Biol. Cybern. 57, 169–185 (1987)

generation of motor command

Biological Cybernetics

A Hierarchical Neural-Network Model for Control and Learning of Voluntary Movement

M. Kawato*, Kazunori Furukawa**, and R. Suzuki

Department of Biophysical Engineering, Faculty of Engineering Science, Osaka University, Toyonaka, Osaka, 560 Japan

Abstract. In order to control voluntary movements, the central nervous system (CNS) must solve the following three computational problems at different levels: the determination of a desired trajectory in the visual coordinates, the transformation of its coordinates to the body coordinates and the generation of motor command. Based on physiological knowledge and previous models, we propose a hierarchical neural network model which accounts for the generation of motor command. In our model the association cortex provides the motor cortex with the desired trajectory in the body coordinates, where the motor command is then calculated by means of long-loop sensory feedback. Within the spinocerebellum – magnocellular red nucleus system, an internal neural model of the dynamics of the musculoskeletal system is acquired with practice, because of the heterosynaptic plasticity, while monitoring the motor command and the results of movement. Internal feedback control with this dynamical model updates the motor command by predicting a possible error of movement. Within the cerebrocerebellum – parvocellular red nucleus system, an internal neural model of the inverse-dynamics of the musculo-skeletal system is acquired while monitoring the desired trajectory and the motor command. The inverse-dynamics model substitutes for other brain regions in the complex computation of the motor command. The dynamics and the inverse-dynamics models are realized by a parallel distributed neural network, which comprises many sub-systems computing various nonlinear transformations of input signals and a neuron with heterosynaptic plasticity (that is, changes of synaptic weights are assumed proportional to a product of two kinds of synaptic inputs). Control and learning performance of the model was investigated by computer simulation, in which a robotic manipulator was used as a controlled system, with the following results: *(1)* Both the dynamics and the inverse-dynamics models were acquired during control of movements. *(2)* As motor learning proceeded, the inverse-dynamics model gradually took the place of external feedback as the main controller. Concomitantly, overall control performance became much better. *(3)* Once the neural network model learned to control some movement, it could control quite different and faster movements. *(4)* The neural network model worked well even when only very limited information about the fundamental dynamical structure of the controlled system was available. Consequently, the model not only accounts for the learning and control capability of the CNS, but also provides a promising parallel-distributed control scheme for a large-scale complex object whose dynamics are only partially known.

1 Introduction

Although the neural connections in the brain are basically stable and rigid after their formation at an early developmental stage, some of them have plasticity. This "plasticity" is thought to be the neural basis for adaptive behavior. Most voluntary movements, which are executed at will, are learned movements. It is very important and interesting to investigate how the central nervous system (CNS) acquires the ability to control movements by making use of synaptic plasticity, not only from the standpoint of neuroscience but also from that of robotics.

Based on detailed knowledge on the neural circuits in the cerebellum, Marr (1969) and Albus (1971) proposed learning network models of the cerebellum. In these "perceptron" models, the efficacy of a parallel

* To whom correspondence should be addressed

** Present address: Fujitsu Limited, Kawasaki, Japan

fiber-Purkinje cell synapse was assumed to change when conjunction of the parallel-fiber input and the climbing-fiber input occurs. Ito et al. (1982) demonstrated the presence of the putative heterosynaptic plasticity of Purkinje cells in the flocculus of the cerebellum, which plays an essential role in the adaptive control of the vestibulo-ocular reflex. Although the Marr-Albus model of the cerebellum as a spatial pattern discriminator does not give a sufficient account of the processing of temporal patterns, which is essential for the control of movement, Fujita (1982a) expanded the Marr-Albus model and proposed an adaptive filter model of the cerebellar cortex. His model reproduced several experimental features in an adaptive modification of the vestibulo-ocular reflex (Fujita 1982b). Consequently, a splendidly comprehensive understanding of the adaptive control of the vestibulo-ocular reflex was provided by these works (see Ito 1984 for a review), which accounts for all of the following three levels for understanding complex information-processing systems proposed by Marr (1982): *(1)* computational theory, *(2)* representation and algorithm, and *(3)* hardware implementation.

The investigation of neural mechanisms involved in the control and learning of voluntary movement seems much more difficult than that of the vestibulo-ocular reflex for the following reasons. First, the control object of voluntary movement (e.g. hand, leg or trunk) has highly nonlinear dynamics with multiple degrees of freedom. Second, many neural networks and pathways are hierarchically involved (Allen and Tsukahara 1974). Third, volition participates in the highest level.

We propose a computational model of voluntary movement in Fig. 1, which accounts for Marr's first level (1982). Consider a thirsty person reaching for a glass of water on a table. The goal of the movement is moving the arm toward the glass to reduce thirst. First, one desirable trajectory in the task-oriented coordinates must be selected from out of an infinite number of possible trajectories which lead to the glass, whose spatial coordinates are provided by the visual system (determination of trajectory). Second, the spatial coordinates of the desired trajectory must be reinterpreted in terms of a corresponding set of body coordinates, such as joint angles or muscle lengths (transformation of coordinates). Finally, motor commands (e.g. torque) must be generated to coordinate the activity of many muscles so that the desired trajectory is realized (generation of motor command). We do not adhere to the hypothesis of the step-by-step information processing shown by the three straight arrows in Fig. 1. Rather, Uno et al. (1987) proposed a learning algorithm which calculates the motor command directly from the goal of the movement represented by

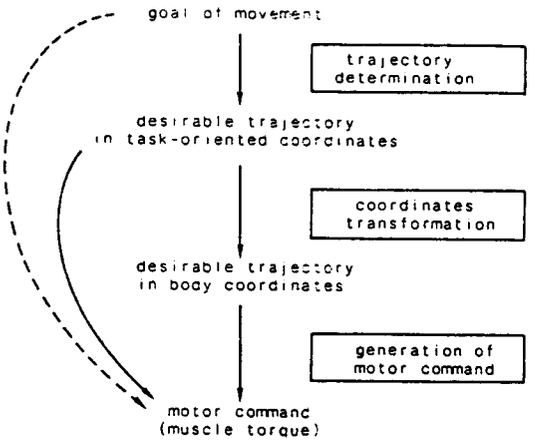

Fig. 1. A computational model for control of voluntary movement

some performance index (broken and curved arrow in Fig. 1). Further, as shown by a curved arrow in Fig. 1, motor command can be obtained directly from the desired trajectory represented in the task-oriented coordinates by an iterative learning algorithm (Kawato et al. 1987). In this respect, our model differs from the three-level hierarchical movement plan proposed by Hollerbach (1982). However, several lines of experimental evidence suggest that the information in Fig. 1 is internally represented in the brain. First, Flash and Hogan (1985) provided strong evidence to indicate that movement is planned at the task-oriented coordinates (visual coordinates) rather than at the joint or muscle level. Second, the presence of the transcortical loop (i.e. the negative feedback loop via the cerebral cortex; Evarts 1981) indicates that the desired trajectory must be represented also in the body coordinates, since signals from proprioceptors are expressed in the body coordinates. Finally, Cheney and Fetz (1980) showed that discharge frequencies of primate corticomotoneuronal cells in the motor cortex were fairly proportional to active forces (torque). Consequently, the CNS must adopt, at least partly, the step-by-step strategy for the control of voluntary movement.

The problem of the determination of the trajectory was investigated by Uno et al. (1987), and the problem of the transformation of the coordinates will be dealt with in our next paper (Kawato et al., in preparation). In this paper, we concentrate on the problem of the generation of motor command. First, a hierarchical neural network model with heterosynaptic plasticity is proposed for the control and learning of voluntary movement. Second, the capability of learning control of a robotic manipulator is demonstrated by computer simulation.

2 Hierarchical Neural Network for the Control of and Learning of Voluntary Movement

In learning a movement, we first execute the movement very slowly because it cannot be adequately preprogramed. Instead, it is performed largely by cerebral intervention with use of long-loop sensory feedback. With practice, a greater amount of the movement can be preprogramed and the movement can be executed more rapidly. Ito (1970) proposed the hypothesis that the cerebrocerebellar communication loop is used as a reference model for the open-loop control of voluntary movement. Allen and Tsukahara (1974) proposed a comprehensive model, which accounts for the functional roles of several brain regions (association cortex, motor cortex, lateral cerebellum, intermediate cerebellum, basal ganglia) in the control of voluntary movement. Tsukahara and Kawato (1982) proposed a theoretical model of the cerebro-cerebello-rubral learning system based on recent experimental findings of the synaptic plasticity in the red nucleus, especially on the sprouting phenomena (see Tsukahara 1981, for a review). Expanding on these previous models, we propose a hierarchical neural network model for the control of and learning of voluntary movement, shown in Fig. 2. This model provides concrete algorithms and neural networks for the problem of the generation of motor command, raised in Fig. 1. In this section we explain the global structure of the model and the information flow in it.

In our model, the association cortex sends the desired motor pattern, that is trajectory x_d expressed in the body coordinates, to the motor cortex, where the motor command, that is torque u to be generated by muscles, is then somehow computed. Here, for simplicity, we identify the motor command with the active torque based on the experimental data of Cheney and Fetz (1980) that discharge frequencies of primate corticomotoneuronal cells in the motor cortex were fairly proportional to active forces. The motor command is transmitted to muscles via spinal motoneurons. The musculoskeletal system interacts with its environment and realizes some kind of motor pattern, x. In general, x does not coincide with x_d. The actual motor pattern (x) and its time derivative (dx/dt) (e.g. muscle length and its derivative) are measured by proprioceptors and sent back to the motor cortex, for example via the transcortical loop. Then, feedback control can be performed utilizing error in the movement trajectory ($x_d - x$). However, severe limitations are imposed on the biological feedback system. There are substantial delays in the feedback loop; for example, the transcortical loop requires 40–60 ms (Evarts 1981). Further, experimental results indicate that the contribution of the supraspinal loop to load compensation is insubstantial (Evarts 1981). Feedback delays and small gains both limit controllable speeds of motions. Consequently, in learning a movement, we first must execute the movement very slowly, because otherwise the control system becomes unstable.

