(IEEE Trans. on Syst. Man Cybern. Part A-Systems and Humans, Vol.33, No.4, pp.481-488, 2003)

Self-Organization of Behavioral Primitives as Multiple Attractor Dynamics: A Robot Experiment

(in press IEEE Trans. on System Man and Cybernetics B)

Jun Tani
Brain Science Institute, RIKEN
2-1 Hirosawa, Wako-shi, Saitama, 351-0198 Japan
Tel +81-48-467-6467, FAX +81-48-467-7248
E-mail tani@brain.riken.go.jp
http://www.bdc.brain.riken.go.jp/ tani

Masato Ito
Digital Creatures Lab.
Sony Corp.
Gotanda, Shinagawa-ku, Tokyo

Abstract
This paper investigates how behavior primitives are self-organized in a neural network model utilizing a distributed representation scheme. The model is characterized by so-called parametric biases which adaptively modulate the encoding of different behavior patterns in a single recurrent neural net (RNN). Our experiments, using a real robot arm, showed that a set of end-point and oscillatory behavior patterns are learned by self-organizing fixed points and limit cycle dynamics that form behavior primitives. It was also found that diverse novel behavior patterns can be generated by modulating the parametric biases arbitrarily. Our analysis showed that such diversity in behavior generation emerges because a nonlinear map is self-organized between the space of parametric biases and that of the behavior patterns. The origin of the observed nonlinearity from the distributed representation is discussed.

1 Introduction

Many researchers have studied how to identify motor primitives for real biological systems utilizing the language of dynamical systems. Some biological models [1, 2] explain that the end-point motor behavior, such as reaching with legs or arms to a certain position, can be generated by having fixed point dynamics as motor primitives in spinal reflexes. By vector summing among multiple fixed point dynamics, end-point

movement to an arbitrary position can be generated. Others [3, 4, 5] have shown that rhythmic movement primitives for locomotor activity can be embedded in oscillatory dynamics embodied by neural circuits at the spinal level. Taga [6] showed that bi-pedal walking can be controlled by modulating values of interference signals to oscillatory dynamical systems. It was shown that parametric interference to limit cycling attractor dynamics generates diverse transient motor trajectories which can be utilized during various behaviors, such as collision avoidance during walking. Kotosaka and Schaal [7] suggested that both fixed point dynamics and limit cycling dynamics should be utilized as motor primitives simultaneously in order to generate diverse behaviors. However, their work did not clarify well how such motor primitives could be learned or acquired.

Previously, we proposed the so-called "Forwarding Forward model", or FF-model [8], in which we explained how behavior primitives can be self-organized hierarchically and how sensory-motor flow can be recognized and articulated, i.e., segmented into reusable chunks. This model is characterized by its distributed representation scheme, in contrast to prior localist models [9, 10] that use the mixture of experts scheme [11]. While each behavior primitive is individually stored in its corresponding local module in the mixture of experts scheme, in the FF-model multiple behavior primitives can be stored in an overlapping fashion in a single network utilizing a specific mechanism of behavior modulation. However, our study of the FF-model [8] did not examine well how the characteristics of the distributed representation actually affect the organization of the primitives.

The current paper will show that, in the distributed representation scheme, the behavior primitives can be generated and utilized more flexibly if they are organized in the forms of various attractor dynamics. This work seeks to answer the following questions. (1) How can different attractors with fixed point and limit cycling dynamics be self-organized simultaneously in the network through supervised learning of end-point and oscillatory behaviors? (2) Can the network generate diverse behavior patterns other than learned ones? If so, how do the novel behaviors correspond to the learned behavioral primitives stored in the model?

These issues are examined by conducting robot imitation learning experiments using a 4 degree of freedom robot arm.

2 Model

2.1 Overview

We explain the basic ideas behind our model first with an overview of the FF-model described in [8]. Although the original model is characterized by two levels of forward

models [12, 13] which interact with each other both in a top-down and bottom-up manner, the current study focuses only on the lower level forward model.

First, the diagram of the behavior generation is shown in Figure 1 (a). The lower level forward model receives the actual current sensory-motor state x_t as input and generates predictions of the next sensory-motor state $\hat{x_{t+1}}$ as output by utilizing the internal state c_t and the parametric bias p_t. Here, x_t, c_t and p_t are vectors. The sensory-motor sequences generated by the forward model as x_t are utilized for predicting the sensory inputs as well as generating the desired motor outputs. The internal state c_t is necessary in order to avoid sensory aliasing problems. (For example when drawing a figure "8", the internal state disambiguates which way to go at the crossing point.) The parametric bias plays the role of a behavior modulator in a manner analogous to parameter changes that cause bifurcations of nonlinear dynamical systems. A specific behavior pattern is generated while the parametric bias is clamped to a constant value. The value of the parametric bias is determined for each behavior primitive through prior learning processes (as will be described later) and its value is assumed to be stored at a higher level. The idea is that the parametric biases are keys for behavior primitives, sent one by one from the higher level to the lower level, by which corresponding behavior patterns are generated as sequences in the lower level. Although the previous work [8] utilized an RNN in the higher level to memorize such sequences of the parametric bias, the current work simplifies this part by storing the values of the parametric bias in a database.

Next, the recognition process, shown in Figure 1 (b), is discussed. In this process, the system recognizes manually guided behaviors that have already been learned. A sensory-motor sequence is generated while the robot arm is manually guided through a behavioral movement pattern. The problem is to find the value of the parametric bias needed to regenerate the given sensory-motor sequence. The parametric bias p_t can be determined by solving the inverse problem of the lower level forward model with regression of the sensory-motor sequences obtained through the guided behaviors. More specifically, the optimal parametric bias p_t is iteratively searched such that the forward model can regenerate the sensory-motor sequence with minimum error relative to the observed target sequence. In this computation, p_t is obtained under the condition that its temporal profile be generated as flat, or constant over time, as possible while a specific behavior pattern is being produced. On the other hand, p_t should change in a stepwise manner when the behavior pattern is changed. The higher level can categorize the pattern by comparing the obtained parametric bias to the ones stored in the database from prior learning. This is the process whereby given behavior patterns are recognized. However, the categorization process after obtaining the parametric bias

is not implemented in the current study.

The learning process goes through a set of manually guided behavioral patterns. The objective of learning is to find optimal connective weights which are responsible for all trained patterns and an optimal parametric bias for each trained pattern. Through the repeated learning of various guided behavior patterns, the lower level forward model adapts through synaptic changes such that each guided behavior pattern can be regenerated with the self-associating, corresponding parametric bias value. We will now describe each process in detail.

