Advanced Robotics, Vol. 21, No. 10, pp. 1177–1199 (2007)

Also available online - www.brill.nl/ar

Full paper

Incremental acquisition of behaviors and signs based on a reinforcement learning schemata model and a spike timing-dependent plasticity network

TADAHIRO TANIGUCHI * and TETSUO SAWARAGI

Graduate School of Engineering, Kyoto University Yoshida-honmachi, Sakyo, Kyoto, 606-8501, Japan

Received 17 October 2006; accepted 28 January 2007

Abstract—A novel integrative learning architecture based on a reinforcement learning schemata model (RLSM) with a spike timing-dependent plasticity (STDP) network is described. This architecture models operant conditioning with discriminative stimuli in an autonomous agent engaged in multiple reinforcement learning tasks. The architecture consists of two constitutional learning architectures: RLSM and STDP. RLSM is an incremental modular reinforcement learning architecture, and it makes an autonomous agent acquire several behavioral concepts incrementally through continuous interactions with its environment and/or caregivers. STDP is a learning rule of neuronal plasticity found in cerebral cortices and the hippocampus of the human brain. STDP is a temporally asymmetric learning rule that contrasts with the Hebbian learning rule. We found that STDP enabled an autonomous robot to associate auditory input with its acquired behaviors and to select reinforcement learning modules more effectively. Auditory signals interpreted based on the acquired behaviors were revealed to correspond to 'signs' of required behaviors and incoming situations. This integrative learning architecture was evaluated in the context of on-line modular learning.

Keywords: Reinforcement learning; symbol emergence; spike timing-dependent plasticity; operant conditioning; modular learning.

1. INTRODUCTION

Can we create a robot that we can teach tricks and their corresponding 'signs'? What neural computations are important for operant conditioning with a discriminative stimulus (SD)? These two questions, although different, need to be addressed to understand the learning process. The first question is an engineering problem, and the second question is a problem in cognitive neuroscience and psychology.

*To whom correspondence should be addressed.
E-mail: tanichu@groove.mbox.media.kyoto-u.ac.jp

In this paper, our goal is to provide a possible computational model for operant conditioning with SD by modeling a novel machine learning architecture. The computational model will enable us to teach autonomous robots tricks incrementally and provide us with a clear understanding of the learning process in operant conditioning.

1.1. Operant conditioning with SD

Numerous experiments related to operant conditioning have shown that animals become able to switch their behaviors based on an incoming SD after some operant conditioning learning phases. The operant conditioning theory involves four types of stimuli [1]: an eliciting stimulus, a reinforcing stimulus, a discriminative stimulus and a neutral stimulus. An eliciting stimulus is a stimulus that evokes a fixed, sometimes innate, response. A neutral stimulus is a stimulus that has no effect on an animal's response. The other two stimuli are more important in operant conditioning. Reinforcing stimuli applied to an animal after certain actions increase the probability with which the agent will perform the same actions. An organized series of actions are called an operant. In addition, an SD can increase the appearance ratio of an operant acquired under the same conditions that the SD was provided. It is important that the SD is associated with an operant and not an action. For example, a dog's keeper rewards a dog with food when the dog runs into the keeper's house under the condition that the keeper toots a trumpet or the dog keeper gives the dog some other rewards when the dog does tricks under the condition that the keeper says the trick's name. In these examples, the sound of the keeper's trumpet and the words are considered as the SD, and the food and other rewards are the reinforcing stimuli. With these points in mind, we classify operant conditioning tasks into two classes. One is simple operant conditioning without SD and the other is operant conditioning with SD. The former is characterized simply by rewards and penalties. This learning process is usually termed reinforcement learning in the field of machine learning. The learning process involves an animal acquiring an accepted behavior, e.g., to run into a house or to do a trick, that maximizes rewards and minimizes penalties. The reinforcement learning procedure of machine learning has been thoroughly discussed and significant progress has been made in the last two decades [2–6]. Operant conditioning with SD is where an animal is required to acquire not one, but several different behaviors. In addition, the animal has to associate incoming SDs with the acquired behaviors. Once learning is complete, the animal has to recall and perform the corresponding task and its solution when an SD is displayed. If the recalled behavior is correct, the animal will obtain adequate rewards.

1.2. Reinforcement learning and brain science

Simple operant conditioning without SD has been computationally modeled as a reinforcement learning theory [2, 3]. This learning method enables an agent to

perform as it maximizes a cumulative future discounted reward based on a Markov decision process (MDP) model. Computational research involving reinforcement learning has enabled a better understanding of neuroscience and psychology [7–9]. The computational theory of reinforcement learning helps to reveal how the human brain operates when we solve reinforcement learning tasks. Research results suggest that the basal ganglia is involved in reinforcement learning and dopamine neuron activity encoding of the reward prediction error, $\delta(t)$ (refer to (1)) [10–12]. Research has given us good examples that robotic and/or computational models may possibly help neuroscience research. However, an adequate computational model, which describes operant conditioning with an SD, has not been proposed yet. Therefore, a computational model for operant conditioning is needed to enable better understanding of neuroscience.

1.3. Associative memory and reinforcement learning

Reinforcement learning is a satisfactory computational model for operant conditioning without SD. However, reinforcement learning architectures based on MDP is unsuitable for the computational model of operant conditioning with an SD. The reason is that SD is often given as a transient stimulus, e.g. alarm and human voice. If the state expressions in MDP are adopted, such transient inputs willl be difficult to treat in the MDP framework. If we define the incoming stimulus as one dimension of a multi-dimensional state vector, the state value will be the same before and after an SD passes on in MDP. Therefore, the framework should be extended to a type of a partially observed Markov decision processes (POMDP) [13] or an integrative learning architecture with an associative memory architecture and several separated reinforcement learning modules, i.e., modular reinforcement learning architecture [6, 14]. In the current research, we chose integrative learning architecture. The associative memory in this situation has to learn the relationship between the SD and the incoming task environment. The popular Hebbian rule [15] is not applicable in this stuation. The reason is that the Hebbian rule assumes that two related signals temporarily cooccur, but the SD is usually given before the tasks change. There is not a temporarily coocurance, but a temporal ordinarity. To learn the temporal ordinarity, a temporally asymmetric associative learning rule is required. We found that spike timing-dependent plasticity (STDP) can be used to learn the ordinal relationships of incoming SD and to select the next required reinforcement learning module (see Section 3).

1.4. Emergence of behaviors and signs

Since Sony developed AIBO as a pet robot several years ago [16], many different pet robots have been developed. Most of them interact with people by using several preprogrammed behaviors. However, they cannot acquire behaviors incrementally through the interactions with their owner, which is different from real pets. A lack in adaptability and limited behaviors have led to their owners becoming bored. Asada

highlighted the importance of emerging intelligent behaviors [17]. Therefore, to develop adaptive autonomous robots capable of acquiring several behaviors like real pets is an important issue to maintain lifelong human–machine interactions. Using our example of a dog, we often teach them tricks, e.g., 'sit down!', 'fetch me the ball!' and 'stand up!', through interactions with some rewards. Psychologically, this learning process is based on operant conditioning with an SD. Developing an algorithm for such a learning process is a challenging problem for engineers. Sutton proposed options to achieve multiple skills acuisition with reinforcement learning [18]. Based on this idea, intrinsically motivated learning to enable a robot to acquire hierarchical collections of skills was proposed [19, 20]. However, the number of options and the subgoal of each option is fixed in an options framework. To achieve incremental acquisition of behaviors without an explicit supervised learning process, an incremental modular reinforcement learning architecture is needed. Takahashi [21] proposed a modular reinforcement learning system in which Q-tables corresponding to several environmental dynamics, and not behaviors or skills, are acquired by a robot. In contrast, Taniguchi [14] proposed a reinforcement learning schema model (RLSM) as an incremental modular reinforcement learning architecture that enables a robot to obtain several behaviors incrementally. We used this learning architecture in this paper. We focused on a learning rule for neural networks termed STDP, which is a temporally asymmetric learning rule recently found to occur in the cerebral cortex and the hippocampus [22, 23]. STDP is believed to be related to associative memory; thus, several statistical analyses have been performed [24]. However, the possibile functions of STDP are still under investigation. In particular, what roles STDP plays in an animal's behavioral learning process is unclear. Therefore, constructing a learning rule, which includes STDP, will indicate the advantages of this learning rule in understanding the problems of neuroscience. In this paper, we developed an integrative learning architecture that enabled an autonomous robot to perform operant conditioning with an SD by combining an STDP learning rule and an incremental reinforcement learning architecture based on an RLSM [14] (Fig. 1).