The spinocerebellum (vermis and intermediate part of the hemisphere) – magnocellular part of the red nucleus system receives information about the results of the movement (x) as afferent input from the proprioceptors, as well as an efference copy of the motor command (u). Within this spinocerebellum – magnocellular red nucleus system, an internal neural model of the musculoskeletal system is acquired. Once the

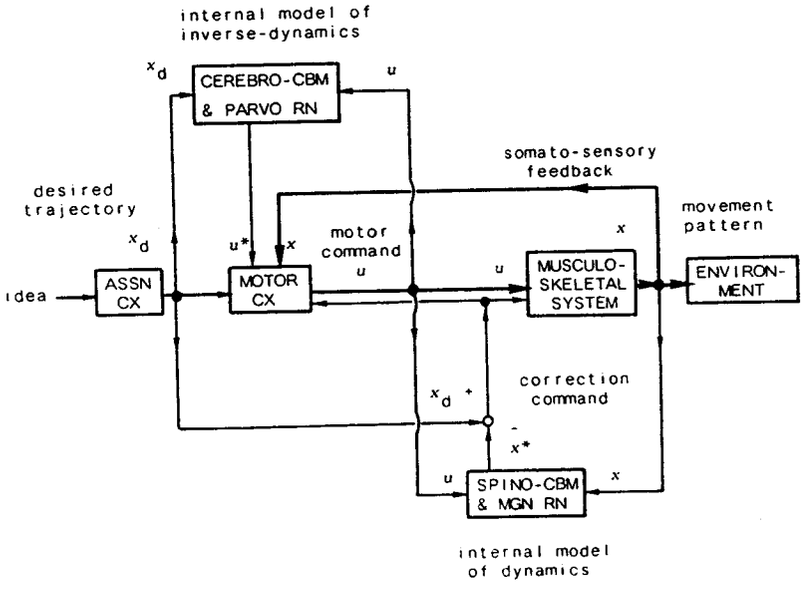

Fig. 2. A hierarchical neural network model for control and learning of voluntary movement. The model is composed of the following three parts: *(1)* The main descending pathway and the transcortical loop designated by heavy lines, *(2)* The spinocerebellum – magnocellular red nucleus system as an internal nerual model of dynamics of the musculoskeletal system, *(3)* The cerebrocerebellum – parvocellular red nucleus system as an internal neural model of inverse-dynamics of the musculoskeletal system. See text for detail

internal model is formed by motor learning, it can provide an approximated prediction x^* (see Appendix) of the actual movement x when it receives the motor command u. A predicted possible movement error $x_d - x^*$ is transmitted to the motor cortex and to muscles via the rubrospinal tract. Since the loop time of the cerebro-cerebellar communication loop is 10–20 ms (Eccles 1979) and is shorter than that of the supraspinal loop, the performance of the feedforward control with the internal model and the internal feedback loop is better than that of long-loop sensory feedback. In summary, the spinocerebellum – magnocellular red nucleus system updates the motor command by predicting a possible error of movement.

The cerebrocerebellum (lateral part of the hemisphere) – parvocellular part of the red nucleus system, which develops extensively in primates, especially in man, receives its input from wide areas of the cerebral cortex and does not receive peripheral sensory input. That is, it monitors both the desired trajectory x_d and the motor command u but it does not receive information about the actual movement x. Within the cerebro-cerebellum – parvocellular red nucleus system, an internal neural model of the inverse-dynamics of the musculoskeletal system is acquired, with practice, by making use of the synaptic plasticity. The inverse-dynamics of the musculoskeletal system is defined as the dynamical system whose input and output are inverted (trajectory x is the input and motor command (i.e. torque) u is the output). Note that the spinocerebellum – magnocellular red nucleus system provides a model of the dynamics of the musculoskeletal system (motor command (i.e. torque) u is the input and trajectory x is the output). The inverse-dynamics model is not a model of the external world; rather it is a model of the information processing done in other brain regions such as the motor cortex and the spinocerebellum which computes the motor command from the desired trajectory. A germ of this idea can already be seen in Ito (1970). Once the inverse-dynamics model is acquired by motor learning, it can compute a good motor command u^* directly from the desired trajectory x_d. This motor command is transmitted to the motor cortex via the ventrolateral nucleus of the thalamus. In summary, the cerebrocerebellum – parvocellular red nucleus system substitues for other brain regions in the complex computation of motor commands.

The neural network model shown in Fig. 2 is based on various physiological and morphological information (Ito 1984; Ghez and Fahn 1985), especially on the importance of synaptic plasticity (Gilbert and Thach 1977) and of the cerebro-cerebellar communication loop (Sasaki et al. 1982; Sasaki and Gemba 1982) in the motor learning of voluntary limb movements. The model predicts that if the rubro-olivo-cerebellar loop is destroyed, the internal model would be destroyed also, and motor performance would be severely disturbed. This prediction is in accord with the frequently reported symptoms of "tremor" after lesions anywhere in the rubro-olivary pathway (Poirier et al. 1969). This may be interpreted as being the oscillation due to the delay of feedback. Another prediction of the model is that after lesions of the rubro-olivo-cerebellar loop, motor learning does not take place. This, of course, is the deficit of motor learning as reported by Ito and his collaborators (Ito et al. 1974, 1982) and also by Llinás (Llinás et al. 1975), although the interpretation of this deficit has been related to Marr's hypothesis in the case of Ito's paradigm (Ito 1984).

3 Internal Neural Model with Heterosynaptic Plasticity

The internal dynamics model and the internal inverse-dynamics model proposed in the previous section can be realized by a parallel-distributed-processing neural network with hetero-synaptic plasticity. They can be regarded, from an engineering point of view, as identifiers of unknown dynamics and inverse-dynamics of the musculoskeletal system. Arbib (1981) pointed out that such a neural identifier is essential for motor learning. Let us consider an identifier which approximates the output $z(t)$ of an unknown nonlinear system by monitoring both the input $u(t)$ and the output $z(t)$ of this system (Fig. 3). This type of identifier can be realized by a neural network, as shown in Fig. 3 (Tsukahara and Kawato 1982), which comprises many subsystems computing various nonlinear transformations of the input $u(t)$, and a neuron with heterosynaptic plasticity.

The input $u(t)$ to the unknown nonlinear system is also fed to n subsystems and is nonlinearly transformed into n different inputs $x_l(t)$ $(l=1, \ldots, n)$ to the neuron with plasticity. That is, instantaneous firing frequencies of n input fibers to the neuron are designated by $x_1(t), \ldots, x_n(t)$. Let w_l denote a synaptic weight of the l-th input. Membrane potential $y(t)$ of the neuron is the sum of n postsynaptic potential. For simplicity, we assume that the output signal of the neuron is equal to its membrane potential $y(t)$. In vector notation, we have the following equations.

$$\begin{aligned} &\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T, \\ &\mathbf{w} = [w_1, w_2, \ldots, w_n]^T, \\ &y(t) = \mathbf{w}^T\mathbf{x}(t) = \mathbf{x}(t)^T\mathbf{w}. \end{aligned} \tag{1}$$

Here, T denotes transpose. The second synaptic input to the neuron is an error signal (e.g. climbing fiber input for the Purkinje cell), and is given as an error

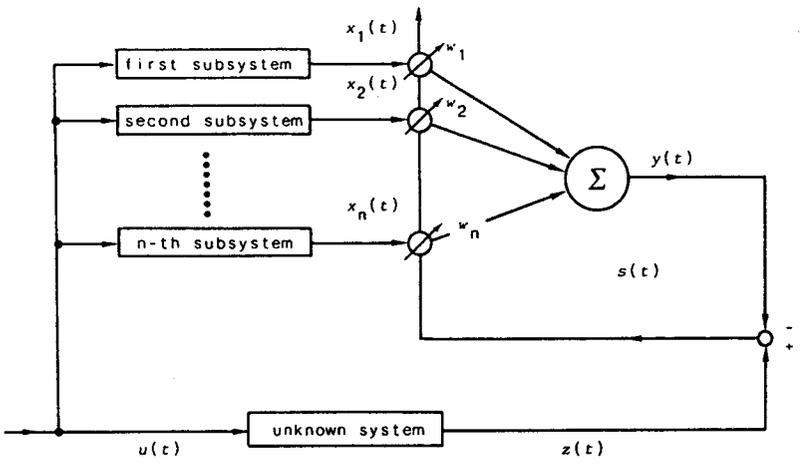

Fig. 3. A neural identifier of an unknown system comprised of n subsystems and a neuron with heterosynaptic plasticity

between the output of the neuron and the output from the unknown system, $s(t)=z(t)-y(t)$. Based on physiological knowledge of the heterosynaptic plasticity of the red nucleus neurons (Tsukahara et al. 1981) and the Purkinje cells (Ito 1984), we assume that the l-th synaptic weight w_l changes when the conjunction of the l-th input $x_l(t)$ and the teaching signal $s(t)$ occurs. Further, the rate of change of the synaptic weight is assumed proportional to the product of the two inputs:

$$\tau \mathrm{d}\mathbf{w}(t)/\mathrm{d}t=\mathbf{x}(t)s(t)=\mathbf{x}(t)[z(t)-\mathbf{x}(t)^{\mathrm{T}}\mathbf{w}(t)]\,. \tag{2}$$

Here, τ is a time constant of change of the synaptic weight. This learning rule is closely related to Amari's (1977) orthogonal learning, although it was used for the association of static spatial patterns.

