an Article from

Journal of Robotics and Mechatronics

7F Dai-ni Bunsei Bldg., 11-7, Toranomon 1-chome, Minato-ku, Tokyo 105, Japan
Tel: 03-508-0051 Fax: 03-592-0648

Papers

Motor Schema Model Learned by Structural Neural Networks

Toshio Tsuji*, Yusuke Ishida*, Koji Ito*, Mitsuo Nagamachi*

and Tatsuo Nishino**

* Faculty of Engineering, Hiroshima University
Higashi-Hiroshima-shi, 724 Japan
** Department of Industrial Engineering, Hiroshima Institute of Technology
Hiroshima-shi, 731-51 Japan

Human beings remember plans concerning typical motions which occur frequently as schema, and by selecting suitable schema depending on conditions, generate muscular motion almost unconsciously. Though a motor schema represents typical motions, it is equipped with superior plan structure taking into consideration the concurrency and seriality of motions as seen in grasping actions and walking motions, and the structure of plans can be acquired by learning. In this paper, a study is made of the modeling of such motor schema with the use of neural networks. For this purpose, the neural network is structured beforehand into the part which generates action sequences in the form containing concurrency (concurrent action generation part) and the part which modifies the action sequences to satisfy constraints which cannot be executed concurrently (constraint representation part). After learning in each part model the neural network can generate motion sequences while taking into consideration the seriality and concurrency of motion by combining the parts at the time of execution. Finally, this model is applied to the formation of typewriting action motor schema, and it is demonsted that generates motion sequences which take into consideration the constraint of the motion system accompanying the execution of motion.

1. Introduction

Human beings can perform actions having several simultaneous purposes. For example, in the case of conversation while eating, consciousness is oriented for conversation and the motion of eating is done almost unconsciously. This is because the plans concerning typical activities which take place frequently are remembered as motor schema. By selecting suitable schema depending on conditions at the time of motion and switching to meet them, muscular motion necessary for the action can be generated almost without conscious effort.

Of course, such unconscious action, though typical, has to be equipped with fairly complicated plan structures.[1] For example, when considering the plan of grasping a cup, if the hand is made into a form for grasping a cup during the reaching motion, effective execution is possible compared to the case when the next action is started after prior action is completed (**Fig.1(a)**). On the other hand, in case the cup is going to be placed on a table, the hand cannot come away from the cup until the motion of placing the cup has been completely finished (**Fig.1(b)**). Also, the dependent relation between each motion must be represented in the plan consisting of statically unstable motions such as walking in order to keep stability and feasibility.

As mentioned above, it is known that human motor schema skillfully assembles the concurrency and seriality of each motion to constitute a plan in compliance with the purpose of action and, in addition, the structure of a plan is formed by learning while typical activities are repeated many times.

In this paper, a motor schema model is studied with the object of applying it to control of robots which have autonomous motion capability. To realize such motor schema by learning, the use of a system in which concurrent mechanisms operate naturally like neural networks is considered more suitable than information processing which uses sequential symbol operation. A neural network has the following advantages:

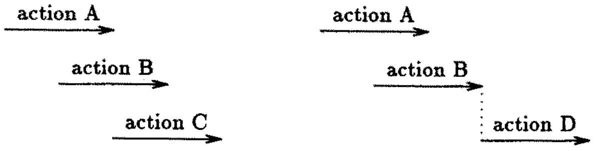
1) Realization of highly concurrent processing is possible;

2) Knowledge can be acquired naturally by learning; and

3) It is resistant to noise and failure

The neural network has been noted as a new concurrent decentralization type of information processing mechanism taking the place of serial symbol processing.[2,3] However, because of its highly concurrent mechanism, it cannot readily handle data which changes over time and is not suitable for the serial and logical information processing necessary for high-level inference.[4]

Therefore, in realizing motor schema with the use of



(a) An action sequence with concurrency

(b) The action D cannot be executed concurrently with the action B

Fig. 1. Action sequences with concurrency

neural networks, the following points become problematic:

1) How to describe the concurrency and seriality of motion;

2) How to generate motion sequences by time; and

3) How to represent constraints in problem regions.

In this paper, an approach for composing motor schema based on such neural networks is taken. For this purpose, neural networks are structured beforehand into the part which generate action sequences in the form containing concurrency (concurrent action generation part) and the part which modifies action sequences to satisfy constraints which cannot be executed concurrently (constraint representation part). After learning in each part, the neural network model can generate action sequences taking into consideration the seriality and concurrency of action by combining both parts at the time of execution. In Chapter 2, an outline of the conventional method proposed for serial and time-series processing with the use of neural networks is explained. In Chapter 3, a neural network for constituting motor schema is proposed. In Chapter 4, this model is applied to the formation of schema for typewriting action, and it is demonstrated that action sequences which take into consideration the constraint of action systems can be generated.