2. RLSM

In this section, RLSM [14] is explained in detail. A simple reinforcement learning architecture is insufficient for a robot to learn several behaviors incrementally. If a robot learns another behavior after it has acquired one, the acquired behavior will be overwritten by the new one. Therefore, a dynamically distributed memory architecture is necessary to obtain multiple behaviors in reinforcement learning tasks.

2.1. Basic concepts

Several modular learning architectures have been proposed. Wolpert *et al.* proposed MOSAIC as a model for the cerebellum obtaining a multiple internal model [25].

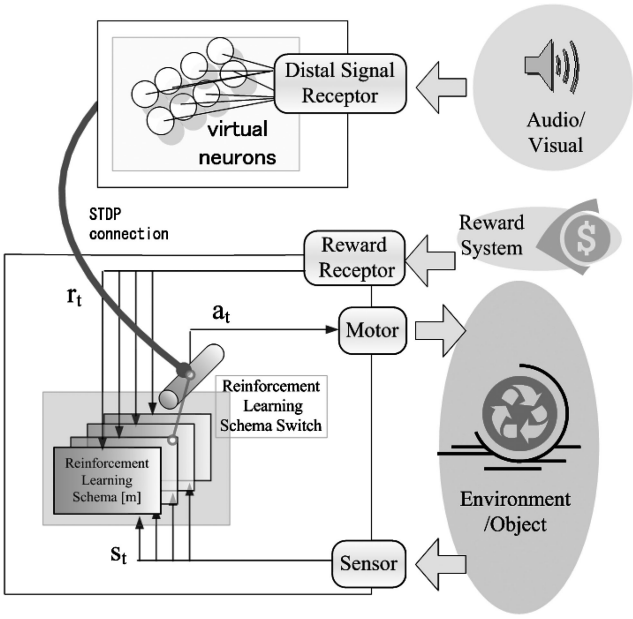

Figure 1. RLSM with an STDP network.

Tani *et al*. proposed RNNPB, which makes robots obtain several behaviors through a supervised learning phase [26]. Strictly speaking, RNNPB is not a modular learning architecture, but dynamically manages its memory based on parametric biases. Jacobs *et al*. proposed the mixture of experts [27], and this method is widely used in pattern-matching problems and state-prediction problems. However, these modular learning methods use gating or switching their modules based on state prediction errors. Singh extended this idea to reinforcement learning and proposed compositional Q-learning (CQL) [6]. Modules in this method are selected not on their state prediction errors, but on their reward prediction errors $\delta(t)$. However, these modular learning architectures, besides RNNPB, did not achieve adequate on-line incremental modular acquisition.

To achive adequate on-line incremental acquisition of learning modules learning, Taniguchi *et al.* proposed a dual-schemata model based on Piaget's schema theory [28, 29]. The schema system is described as an autonomous distributed cognitive system in Piaget's developmental psychology. The schema system explains human infant development, especially in the sensorimotor period. Taniguchi also extended that computational schema model to reinforcement learning and proposed RLSM as Singh extended mixture of experts to CQL [14].

The basic concepts of the Taniguchi's computational schema model are as follows. The schema model is characterized by two pairs of processes driven by incoming experiences. A schema assimilates experiences that are predicted with sufficient accuracy by its inner prediction function. The experiences accommodate its inner function and variance predictor. This cyclic process is called an equilibration

process. However, if all schemata refuse to assimilate a new experience, a new schema is then created for the situation producing the experience. This process is called differentiation. Equilibration and differentiation are the basic concepts of the schemata model. Other approachs to modular learning system are usually based on Bayes' rule to select an adequate module [25, 27, 30–32]. However, RLSM selects the most adequate module based on the hypothesis testing theory. RLSM is basically characterized by the difference.

2.2. Algorithm

An RLSM is formulated based on Q-learning [4] and has several reinforcement learning modules termed reinforcement learning schemata. The λ-th schema has two functions: state-action-value function Q^λ and Q^λ's second-order statistics function $Q^{(2)\lambda}$. To calculate the standard deviation of Q-value (σ^λ) $Q^{(2)\lambda}$ is used as a supplementary function. Temporal difference (TD) learning, including Q-learning, does not enable errors in the value function to be observed directly, thus temporal difference errors must be considered. $Q^{(2)\lambda}$ functions are used to estimate Q^λ's standard deviation, $\hat{\sigma}^\lambda$. Second-order statistics of a value function are also considered in Bayesian Q-learning [33]. The TD-error δ_t and secondary TD-error $\delta_t^{(2)}$ for each λ-th schema are calculated using:

$$\delta_t^\lambda = r_t + \gamma V^\lambda(s_{t+1}) - Q^\lambda(s_t, a_t) \quad \text{and} \tag{1}$$

$$\delta_t^{(2)\lambda} = r_t^2 + 2\gamma r_t V^\lambda(s_{t+1}) + \gamma^2 Q^{(2)\lambda}(s_{t+1}, a_{t+1}^*) - Q^{(2)\lambda}(s_t, a_t), \tag{2}$$

where

$$V^\lambda(s_t) = Q(s_t, a_t^*), \tag{3}$$

$$a_t^* = \arg\max_a Q^\lambda(s_t, a), \tag{4}$$

and γ is a discount parameter. Each function is updated using these errors:

$$Q^\lambda(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\delta_t \tag{5}$$

$$Q^{(2)\lambda}(s_t, a_t) \leftarrow Q^{(2)\lambda}(s_t, a_t) + \alpha\delta_t^{(2)} \tag{6}$$

$$\hat{\sigma}_t^\lambda = \sqrt{Q^{(2)\lambda}(s_t, a_t) - (Q^\lambda(s_t, a_t))^2}. \tag{7}$$

By letting δ_t be divided by the estimated standard deviation $\hat{\sigma}_t$, we define a dimensionless number, R_t^λ, as a subjective error. Subjective error R^λ is a normalized error based on the corresponding λ-th schema:

$$R_t^\lambda \equiv |\delta_t^\lambda / \hat{\sigma}_t^\lambda|^2 \tag{8}$$

$$[R^\lambda]_{-\tau}(t) = \int_{-\infty}^{0} \frac{1}{\tau} \exp\left(\frac{s}{\tau}\right) R^\lambda(t+s)\,\mathrm{d}s \tag{9}$$

$$= \int_{-\infty}^{\infty} \mathbf{W}_{-\tau} R^\lambda(t+s)\,\mathrm{d}s \tag{10}$$

$$\sim (1-p)\sum_{k=0}^{\lambda} p^k R^{\lambda}_{t-k} \tag{11}$$

$$\sim (1-p)R^{\lambda}_t + p[R^{\lambda}]_{-\tau}(t-\Delta t) \tag{12}$$

$$\mathbf{V}^{\lambda}(t) = \chi_{\mathrm{c}}(n_p[R^{\lambda}], n_p) \tag{13}$$

$$n_p = \frac{1+p}{1-p}, \tag{14}$$

where $\chi_{\mathrm{c}}(x,n)$ is a χ^2 one-sided cumulative function, $\mathrm{Prob}(X|X > x)$, which is a monotonously decreasing function and has a max value of 1. The averaged n squared normalized Gaussian probability variables have a $\chi_{\mathrm{c}}(nx,n)$ as a one-sided cumulative function. The weighted average of a squared normalized Gaussian with parameter p has a $\chi_{\mathrm{c}}(n_p x, n_p)$ approximately. (The approximation is derived by fitting their first and second moments.) $[R^{\lambda}]_{-\tau}$ is the temporal weighted average of the subjective errors. R^{λ}_t and $R^{\lambda}(t)$ are the subjective errors of the λ-th schema in continuous and discrete time, respectively. Operator $[*]_{\tau}$ is an operater defined as:

$$[f]_{\tau}(t) = \int_{-\infty}^{\infty} \mathbf{W}_{\tau}(s) f(t+s)\,\mathrm{d}s \tag{15}$$

$$= \int_{-\infty}^{\infty} \frac{1}{\tau}\,\mathrm{sgn}\left(\frac{s}{\tau}\right)\exp\left(-\frac{s}{\tau}\right) f(t+s)\,\mathrm{d}s \tag{16}$$

$$\mathrm{sgn}(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

$$\mathbf{W}_{\tau} = \frac{1}{|\tau|}\,\mathrm{sgn}\left(-\frac{s}{\tau}\right)\exp\left(\frac{s}{\tau}\right). \tag{18}$$

Both $[*]_{\tau}$ and sgn are used throughout this paper. The operator $[*]_{\tau}$ represents the calculated temporally weighted average with a window function, $\mathbf{W}_{\tau}$.