If $u(t)$ is a stochastic process, then $\mathbf{x}(t)$ and $z(t)$ are also stochastic processes and Eq. (2) must be rewritten as a stochastic differential equation:

$$\tau \mathrm{d}\mathbf{w}(t,\omega)/\mathrm{d}t=\mathbf{x}(t,\omega)[z(t,\omega) -\mathbf{x}(t,\omega)^{\mathrm{T}}\mathbf{w}(t,\omega)]\,. \tag{3}$$

Here, ω is a sample point in a stochastic space. Let us consider the following averaged equation of (3), which is obtained by taking an expected value.

$$\tau \mathrm{d}\mathbf{m}(t)/\mathrm{d}t=\mathrm{E}[\mathbf{x}(t,\omega)z(t,\omega)] -\mathrm{E}[\mathbf{x}(t,\omega)\mathbf{x}(t,\omega)^{\mathrm{T}}]\mathbf{m}(t)\,. \tag{4}$$

Here, $\mathbf{m}$ is an averaged vector of $\mathbf{w}$. We can prove the following theorem about the convergence of the synaptic weight $\mathbf{w}(t)$ using Geman's (1979) result, if $\mathbf{x}$ and z are mixing random processes (see Appendix for proof).

Theorem 1. *If the time constant τ of change of the synaptic weight is sufficiently long compared with the "rate" of mixing of $\mathbf{x}$ and z, then the synaptic weight $\mathbf{w}$ converges in mean to the value for which a mean square error of the output $E[(z-y)^2]$ is minimum.*

Because the time constants of physiologically known synaptic plasticities are sufficiently long (from a few hours to a few weeks) compared with temporal patterns of movement (several hundreds ms), the assumption of the theorem is satisfied. It is worthwhile to note that the averaged Eq. (4) gives the steepest descent method and the convergence is global (i.e. there is no local minimum: see Appendix). The basic organization of the neural identifier shown in Fig. 3 is the same as the LMS (Least Mean Square) adaptive filter of Widrow (Widrow et al. 1976) and the adaptive filter model of Fujita (1982a), although they dealt with a linear system and the proof of convergence was different.

Although the best synaptic weights can always be obtained by the learning rule (2) within a given set of subsystems, Theorem 1 does not necessarily guarantee that the output error tends to become zero as learning proceeds. Asymptotic performance of the neural identifier critically depends on the selection of the set of subsystems (see Appendix).

4 Control of Robotic Manipulator by Model Neural Network

We examined whether the proposed neural-network model is efficient in learning control of an object with highly nonlinear dynamics and multiple degrees of freedom. In computer simulation, a usual industrial robotic manipulator was chosen as a controlled system. Although it is much simpler than musculoskeletal systems such as the human arm, they both have several essential features (nonlinear dynamics, multiple degrees of freedom, and interactions between different freedoms) in common.

By computer simulation, Furukawa (1984) and Miyamoto (1985) showed that the internal neural model for the dynamics of a robotic manipulator with two degrees of freedom was actually acquired during a

2000–5000 s learning period while monitoring both the input torque [$u(t)$ in Fig. 3] and the output joint angles [$z(t)$ in Fig. 3] of the manipulator. In both studies, the learning rule in (2) was used. However, as subsystems of the neural identifier, Furukawa used Wilson-Cowan's neuronpool models (1972) with different synaptic-connection parameters, while Miyamoto chose different models of the manipulator dynamics with different viscosity-coefficient parameters at the joints. Once the dynamics model was obtained by learning, the internal feedback loop with this internal model was found to control the manipulator much better than the long-loop feedback via the external world (see the last paragraph of Appendix).

4.1 Hierarchical Control of Manipulator by Inverse-Dynamics Model

To examine learning and control performance, extensive computer simulations were made of the hierarchical neural network model, excluding the spinocerebellum – magnocellular red nucleus system (dynamics model). We omitted the internal dynamics model because of the limitation of computer resources. This part of the model is the most time consuming because we need to numerically integrate many differential equations which describe the subsystems of the dynamics model.

Figure 4a shows a block diagram of a simulated neural network model and a manipulator. Let $T(t)$, $T_i(t)$, and $T_f(t)$ denote a torque fed to the manipulator, a torque calculated by the inverse-dynamics model and a feedback torque, respectively. The total torque was a sum of the feedforward and the feedback torques:

$$T(t) = T_i(t) + T_f(t). \tag{5}$$

The inverse-dynamics model receives the desired trajectory $q_d(t)$ represented as joint angles as an input [$u(t)$ in Fig. 3] and monitors the total torque $T(t)$ as the output of the unknown dynamical system [$z(t)$ in Fig. 3]. If the simulated neural network would be related to the perceptron, the total torque might correspond to the teaching singnal. From (5), the error signal $s(t)$ in the previous section equals the feedback torque $T_f(t)$, and it is expected that $T_f(t)$ tends to zero as learning proceeds. But this is by no means guaranteed because we cannot simply take out an unknown dynamical system from the block diagram of Fig. 4a. In other words, the inverse-dynamics model does not receive the real trajectory $q(t)$ represented as joint angles; instead, it receives only the desired trajectory $q_d(t)$. Consequently, it is very important to examine the

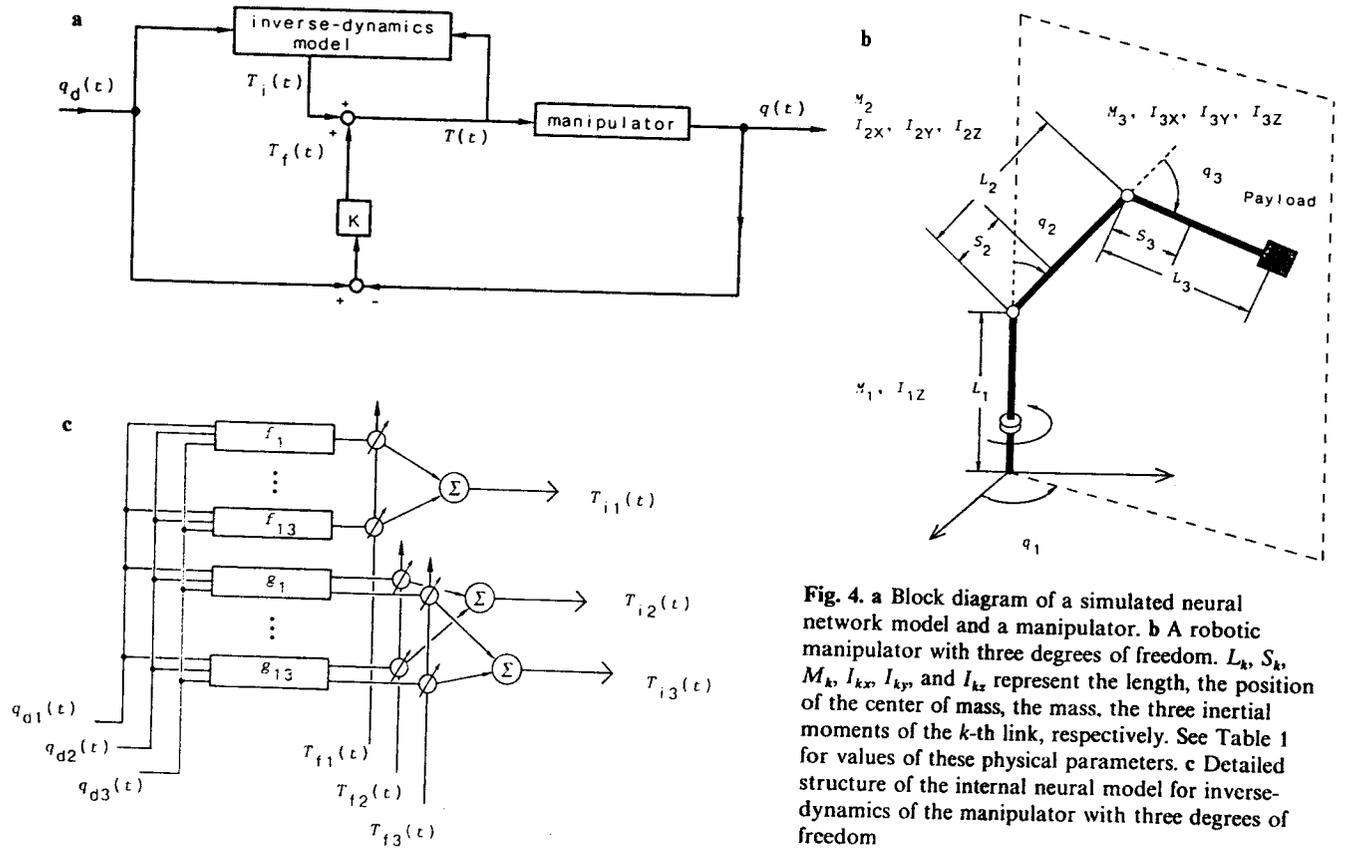

Fig. 4. a Block diagram of a simulated neural network model and a manipulator. **b** A robotic manipulator with three degrees of freedom. L_k, S_k, M_k, I_{kx}, I_{ky}, and I_{kz} represent the length, the position of the center of mass, the mass, the three inertial moments of the k-th link, respectively. See Table 1 for values of these physical parameters. **c** Detailed structure of the internal neural model for inverse-dynamics of the manipulator with three degrees of freedom

learning and control performance of the hierarchical control system shown in Fig. 4a.