2. Serial Processing by Neural Networks

Conventionally, the following two methods were considered for serial and time-series processing by neural networks. The first is a method to develop time series in spatial patterns.[5] This method, using a unit which has a delay characteristic for unit time, prepares layers corresponding to each unit of time and transmits input to the next layer in sequence. However, this method requires the preparation of layers corresponding to the length of the time-series to be recognized, and the handling of excessively long time-series is not possible.

The other method is one in which a model has explicit internal state, and internal units are prepared for representing and keeping the internal state. By renewing the internal state using feedback from the output units and the internal units one unit time before, it acts as a kind of automata and handles time-series data.

Rumelhart et al.,[6] based on the latter approach, proposed a neural network composed of the interpretation network which determines action toward the environment and the world model which simulates the outer world (**Fig.2**). Here, the interpretation network interprets input and determines the action corresponding to the input. The world model, by simulating the environment, predicts what kind of change of environment is brought about by the determined action. By the feedback of the predicted result, it shows the possibility of executing sequential processing resembling human thinking.

Furthermore, Rumelhart and Norman[7] simulated typewriting action as a model to show that neural networks have the capability to do serial and sequential processing. Complying to the spelling of these words, they structured the neural network using units corresponding to the schema of the action of typing words and characters. In this model, the concurrency of actions (such as fingers moving in preparation for impending motion involved in typing the next character) can be represented by partial activation of the units corresponding to the next character and typing mistakes made by humans can also be reproduced by adding adding to the activation value. In this model, however, spelling is required to be given beforehand in the network structure.

On the other hand, Jordan[8] proposed a model which was able to generate learned time series containing concurrency by adding feedback and internal state to multi-layered neural networks. The composition of this model is shown in **Fig.3**. State unit group explicitly represents internal state. Therefore, time series can be generated by computing the next output using the internal state as input. Also, as the computation of output is basically the feed-forward type, the time series given as a teacher signal can be learned with the use of the error back propagation method.[5]

The advantage of this model is the capability of generating concurrent action sequences against the teacher signal which only gives sequential relations. This is because the state which is close in time has similar representation (the humming distance is close) by the interpolation capacity of the error back propagation method and by the work of the feedback loop. For example, the unit taught "1" at a certain timebecomes activated partially (for example, at 0.8) before and after that time. Furthermore, Miyata[8] modeled the learning history of motion sequences by combining Jordan's

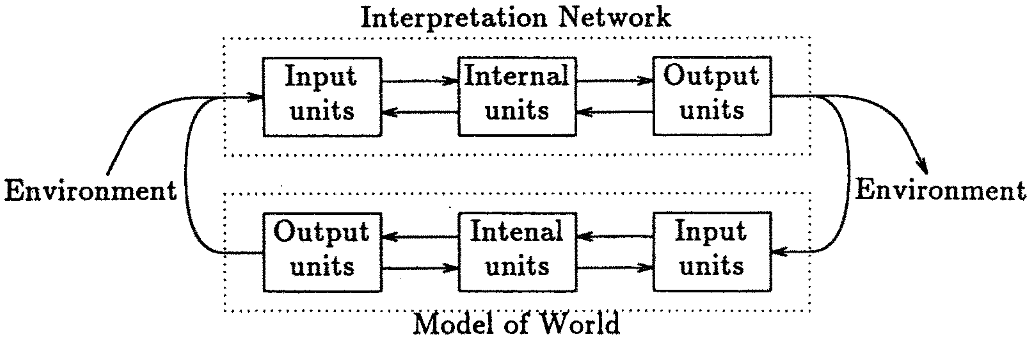


Fig. 2. The model of mental simulations [6]

model in two serial lines. Yet, in Jordan's model, the constraint of seriality which cannot br executed concurrently is explicitly given as a teacher signal. Therefore, even in the case of generating sequences for the same plan, re-learning becomes necessary when action constraints differ. Moreover, as the model uses the error back propagation as it is, learning is difficult for complicated action sequences and the setting of parameters contained in the model is extremely difficult.

Therefore, the network in this paper is structured beforehand into a part which generates action sequences which including concurrency (concurrent action generation part) and a part which modifies the sequences in compliance with the constraints required from the problem region (constraint representation part). Learning occurs for each part, and both parts are combined at the execution time.