2.2 Neural network architecture

In order to implement the conceptual model described above, a neural network architecture based on the Jordan-type RNN [14] is developed. Figure 2 shows the network architecture. In this figure, the RNN receives the current sensory-motor state (s_t, m_t) and outputs their prediction as $(\hat{s_{t+1}}, \hat{m_{t+1}})$. This is the open-loop operation. In the closed-loop operation, copies of the current sensory-motor prediction outputs are fed back to the next sensory-motor inputs by the sensory-motor recurrent loop as shown by the dashed line in Figure 2. This feedback enables look-ahead prediction for an arbitrary number of future steps without perceiving actual sensory-motor inputs. The context units in the input layer represent the internal state c_t. The current internal state c_t is mapped to that in the next time step c_{t+1} that is represented by context units in the output layer. The internal state is recursively computed for future steps utilizing the recurrent feedback loop for context units. There are parametric bias units in the input layer. The parametric bias values are additional network variables that can be manipulated for learning and generating diverse behavior patterns. When a specific behavior is generated, the parametric bias units are clamped to corresponding values as determined by the higher level. The modulation of the parametric bias values shifts the mode of the operation in the lower level network. On the other hand, in the processes of learning and recognition, the parametric bias values are iteratively computed utilizing the error between the target sensory-motor sequence and the predicted sequence.

(1) The learning process

In learning a target sensory-motor sequence, the temporal profile of the parametric bias p_t is computed dynamically in the RNN, corresponding to the sensory-motor profile in the sequence, while the synaptic weights in the RNN are modified. This learning process is conducted in an off-line manner.

The temporal profile of p_t in the sequence is computed via the back-propagation through time (BPTT) algorithm [15], utilizing a working memory that stores the se-

quences of the parametric biases, the teaching targets, and the sensory-motor outputs. For the purpose of updating the parametric bias values the output error signal is back-propagated going through the hidden layer to the input layer where the parametric bias units are allocated. In this computation ρ_t, the internal value of the parametric bias, is obtained first. The internal value ρ_t changes its value due to the force computed by means of the error back-propagated to this parametric bias unit integrated for a specific step length in the sequence. Then the parametric bias, p_t, is obtained by the sigmoid output of the internal value. The utilization of the sigmoid function is just a computational scheme in order to limit the value of the parametric bias to be between 0.0 and 1.0. In this way, the parametric bias is updated in the direction of minimizing the error between the target and the output sequence. The step length of a sequence is denoted by L. For each learning iteration, L steps of look-ahead prediction, starting from the initial sensory-motor state, are computed by the forward dynamics of the RNN through a closed-loop operation. Once the L steps of the prediction sequence are generated, the errors between the teaching targets and the prediction outputs are computed and then back-propagated through time. The error backpropagation updates both the values of the parametric bias at each step and the synaptic weights. The update equations for the ith unit of the parametric bias at time t in the sequence are:

$$\delta {\rho_t}^i = k_{bp} \cdot \sum_{step=t-l/2}^{t+l/2} {\delta_t^{bp}}^i + k_{nb}(\rho_{t+1}^i - 2\rho_t^i + \rho_{t-1}^i) \quad (1)$$

$$\triangle \rho_t^i = \epsilon \cdot \delta {\rho_t}^i + \eta \cdot \triangle \rho_{t-1} \quad (2)$$

$$p_t^i = sigmoid(\rho_t) \quad (3)$$

In Eq. (1), $\delta\rho_t$, the delta component of the internal value of the parametric bias unit, is obtained from the summation of two terms. The first term represents the summation of the delta error, ${\delta_t^{bp}}^i$, in the parametric bias units for a fixed time window l. ${\delta_t^{bp}}^i$, which is the error back-propagated from the output units to the ith parametric bias unit, is summed over the period from $t-l/2$ to $t+l/2$ time step. By summing the delta error, the local fluctuations of the output errors will not affect the temporal profile of the parametric bias significantly. The parametric bias should vary only with structural changes in the sensory-motor sequence. Otherwise it should become flat, or constant, over time. The window step length, l, is taken as 10 steps in the experiment which is close to the time constant of the end-point behavior and the cyclic behavior in the training set. Our preliminary experiments revealed that the parametric bias tends to fluctuate if l is set to fewer than 10 steps. The step length of each training sequence, L, varies from 20 to 60 steps.

The second term plays the role of a low pass filter through which frequent rapid changes of the parametric bias are inhibited. k_{nb} is the coefficient for this filtering effect.

ρ_t is updated based on $\delta\rho_t$ obtained in Eq. (1). The actual update $\triangle\rho_t$ is computed by utilizing a momentum term to accelerate convergence as shown in Eq. (2). Then, the current parametric bias p_t is obtained by means of the sigmoidal outputs of the internal values ρ_t in Eq. (3).

(2) The sensory-motor pattern generation

Once the synaptic weights in the RNN are determined through the learning process, the sensory-motor patterns can be generated either in an open-loop or a closed-loop mode upon receiving the parametric bias from the higher level. In the open-loop mode, the RNN forward dynamics generate the sensory-motor value for the next time step when the network receives the current actual values. In the closed-loop mode, the sensory-motor prediction outputs for the next time step are fed-back into the current step inputs, allowing look-ahead prediction of sensory-motor values for arbitrary future steps without inputs of the actual sensory-motor values. Thus, the closed-loop mode generates "imaginary" motor patterns.

(3) Recognition of the sensory-motor pattern

As noted previously, the recognition of a given sensory-motor sequence is an inverse problem of finding the optimal p_t which regenerates the sequence in the closed loop mode under a smoothness constraint. This recognition process utilizes fixed synaptic weight values which were obtained in the learning process. Eq. (1), utilized during learning, is also used to update p_t in its iterative search computation during recognition. (Note that the synaptic weights are not updated in this recognition process.)

3 Robot Experiments

The proposed model was examined in the context of imitation learning using a robot arm. The robot used in the experiments has 4 degrees of freedom in its arm rotational joints. A hand attached to the arm can sweep over the task table horizontally as shown in Figure 3. A colored mark is attached to the top of the hand for video image processing. The robot is equipped with a color video camera by which positions of the hand can be viewed using color filtering. A handle is attached to the hand so that a trainer can teach behavior to the arm manually. A unix-based computer is connected to the robot controller by a serial line by which the neural network computation is conducted on-line or off-line depending on the operation modes described below.