The configuration of the three-link manipulator shown in Fig. 4b was chosen so that it resembles a human arm (see Table 1 for physical parameters). Let $q_k(k=1,2,3)$ denote the k-th joint angle and $T_k(k=1,2,3)$ denote the torque fed to the k-th joint. Using the Lagrangian, we can derive the following dynamics equation of the manipulator.

$$R(\mathbf{q})\ddot{\mathbf{q}}-\left(\sum_k \dot{q}_k \partial R/\partial q_k\right)\dot{\mathbf{q}} -(1/2)\dot{\mathbf{q}}^{\mathrm{T}}(\partial R/\partial \mathbf{q})\dot{\mathbf{q}}+B\dot{\mathbf{q}}+\mathbf{G}(\mathbf{q})=\mathbf{T}(t), \tag{6}$$

where,

$$\mathbf{q}=(q_1,q_2,q_3)^{\mathrm{T}} \quad \text{and} \quad \mathbf{T}=(T_1,T_2,T_3)^{\mathrm{T}}.$$

$R(\mathbf{q})$ is a 3×3 inertia matrix, which is not diagonal. B is a 3×3 diagonal matrix representing viscosity coefficients, $B=\mathrm{diag}(b_k)$. $\mathbf{G}(\mathbf{q})$ is a 3-dimensional nonlinear vector function and represents gravitational forces. The first term represents the inertia force, the second and third terms represent the centripetal and Coriolis forces, the fourth term is the frictional force and the fifth term is the gravitational force. It can be seen that the manipulator dynamics is nonlinear and there are interactions between different freedoms.

Figure 4c shows the detailed structure of the internal model for the inverse-dynamics of the manipulator. It receives the three inputs $[q_{d1}(t), q_{d2}(t), q_{d3}(t)]$ and the three error signals $[T_{f1}(t), T_{f2}(t), T_{f3}(t)]$, and outputs the three torques $[T_{i1}(t), T_{i2}(t), T_{i3}(t)]$. $q_{dk}(t)$, $T_{fk}(t)$, and $T_{ik}(t)$ $(k=1,2,3)$ denote the desired joint angle, the feedback torque and the torque generated by the inverse-dynamics model, respectively, regarding the k-th joint. This is an expansion of the neural identifier of Fig. 4a to the multiple inputs-outputs case. Each subsystem receives the three inputs q_{d1}, q_{d2}, and q_{d3}, and nonlinearly transforms them into its output $f_l(q_{d1}, q_{d2}, q_{d3})$ for the first joint, or $g_l(q_{d1}, q_{d2}, q_{d3})$ for the second and the third joints. The subsystems f_l and g_l were conveniently chosen as shown in Table 2 from manipulator dynamics Eq. (6). That is, each subsystem corresponds to an expanded term of the left-hand side of (6). Although, in general, different sets of subsystems need to be prepared for different outputs $T_{ik}(k=1,2,3)$, the same set of subsystems $g_l(l=1,\ldots,13)$ were used in common by the second and the third outputs since their subsystems almost overlap with each other. In summary, the torques generated by the inverse-dynamics model are expressed as follows.

$$\begin{aligned} T_{i1}(t)&=\sum_{l=1}^{13} w_{1l} f_l(q_{d1}(t), q_{d2}(t), q_{d3}(t)) \\ T_{ik}(t)&=\sum_{l=1}^{13} w_{kl} g_l(q_{d1}(t), q_{d2}(t), q_{d3}(t)) \quad (k=2,3). \end{aligned} \tag{7}$$

Table 1. Values of physical parameters of the robotic manipulator shown in Fig. 4b. b_k is a viscosity coefficient at the k-th joint. See explanation of Fig. 4 for definitions of other symbols

Parameter	First link	Second link	Third link
L_k(m)	0.4	0.4	0.4
S_k(m)	–	0.15	0.15
M_k(kg)	15.0	7.0	3.0
I_{kx}(kg·m^2)	–	0.589	0.251
I_{ky}(kg·m^2)	–	0.584	0.253
I_{kz}(kg·m^2)	0.0170	0.00673	0.00340
b_k(kg·m/s)	20.0	15.0	5.0
Mass of payload (kg)	1.0		

Table 2. Nonlinear transformations of 26 subsystems used in the inverse-dynamics model shown in Fig. 4c. q_{dk} is simply denoted as q_k here

l	$f_l(q_1,q_2,q_3)$	$g_l(q_1,q_2,q_3)$
1	$\ddot{q}_1$	$\ddot{q}_2$
2	$\sin^2 q_2\cdot\ddot{q}_1$	$\ddot{q}_3$
3	$\cos^2 q_2\cdot\ddot{q}_1$	$\cos q_3\cdot\ddot{q}_2$
4	$\sin^2(q_2+q_3)\cdot\ddot{q}_1$	$\cos q_3\cdot\ddot{q}_3$
5	$\cos^2(q_2+q_3)\cdot\ddot{q}_1$	$\sin q_2\cdot\cos q_2\cdot\dot{q}_1^2$
6	$\sin q_2\sin(q_2+q_3)\cdot\ddot{q}_1$	$\sin(q_2+q_3)\cos(q_2+q_3)\cdot\dot{q}_1^2$
7	$\sin q_2\cos q_2\cdot\dot{q}_1\dot{q}_2$	$\sin q_2\cos(q_2+q_3)\cdot\dot{q}_1^2$
8	$\sin(q_2+q_3)\cos(q_2+q_3)\cdot\dot{q}_1\dot{q}_2$	$\cos q_2\sin(q_2+q_3)\cdot\dot{q}_1^2$
9	$\sin q_2\cos(q_2+q_3)\cdot\dot{q}_1\dot{q}_2$	$\sin q_3\cdot\dot{q}_2^2$
10	$\cos q_2\sin(q_2+q_3)\cdot\dot{q}_1\dot{q}_2$	$\sin q_3\cdot\dot{q}_3^2$
11	$\sin(q_2+q_3)\cos(q_2+q_3)\cdot\dot{q}_1\dot{q}_3$	$\sin q_3\cdot\dot{q}_2\dot{q}_3$
12	$\sin q_2\cos(q_2+q_3)\cdot\dot{q}_1\dot{q}_3$	$\dot{q}_2$
13	$\dot{q}_1$	$\dot{q}_3$

Here w_{kl} is the weight of a synapse from the l-th subsystem to the k-th output neuron. If a perfect inverse-dynamics model is formed, Eq. (7) must coincide with Eq. (6). Let the left-hand side of (6) be expanded by the same functions f_l and g_l as in (7), and w_{kl}^* denote a coefficient of the l-th function f_l or g_l in this expansion of the k-th torque. If uniqueness of convergence of $\mathbf{w}$ holds (see Appendix), we expect that each w_{kl} converges to w_{kl}^* as learning proceeds.

The fourth order Runge-Kutta-Gill method with a time step of 2 ms was used to numerically integrate the learning Eq. (2) and the manipulator dynamics Eq. (6). The first and the second time derivatives of joint angles, which were used in nonlinear transformations of subsystems, were calculated by central difference methods. Initial values of the synaptic weights w_{kl} at the beginning of learning were all set at 0; that is, the inverse-dynamics model did not output any torque at the beginning of learning. The program was written in Fortran 77 for a ACOS 1000 computer at Osaka University.

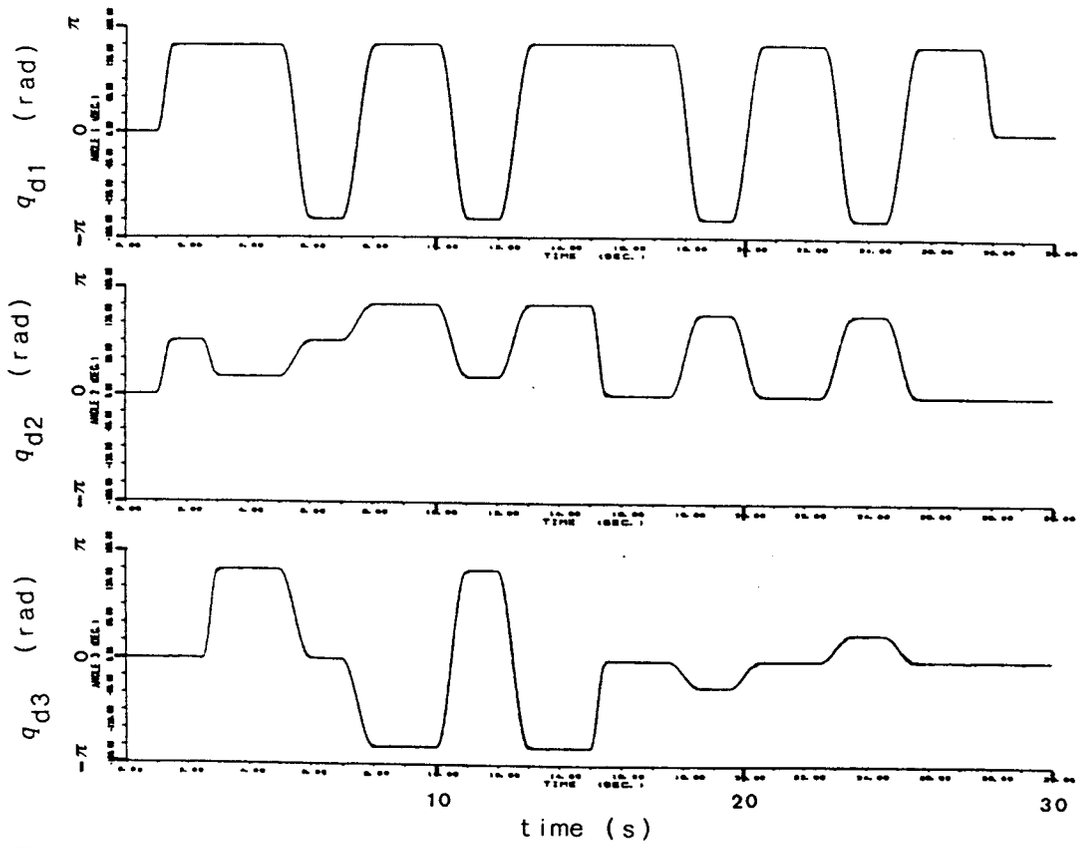

Fig. 5. Time courses of the three joint angles during the desirable movement pattern. The realized trajectory after 20 min learning is also plotted but can not be separately seen

4.2 Learning with Repeated Movement Pattern

We first studied control performance of the neural network model when one movement pattern was repeatedly learned. A desired trajectory, shown in Fig. 5, with a duration of 30 s was given for 20 min (i.e. for 40 times). In this pattern, the three joints moved cooperatively at a maximum speed of about 500 deg/s, which was quite fast for industrial robotic manipulators. As can be seen in Fig. 5, the time course of each joint angle $q_{dk}(t)$ was smooth, since it was composed of trigonometric functions smoothly jointed with constant parts, while the manipulator held a fixed posture. The time constant τ of learning was chosen as 1000 s. For simplicity, the graviational force $\mathbf{G}(\mathbf{q})$ in (6) was compensated beforehand.