In this method, the constraint representation part dependent on regions can be switched over as a module, and it becomes possible to use that part as the motor schema of other regions (other environment, other task condition and other purpose).

3. Structured Neural Network for Motor Schema

This model is composed of a concurrent action generation part and a constraint representation part (**Fig.4**). The concurrent action generation part learns the sequential relation of action sequences for a certain plan and generates action sequences which contain concurrency (refer to **Fig.1(a)**. The constraint representation part learns the constraints dependent on the problem region, gives inhibition to the output of the concurrent action generation part which does not satisfy the constraints, and modifies it to satisfy the constraints. The input having the form of representation, which is irrelevant to the region, is transformed into a form which corresponds to thedisired signal of the motion system.

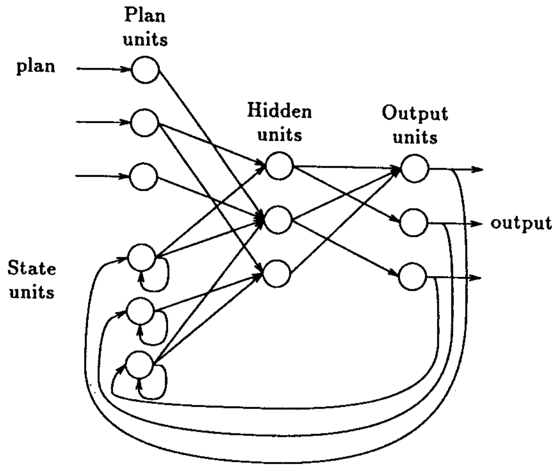


Fig. 3. The sequential network [8]

In this paper, a study is made of the problem described as follows:

1) There is no exchange of the sequential relation of each action. For example, the sequence of the letter line is given as the spelling of a word.

2) There are actions which can be executed concurrently and those which cannot. In typing, for example, plural letters assigned to the same finger cannot be typed concurrently.

3-1. Concurrent Action Generation Part

In this paper, a neural network with the same construction as Jordan's model[8] is used as the concurrent action generation part, where it is given only the sequential relation of actions by the teacher signal and the constraints for concurrent execution are not given. As the result, the concurrent action generation part comes to generate perfectly concurrent action sequences. Therefore the non-linearity of learning and the difficulty of setting parameters which Jordan's model entails are avoided to a certain extent.

Figure 4 shows the construction of the concurrent action generation part. The input to the plan unit group is given as a vector $\vec{p}$ (unit vector). Output vector element $x_{n,i}$ of output unit group at time n expresses the activation value of each action and takes the value from 0 to 1.

Output vector $\vec{x_n}$ at time n is obtained as the function of vector $\vec{p}$ and state vector $\vec{s_n}$ of the state unit group,

$$\vec{x^n} = \vec{f}(\vec{s_n}, \vec{p}) \quad \cdots\cdots\cdots\cdots\cdots\cdots (1)$$

where the function $\vec{f}$ is represented by the multi-layered network from the plan unit and condition unit groups to the output unit group. The action of each unit is defined as follows. Here, $u_{n,i}$, $v_{n,i}$ are the input and output of unit i; w_{ij} is the weight of the link from unit j to unit i.

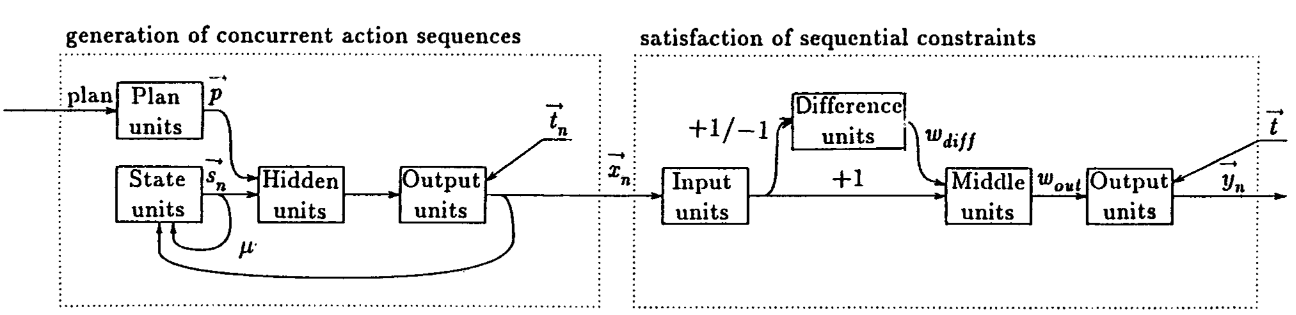


Fig. 4. The structured neural network for motor schemas

$$u_{n,i} = \begin{cases} p_i & \text{(plan unit)} \\ s_{n,i} & \text{(state unit)} \\ \sum_j w_{ij} v_{n,j} & \text{(hidden unit, output unit)} \end{cases}$$