The robot system is operated in four different modes: a manual guidance mode, an off-line learning mode, a behavior generation mode, and a behavior recognition mode. In the manual guidance mode, the arm is guided manually with the handle. All motors are set to free-run and the video camera to on. The sensory-motor sequences, in the

form of visually processed inputs and motor positions (angular positions of four motors) sampled by encoders, are recorded. In the off-line learning mode, training of the network is conducted for a set of sensory-motor sequences that have been recorded during the manual guidance mode. After the learning is completed, the synaptic weights as well as the parametric bias for each training sequence are saved. In the behavior generation mode, the parametric bias is set with a value that was obtained during learning of a specific pattern. Then the open-loop forward computation of the network is conducted on-line for regenerating the pattern. The parametric bias can also be set arbitrarily if novel pattern generation is attempted.

In the recognition mode, the arm is manually guided through a sequence of movements while all motors are set to free-run. The sensory-motor sequence generated during this manual guidance is recorded once. The network then computes, off-line, the optimal value of the parametric bias needed to generate that sensory-motor sequence based on the previously learned biases.

In the following experiments, we will address the following issues. (1) How can different dynamic structures, corresponding to end-point and cycling behaviors, be learned in the network? (2) How much can the trained network be adapted to novel behavior patterns based on learned ones? (3) What type of mapping between parametric bias values and behaviors can self-organize?

3.1 Training of end-point and cycling behaviors

The RNN was trained with three end-point behaviors and two cycling behaviors which were arbitrarily chosen. The end-point behavior patterns are manually generated by starting to move the arm at one position and stopping at another. The cycling patterns are generated by moving the arm back and forth between two specific positions while all four motor positions cycle with a constant period. The trajectory in the four-dimensional motor coordinate systems would be seen as circular. (Figures 4(a) and (b) show typical end-point and cycling behaviors of the arm robot associated with their trajectories shown in the 2-dimensional projection of the motor coordinate systems.) Each end-point in the three end-point behaviors is distant to each other and the two cycling behaviors are not correlated or similar. The training was conducted in a parallel manner for five sequences. The RNN had 20 hidden units and 8 context units. It also had 4 parametric bias units in the input layer. The BPTT learning for all the training sequences was iterated for 20,000 times starting from randomly set initial synaptic weights.

The learning results are summarized in Figure 5. The figure shows the recall process of the network after it has been trained for each target pattern. The plots in the top,

middle, and bottom rows in this figure show the change over time of four target motor outputs, the motor outputs learned by imitation, and four parametric bias values after learning, respectively.

Observe that the target motor patterns are imitated well for all sequences. Observe also that the parametric biases become mostly flat, or constant over time, in the latter half of each target sequence. One may conclude that each different end-point and cyclic behavior is learned by association with different parametric bias values.

We tested the robot's ability to successfully regenerate each trained behavior pattern by setting the corresponding parametric bias value. In this behavior regeneration test, the parametric bias values were sequentially switched from those obtained for one cyclic behavior to those for another cyclic behavior, and then to those for an end-point behavior. This sequential switching of the parametric bias was done manually in the current experiment, although it could be done by using a higher level RNN, as shown in the previous study [8]. Figure 6 shows motor pattern generation in the open-loop mode over time and the corresponding parametric bias values in the top and bottom rows, respectively. Observe that the trained behavior patterns appear one by one, corresponding to the switching of the parametric bias values. From these results, one may conclude that different dynamic structures, corresponding to end-point and cyclic behaviors, can be learned simultaneously in a single RNN by changing the parametric bias values.

3.2 Recognition and adaptation to learned or modulated patterns

The trained network's abilities to recognize learned patterns as well as to adapt to patterns modulated from the learned ones were examined next. As previously described, recognition is a process of finding an optimal parametric bias to regenerate a given, previously learned target pattern. An interesting question is how much the same network can "recognize" or adapt to modulated patterns by changing the parametric bias, but without changing the synaptic weights. To investigate this question, we prepared sets of test patterns by modulating one of the trained patterns. One of the cyclic patterns used in the previous training was modulated in two ways – first its period and then its amplitude was varied. (It is noted that the period and the amplitude are defined for the cyclic trajectory in the four-dimensional motor coordinate.) In the first target set, three modulated patterns were generated by increasing the period of the original training sequence by 10, 20 and 30 percent for all motor outputs. Another three modulated patterns were generated in the second set by decreasing the amplitude of the original one by 10, 20 and 30 percent for all motor outputs. The ability of the network

to imitate the modulated patterns by adapting its parametric biases was then tested.

The experimental results are summarized in Figure 7. Figure 7 (a) shows the result of imitating the original training sequence. The plots in the top and bottom rows show the target and the imitated motor outputs, respectively. Observe that the motor outputs were successfully regenerated to follow the target. Figure 7 (b) shows the results of period modulation. Three plots are shown for the cases of 10, 20 and 30 percent increments of the target pattern's period relative to the original training sequence. Observe that the motor outputs generally follow each target pattern. The period of the regenerated motor outputs increases by 27 percent in the case of 30 percent target modulation. However, Figure 7 (c) shows worse results in the case of amplitude modulation. Three plots are shown for the cases of 10, 20 and 30 percent amplitude decrements in the target pattern relative to the original training sequence. Note that the motor outputs cannot follow each modulated pattern. The amplitude in the regenerated motor outputs decreases by only 5 percent in the case of a 30 percent decrease in the target amplitude. Figure 8 (a) and (b) show the change of the parametric bias as period and amplitude are modulated. Four values of the parametric bias change monotonically relative to the modulation rate of the target in both cases.

The experimental results obtained so far indicate that the network recognizes patterns of previously trained movements and can also adapt to some modulated patterns but not to all. This result is a natural consequence of the employed scheme since the network maintains 4 degrees of freedom for adaptation, as is determined by the number of the parametric bias units. However, this adaptation cannot cover all possible modulations in the target patterns. At this point an essential question is what type of mapping is generated between the parametric bias and the behavioral patterns. The following experiment will address this question.