$\mathbf{V}^{\lambda}$ is the λ-th schema activity. Schema activity quantifies how well fitted the robot's facing environment and/or reward function are to the λ-th reinforcement learning schema. Parameter p represents how long a schema retains a previous recognition. Equation (12) shows the expression in continuous time where time constant $\tau = \Delta t/(1-p)$ corresponds to p; Δt is the continuous time for one step in discrete time. Furthermore, a reinforcement learning schema determines whether to assimilate or reject an experience by referring it to schema activity $\mathbf{V}^{\lambda}$. If all existing schemata reject an incoming experience, differentiation is initiated and a new schema is then created. This algorithm enables an autonomous robot to notice qualitative changes in a time series of s_t, a_t and r_t, and to obtain new behavioral concepts incrementally. Significance parameter $\mathbf{V}_{\alpha}$ is set, and the probability, $P(\lambda)$,

with which the λ-th schema is selected, are defined as:

$$\mu(H^{\lambda}) = \operatorname{sgn}\left(\mathbf{V}^{\lambda}(t) - \mathbf{V}_{\alpha}\right) \tag{19}$$

$$P(\lambda) = \mu(H^{\lambda}) \prod_{k=0}^{\lambda-1} \left(1 - \mu(H^{k})\right), \tag{20}$$

where $\mu(H^{\lambda})$ is the truth value of hypothesis H^{λ}, which means the degree to which the robot's facing environment corresponds to the λ-th schema. H^{0} is a dummy hypothesis defined to simplify the equation ($\mu(H^{0}) = 0$). As the above expressions indicate, the schema model is based on a testing statistical hypothesis theory mathematically. When the the errors are assumed to have a normal distribution, a χ^2 test is available to determine if the test data correspond to a preexisting hypothesis, i.e., reinforcement learning schema.

This learning architecture has been evaluated elsewhere [41] and shown to enable a robot to obtain behaviors incrementally. However, a problem exists with this kind of modular learning architecture. Window function $\mathbf{W}_{-\tau}$ in (10) has its center of gravity at a t of $-\tau$. This means that the schemata are selected based on past subjective errors. The schema selection delay depends on time constant τ. If the constant is too small, the RLSM becomes sensitive to incoming noise. Therefore, a trade-off exists between robustness and quickness. To overcome this trade-off, we introduced an STDP learning rule to the RLSM, as described in the next section.

3. STDP

An STDP learning rule has been found to operate in the cerebral cortex and hippocampus [22, 23]. Before this finding, the Hebbian learning rule [15] was considered to play the main role in associative learning. What is a the important difference between the STDP rule and the Hebbian rule? The STDP rule is a temporally asymmetric rule, whereas the Hebbian rule is a temporally symmetric rule (Fig. 2). The functions that STDP possess have been studied by many researchers [34–37]. However, the functions relating to an agent's behavior and its acquisition of behaviors are still unclear.

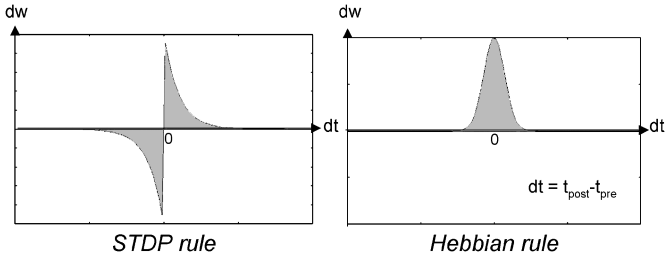

Figure 2. Difference between the asymmetric STDP learning rule and the symmetric Hebbian learning rule.

3.1. STDP learning rule

Various studies on computational STDP learning rules were conducted after neurobiological experiments found the asymmetric learning rule [38, 39]. However, such experiments could not clarify the exact mathematical formula of an STDP learning rule. Therefore, many different expressions of STDP rules have been proposed and discussed [24]. The most common STDP rule is described as:

$$\Delta w = \sum_{t_\mathrm{i}\in\mathcal{T}_I, t_\mathrm{o}\in\mathcal{T}_O} \mathbf{W}^{\mathrm{STDP}}(w, t_\mathrm{o} - t_\mathrm{i}) \tag{21}$$

$$\begin{aligned}\mathbf{W}^{\mathrm{STDP}}(w, \Delta t) &= \begin{cases} \frac{S_+(w)}{\tau_+}\exp(-|\Delta t|/\tau_+) & \text{if } t > 0, \\ -\frac{S_-(w)}{\tau_-}\exp(-|\Delta t|/\tau_-) & \text{otherwise} \end{cases} \\ &= S_+(w)\mathbf{W}_{\tau_+} - S_-(w)\mathbf{W}_{-\tau_-}, \end{aligned} \tag{22}$$

where w is the synaptic weight between the pre and post neuron, Δw is the change in the synaptic weight, $\mathcal{T}_O$ is a set of time points when the post neuron fires, and $\mathcal{T}_I$ is a set of time points when the pre neuron fires. $\mathbf{W}(w, \Delta t)$ is an asymmetric function with respect to Δt, as shown in Fig. 2. τ_+ and τ_- are the time constants, and S_+ and S_- are the learning gains in the STDP. Generally, gain parameters S_+ and S_- depend on the w. However, we only treated the case when the gain parameters were independent of the w in this paper. Song *et al.* made the same assumption [39].

However, a simple learning rule, as shown in (21), is obviously insufficient. If orderly input–output spikes were given to an STDP synaptic connection without other additional learning rules, the w would diverge. Therefore, a hard boundary [39], which restricts the w between a minimum and maximum value, or a soft boundary [23], which includes an additional multiplicative rule to the additive STDP rule, should be designed. We introduced a multiplicative decay term in proportion to the w:

$$\Delta w = \sum_i -S_{\mathrm{decay}} w(t_\mathrm{i}). \tag{23}$$

S_{decay} is the gain parameter of the decay term. By using the operator $[*]_\tau$, the total modified STDP rule can be transformed into a simple form like a dynamical system:

$$\dot{w} = S_+ I[O]_{-\tau_+} - S_- O[I]_{-\tau_-} - S_{\mathrm{decay}} w I, \tag{24}$$

where I and O are input and output functions, respectively (see Appendix A). Each function is defined as:

$$I(t) = \sum_{t_\mathrm{i}\in\mathcal{T}_I(T)} \delta(t - t_\mathrm{i}) \tag{25}$$

$$O(t) = \sum_{t_\mathrm{o}\in\mathcal{T}_O(T)} \delta(t - t_\mathrm{o}), \tag{26}$$

where $\mathcal{T}_I(T)$ and $\mathcal{T}_O(T)$ are sets of time points when the pre neuron and the post neuron fires, respectively, until time T. Dirac's $\delta(t)$ represents a spike.

3.2. What does STDP encode to synaptic weight?

RLSM selects the reinforcement learning schema, i.e., the Q-table, based on the $\mathbf{V}^{\lambda}(t)$ defined in (13). $\mathbf{V}^{\lambda}(t)$ is originally from $[R^{\lambda}]_{-\tau}(t)$. However, as mentioned, the window function $\mathbf{W}_{-\tau}$ in (10) has its center of gravity at a t of $-\tau$. MOSAIC [25, 31] also has the same time delay, and RNNPB [26] and the mixture of experts [27, 32] also has similar characteristics. To reduce this time delay, τ should be small. However, the modular selection and organization would become unstable if the τ were too small. Therefore, τ causes a trade-off between the time delay of the modular selection and the stability in an on-line modular organization. If an off-line learning rule were adopted, τ does not always make modular organization worse.