. . . . (2)

$$v_{n,i} = \begin{cases} u_{n,i} & \text{(plan unit, state unit)} \\ \dfrac{1}{1+e^{-u_{n,i}}} & \text{(hidden unit, output unit)} \end{cases}$$

. . . . (3)

In other words, each unit of the plan unit group and state unit group outputs its input as it is, and each unit of hidden unit group and output unit group outputs activation value in accordance with Sigmoid-type characteristic functions.

State $\vec{s_n}$ of the state unit group is renewed as the following equation with the use of feedback from itself and output from one unit time before,

$$\vec{s}_{n+1} = \mu \vec{s_n} + \vec{x_n} \quad \dots\dots\dots\dots\dots (4)$$

where, $\vec{s_o} = [1, 1, \dots\dots, 1]^T$, and μ is the weight of the state unit's self feedback loop.

The learning rule of this model is based on the error back propagation method;[5)] and, as it contain a feedback loop and state units, the following method is used:

1) The weight of each link is initialized by random num bers.
2) The procedures of the following (2a) to (2f) are taken for each plan.
(2a) Plan input $\vec{p}$ given to the plan unit group and at the same time, the state unit group is initialized.
(2b) Output $\vec{x_n} = \vec{f}(\vec{s_n}, \vec{p})$ is calculated .
(2c) Error back propagation learning is performed with the use of output $\vec{x_a}$ and a teacher signal t_n for the time n and the weight of link w_{ij} is renewed.

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial v_{n,i}} \frac{dv_{n,i}}{du_{n,i}} v_{n,j} \quad \dots\dots\dots\dots (5)$$

Here, for the output units,

$$\frac{\partial E}{\partial v_{n,i}} = \begin{cases} v_{n,i} - t_{n,i} \quad (t_{n,i} = 0, 1) \\ \max(v_{n,i} - 0.8, 0) \\ \quad (t_{n,i}:\ \text{imperfect activation value}) \end{cases} \qquad (6)$$

and, for the hidden units,

$$\frac{\partial E}{\partial v_{n,i}} = \sum_k \frac{\partial E}{\partial v_{n,k}} \frac{dv_{n,k}}{du_{n,k}} w_{ki} \quad \dots\dots\dots\dots (7)$$

If $t_{n,i}$ is the teacher signal which represents imper fect activation, learning is made so that output becomes less than 0.8.

(2d) $\vec{x_n} = \vec{f}(\vec{s_n}, \vec{p})$ is obtained with the use of the new w_{ij}.
(2e) Internal state $\vec{s}_{n+1}$ is renewed by equation (4).
(2f) (b) to (e) are repeated until the sequence of the teacher signal for the plan comes to an end. During the interval, $\vec{p}$ is kept fixed.

3) Among the plans learned in (2), the learning procedures (a) to (f) are applied again to the one having the largest error (reinforced learning).

4) 2) to 3) are repeated for all the plans, times and actions until error becomes smaller than ε.

Here, the following values are used as teacher signals.
(i) Action to be perfectly activated in plan l, time n:

$$t^{(l)}_{n,i} = 1$$

(ii) Action contained in plan l:

$$t^{(l)}_{m,i} = *(\text{imperfect activation value}) \quad (m \neq n)$$

(iii) Action which does not appear in plan l:

$$t^{(l)}_{m,i} = 0 (m = 1, 2, \cdots\cdots, N)$$

(iv) All the actions at the time of starting and finishing plan:

$$t_{m,i} = 0 \ (m = 1, N)$$

For example, as for the plan of "abc", the sequence of the teacher signal shown in **Fig.5** is given. The mark * in the figure shows imperfect activation value.

3-2. Constraint Representation Part

When a motion such as typewriting action is considered, the action of typing some letter allocated to the same finger have the constraint of impossibility of concurrent execution. In other words, in the action group corresponding to the same output organ of the motion system, it is necessary that the action which has the strongest activation value be executed and other actions inhibited.