The feedback torque was made of a proportional component and a local derivative component:

$$T_{fk}(t)=K_{pk}[q_{dk}(t)-q_k(t)] + K_{vk}dq_k(t)/dt, \quad k=1,2,3,$$

$K_{vk}=0$ unless $|q_k(t)-q_{dk}$ (objective point)$|<\varepsilon$.

Proportional and derivative feedback gains K_{pk} and K_{vk} were selected as (517.2, 746.0, 191.4) and (16.2, 37.2, 8.4) so that the natural angular frequency $\omega_n=20$ and the damping ratio $\xi=0.7$ were attained. These values were calculated based on a linearization of the manipulator dynamics Eq. (6). The velocity feedback was applied only around the stopping points (i.e. only after the joint angle got into some small bounds of the objective point). Organisms hold a posture by the cocontraction of flexor and extensor muscles around the same joint, which induces an increase of viscous friction of the muscles. The local velocity feedback simulates this effect and reduces overshoots of movements.

Figure 6 compares the total torques fed to the third joint (T_3, top), feedback torques (T_{f3}, middle) and the torques generated by the inverse-dynamics model (T_{i3}, lower), during the first 30 s at the beginning of learning (left) and during the final 30 s at the end of 20 min learning (right). At the beginning of learning, T_3 was composed mainly of T_{f3} and was considerably spiky. As the learning proceeded, T_{f3} decreased while T_{i3} gradually increased. After 20 min of training, T_{f3} was very small and was composed only of the local velocity feedback; hence T_3 was almost identical to T_{i3}. The time course of T_3 was smoother at the end of learning than at the beginning.

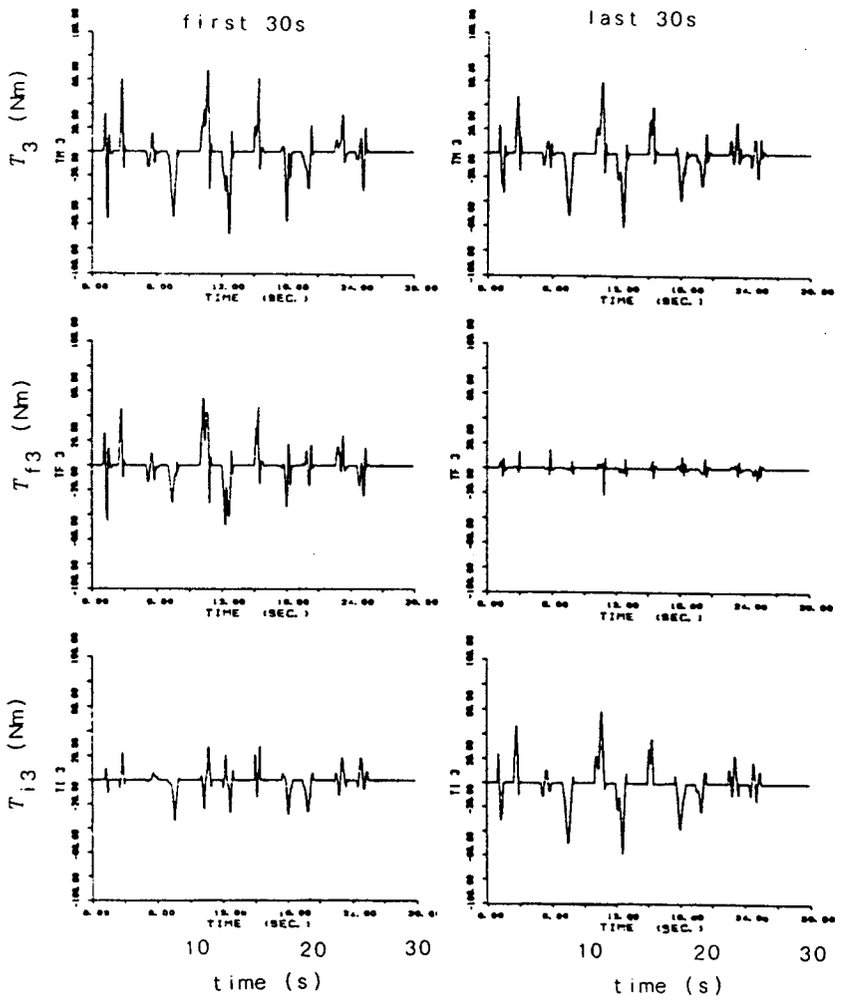

Fig. 6. The total torque (T_3), the feedback torque (T_{f3}) and the torque computed by the inverse-dynamics model (T_{i3}), which were fed to the third joint during the first 30 s of learning (left) and the last 30 s (right)

Changes of mean square errors of the outputs (i.e. mean square of feedback torques) during 20 min of learning are semilogarithmically plotted in Fig. 7, top; and mean square errors of the joint angles are plotted in Fig. 7, bottom. They were averaged values over 30 s of one training session. Note that scales of the ordinates for the three different joints were different. The mean square errors of both the torques and trajectories decreased gradually, but those of the first joint decreased more rapidly than the second and the third joints. At the end of 20 min of learning, the errors of trajectory were considerably small. Figure 5 actually plots not only the desired trajectory but also a realized trajectory during the last 30 s of the learning. But desired and real trajectories almost overlapped and could not be seen separately at this resolution.

Figure 8 shows a change of the synaptic weight w_{14} during the first 5 min of learning. The synaptic weight gradually approached an asymptotic value while showing a damped oscillation with a 30 s period of one training session. Table 3 compares the values of the synaptic weights w_{1l} at the end of 20 min training with the corresponding expansion coefficients w_{1l}^* of (6), which were calculated from physical parameters in Table 1. Some synaptic weights (e.g. w_{14}, w_{16}, w_{19}, w_{113}) have already converged quite closely to the corresponding values of w_{kl}^*; but some other synaptic weights (e.g. w_{11}, w_{13}, w_{17}) were still considerably

Table 3. Comparison of synaptic weights w_{1l} at the end of 20 min training with the corresponding expansion coefficients w_{1l}^* of (6)

l	w_{1l}	w_{1l}^*
1	0.526	0.017
2	0.976	0.382
3	− 0.451	0.007
4	0.478	0.480
5	0.048	0.003
6	0.655	0.680
7	0.655	2.75
8	1.955	0.953
9	0.718	0.680
10	0.817	0.680
11	0.466	0.953
12	1.040	0.680
13	19.917	20.0

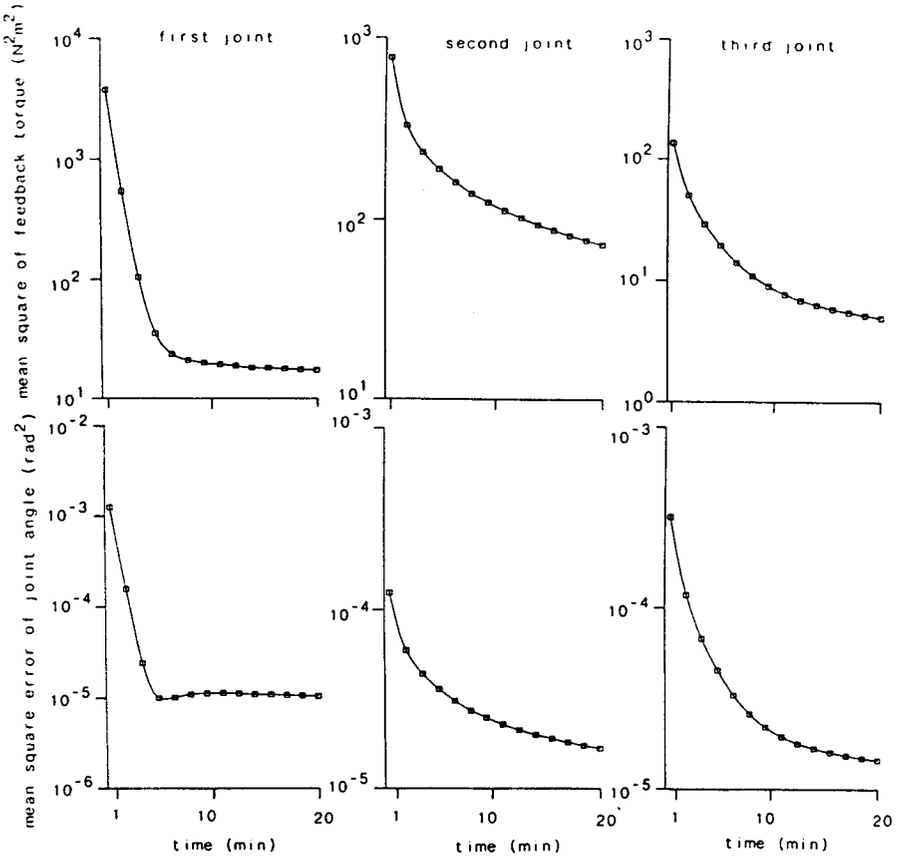

Fig. 7. Mean squares of feedback torques (top) and mean square errors of joint angles (below) during 20 min learning

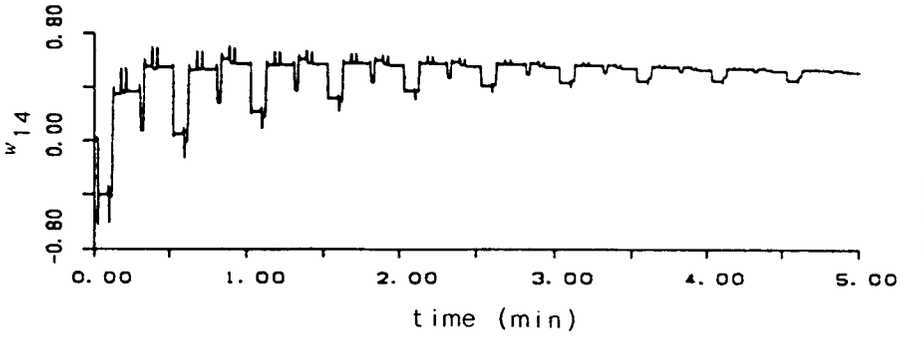

Fig. 8. Change of the synaptic weight w_{14} during the first 5 min of learning

different from them. These discrepancies could, of course, be ascribed to insufficient learning time; but they were partially due to the linear dependency of outputs of different subsystems (e.g. $f_1 = f_2 + f_3$). This linear dependency is irrelevant, since our concern is not whether w_{kl} converges to w^*_{kl}, but whether the inverse-dynamics model generates good torque (see Appendix).