We estimated the future weighted averaged subjective error $[R]_{\tau}(t)$. This error cannot usually be calculated because it requires future information about R. We introduce an STDP neuronal network to RLSM, as shown in Fig. 3. The post neurons indicated in Fig. 3 correspond to schemata that fire spikes coding subjective errors. For this paper, we assumed an input neuron fires when the sound characteristics of each neuron come in. We obtained the estimated value by averaging $[R]_{\tau}$ observed after a certain pre-neuron fired. If the pre-neuron fires at t:

$$\widehat{[R]}_{\tau_+} = E\big[[R]_{\tau}(t_{\mathrm{i}})|t_{\mathrm{i}} \in \mathcal{T}_I\big] \tag{27}$$

$$= E\left[\int_{-\infty}^{T} \mathbf{W}_{\tau_+}(s)R(t_{\mathrm{i}}+s)\,\mathrm{d}s\,\middle|\,t_{\mathrm{i}} \in \mathcal{T}_I\right] \tag{28}$$

$$\sim E\left[\int_{-\infty}^{T} (\mathbf{W}_{\tau_+}(s) - \mathbf{W}_{-\tau_-}(s))R(t_{\mathrm{i}}+s)\,\mathrm{d}s\,\middle|\,t_{\mathrm{i}} \in \mathcal{T}_I\right] + [R]_{-\tau}(t). \tag{29}$$

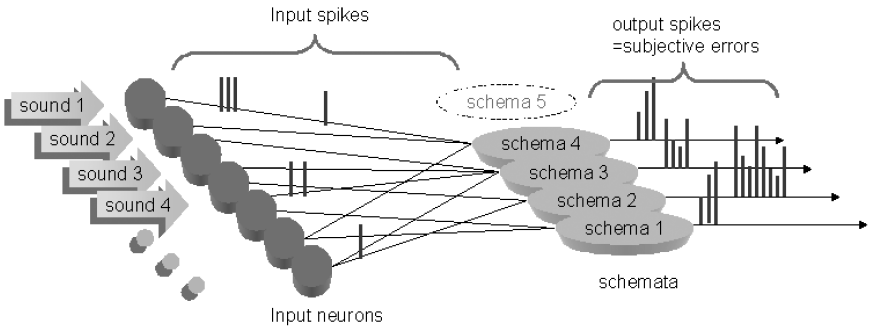

Figure 3. STDP neuronal network connected to reinforcement learning schemata firing spikes that encode subjective errors.

The first term in (29) represents the estimated difference between $[R]_{\tau}$ and $[R]_{-\tau}$. By transforming the first term, we obtain:

$$\sim \frac{\int_{-\infty}^{T}\int_{-\infty}^{T} I(t)(\mathbf{W}_{\tau_+}(s-t)+\mathbf{W}_{-\tau_-}(s-t))R(s)\,\mathrm{d}s\,\mathrm{d}t}{\#(\mathcal{T}_I(T))} \tag{30}$$

$$= \frac{\int_{-\infty}^{T} R[I]_{-\tau_+} - I[R]_{-\tau_-}\,\mathrm{d}t}{\int_{-\infty}^{T} I\,\mathrm{d}t}. \tag{31}$$

By differentiating this formula with respect to T, we obtain:

$$\dot{w} = \frac{1}{\#(\mathcal{T}_I(T))}(R[I]_{-\tau_+} - I[R]_{-\tau_-} - wI) \tag{32}$$

$$= S_+ I[R]_{-\tau_+} - S_- R[I]_{-\tau_-} - S_{\mathrm{decay}} wI, \tag{33}$$

where $S(T) \equiv S_+(T) = S_-(T) - = S_{\mathrm{decay}}(T) = 1/\#(\mathcal{T}_I(T))$ (see Appendix B). However, the STDP gain parameters explicitly depend on T. Therefore, we modify the estimation method in (27) to a weighted average. The parameters in such cases become constant values (see Appendix C). This shows that the STDP learning rule can encode differences between estimated future averaged subjective errors and past averaged subjective errors to the synaptic weight by considering R as output spikes of the post-neuron.

3.3. Integrative learning architecture

Previous research investigated hierarchical learning architectures. In such hierarchical learning architectures, the higher-level layer can learn lower learners' activities and lower-level learners can learn practical dynamics or tasks [32, 40, 41]. However, most researchers have assumed incoming signs are not a transient stimulus, but a sustained stimulus. Takamuku [42] proposed a hierarchical learning architecture, which has a Hebbian network for the learners in the higer-level layer and Q-tables for the learners in the lower-level layer. However, they also assumed that the higher-level layer's input is a sustained stimulus. To treat transient input, an STDP rule is better than a Hebbian rule. Therefore, we integrated an STDP neuronal network and an RLSM to model operant conditioning with SD.

We replaced the post neurons in the STDP network by a reinforcement schemata firing R^{λ} as output spikes (Fig. 3). We assumed that several different sound cues were provided as SD. The reason for this is as mentioned in the previous subsection; the STDP network automatically encodes information from the estimated $[R^{\lambda}]_{\tau_+}$ to the synaptic weight, w_i^{λ}, between the i-th input neuron and the λ-th schema. This integrative learning architecture has an STDP learning rule and an RLSM learning rule running. Under these conditions, the architecture can make sense out of incoming stimuli related to changes in schemata activities. Window function $\mathbf{W}_+$ has its center of gravity at a t of τ. Therefore, the RLSM becomes able to select an appropriate schema by referring to the incoming stimuli interpreted as a 'sign'

of the change that will happen in the future. In terms of operant conditioning, the incoming sound will be referred to as an SD [1].

To modulate the schema activity, $\mathbf{V}^{\lambda}$, by using the information extracted from the incoming 'sign', the inner states of the schemata need to be defined. The λ-th schema's inner state, Ψ^{λ}, changes as:

$$\dot{\Psi}^{\lambda} = \sum_{i} (w_i^{\lambda} - \Psi^{\lambda}) \operatorname{sgn}(w_i^{\lambda} - w_{\mathrm{dz}}^{\lambda}) I_i(t) - \frac{1}{\tau}\Psi^{\lambda}, \tag{34}$$

where $w_{\mathrm{dz}}^{\lambda}$ denotes the dead zone of the synaptic weight. $w_{\mathrm{dz}}^{\lambda}$ is defined as $w_{\mathrm{dz}}^{\lambda} = k\sqrt{2\,\mathrm{var}[R^{\lambda}]S_i^{\lambda}}$. k is a constant, and S_i^{λ} is the gain parameter of the connection between the i-th input neuron and the λ-th schema in the STDP learning rule. If incoming spikes and the firing subjective errors are not temporally correlated, w will distribute around 0. The first term reflects the synaptic weight, w_i^{λ}, of the nearest input spike to the λ-th schema's inner state. The second term denotes the decay term of the inner state. The inner state is designed to reflect the w of the latest recieved input spike. By using this inner state, the RLSM can select an appropriate schema more effectively without wasting any time. The modulated schema activity is defined as:

$$\mathbf{V}_{(\tau_+)}^{\lambda} = \chi_{\mathrm{c}}\big(n_p[\widehat{R^{\lambda}}]_{\tau_+}, n_p\big) \tag{35}$$

$$\sim \chi_{\mathrm{c}}\big(n_p(\Psi^{\lambda} + [R^{\lambda}]_{-\tau_-}), n_p\big). \tag{36}$$

4. EVALUATION

First, we evaluated the RLSM by using a two-dimentional (2-D) simulation for a Khepera mobile robot [43].

4.1. Conditions

The simulation space and a Khepera's sensory-motor system are illustrated in Fig. 4. We used Webots, produced by Cyberbotics, to simulate Khepera's dynamics. The square simulation space was enclosed by walls 2 m long and 10 cm high. A light source was located 10 cm above the center of the space. The Khepera motor system has two wheels and their rotational velocities can be set independently for each time step. Khepera's sensory system is comprised of an infrared sensor, a light sensor and a GPS. Q-learning usually requires a discrete state space. Therefore, we divided Khepera's x, y coordinates and its angle of direction, obtained from the GPS, into six parts and defined 216 (=6 × 6 × 6) states. The action space was also made discrete by defining five representative motor outputs: forward, back, right, left and stop. Forward and back move the robot about 30 cm per step, and right and left rotate it about 60° per step. The infrared sensor (ds) and the light sensor (ls) were limited to between 0 and 1. They were used only to calculate the rewards. To detect

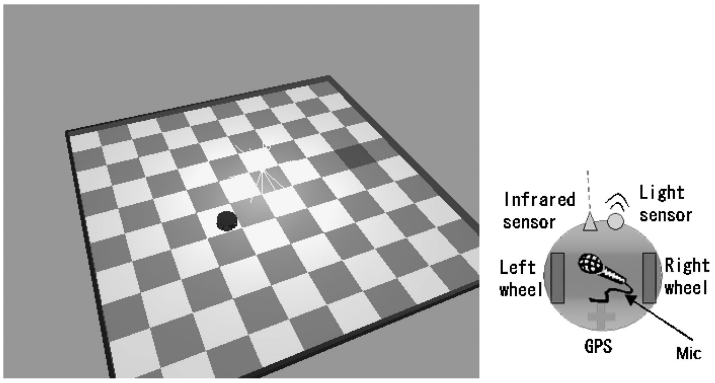

Figure 4. (Left) Simulation space, and (right) Khepera's sensory-motor system.