As such a constraint (different from the concurrent action generation part) need not give consideration to dynamics by time, it can be achieved by the mapping of multi-layered networks. However, it is highly likely to take time for learning and of having the learning fall into the local minimum, as the nonlinearity of mapping is considerably strong. Therefore, to make learning efficient it is separated into a part which gives constraint and one which makes the transformation of representation; the network is structured into each part beforehand. **Figure 4** shows the composition of the constraint representation part. In the figure, the constraints from the problem region is represented by the mapping from an input unit group to a middle unit group, and the transformation of representation is made by mapping from the middle unit to the output unit group.

A difference detection unit group is prepared for all the input unit pairs, and the difference diff_{ij} of activation values x_i and x_j of the unit pairs is detected.

$$\begin{array}{c} m= \\ a: \\ b: \\ c: \\ d: \end{array} \begin{array}{c} 1 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} 2 \\ \begin{bmatrix} * \\ * \\ * \\ 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} 3 \\ \begin{bmatrix} 1 \\ * \\ * \\ 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} 4 \\ \begin{bmatrix} * \\ 1 \\ * \\ 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} 5 \\ \begin{bmatrix} * \\ * \\ 1 \\ 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} 6 \\ \begin{bmatrix} * \\ * \\ * \\ 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} 7 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array}$$

Fig. 5. An example of teacher singnals for concurrent action generation part

$$\mathrm{diff}_{ij} = \begin{cases} 0 & (\text{if} \quad x_i \leq x_j) \\ 1 & (\text{if} \quad x_i > x_j) \end{cases} \quad \cdots\cdots (8)$$

The middle unit group forms subgroups corresponding to the output unit. If there is a unit with a stronger activation value than other units in the same sub group, other units receive constraints from the difference detection units through w_{diff},

$$x_k' = \min(x_k + \sum_{i,j} w_{\mathrm{diff}k,ij} \mathrm{diff}_{ij}, 0) \quad \cdots\cdots (9)$$

Where, $W_{\mathrm{diff}\ k,ij} \leq 0$ is the weight of link from difference detection unit ij to the middle unit k and represents the constraint among the actions in the sub group. The output unit group receives x_j' through w_{out}, and outputs the sum y_i

$$y_i = \sum_j w_{\mathrm{out}ij} x_j' \quad \cdots\cdots (10)$$

The action of the constraint representation part is explained with the use of **Fig.6**. Action a and action b correspond to the same unit O_1, and action c corresponds to the other output unit O_2. Here, the activation values of each action given by the concurrent action generation part to input unit are a=1.0, b=0.8 and c=0.6. In this case, a-b, a-c, b-c become activated out of the difference detection units, and middle unit b′=0. Yet, there is no constraint between unit a′ and c′ as the link is not connected (w_{diff}=0), and the input value is transmitted to the activation value of middle unit as it is. These activation values are then transmitted to the output unit through link w_{out}.

As for the learning rules of the network, the following method is used:

(1) w_{diff}, w_{out} are set to 0.

(2) Procedures (2a) to (2b) are applied to all actions which are output from the concurrent action generation part.

(2a) Unit vector $\vec{x^{(c)}}$, which represents action c, is given to the input unit group, and teacher signal $\vec{t^{(c)}}$ for c is given to the output unit group.

(2b) w_{out} is modified by the use of equation (11). At this time, $\vec{x}' = \vec{x^{(c)}}$ as w_{diff} are all 0. By this modification, the middle units are subgrouped implicitly.

$$\Delta w_{\mathrm{out}ij} = x_j' t_i \quad \cdots\cdots (11)$$

(3) The procedures (3a) to (3c) are applied to all the actions.

(3a)$\vec{x^{(c)}}$ is given to the input unit group, and output $\vec{y}$ is calculated.

(3b) y_i at this time is back-propagated to the middle unit through $w_{\mathrm{out}\ ij}$.

$$d_i = \sum_j w_{\mathrm{out}ij} y_j \quad \cdots\cdots (12)$$

(3c) Here, action i (which becomes $d_i \neq 0$) belongs to the same subgroup as action c, and thus has the possibility of being inhibited by action c. To represent the constraint, $w_{\mathrm{diff}\ k,ij}$ is renewed according to the following equation.

$$\Delta w_{\mathrm{diff}k,ij} = \begin{cases} -1 & (\mathrm{diff}_{ij} d_k \neq 0) \\ 0 & (\mathrm{diff}_{ij} d_k = 0) \end{cases} \quad \cdots\cdots (13)$$

By the above procedures, the output of the concurrent action generation part is modified, and the transformation to the representation corresponding to the desiored value of the motion system becomes possible.