We then examined whether the neural network model after 20 min of learning a single pattern could control a quite different movement which was about twice as fast as the training pattern. Figure 9 shows the desirable trajectory of this test movement (the third joint angle, broken curve), and trajectories (solid curves) realized by the neural network model without learning (top), and after learning (bottom). Before learning, delays and overshoots were often observed, and the realized trajectory deviated considerably from the desired trajectory. However, after learning, the actual trajectory almost coincided with the desired trajectory. This control capability for quite different and faster movements than the training pattern is one of the most outstanding features of our neural network model (see Discussion).

In the above experiments, we incorporated the local velocity feedback into the feedback torque, and hence we could set comparatively high feedback gains. So, the feedback control before learning was reason-

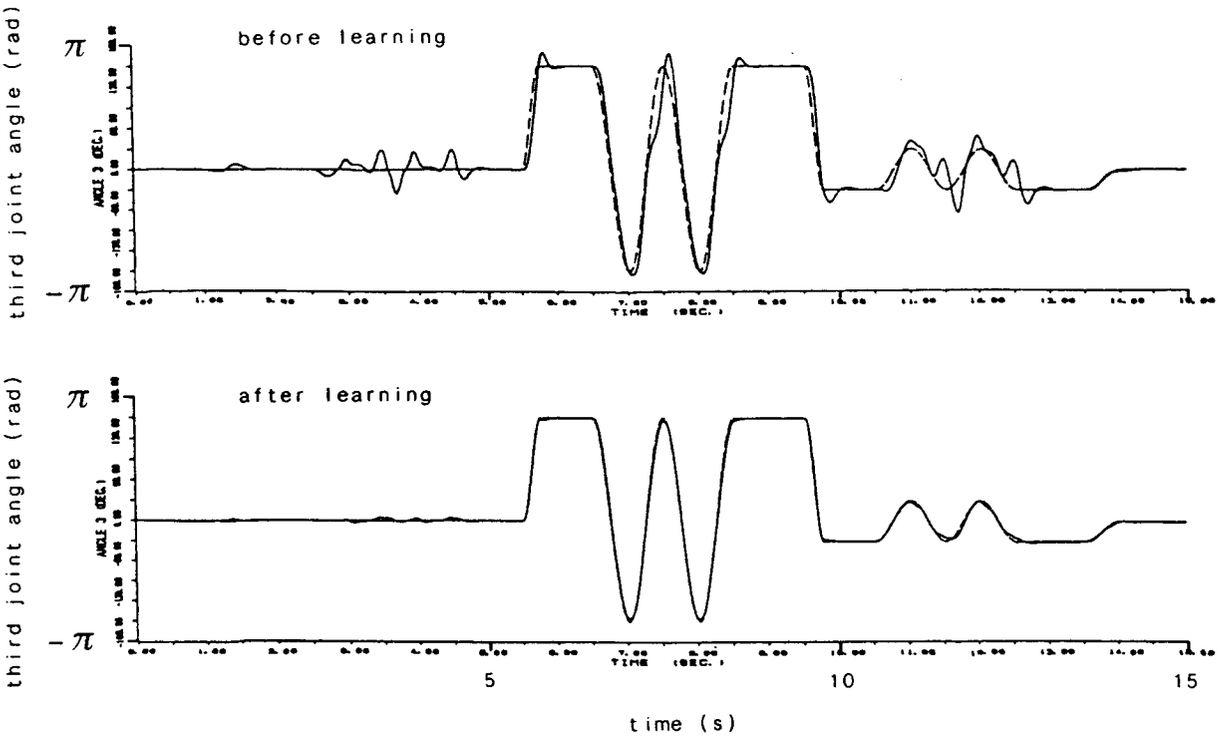

Fig. 9. Results of control for a faster and different movement from the training pattern, before (top) and after (bottom) the 20 min of learning. The desired trajectory of the third joint angle is plotted by a broken curve and the realized time courses are plotted by solid curves. They almost overlapped after learning

ably good (see Fig. 9 top). We examined whether the neural network model could learn movements even when the feedback was poorer. The feedback torque was generated only by the proportional term, and the feedback gains K_{p1}, K_{p2}, and K_{p3} were set (309.4, 120.6, 52.3) so that the damping ratio ξ was 0.5. The resulting natural frequencies ω_n were between 8 and 15. With this poorer feedback, the neural network model could not learn the desirable movement shown in Fig. 5; that is, the synaptic weights diverged. However, when a training movement pattern the same as that shown in Fig. 5 but with a duration of 1 min was chosen (i.e. the speed of the movement was reduced by half), the learning went well. After 40 min of learning, the network controlled the test pattern shown in Fig. 9 quite well. Consequently, even if the feedback was poor, the neural network model could learn slow movements, and after learning, it could also control different and faster movements.

Although the gravitational force was compensated for beforehand in almost all computer simulations, this is not indispensable. When additional subsystems which correspond to expanded terms of the gravitational force $\mathbf{G}(\mathbf{q})$ were prepared in the inverse-dynamics model, the neural network model without gravity compensation could learn the movement shown in Fig. 5.

4.3 Learning with Quasi-Periodic Movement Pattern

The movement pattern in Fig. 5 contained various elements of coordinated multi-joint movements. But, in general, it is not easy to determine such a pattern; and further, it is very unnatural for organisms to learn movement by repetitions of a single pattern. We examined a learning performance when a simple and quasi-periodic movement pattern was given as a desirable trajectory. In this pattern, the three joint angles changed as $\sin(\omega_k t)$ and the ratio of angular frequencies $\omega_1 : \omega_2 : \omega_3$ were set as $1 : \sqrt{2} : \sqrt{3}$, so that various coordinated movements were experienced during learning. The feedback gains, the time constant of learning and the learning duration were the same as the first simulation experiment (Figs. 6–8). However, since the manipulator did not hold a constant posture in this training pattern, the local velocity feedback did not work during learning.

Figure 10 shows changes of mean square errors of the output (i.e. the mean square of the feedback torques) during 20 min of learning. The mean square errors about the three joints all decreased monotonically and the learning went well. Since the final feedback torques were smaller than those in Fig. 7 by order of 10^{2-3}, it first appears that the learning with the quasi-periodic pattern was much more efficient.

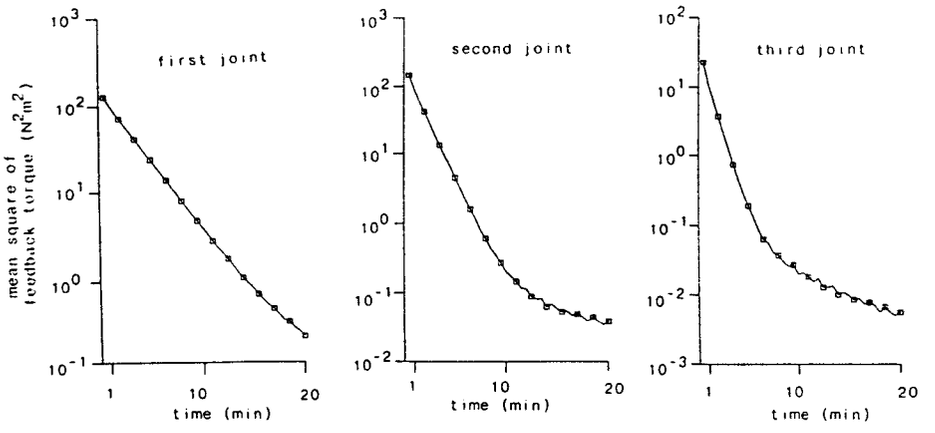

Fig. 10. Changes of mean squares of the feedback torques during 20 min of learning with the quasi-periodic training pattern

However, we can not compare control performances using the two different training patterns. Thus, we examined control performance using the test pattern of Fig. 9 after 20 min of quasi-periodic learning. The trajectory realized was almost identical to the desired trajectory, and we could not simply determine whether this result was better than the result shown in Fig. 9 (repetitive learning). Regarding the first and the second joints, the quasi-periodic learning seemed better; while for the third joint, the repetitive learning seemed better.

4.4 Learning with Redundant Subsystems

We conveniently chose necessary and sufficient subsystems, as shown in Table 2, from an expansion of the manipulator dynamics Eq. (6) (although some of them were linearly dependent; see Sect. 4.2 and Appendix). From an engineering point of view, we sometimes need to control a very complex system, even when its fundamental dynamical structure is unknown. Furthermore, as a neural network model, it is too idealistic to assume that necessary and sufficient subsystems are inherently prepared in the CNS. Thus, we examined learning performance when 20 extra subsystems, shown in Table 4 (f_l and g_l were the same) were added to those of Table 2. The idea is that the CNS can inherently prepare a very large number of subsystems which are highly redundant but which include a relatively small number of the essential subsystems.