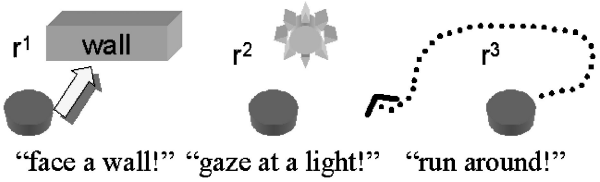

Figure 5. Reward functions.

incoming sounds, a microphone sensor was also provided and was connected to input neurons, which are modeled in Fig. 3.

We prepared three reward functions: $r^1 = ds$, $r^2 = 1.5ls$ and $r^3 = 1.8v_{\text{forward}}$, where $0 \leqslant v_{\text{forward}} \leqslant 1$ is the value given when Khepera advances. These reward functions indicate the situation when the robot should face a wall, gaze at a light, and run around, respectively (Fig. 5). The parameters for reinforcement learning were set to an α of 0.2 and a γ of 0.8. The parameters for the schemata model were set to a p of 0.999 and a $\mathbf{V}_\alpha$ of 0.0001. The parameter for the STDP was set to a k of 0.75. We previously investigated whether the RLSM enabled Khepera to obtain and recall several behavioral concepts, i.e., reinforcement learning schemata while it was interacting with a specific environment. The reward functions in this environment were altered. Each trial consisted of 200 steps Khepera's action at each step was determined by Boltzmann selection. The inverse temperature in each trial was set to a β of 0 during the first 100 steps, to a β of 1 during the next 50 steps and to a β of 3 (almost greedy) during the final 50 steps. The reward functions were set to r^1 for ($0 <$ trial $\leqslant 750$), r^2 for ($750 <$ trial $\leqslant 1500$) and r^3 for ($1500 <$ trial $\leqslant 2250$). Subsequently, each reward function was used alternately every 250 trials.

As shown in Fig. 6, each schema activity, $\mathbf{V}^\lambda$, had a successful transition. Also, the schema, initially only one, differentiated into three separate schemata. Three behavioral concepts corresponding to the three reward functions were eventually organized. Each schema was selected, as shown in Fig. 6.

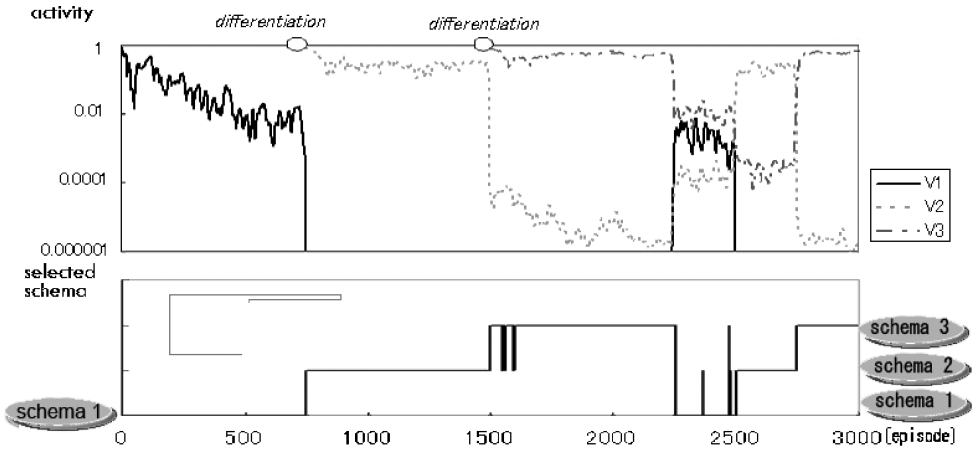

Figure 6. (Top) Schema differentiation process and transition of schema activities, and (bottom) selected reinforcement learning schema.

However, an average time of 2000 steps had already passed when RLSM recalled an appropriate schema when switching reward functions. To overcome this time delay, Khepera must be able to identify sounds when the corresponding reward functions are selected as 'signs' for upcoming situations. By using the STDP learning rule, Khepera can learn the relationship between sounds and when the next schema should be activated, and should be able to recall an appropriate schema effectively. We prepared 36 (6×6) different sounds in the simulation space to evaluate the STDP learning rule. After the three schemata were acquired, we continued to alternate the reward functions for each of the 25 trials. Sounds 1, 2 and 3 were rung when r^1, r^2 and r^3 were selected, respectively. In addition, sounds 4–36 were rung randomly as noise. Under this noisy condition, an autonomous robot had to determine meaningful sounds related to its organized schemata.

4.2. Results

As shown in Fig. 7, synaptic weights encoded the relationship between the sounds and schema 1. The results in Fig. 7 show that the synaptic weights around the third and fourth sound, which have no relationship to schema 1, converged to zero. However, the synaptic weight around the second sound was large because schema 1 started to output big subjective errors when its facing reward function was switched from r^1 to r^2. The synaptic weight around the first sound obtained a large negative value because schema 1 reduced its subjective errors when its reward function was switched from r^3 to r^1. This is in contrast to the second sound.

By exploiting these obtained synaptic weights, Khepera became able to interpret incoming 'signs' required to select the next schema. As shown in Fig. 8, the required time delays when Khepera switched its schema corresponding to external rewards decreased gradually as it learned and exploitied incoming 'signs'. Figure 9 shows the acquired synaptic weight on the network. To facilitate visualization,

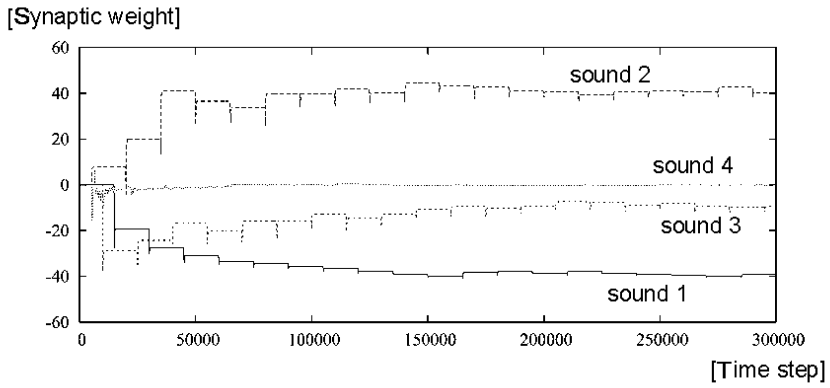

Figure 7. Transition of synaptic weight between input neurons and schema 1.

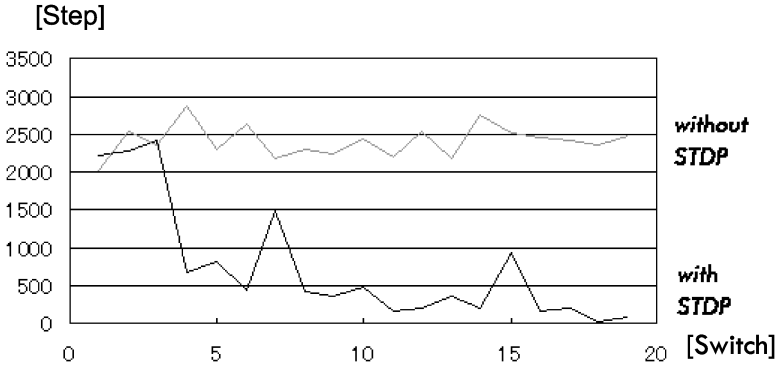

Figure 8. Course of time delays in switching schemata.

the 36 input neurons were placed in 2-D maps, as shown in Fig. 9. Each map shows the acquired synaptic weight between all input neurons and each schema. Sounds 1, 2 and 3 correspond to (1, 2), (2, 4) and (3, 6), respectively. Both negative and positive relationships between the changes in the situations and descriminative stimuli are clearly acquired by the STDP network. In addition, other synaptic weights corresponding to the other random sounds converged to alomost zero. This means the network can distinguish meaningful sounds, which relate to changes in situations, from other meaningless noise.