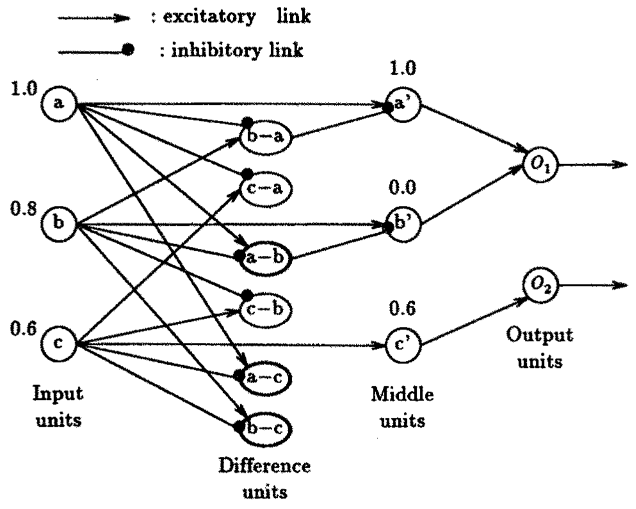


Fig. 6. The network structure for the constraint representation part

4. Application to Typewriting Action

As an application example of the proposed model, the generation of motor schema concerning typewriting action was attempted.

Table 1 shows the input/output of typewriting action. As for input of the plan of the concurrent action generation part, M kinds. of words are assumed. The output action of the concurrent action generation part (which becomes the input of the constraint representation part) is to be the action of typing a certain letter (26 alphabetical letters). As for the constraint representation part output and teacher signal, the target position in the task coordinates of each finger and key hitting instruction considered. Here, as forefingers require x and y coordinates and fingers from middle fingers to little fingers require only a y coordinate, a total of ten positions for right and left hands are output. **Table 2** is the alphabet corresponding to each finger. The letter corresponding to one finger cannot act concurrently. It should be noted that the motion control of moving fingers to the target position is not treated here. Key hitting instruction units represent the timing of typing by fingers. When the activation value of this typing unit becomes 1, actual typing action is generated. Here, the parameters of the concurrent action generation part is μ=0.8, η=1.0, ε=0.01, and 52 pieces of hidden units were prepared.

A part of the simulation results with the numbers of words M=10 is shown hereunder. The words used are 10 German numerals, "ein", "zwei", "drei", "vier", "fuenf", "sechs", "sieben", "acht", "neun", and "zehn". First, the outputs for a word "neun" after learning without reinforced learning (refer to 3.1) in the concurrent action generation part are shown in **Fig.7**. **Fig.7(a)** shows the output unit of the concurrent action generation part, **(b)** shows the middle unit of the constraint representation part and **(c)** shows a part of the output unit of the constraint representation part.

In Fig.7(a), the activation value of unit 'n' is 1 four times in succession. As a result of this, in the middle unit group of the constraint representation part, unit 'n' inhibits unit 'u' (Fig.7(b)). Therefore, a wrong typing action sequence is generated(Fig.7(c)).

Simulation was repeated with the use of reinforcement learning based on the learning rule of the concurrent action generation part. **Figure 8** shows the result of learning (about 6000 times). As the result of reinforcement learning, the output unit 'n' of the concurrent action generation part decreases once after it takes the largest activation value, and finally takes the largest value again (**Fig.8(a)**). Therefore, in the middle unit group of the constraint representation part, the part where unit 'u' inhibits unit 'n' appears contrary to Fig.7(b), and, as a result, the correct typing action sequence is output. As for the other 9 words, correct typing action sequences were obtained.