3.3 Examination of mappings between parametric biases and behaviors

In this experiment, we examined how behavior patterns were modulated as the values of the parametric biases were gradually changed. The parametric bias values were initially set as (0.32 0.77 0.41 0.88), which are the exact values obtained for learning the second cyclic behavior in the training set. Then the behavior patterns were generated while the third value of the parametric bias was incremented from 0.0 to 1.0 with a 0.2 step size at each behavioral trial. Figure 9 shows the generated behavioral patterns corresponding to each parametric bias value. Observe that the behavior patterns can be modulated significantly even with small changes of the parametric bias, although they are less sensitive to change in different ranges of parametric bias.

In order to clarify the detailed structures of the mappings between the parametric bias and corresponding behavior characteristics, phase analyses of the parametric bias space were conducted for the RNN learned. Since our preliminary examinations revealed that the network generally converges to either a fixed point or to limit cycling after 100 time steps of the forward computation (although non-periodic oscillations are sometimes generated), we decided to characterize the dynamic system in the parametric bias space by period and amplitude if it converges to limit cycling. Motor sequences of 200 time steps were generated by the forward dynamics in the closed-loop mode while the third and fourth values of the parametric bias were gradually changed with a 0.1 interval. Upon looking at the sequence of the motor output activation of the RNN from the 100th time step to 200th time step, the period and amplitude were measured if the dynamics converged into a limit cycle. In this analysis, if all 4 motor output values that correspond to 4 angular joint positions happen to cycle during the sequence, limit cycle dynamics are generated. On the other hand, fixed point dynamics occur if all the motor output values stop changing during the sequence. In the case of limit cycle, the observed step length of one cycle is taken as its period and the difference between the maximum and the minimum values of the first motor output during the cycle is taken as its amplitude. If the dynamics fluctuated without showing any periodicity within this sample sequence, its period was assumed to be more than 100 steps for convenience, although this could be a case of fixed point dynamics having very slow convergence. In this situation the amplitude was computed by taking the difference between the maximum and the minimum values of the first motor output in the sample period.

Figure 10 shows the generated phase plots for two values of the parametric bias (P_2, P_3) where the regions of fixed point dynamics and limit cycling are indicated by hatching and graded colors, respectively. In Figure 10(a) amplitude is indicated by grading from black to white as mapped from 0.0 to 1.0 for each combination of two parametric bias values. (It is noted that the region of fixed point dynamics is indicated by hatched tiles rather than the black ones.) In Figure 10(b) the period is indicated by its log scale measure. The color mapping of each tile is black for 1 and white for 100 steps. (The period could be more than 100 steps for white regions.) The hatching region, which is the same shape as the one in Figure 10(b), again represents the region of fixed point dynamics.

An important observation by looking at these two plots is that the landscape of dynamic characteristics represented by the periodicity and the amplitude is quite rugged in some regions while it is smooth in other regions. This observation suggests that the mapping between the parametric bias and the characteristics of generated behaviors is

quite nonlinear. This result contrasts with the one obtained by Haruno et al. [16] utilizing a localist scheme, the MOSAIC model [10]. They [16] showed that novel behavior patterns can be generated by means of linear interpolations among outputs of local forward model networks by adjusting their gate values. In their localist model, linear mapping occurs between the gate values and the generated behavior patterns. On the other hand, in our model diverse behaviors can be generated quite easily utilizing a self-organized, nonlinear mapping.

4 Discussions and Summary

Our experiments with a robot arm have shown that behavior primitives, based on different dynamic structures of fixed points and limit cycling, can be learned simultaneously in the proposed FF-model. It was further shown that nonlinear mappings are generated between the parametric bias and corresponding behaviors, by which diverse behavior patterns can be generated.

Although it is true that varying a parametric bias with limited degrees of freedom cannot generate every possible movement pattern modulation, this study suggests that a network which has been trained with only a finite set of patterns could potentially generate a large number of novel patterns. When the system is required to generate novel trials in certain situations, it can explore the diversity of behavior patterns just by modulating the parametric bias internally. If certain desired behavior patterns are found during such "internal" explorations, those patterns can be memorized for future use just by storing the corresponding values of the parametric bias in a higher level memory. The essential idea is to look first at the possibility of finding desired patterns that have already been organized implicitly in the memory structure of the network. If the "internal" explorations do not yield a satisfactory match, then supervised retraining of the network with different training sequence patterns might be required. Although some may argue that diverse novel patterns can be generated simply by using a random generator, such a scheme would not always be beneficial. Additional learning of randomly generated patterns (if they are found to be worth memorizing) could be sometime hard and take longer learning iteration steps since the patterns would not be correlated with the current memory contents. Therefore, such learning might cause memory interference problems with previously learned contents. In contrast, the novel patterns to be utilized in our internal exploration scheme exist already, hidden in the current memory structure, and therefore have no additional learning costs. One possible application for this movement learning system would be "user-development" of pet robots in the field of entertainment robotics [17]. Users would teach a robot a

number of behavior patterns. After learning them the robot would start to generate various behavior patterns, of which some might be quite novel and amusing for the users. If the user did not like the current types of movement patterns generated, they could re-train the robot with additional teaching patterns in the hope that the resulting modulation of the synaptic weights altered the pattern generation characteristics. The interleaving mechanism between internal exploration and supervised training is an area for future research.

Discussions continue about whether motor primitives should be represented locally or in distributed networks. Tani and Nolfi [9] as well as Wolpert and Kawato [10] described a localist model in which complex behaviors could be decomposed into sets of reusable behavioral patterns, each of which was stored in a specific, local neural network module. Our FF-model contrasts with this localist view in that various behavior primitives are represented in a distributed manner in a single RNN, where all neurons and synaptic weights participate in representing all trained patterns. These two types of models differ in how the memories of behavior patterns interfere with each other. In the localist network architecture, a novel pattern can be learned by allocating an additional network module. In this way, interference can be minimized between novel patterns and previously memorized patterns. However, in a distributed representation, memory interference would occur since the memories share the same network resources. Nevertheless, as a result of embedding multiple attractors in a distributed network, a global structure that accounts for learned patterns as well as unlearned patterns emerges.