In an additional experiment, we continued to alternate the reward functions every 200 steps. The interval was set to be shorter than the previous task. Without an STDP neuronal network, RLSM took about 2000 more steps than the previous experiment. However, the integrative learning eventually caught up with the frequent environmental changes. This means that the robot can select adequate behavior at every moment (Fig. 10). Figure 11 shows the difference between the weighted average of obtained reward with a simple RLSM learning architecture and with that obtained using integrative learning architecture. This clearly shows that STDP helps RLSM in terms of reward.

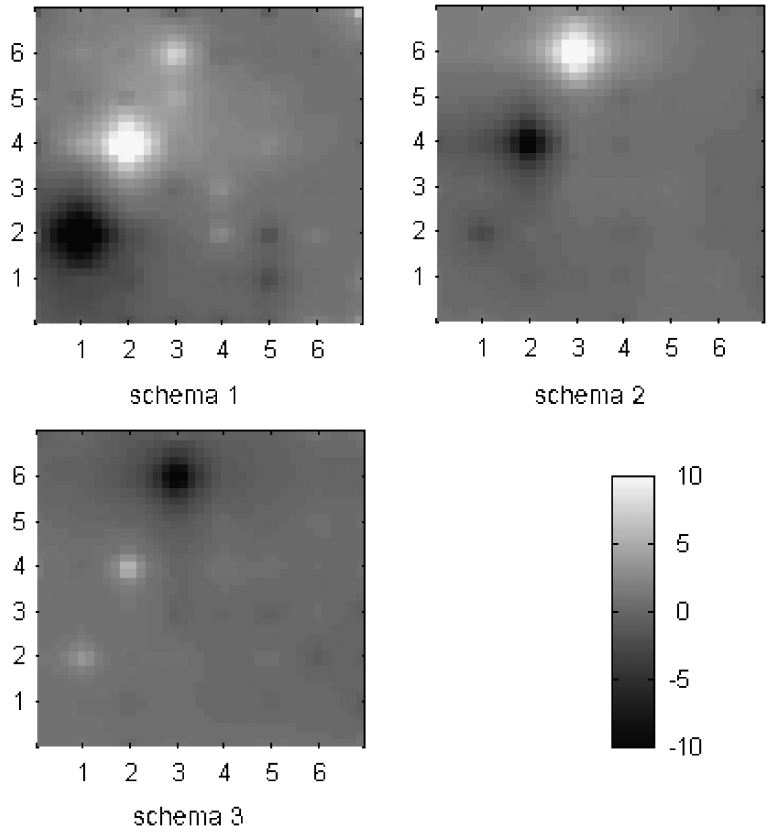

Figure 9. Obtained synaptic weight on the STDP network.

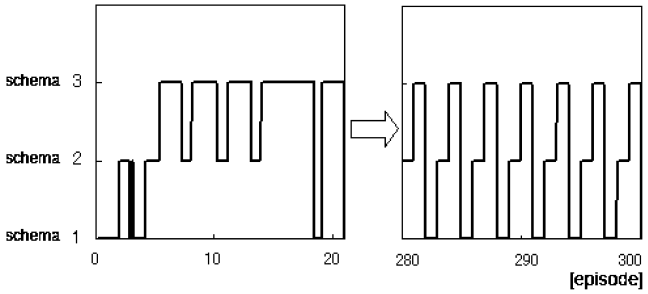

Figure 10. STDP makes RLSM switch adequately even in more frequently changing environment.

5. CONCLUSIONS

We described a novel integrative learning architecture that consists of an RLSM and an STDP neuronal network. This network achieved operant conditioning with SD. Using the RLSM, an autonomous agent could obtain several learning modules, called reinforcement learning schemata, without any explicit indication except for sensor vectors, motor vectors and rewards. After obtaining several schemata, the

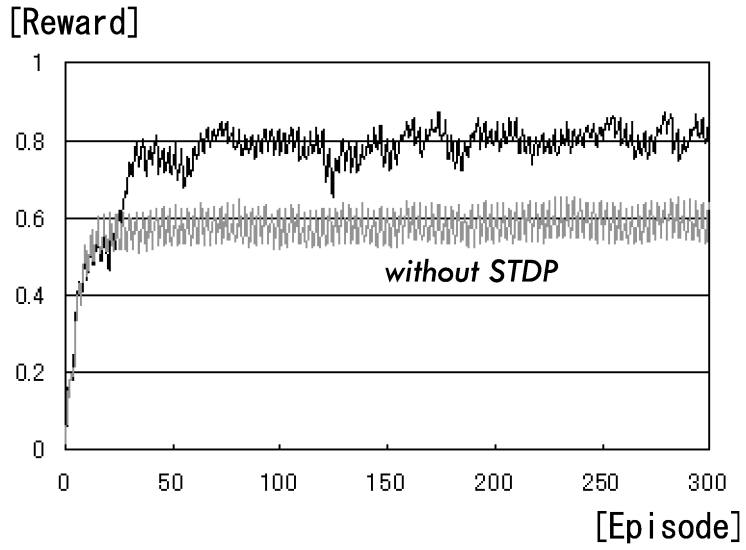

Figure 11. Difference in obtained rewards of two learning schemes.

agent could obtain the relationships between incoming sounds and the agent's obtained schemata, and exploit them as 'signs' when using STDP. Owing to the associative memory, the agent could perform better (Fig. 11). However, qualitative and not quantitative progress is important here. It is important that neither an RLSM learning rule nor an STDP learning rule is classified into supervised-learning rules. Therefore, an integrative learning rule is classified into self-organizational learning rules. The incoming sounds had no meaning before the agent determined their meanings. This learning process is closely related to 'symbol emergence' [17]. To overcome a symbol grounding problem [44], which is a historical problem in AI, symbol emergence is believed to be a promising approach [45]. Takamuku has shown that a robot can acquire several lexicons based on the dynamics of a robot facing objects and through the robot interacting with objects [42]. In contrast, this paper describes a learning architecture that enables a robot to generate several interpretants of behaviors. How we overcome the symbol grounding problem by investigating symbol emergence is one of the important next challenges.

Human beings can learn not only one behavior based on reinforcement learning, but also numerous behaviors and their corresponding SD or the behavior's name. The human brain may bring several different cortices into line to solve this complex problem. Basal ganglia is charged with reinforcement learning [8, 9]. However, a reinforcement learning framework is insufficient to achieve operant conditioning with SD. Our results suggest that cortices, which have STDP as their learning rule, other than basal ganglia also take part in the learning process. We achieved operant conditioning with SD *in silico* by combining a reinforcement learning scheme and an STDP artificial neuronal network. The STDP learning rule is found in the cerebral cortices and the hippocampus of the human brain [23]. This suggests the

possibility that cerebral cortices and/or the hippocampus take part in the learning process of operant conditioning with SD.

Acknowledgments

This work was supported in part by the Center of Excellence for Research and Education on Complex Functional Mechanical Systems (the 21st Century COE Program of the Ministry of Education, Culture, Sports, Science and Technology, Japan).

REFERENCES

1. G. S. Reynolds, *A Primer of Operant Conditioning*. Scott, Foresman, Glenview, IL (1975).
2. R. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA (1998).
3. D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Athena Scientific, Nashua, NH (1996).
4. C. Watkins and P. Dayan, Technical note: Q-learning, *Machine Learn.* **8**, 279–292 (1992).
5. K. Doya, Reinforcement learning in continous time and space, *Neural Comput.* **12**, 219–245 (2000).
6. S. P. Singh, Transfer of learning by composing solutions of elemental sequential tasks, *Machine Learning Arch.* **8**, 323–339 (1992).
7. A. G. Barto, Adaptive critics and the basal ganglia, in: *Models of Information Processing in the Basal Ganglia* (J. C. Houk, J. L. Davis and D. G. Beiser, Eds), pp. 215–232. MIT Press, Cambridge, MA (1995).
8. K. Doya, What are the computations of the cerebllum, the basal ganglia, and the cerebral cortex?, *Neural Networks* **12**, 961–974 (1999).
9. S. C. Tanaka, K. Doya, G. Okada, K. Ueda, Y. Okamoto and S. Yamawaki, Prediction of immediate and future rewards differentially recruits cortico-basal ganglia loops, *Nat. Neurosci.* **7**, 887–893 (2004).
10. T. Aosaki, H. Tsubokawa, A. Ishida, K. Watanabe, A. M. Graybiel and M. Kimura, Responses of tonically active neurons in the primate's striatum undergo systematic changes during behavioral sensorimotor conditioning, *J. Neurosci.* **14**, 3969–3984 (1994).
11. W. Schultz, Predictive reward signal of dopamine neurons, *J. Neurophysiol.* **80**, 1–27 (1998).
12. W. Schultz, P. Dayan and P. R. Montague, A neural substrate of prediction and reward, *Annu. Rev. Neurosci.* **15**, 353 (1992).
13. A. R. Cassandra, L. P. Kaelbling and M. L. Littman, Acting optimally in partially observable stochastic domains, in: *Proc. 12th Natl. Conf. on Artificial Intelligence*, Seattle, WA, vol. 2, pp. 1023–1028 (1994).
14. T. Taniguchi and T. Sawaragi, Incremental acquisition of behavioral concepts through social interactions with a caregiver, in: *Proc. Artificial Life and Robotics* (2006).
15. D. O. Hebb, *The Organization of Behavior*. Wiley, New York, NY (1949).
16. Aibo offcial site: http://www.jp.aibo.com/.
17. M. Asada, K. F. MacDorman, H. Ishiguro and Y. Kuniyoshi, Cognitive developmental robotics as a new paradigm for the design of humanoid robots, *Robotics Autonomous Syst.* **37**, 185–193 (2001).
18. R. S. Sutton, D. Precup and S. Singh, Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning, *Artif. Intell.* **112**, 181–211 (1999).