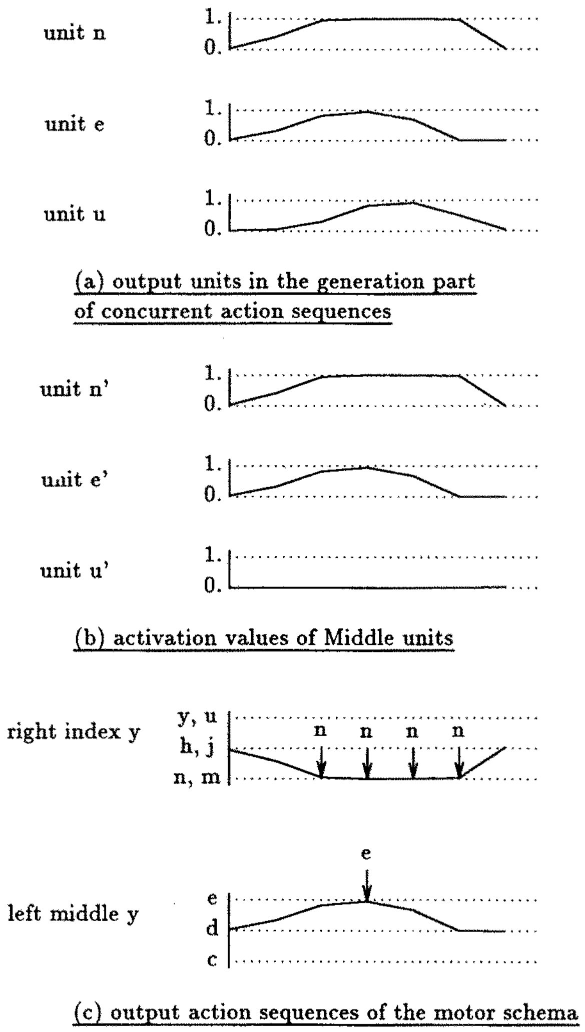
From results of the above simulation, it became clear that the model proposed in this paper can represent seriality and concurrency of typewriting action satisfactorily, and can generate efficient typing action sequences. In this model, it is not necessary to prepare a network structure corresponding to each word beforehand, and arbitrary set of words can be stored by learning. Furthermore, when the learning of the constraint representation part is completed once, the constraint of the motion sequences concerning typewriting action can be represented in a generalized form; and the model can generate all lines of letters.

Table 1 Input and output of typing actions

	Concurrent action generation part	Constraint repre-sentaton part
Input	Words (M kinds)	Letter typing actions (26 kinds)
Output	Letter typing action (26 kinds)	Finger positions (10) Key hitting instruction
Teacher signal	spelling of words	Finger position for letters

5. Conclusion

In this paper, the realization of an efficient motor schema combining seriality and concurrency with the use of a neural network was explained. The neural network which divided its structure into concurrent action generation and constraint representation parts and its learning rule were proposed, and a motor schema of typewriting action was constructed.



(a) output units in the generation part of concurrent action sequences

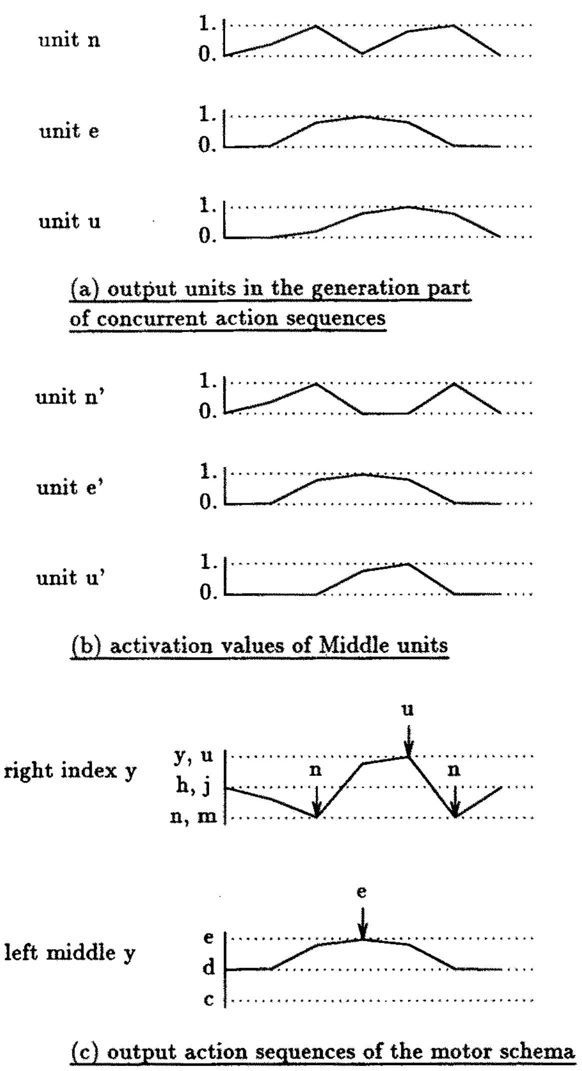
(b) activation values of Middle units

(c) output action sequences of the motor schema

Fig. 7 Simulation results for a word "neun" (without reinforcement learning)

Table 2 Fingers and corresponding letters

Left hand						Right hand				
	Little finger	Third finger	Middle finger	Index finger x =		Index finger x =		Middle finger	Third finger	Little finger
y				0	+1	-1	0			
+ 1	q	w	e	r	t	y	u	i	o	p
0	a	s	d	f	g	h	j	k	l	
-1	z	x	c	v	b	n	m			