The characteristics of the internal structure that emerges through learning in the proposed network seems to depend on interrelations among the training sequence patterns. Our studies [18] have recently shown that the mapping between the parametric bias and generated patterns becomes smooth and mostly linear when patterns in training sets are linearly correlated with each other. In this study, the network was trained with a set of similarly shaped patterns that differed in amplitude and period. Upon looking at the pattern generation capability after learning, it was shown that similarly shaped patterns were generated whose amplitudes and periods could be smoothly modulated with changes of two values of the parametric bias. This prior result, as well as the result in the current paper, indicate that the way the mapping self-organizes depends on the organization of the training sequence patterns. If all training sequence patterns are interrelated such that variations among the patterns can be accounted for by variables with a few degrees of freedom, then such relations can be reconstructed in the network by organizing a smooth and simple mapping between the parametric bias and patterns. On the other hand, if the patterns are all independent or cannot be

simply related, then attempts at reconstructing all the patterns in a small parametric bias space could generate arbitrary complex nonlinear mappings between the parametric bias and the patterns. In the latter case, generalization through learning cannot be expected since there are no general structures to be extracted from the training sequence patterns. Instead, the network will likely generate diverse patterns by utilizing internally organized nonlinearities. In the current experiment, the training sequence patterns consist of two sets of uncorrelated dynamics: fixed point dynamics and limit cycling dynamics. It is highly probable that the observed nonlinearities originated from the simultaneous training of these two distinct types of dynamic patterns.

Which is better, the distributed or the local representation of motor control? This is a trade-off problem. In the localist scheme, as shown in [16], stability of old behavior patterns during new learning can be gained. On the other hand in the distributed representation, the diversity in the behavior generation can be gained utilizing its potential nonlinearity of the system, but at the cost of losing memory stability. It is assumed that biological systems found optimal points between these two extremes. Our future research will focus on possible adaptation mechanisms by which the networks themselves can determine the degree of localization or distribution in their internal representation.

References

[1] E. Bizzi, N. Acornero, W. Chapple, and N. Hogan, "Posture control and trajectory formation during arm movements", *J. Neurosci.*, vol. 4, pp. 2738–2744, 1984.

[2] A. Feldman, "Superposition of motor programs, I. Rhythmic forearm movements in man", *Neuroscience*, vol. 5, pp. 81–90, 1980.

[3] H. Haken, J. Kelso, and H. Bunz, "A theoretical model of phase transition in human hand movements", *Biol Cybern*, vol. 51, pp. 347–356, 1985.

[4] M. Cohen, "The construction of arbitrary stable dynamics in nonlinear neural networks", *Neural Networks*, vol. 5, pp. 83–103, 1992.

[5] J. Kelso, "Elementary coordination dynamics", in *Interlimb Coordination: neural, dynamical, and cognitive constraints*, S. Swinnen, Ed. New York, Akademic Press., 1994.

[6] G. Taga, "A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance", *Biological Cybernetics*, vol. 12, pp. 1131–1141, 1996.

[7] S. Kotosaka and S. Schaal, "Synchronized robot drumming by neural oscillator", in *The International Symposium on Adaptive Motion of Animals and Machines, Montreal, Canada*, 2000.

[8] J. Tani, "Learning to generate articulated behavior through the bottom-up and the top-down interaction processes", Tech. Rep. RIKEN-BSI-BDC-TR2001-001, RIKEN, BSI, 2001, to appear in Neural Networks.

[9] J. Tani and S. Nolfi, "Learning to perceive the world as articulated: an approach for hierarchical learning in sensory-motor systems", in *From animals to animats 5*, R. Pfeifer, B. Blumberg, J. Meyer, and S. Wilson, Eds. Cambridge, MA: MIT Press., 1998, later published in Neural Networks, vol12, pp1131–1141, 1999.

[10] D. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control", *Neural Networks*, vol. 11, pp. 1317–1329, 1998.

[11] R.A. Jacobs and M.I. Jordan, "Adaptive mixtures of local experts", *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[12] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural network model for the control and learning of voluntary movement", *Biological Cybernetics*, vol. 57, pp. 169–185, 1987.

[13] M.I. Jordan and D.E. Rumelhart, "Forward models: supervised learning with a distal teacher", *Cognitive Science*, vol. 16, pp. 307–354, 1992.

[14] M.I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine", in *Proc. of Eighth Annual Conference of Cognitive Science Society.* 1986, pp. 531–546, Hillsdale, NJ: Erlbaum.

[15] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation", in *Parallel Distributed Processing*, D.E. Rumelhart and J.L. Mclelland, Eds. Cambridge, MA: MIT Press, 1986.

[16] M. Haruno, D. Wolpert, and M. Kawato, "MOSAIC model for sensorimotor learning and control", *Neural Computation*, vol. 13, pp. 2201–2220, 2001.

[17] M. Fujita and K. Kageyama, "An Open Architecture for Robot Entertainment", in *Proc. of the First Int. Conf. on Autonomous Agents*, 1997, pp. 435–442.

[18] M. Ito and J. Tani, "Analysis of generalization in dynamic pattern learning", submitted to IEEE Trans. on Neural Networks, 2002.

List of Figures

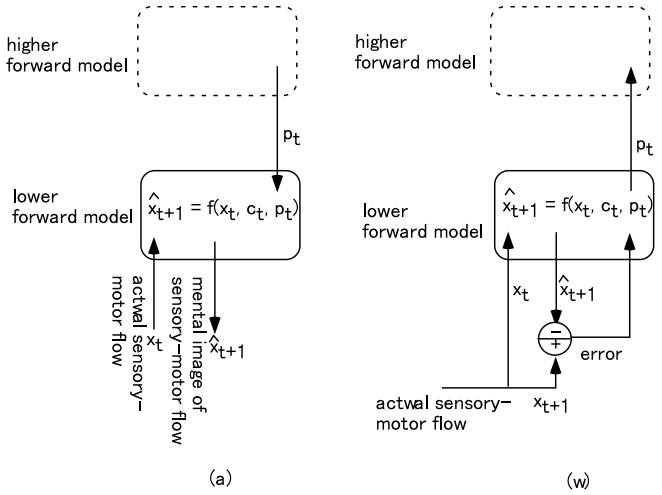

Figure 1: (a) and (b) illustrate the behavior generation and the recognition processes, respectively in the FF-model. The dashed boxes represent the higher level forward model which is not utilized in the current study.

