of one skill would be different from that of another. For closer or similar two tasks, the two manifold may have a manifold of intersection, which implies that any vector \(b\) on the intersection manifold guarantees emergence of the two skills. We consider it the representation of a higher skill and call its computation process the skill abstraction. It is important but beyond the current study to discuss what kind of two skills allow the skill abstraction, how many skills can be abstracted as a higher level skill, and so on. In the rest of this section we propose a simple method for the skill abstraction taking as an example the case of abstracting two skills into a higher skill.

The performance index vector is defined as follows:

\[
I_t(b) = \int_0^T \left( \alpha - \alpha_d^{(i)} \right)^T Q \left( \alpha - \alpha_d^{(i)} \right) dt
\]

(31)
where $a_i(t) (i = 1, 2)$ is the desired sensory intensity of the $i$-th task. The manifold of the $i$-th skill consists of solutions of $b$ satisfying $l_i(b) = 0$. The intersection manifold of the two implies the abstracted skill. We adopt the following method to search for a vector $b$ in the intersection manifold:

1. Set $i$ either 1 or 2. Search for a vector $b$ satisfying $l_i(b) = 0$ by iterative learning. We iteratively increment $b$ by:

$$\Delta b = -k_i \left( \frac{\partial l_i}{\partial b} \right)^T$$

where $k_i$ is a non-negative constant. The index $l_i$ will then update itself as:

$$\Delta l_i = \frac{\partial l_i}{\partial b} \Delta b$$

$$= -k_i \frac{\partial l_i}{\partial b} \left( \frac{\partial l_i}{\partial b} \right)^T \leq 0$$

2. Now we search for a vector $b$ satisfying both $l_1(b) \equiv 0$ and $l_2(b) \equiv 0$. We attempt this search by alternately repeating the following two processes.

This process is conceptually shown in Fig.3. The above algorithm is a primitive and heuristic one to show the idea of abstraction. More sophisticated algorithms need to be developed.

5 Robot Hand Experimental Setup

5.1 Robot hand and sensors

Fig.4 shows the overview of the experimental setup.

5.2 Sensory information

With internal sensors of the robot hand, we prepared 48 kinds of sensory information. They were normalized and used in the experiments. The list of sensory information follows and their reference
numbers are shown in Table 1. Fig.6 illustrates the physical meaning of the main information.

| Table 1: Reference number of sensors |
|---|---|---|---|---|---|---|---|
| 1 |  $S_u$ | 13 | $d_{f2}$ | 25 | $h_{o_{f2}}$ | 37 | $\theta_{12}$ |
| 2 |  $S_o$ | 14 | $d_{f3}$ | 26 | $h_{o_{f3}}$ | 38 | $\theta_{13}$ |
| 3 |  $\gamma$ | 15 | $h_{f1}$ | 27 | $\theta_{11}$ | 39 | $\theta_{21}$ |
| 4 |  $H$ | 16 | $h_{f2}$ | 28 | $\theta_{12}$ | 40 | $\theta_{22}$ |
| 5 |  $O_{d}$ | 17 | $h_{f3}$ | 29 | $\theta_{13}$ | 41 | $\theta_{23}$ |
| 6 |  $O_{st}$ | 18 | $h_{p1}$ | 30 | $\theta_{21}$ | 42 | $\theta_{31}$ |
| 7 |  $O_{st}$ | 19 | $h_{p2}$ | 31 | $\theta_{22}$ | 43 | $\theta_{32}$ |
| 8 |  $O_{st}$ | 20 | $h_{p3}$ | 32 | $\theta_{23}$ | 44 | $\theta_{33}$ |
| 9 |  $\phi_{1}$ | 21 | $h_{m1}$ | 33 | $\theta_{31}$ | 45 | $F_{11}$ |
| 10 |  $\phi_{2}$ | 22 | $h_{m2}$ | 34 | $\theta_{32}$ | 46 | $F_{12}$ |
| 11 |  $\phi_{3}$ | 23 | $h_{m3}$ | 35 | $\theta_{33}$ | 47 | $F_{13}$ |
| 12 |  $d_{f1}$ | 24 | $h_{o_{f1}}$ | 36 | $\theta_{11}$ | 48 | $T$ |

6 Experiments

6.1 Self-organization of behavioral network through learning

The theory of section 3 was experimentally investigated. The ideal values of sensor intensity and the corresponding behavioral intensity were chosen as seen in Fig.7.

We changed and moved the behavioral point-mass configuration from the ideal configuration that results in Fig.7.

Figure 7: Desired sensor intensity (teacher) in grasping a sphere
object \((H)\).

The convergence rate of the performance index can be found in Fig.8, which implies that learning successfully proceeded in spite of the computational approximation in learning algorithm.

![Figure 8: Convergence of the learning index](image)

**6.2 Experiment of Skill Abstraction**

We also conducted experiments of skill abstraction proposed in section 4. Grasping a ball and grasping a rectangular prism are chosen as the two skills to be abstracted. We search for a single vector \(\mathbf{b}\) that realize the two tasks.

**Fig.9** shows the results of two different approaches. The axes show the performance indexes of grasping a sphere (horizontal) and grasping a rectangular-prism (vertical). In the top figure, starting from the initial set of parameters \(\mathbf{b}\) (Initial), we repeated skill learning of grasping a ball and reached \(\text{Sph}\). From this point, we repeated skill learning of grasping a rectangular-prism and grasping a sphere alternately, and arrived at \(\text{Sph&Rect}\), which implies the corresponding parameter \(\mathbf{b}\) acquired two skills. The bottom figure of **Fig.9** shows the result of another approach, where the skill of grasping a rectangular-prism was first acquired.

![Figure 9: Changes of two indexes in the skill abstraction](image)

**7 Conclusion**

In this paper, we introduced the behavioral networks with smooth mapping from sensory information to behaviors, and proposed algorithms of skill learning and skill abstraction. Experiments of the proposed algorithms were successfully done.

This work was supported by the Ministry of Education, Culture, and Sports through an Inter-university Research Program (PI: H. Inoue, University of Tokyo).

**References**