Table 4. Nonlinear transformations of 20 extra subsystems which were added to the original subsystems of Table 2

l	$f_l(q_1, q_2, q_3)$, $g_l(q_1, q_2, q_3)$
14	$0.01 \cdot \dot{q}_1^3$
15	$0.01 \cdot \dot{q}_2^3$
16	$0.01 \cdot \dot{q}_3^3$
17	q_1^2
18	q_2^2
19	q_3^2
20	$\sin q_2 \cos(q_2+q_3)$
21	$\cos q_2 \sin(q_2+q_3)$
22	$\sin q_2 \cos q_3$
23	$\cos q_2 \sin q_3$

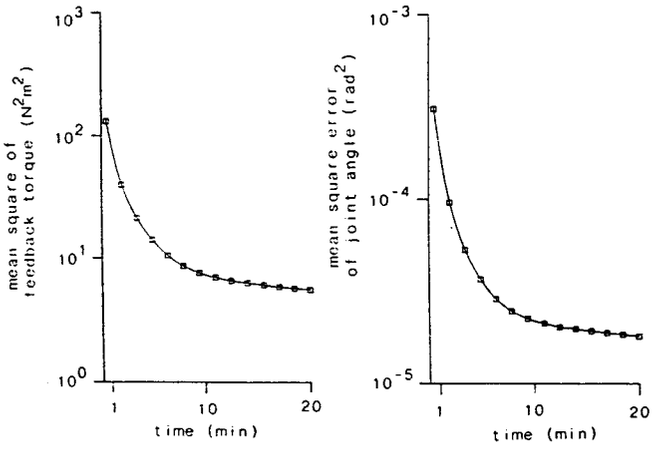

Fig. 11. Changes of the mean square of the feedback torque fed to the third joint (left) and the mean square error of the third joint angle (right) when the extra subsystems were added to the original subsystems

The feedback torques, the feedback gains, the time constant of learning and the desirable trajectory were the same as the first simulation experiment (Figs. 5–7). Figure 11 shows changes of the mean-square feedback torque fed to the third joint (left), and the mean square error of the third joint angle (right) during 20 min of learning. Both of them decreased monotonically and were not significantly larger than the errors shown in Fig. 7. In Table 5, 23 synaptic weights w_{1l} at the end of learning are shown. The corresponding coefficients w_{1l}^* are the same as in Table 3 for $l=1-13$ and are 0 for $l=14-23$. Most of the unnecessary synaptic weights were close to zero; but, for example, w_{114} was considerably large. Probably, this did not severely interfere with the overall performance of the inverse-dynamics

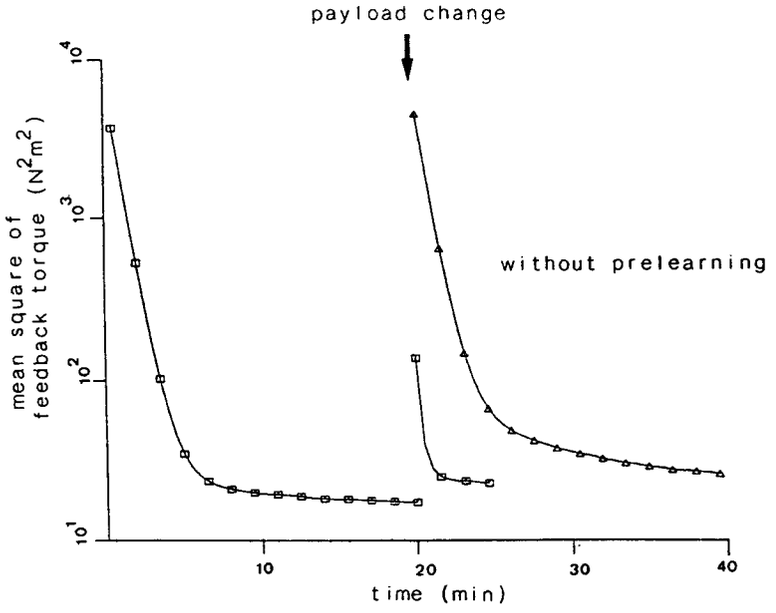

Fig. 12. Changes of the mean square of the feedback torque fed to the first joint during 40 min of learning. The mass of the payload suddenly changed from 1 kg to 3 kg, 20 min after the beginning of learning. Triangles show results without the 20 min of pre-learning

Table 5. Values of 23 synaptic weights w_{1l} regrading the first joint at the end of 20 min of learning when the extra subsystems were added to the original subsystems

l	w_{1l}	l	w_{1l}
1	0.440	13	16.472
2	0.892	14	6.417
3	− 0.452	15	− 0.350
4	0.365	16	− 0.575
5	0.075	17	− 0.001
6	0.933	18	0.099
7	2.065	19	− 0.065
8	0.516	20	0.240
9	1.064	21	− 0.034
10	1.506	22	0.142
11	0.891	23	0.011
12	0.824		

model as shown in Fig. 11, since the following approximation might hold within a working range of the manipulator: $20q_1 = 16.47q_1 + 6.42q_1^3$ (compare $w^*_{113} = 20$ and $w^*_{114} = 0$ with w_{113} and w_{114} of Table 5). In summary, if a sufficient number of subsystems are prepared, which can be done even from a very incomplete knowledge of the dynamics of a controlled system, the neural network model can efficiently learn and control movements.

4.5 Change of Manipulator Dynamics during Learning

We studied the control performance of the neural network model when a physical parameter of the manipulator suddenly changed. The feedback torques, the feedback gains, the time constant of learning and the desirable trajectory were the same as the first simulation experiment (Figs. 5–7). After 20 min of learning, the mass of the payload changed from 1 kg to 3 kg, and the simulation was continued for another 20 min. Figure 12 shows a change of the mean square of the feedback torque fed to the first joint (rectangles), and that of a naive neural network model without pre-learning (triangles). As can be seen, with pre-learning, the error quickly returned to a previous level within 5 min, and it was significantly smaller than the error without pre-learning. It can be said that the model has good adaptability to changes of the physical parameters of the manipulator.

5 Discussion

A hierarchical neural network model was proposed, based on physiological information and previous models. It contains internal neural models of dynamics and inverse-dynamics of the musculoskeletal system as essential learning parts. The potential of this model to control a complex nonlinear object was demonstrated by computer simulation with the following results.

(1) The dynamics model (spinocerebellum – magnocellular red nucleus) was acquired by learning while monitoring the motor command and the resulting movement.

(2) The inverse-dynamics model (cerebrocerebellum – parvocellular red nucleus) was acquired by repetitively experiencing a single motor pattern, while receiving the desired trajectory and monitoring the feedback torque as an "internal" error signal.

(3) As motor learning proceeded, the inverse-dynamics model gradually took the place of the external feedback as a main controller.

(4) Once the neural network model learned some movement, it could control quite different and faster movements. So, the present model is totally different from previous "table look-up" learning proposed by Albus (1975) or Raibert (1978), because of its capability of generalizing learned movements. The reason is because the present model learns the dynamics and inverse-dynamics of a control object instead of a specific motor command for a specific movement pattern.

(5) The total torque fed to the manipulator was by no means a good teacher at the beginning of learning; hence, quick movements could not be realized by it. However once the inverse-dynamics model was acquired while being supervised by this teacher, it could control fast movements smoothly. The student eventually surpassed the teacher.

(6) The model had adaptability to a sudden change in the dynamics of the controlled system.

(7) Even when redundant subsystems were added to the inverse-dynamics model, learning performance remained essentially unchanged.

Although we did not simulate the learning performance of the whole neural network model shown in Fig. 2, as learning proceeds, the internal feedback loop is first expected to take the role of the external feedback loop as the main controller; then the inverse-dynamics model is expected to take the part of the internal feedback loop. This upward shift of a dominant controller in the hierarchical neural network might reflect the phylogenesis of the motor nervous system in vertebrates. If a perfect inverse-dynamics model is formed, neither the internal nor the external feedback controls function, since movement error is absent. On the other hand, even when a perfect dynamics model is formed, the internal feedback loop suffers from limitations inherent in feedback control. So, at first sight, the dynamics model seems to do things by halves. However, it plays an essential role in providing the inverse-dynamics model with a good teaching signal. In the second simulation experiment described in Sect. 4.2, the synaptic weights diverged because the feedback torque was too poor. In order to avoid the failure of learning, either the training movement must be slowed down, the time constant of learning must be lengthened or the teaching signal must be improved. Consequently, the dynamics model makes it possible for the inverse-dynamics model to learn a relatively quick movement during a relatively short time.

In this paper, we have mainly studied free movements. It is worthwhile considering constrained movements also, such as carrying a book or exerting a force against a door, since most of our voluntary movements are related to the manipulation of external objects. In Fig. 13 we propose an expansion of the inverse-dynamics model of Fig. 4c. Figure 13 shows the single input-output case; but it is easily expanded to the multiple inputs-outputs case. The basic idea is that several sets of subsystems are prepared for different situations of movement. The selection of the sets used in control and learning of a certain movement is specified by higher motor centers (e.g. association cortex). During free movement, only a fundamental set, which is equivalent to Fig. 4c, is used both for control and learning. On the other hand, during movement when holding a light-weight object, the second set of subsystems is also recruited, and only the synaptic weights of this set are subjected to modification, while the synaptic weights of the fundamental set remain unchanged. The second set compensates for torques necessary to carry a light-weight object. When the hand releases the object, the second set of subsystems is instantaneously suppressed (or not excited); then, the inverse-dynamics model can control free arm-movement as well as before. Similarly, during movement while gripping a heavy object, the third set is recruited (it might be even more efficient if higher motor centers transmit the estimated mass of a gripped object to the inverse-dynamics model). Further, if a hand needs to exert a force on an object, a quite different set of subsystems is recruited, which receives informations about the desirable forces as well as the desirable trajectory, and generates torques necessary to exert the required force at the hand. The merit of possessing several sets of subsystems is twofold. First, the internal model can preserve various dynamical information about the musculoskeletal system combined with its different environments (e.g. grasped objects or the knob of a door) as synaptic weights (on the contrary, dynamical information about free movement was lost in Fig. 12 after the payload changed). Second, a good motor performance is instantaneously realized as soon as an experienced behavioral situation is once again given.