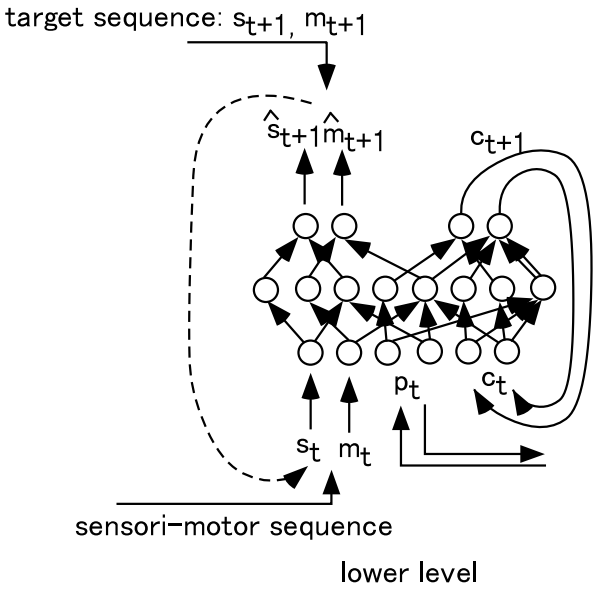

Figure 2: The RNN associated with the parametric bias inputs p_t. s_t and m_t are sensor and motor sequences and c_t is context unit activation.

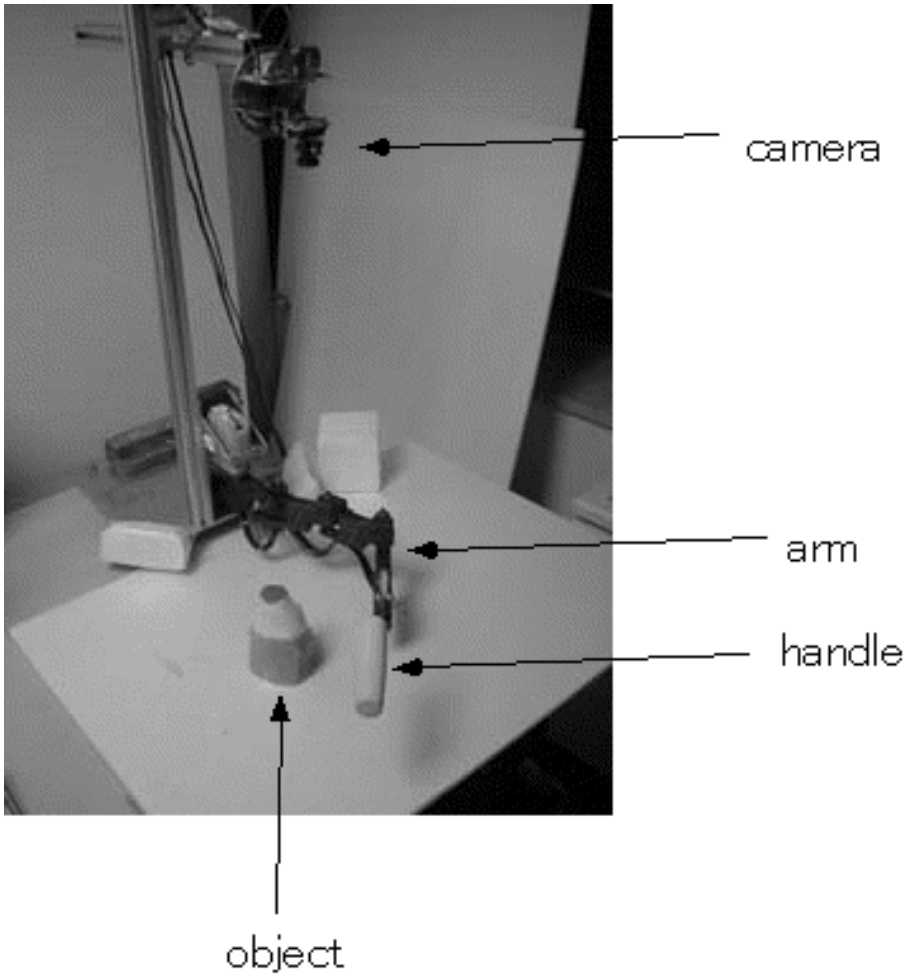

Figure 3: The arm robot used in the imitation learning experiments.

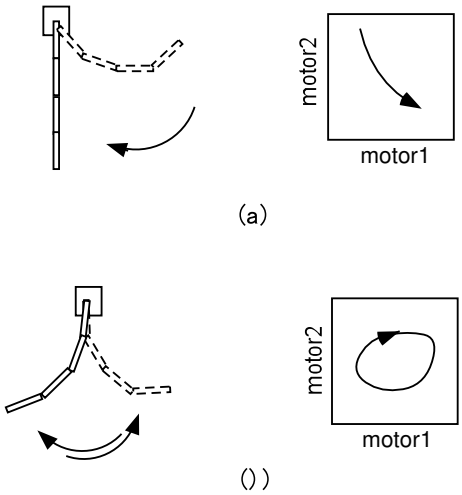

Figure 4: Typical (a) end-point and (b) cycling behavior of the arm robot associated with their trajectories shown in the 2-dimensional projection of the motor coordinate systems