19. A. G. Barto, S. Singh and N. Chentanez, Intrinsically motivated learning of hierarchical collections of skills, in: *Proc. Int. Conf. on Development and Learning* (2004).
20. V. Soni and S. Singh, Reinforcement learning of hierarchical skills on the sony Aibo robot, in: *Proc. Int. Conf. on Development and Learning* (2006).
21. Y. Takahashi *et al*., Modular learning syatem and scheduling for behavior acquisition in multi-agent environment, in: *RoboCup 2004 Symp. Papers and Team Description Papers*, CD-ROM (2004).
22. H. Markram *et al*., Regulation of synaptic efficacy by coincidence of postsynaptic aps and epsps, *Science* **275**, 213–215 (1997).
23. L. F. Abbott and S. B. Nelson, Synaptic plasticity: taming the beast, *Nat. Neurosci. Suppl.* **3**, 1178–1182 (2000).
24. Y. Sakai, K. Nakano and S. Yoshizawa, Synaptic regulation on various stdp rules, *Neurocomputing* **58–60**, 351–357 (2004).
25. D. M. Wolpert and M. Kawato, Multiple paired forward and inverse models for motor control, *Neural Networks* **11**, 1317–1329 (1998).
26. J. Tani, M. Ito and Y. Sugita, Seif-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robots using RNNPB, *Neural Networks* **17**, 1273–1289 (2004).
27. R. A. Jacobs, M. I. Jordan *et al*., Adaptive mixtures of local experts, *Neural Comput.* **3**, 79–87 (1991).
28. T. Taniguchi and T. Sawaragi, Design and performance of symbols self-organized within an autonomous agent interacting with varied environments, in: *Proc. IEEE Int. Workshop on ROMAN*, CD-ROM (2004).
29. T. Taniguchi and T. Sawaragi, Self-organization of inner symbols for chase: symbol organization and embodiment, in: *Proc. IEEE Int. Conf. on SMC*, CD-ROM (2004).
30. M. Haruno, D. M. Wolpert and M. Kawato, Mosaic model for sensorimotor learning and control, *Neural Comput.* **13**, 2201–2220 (2001).
31. K. Doya *et al*., Multiple model-based reinforcement learning, *Neural Comput.* **14**, 1347–1369 (2000).
32. J. Tani and S. Nolfi, Learning to perceive the world as articulated: an approach for hierarchical learning in sensory-motor systems, *Neural Networks* **12**, 1131–1141 (1999).
33. R. Dearden *et al*., Bayesian Q-learning, in: *Proc. AAAI-98*, pp. 761–768 (1998).
34. R. Legenstein, C. Naeger and W. Maass, What can a neuron learn with spike-timing-dependent plasticity?, *Neural Comput.* **17**, 2337–2382 (2005).
35. H. D. I. Abarbanel, R. Huerta and M. I. Rabinovich, Dynamical model of long-term synaptic plasticity, *Proc. Natl. Acad. Sci. USA* **99**, 10132–10137 (2002).
36. H. Cateau and T. Fukai, A stochastic method to predict the consequence of arbitrary forms of spike-timing-dependent plasticity, *Neural Comput.* **15**, 597–620 (2003).
37. S. M. Bohte and M. C. Mozer, Reducing spike train variability: a computational theory of spike-timing dependent plasticity, *Adv. Neural Inform. Process. Syst.* **17**, 201–208 (2005).
38. G. Q. Bi and M. M. Poo, Synaptic modification in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type, *J. Neurosci.* **18**, 10464–10472 (1998).
39. S. Song *et al*., Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience* **3**, 919–926 (2000).
40. M. Haruno, D. M. Wolpert and M. Kawato, Hierarchical mosaic for movement generation, *Int. Congr. Ser.* **1250**, 575–590 (2003).
41. S. R. Waterhouse and A. J. Robinson, Constructive algorithms for hierarchical mixtures of experts, *Adv. Neural Inform. Process. Syst.* **8**, 584–590 (1996).
42. S. Takamuku, Y. Takahashi and M. Asada. Lexicon acquisition based on behavior learning, in: *Proc. 4th IEEE Int. Conf. on Development and Learning* (2005).

43. Webots, http://www.cyberbotics.com (commercial mobile robot simulation software).
44. S. Harnad, The symbol grounding problem, *Physica D* **42**, 35–346 (1990).
45. T. Taniguchi and T. Sawaragi, Symbol emergence by combining a reinforcement learning schema model with asymmetric synaptic plasticity, in: *Proc. 5th Int. Conf. on Development and Learning* (2006).
46. S. Song and L. F. Abbott, Cortical development and remapping through spike timing-dependent plasticity, *Neuron* **32**, 339–350 (2001).

APPENDIX A

In this section, we derive on-line STDP learning rule in continuous time. An STDP learning rule is often expressed as an off-line process as (21) [23, 24, 39, 46]. However, this learning rule is complicated when we apply STDP learning dynamics in an on-line learning agent because the timings of all input spikes and output spikes must be stored. When we apply the STDP learning dynamics to on-line experiments, a learning rule should be expressed as a dynamical system by using differential equations. Therefore, we transform (21) to obtain differential equations, i.e. an on-line STDP learning rule.

First, the total amount of change in synaptic weight w until time T can be rewritten by using Dirac's delta function $\delta(t)$.

$$w = \sum_{t_\mathrm{i}\in\mathcal{T}_I(T), t_\mathrm{o}\in\mathcal{T}_O(T)} W^{\mathrm{STDP}}(t_\mathrm{o}-t_\mathrm{i}) \tag{A1}$$

$$= \sum_{t_\mathrm{i}\in\mathcal{T}_I(T)} \sum_{t_\mathrm{o}\in\mathcal{T}_O(T)} \left[\int_{-\infty}^{T}\int_{-\infty}^{T} \delta(s-t_\mathrm{o})\delta(t-t_\mathrm{i}) W^{\mathrm{STDP}}(s-t)\,\mathrm{d}s\,\mathrm{d}t\right] \tag{A2}$$

$$= \int_{-\infty}^{T}\int_{-\infty}^{T} \sum_{t_\mathrm{o}\in\mathcal{T}_O(T)} \delta(s-t_\mathrm{o}) \sum_{t_\mathrm{i}\in\mathcal{T}_I(T)} \delta(t-t_\mathrm{i}) W^{\mathrm{STDP}}(s-t)\,\mathrm{d}s\,\mathrm{d}t \tag{A3}$$

$$= \int_{-\infty}^{T}\int_{-\infty}^{T} O(s) W^{\mathrm{STDP}}(s-t) I(t)\,\mathrm{d}s\,\mathrm{d}t. \tag{A4}$$