(a) output units in the generation part of concurrent action sequences

(b) activation values of Middle units

(c) output action sequences of the motor schema

Fig. 8. Simulation results for a word "neun" (with reinforce ment learning)

This model, by the exchange of the constraint representation part as a module, can correspond to plural problem regions. For example, action sequences corresponding to the spellings of words are generated in the concurrent action generation part, and the action sequences are modified to the motion command for several tasks such as typewriting, handwriting, pronunciation, etc. (**Fig.9**). Conversely, once the constraint representation part concerning a certain problem region is obtained by learning, the learning of the concurrent action generation part only suffices if a new plan is given. The flexible planning capability was obtained by structuring the network.

Nevertheless, some problematic points remain in this model.

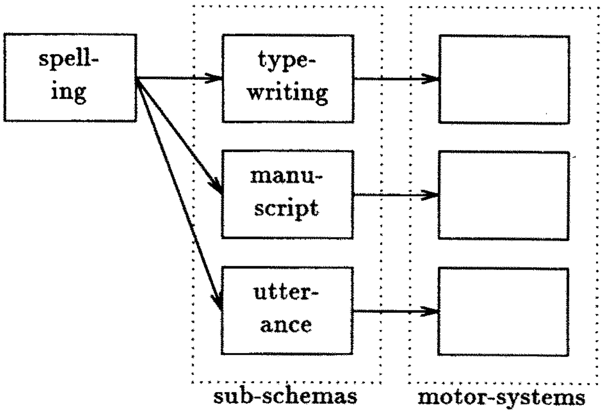


Fig. 9 The hierarchical structure of motor schemas

1) It takes long time to learn concurrent action generation, and learning has the possibility of running into a local minimum, which may be true of the model which uses the error back propagation method in common. In this paper, this problem was settled to a certain extent by introducing reinforcement learning into learning rules, but there is no guarantee that this learning always produces the most suitable solution, so further superior learning rules must be studied.

2) The kinds of constraints which can be represented in the constraint representation part are limited. In this paper, the constraint from problem regions was realized by imbedding the property of constraint into the structure of the model beforehand, which simplified learning. On the other hand, the problem region which is made a subject becomes limited to a considerable extent. It is thus necessary to study the structure of a model which can represent constraints from various regions.

3) The action sequence contained in the plan is not exchangeable. In this paper, only the action sequence determined by the object of the plan is treated and an exchange of action sequence is not considered.

The above problems must be studied in the future to achieve a wider range of motor schema and to apply it to robot control.

Appreciation: A part of this study was funded by the Ministry of Education, Science and Culture under project number (62460142, 63750415), and our thanks go to the Ministry.

References:

1) M.A. Arbib: "Schema and Perception: Perspectives from Brain Theory and Artificial Intelligence", Pattern Recognition by Humans and Machines, Vol.2, eds. E.C. Schewb and H.C. Nusbaum, pp.121-157, Academic Press, New York (1986).

2) D.E. Rumelhard and J.L. McClelland: "Parallel Distributed Processing, Vol.1 & 2", MIT Press, Cambridge (1986).

3) H. Aso: "Neural Network Information Processing", Sangyotosho (1987).

4) S. Kunifuji: "Connectionist Model and its Environment", Measurement and Control, 27, 10, pp.935-938 (1987).

5) D.E. Rumelhart, G.E. Hinton and R.J. Williams: "Learning Internal Representations by Error Propagation", Parallel Distributed Processing, Vol.1, eds. D.E. Rumelhard and J.L. McClelland, MIT Press Cambridge (1986).

6) D.E. Rumelhart, P. Smolensky, J.L. McClelland and G.E. Hinton: "Schemata and Sequential Thought Processes in PDP Models", Parallel Distributed Processing, Vol.2, eds. D.E. Rumelhard and J.L. McClelland, pp.7-57, MIT Press, Cambridge (1986).

7) D.E. Rumelhart and D.A. Norman: "Simulating a Skilled Typist: A

Study of Skilled Cognitive-Motor Performance", Cognitive Sci., 6, 4, pp.1-36 (1982).

8) M.I. Jordan: "Attractor Dynamics and Parallelism in a Connectionist Sequential Machine", Proc. of the 8th Annual Conference of the Cognitive Science Society, pp.531-546 (1986).

9) Y. Miyata: "Organization of Action Sequence in Motor Learning", Proc. 9th Annual Conference of the Cognitive Science Society, pp.496-507 (1987).