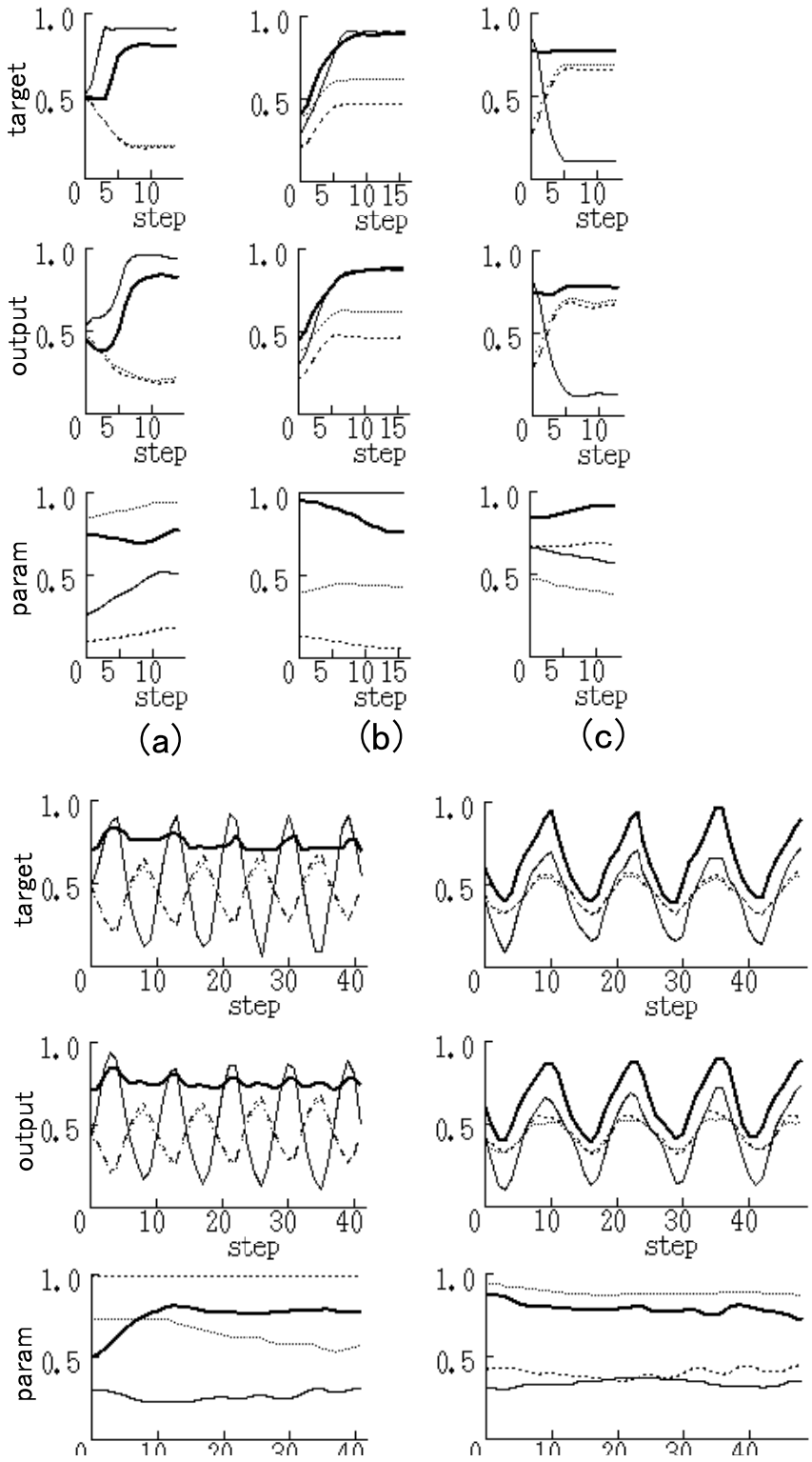

Figure 5: The results of training of three end-point behaviors in (a) (b) and (c) and two cyclic behaviors in (d) and (e). The change over time of the four target motor outputs, the imitated outputs, and four parametric bias values are shown in the top, middle, and bottom rows, respectively, for each pattern. Time steps are shown in the abscissa.

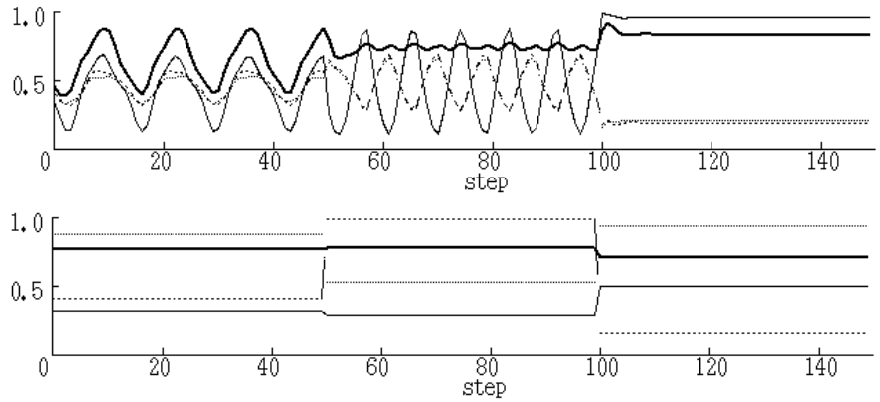

Figure 6: The results of generating two oscillatory movements followed by one end-point movement. The change over time of the motor outputs and the parametric biases are shown in the top and bottom rows, respectively. Time steps are shown in the abscissa.