In the surface integral, the area for integral is divided into LTP area $S_{\mathrm{LTP}} = \{(s,t)|s > t\}$ and LTD area $S_{\mathrm{LTD}} = \{(s,t)|s \leqslant t\}$:

$$= \iint_{S_{\mathrm{LTP}}} \frac{S_+}{\tau_+} \exp\left(\frac{t-s}{\tau_+}\right) O(s) I(t)\,\mathrm{d}s\,\mathrm{d}t + \iint_{S_{\mathrm{LTD}}} -\frac{S_-}{\tau_-} \exp\left(\frac{s-t}{\tau_-}\right) O(s) I(t)\,\mathrm{d}s\,\mathrm{d}t \tag{A5}$$

$$= S_+ \int_{-\infty}^{T} O(s) \int_{-\infty}^{s} \frac{1}{\tau_+} \exp\left(\frac{t-s}{\tau_+}\right) I(t)\,\mathrm{d}t\,\mathrm{d}s - S_- \int_{-\infty}^{T} I(t) \int_{-\infty}^{t} \frac{1}{\tau_-} \exp\left(\frac{s-t}{\tau_-}\right) O(s)\,\mathrm{d}s\,\mathrm{d}t. \tag{A6}$$

The inner integral of the first term can be transformed into:

$$\int_{-\infty}^{s} \frac{1}{\tau_+} \exp\left(\frac{t-s}{\tau_+}\right) I(t)\,\mathrm{d}t \tag{A7}$$

$$= \int_{-\infty}^{\infty} \frac{1}{\tau_+} \operatorname{sgn}\left(\frac{s-t}{\tau_+}\right) \exp\left(\frac{t-s}{\tau_+}\right) I(t)\,\mathrm{d}t \tag{A8}$$

$$= \int_{-\infty}^{\infty} \frac{1}{\tau_+} \operatorname{sgn}\left(\frac{q}{-\tau_+}\right) \exp\left(\frac{-q}{-\tau_+}\right) I(s+q)\,\mathrm{d}q \tag{A9}$$

$$= [I]_{-\tau_+}(s), \tag{A10}$$

by using operator $[*]_\tau$. Here, $q = t - s$. After the same transformation of the second term, we obtain:

$$= \int_{-\infty}^{T} S_+ O(t)[I]_{-\tau}(t) - S_- I(t)[O]_{-\tau_-}(t)\,\mathrm{d}t. \tag{A11}$$

If we differentiate the equation by T, we finally obtain:

$$\dot{w} = S_+ O[I]_{-\tau_+} - S_- I[O]_{-\tau_-}. \tag{A12}$$

APPENDIX B

In this section, an STDP learning rule is derived from the expression of the expected output spike value after input spikes from a certain neuron. The transformation from (30) to (31) can be performed by using similar procedures outlined in Appendix A. Equation (31) is transformed to (32) as follows.

$$\frac{\mathrm{d}w}{\mathrm{d}T} = \frac{\mathrm{d}}{\mathrm{d}T}\left\{\frac{\int_{-\infty}^{T} O[I]_{-\tau_+} - I[O]_{-\tau_-}\,\mathrm{d}t}{\int_{-\infty}^{T} I\,\mathrm{d}t}\right\} \tag{B1}$$

$$= \frac{(O[I]_{-\tau_+} - I[O]_{-\tau_-})(\int_{-\infty}^{T} I\,\mathrm{d}t) - (\int_{-\infty}^{T} O[I]_{-\tau_+} - I[O]_{-\tau_+}\,\mathrm{d}t)I}{(\int_{-\infty}^{T} I\,\mathrm{d}t)^2} \tag{B2}$$

$$= \frac{O[I]_{-\tau_+} - I[O]_{-\tau_-}}{\int_{-\infty}^{T} I\,\mathrm{d}t} - \frac{1}{\int_{-\infty}^{T} I\,\mathrm{d}t} \frac{\int_{-\infty}^{T} -O[I]_{-\tau_+} + I[O]_{-\tau_-}\,\mathrm{d}t}{\int_{-\infty}^{T} I\,\mathrm{d}t} I \tag{B3}$$

$$= \frac{1}{\#(\mathcal{T}_I(T))}(O[I]_{-\tau_+} - I[O]_{-\tau_-} - wI). \tag{B4}$$

This expression equals (32).

APPENDIX C

An STDP learning rule whose gain term is constant can be derived by differentiating the weighted average of the difference between future and past output spikes.

The weighted average is defined as:

$$w = \frac{\int_{-\infty}^{T} (1/\nu) \exp(-(1/\nu) \int_{t}^{T} I(s)\,\mathrm{d}s)(O(t)[I]_{-\tau_+}(t) - I(t)[O]_{-\tau_-}(t))\,\mathrm{d}t}{\int_{-\infty}^{T} (1/\nu) \exp(-(1/\nu) \int_{t}^{T} I(s)\,\mathrm{d}s) I(t)\,\mathrm{d}t} \tag{C1}$$

$$= \left(\exp(-(1/\nu)\overline{I}(T)) \int_{-\infty}^{T} (1/\nu) \exp((1/\nu)\overline{I}(t) \times (O(t)[I]_{-\tau_+}(t) - I(t)[O]_{-\tau_-}(t))\,\mathrm{d}t \right) \times \left(\exp(-(1/\nu)\overline{I}(T)) \int_{-\infty}^{T} \frac{1}{\nu} \exp((1/\nu)\overline{I}(t)) I(t)\,\mathrm{d}t \right)^{-1} \tag{C2}$$

$$= \frac{\int_{-\infty}^{T} (1/\nu) \exp((1/\nu)\overline{I}(t)(O(t)[I]_{-\tau_+}(t) - I(t)[O]_{-\tau_-}(t))\,\mathrm{d}t}{\int_{-\infty}^{T} (1/\nu) \exp((1/\nu)\overline{I}(t)) I(t)\,\mathrm{d}t}, \tag{C3}$$

where ν is a forgeting time constant. Term $(1/\nu)\exp(-(1/\nu)\int_t^T I(s)\,\mathrm{d}s)$ means that the effect of the input spike becomes less as the input spike gets older.

$$\overline{I}(t) \equiv \int_{0}^{t} I(s)\,\mathrm{d}s \tag{C4}$$

$$\lim_{t \to -\infty} \overline{I}(t) = -\infty. \tag{C5}$$

The boundary condition assumes that incoming spikes have been continuously incoming. We represent the denominator of (C3) by p and the numerator by q. By differentiating the formula with T, we obtain

$$\frac{\mathrm{d}w}{\mathrm{d}T} = \frac{q'p - p'q}{p^2} \tag{C6}$$

$$= \frac{q'}{p} - \frac{p'}{p}\frac{q}{p} \tag{C7}$$

$$= \frac{1}{p}(q' - wp') \tag{C8}$$

$$= \frac{((1/\nu)\exp((1/\nu)\overline{I}(T))(O[I]_{-\tau_+} - I[O]_{-\tau_-}) - w(1/\nu)\exp((1/\nu)\overline{I}(T))I)}{\exp((1/\nu)\overline{I}(T)) - \exp((1/\nu)\overline{I}(-\infty))} \tag{C9}$$

$$= (1/\nu)(O[I]_{-\tau_+} - I[O]_{-\tau_-} - wI). \tag{C10}$$

This equals an STDP learning rule whose gain parameters are constant.

ABOUT THE AUTHORS

Tadahiro Taniguchi was born in 1978. He received the ME and PhD degrees from Kyoto University in 2003 and 2006, respectively. From April 2005 to March 2006, he was a Japan Society for the Promotion of Science (JSPS) Reserach Fellow (DC2) at the Department of Mechanical Engineering and Science, Graduate School of Engineering, Kyoto University. From April 2006 to March 2007, he was a JSPS Reserach Fellow (PD) at the same department. Since April 2007, he is a JSPS Research Fellow at the Department of Systems Science, Graduate school of Informatics, Kyoto University. He has been engaged in research on machine learning, emergent systems and semiotics.

Tetsuo Sawaragi was born in 1957, and is now a Professor at the Department of Mechanical Engineering and Science, Graduate School of Engineering, Kyoto University, Japan. He received his BS, MS and PhD degrees in Systems Engineering from Kyoto University in 1981, 1983 and 1988, respectively. From 1986 to 1994, he was a Research Associate at the Department of Precision Mechanics, Faculty of Engineering, Kyoto University, wherein he was an Associate Professor and a Professor in 1994 and 2002, respectively. In 2005 he was in the current department as a Professor. From 1991 to 1992, he was a Visiting Scholar at the Department of Engineering-Economic Systems, Stanford University, USA. He has been engaged in research on systems engineering, cognitive science and artificial intelligence, particularly in the development of human–machine collaborative systems, modeling the transfer of human cognitive skills into machines. He was a chair of IEEE SMC Japan Chapter and a Board Member of the Institute of Systems, Control and Information Engineers, Human Interface Society and Japan Society for Fuzzy Theory and Systems. He is a member of the Japanese Society for Artificial Intelligence, JSME and IEEE.