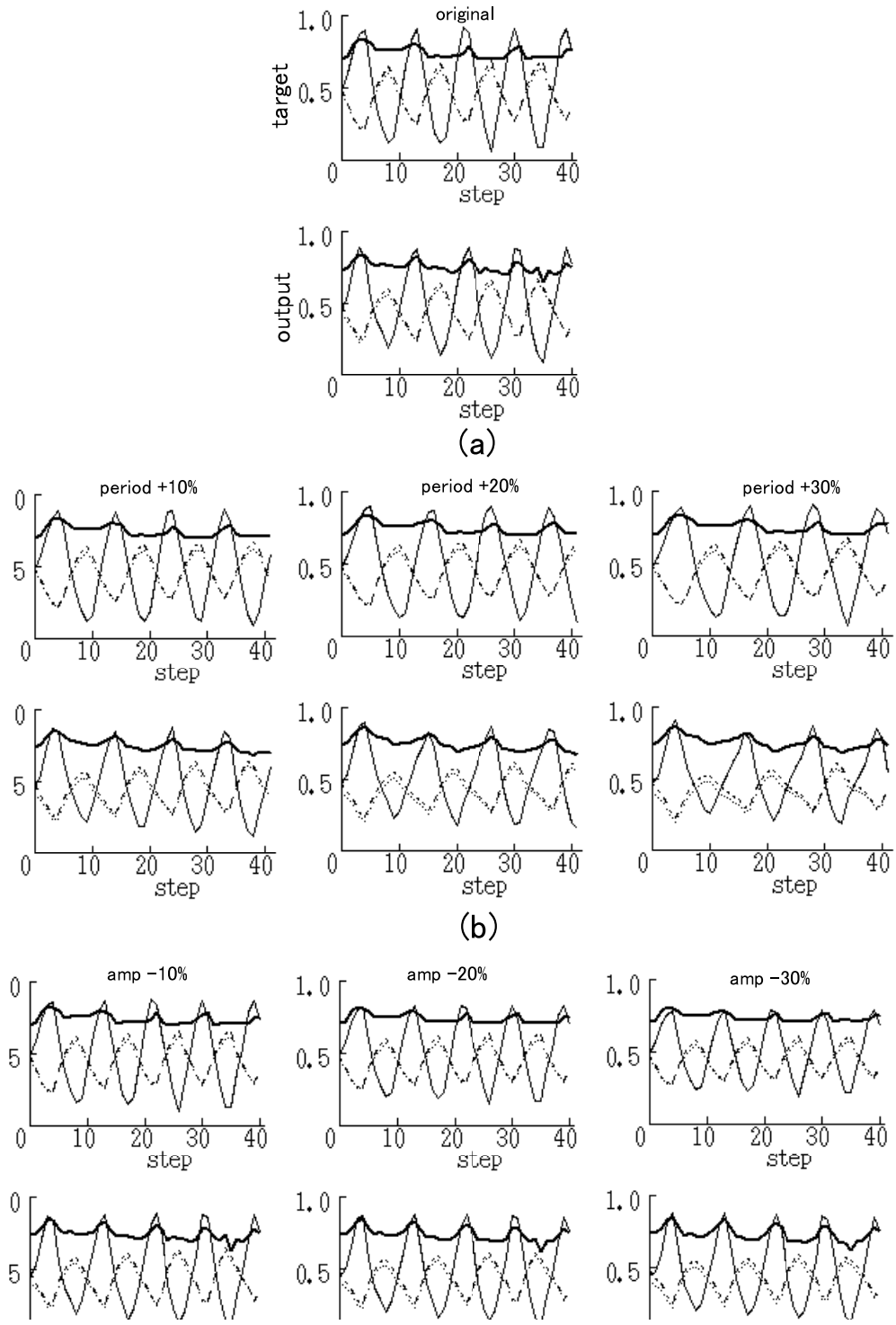

Figure 7: The results of adaptation to patterns modulated from learned ones. Comparison between the target motor output and the regenerated one is shown in (a). (b) shows the same comparison when period is increased from 10 percent to 30 percent in the target pattern. (c) shows the case of 10 percent to 30 percent decrease in the target amplitude.

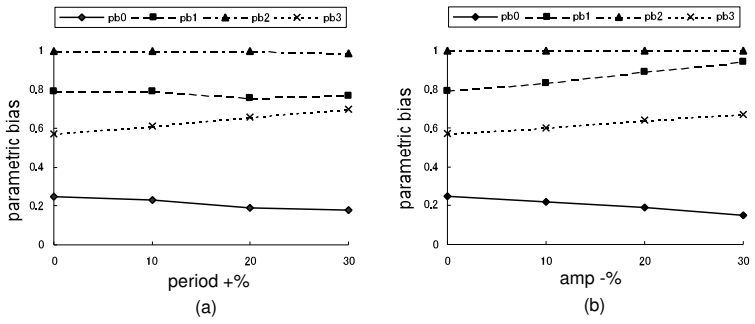

Figure 8: Modulation of target sequence period (a) and amplitude (b) versus corresponding parametric bias values.

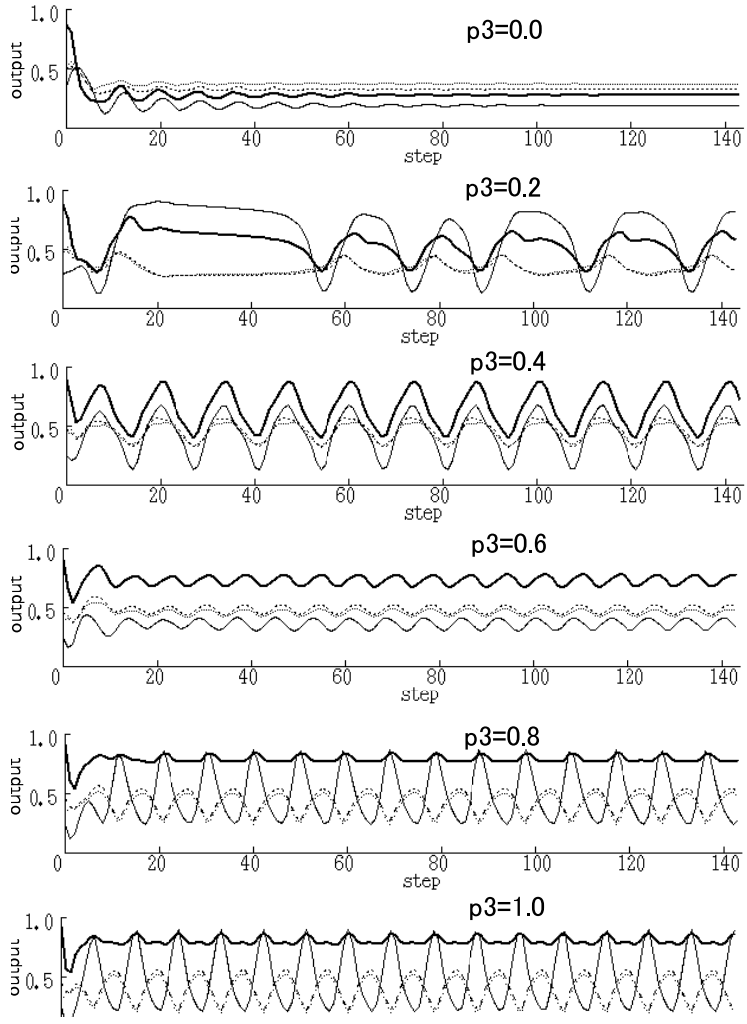

Figure 9: 6 motor activity patterns are plotted with the parametric bias values incrementally increased from top to bottom. Ordinate: Motor Output; Abscissa: Time Step.

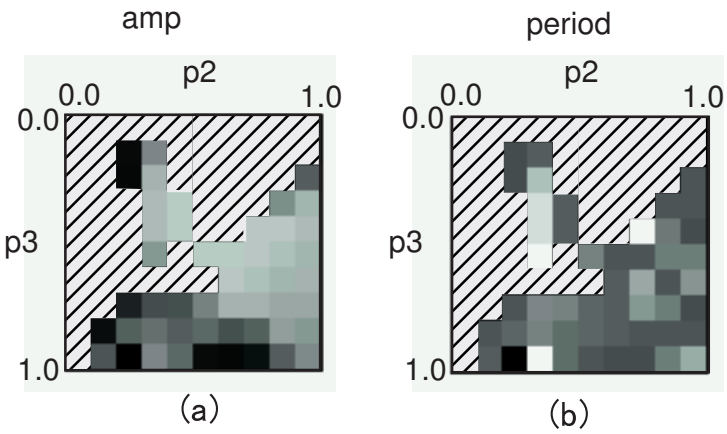

Figure 10: The phase plots for (a) the amplitude and (b) the period using two values of the parametric biases. In (a), amplitude is graded from black=0 to white=1. In (b), period is graded from black=1 to white=100 time steps. The hatched regions correspond to the region of a fixed point dynamics in both figures.