For control of the 3 degrees of freedom (d.g.f.) manipulator, 3 output neurons and 26 subsystems were used in the inverse-dynamics model. In general, n output neurons are required to control an object with n d.g.f. For a 6 d.g.f. manipulator, about 900 subsystems are required (Setoyama 1987). The number of necessary subsystems is expected to increase in an order of n^4. A human arm has 7 d.g.f. but the number of muscles is much higher. So, the number of necessary subsystems for the inverse-dynamics model of a human arm may reach astoronomical figures. Based on these considerations we can predict possible divergence and convergence of neural projections within the internal neural models. Each subsystem needs to receive the

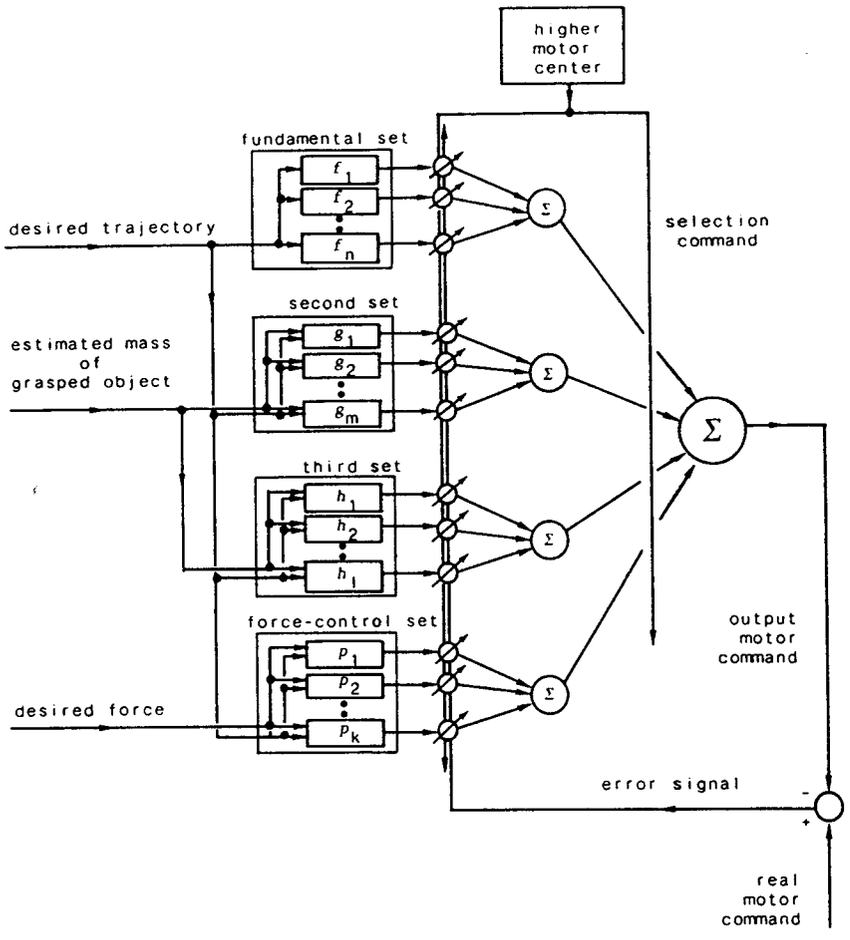

Fig. 13. Internal structure of an expanded inverse-dynamics model, which can cope with various behavioral situations such as carrying an object or exerting a force against an object, as well as with free movement

desirable time courses of lengths of several muscles whose contractions interact with each other in movements. Hence, input signals to the subsystems (desirable trajectory) extensively diverge with moderate convergence. Then, the outputs of the subsystems intensively converge onto a relatively small number of output neurons. There are two possibilities about how the CNS computes nonlinear transformations of the subsystems. One is that they are realized by neural circuits. The other is that they are computed by nonlinear information processing within the dendrites of neurons (Poggio and Torre 1981; Kawato et al. 1984). Although it is possible that the subsystems are genetically prepared, the other possibility, that the subsystems themselves are acquired by the synaptic plasticity of the Purkinje cells is appealing, since within the internal neural models both the Purkinje cells and red nucleus neurons have heterosynaptic plasticity. In this case, we need to investigate the performance of learning in the two-layered adaptive neural network. This is one of our problems for the future.

Finally, we assess the present model from an engineering point of view. Feedback controls have been nearly adequate for the slow control of usual industrial robotic manipulators with high reduction ratios between joints and actuators, which dramatically reduces the nonlinearity of manipulator dynamics and interactions between different freedoms. However, for a directdrive manipulator such as used in the present simulation study, new control methods have been demanded. Although several learning and adaptive control schemes were proposed (model reference adaptive control: Dubowsky and DesForges 1979; betterment process: Arimoto et al. 1984a and 1984b; table look-up method: Albus 1975; Raibert 1978) they were at most perturbation-learning schemes. That is, experiences obtained during learning can not be used for the execution of a quite different movement. The method of computed torque (Luh et al. 1980; Hollerbach 1980) requires both strict modelling of the manipulator dynamics and the precise estimation of physical parameters, which are difficult in practice. In contrast, the present method requires neither an accurate model (see Sect. 4.4) nor parameter estimation. Further, it posseses a great ability to generalize learning. We also note that it can be easily implemented in a parallel distributed processing machine, since both the nonlinear transformations in sub-

systems and the synaptic modifications are essentially in parallel. Consequently, the computation time in a parallel machine using the present control scheme is expected to be much shorter (shorter than 1/100) than the serial methods such as the recursive scheme for computed torque of Luh et al. (1980). Moreover, it does not require the enormous memory size of the table look-up method (only 925 synaptic weights are necessary for a 6 d.g.f. manipulator; Setoyama 1987). Miyamoto, Kawato and Suzuki (1987) has successfully applied the present method to control an industrial robotic manipulator (Kawasaki-Unimate PUMA 260) with the neural network model in a microcomputer (Hewlett Packard 9000-300-320). In summary, the present method is one of the promising schemes for the future control of not only a direct drive manipulator, but also of a large-scale complex system, whose dynamics is only partially known.

Acknowledgement. We would like to thank Drs. F. Murakami and Y. Oda for discussing this work on several occasions and reading earlier versions of the manuscript.

Appendix

In this appendix we prove the convergence of the synaptic weights to optimal values, for which the mean square error of the output of the neural identifier shown in Fig. 3b is minimum. We assume that $u(t,\omega)$, and hence $x(t,\omega)$ and $z(t,\omega)$ of Fig. 3b, are strongly mixing stochastic processes (stochastic processes for which the "past" and the "future" are asymptotically independent). This assumption seems sufficiently reasonable when we consider the "randomness" of our voluntary movements. Geman (1979) showed that when the "rate" of mixing is rapid relative to the rate of change of the solution processes, the averaged deterministic Eq. (4) is a good approximation of (3). Geman proved the following theorem in a general manner.

Theorem. For all τ sufficiently large, $\sup_{t\geq 0} E[W(t,\omega)^2] < \infty$, and $\lim_{\tau\to\infty} \sup_{t\geq 0} E[\{M(t)-W(t,\omega)\}^2]=0$.

Consequently, we only need to study the averaged Eq. (4) if the time constant τ of synaptic modification is sufficiently long. Let P denote the cross-correlation vector between the input signals $X(t,\omega)$ and the desired output $z(t,\omega)$:

$$P=E[x_i(t,\omega)z(t,\omega)]^T.$$

Q denotes the symmetric and positive definite correlation matrix of the input signals $X(t,\omega)$.

$$Q\equiv\{q_{ij}\}=\{E[x_i(t,\omega)x_j(t,\omega)]\}.$$

Then, Eq. (4) can be rewritten as follows.

$$\tau dM(t)/dt=P-QM(t).$$

For simplicity, P and Q are assumed to be constant in time. This is equivalent to assuming the stationarity of the stochastic processes $X(t,\omega)$ and $z(t,\omega)$. Further, if Q is invertible, the averaged equation has the following solution.

$$M(t)=\{1-\exp(-Qt/\tau)\}Q^{-1}P.$$

Therefore, $M(t)$ asymptotically converges to $Q^{-1}P$ since Q is positive definite. From the above theorem, the synaptic weights $W(t,\omega)$ also converge to $Q^{-1}P$ in mean.

The mean square error of the output is given as follows:

$$\text{m.s.e.}=E[s(t)^2]=E[\{z(t,\omega)-X(t,\omega)^TW(t,\omega)\}^2]$$
$$=E[z^2]-2P^TW^+W^TQW.$$

Substituting the solution $M(t)$ into this equation we obtain:

$$\text{m.s.e.}_{\text{mean}}=E[z^2]-P^T\{1-\exp(-2Qt/\tau)\}Q^{-1}P.$$

It may be observed that the m.s.e. performance is a quadratic function of the synaptic weights, that is, a "bowl-shaped" surface. The gradient at any point of the performance surface is obtained by differentiating the m.s.e. with respect to the synaptic weights as follows.

$$\nabla=-2P+2QW.$$

From this equation, it is easy to see that the averaged Eq. (4) gives the steepest descent method. The optimal synaptic weights W^* and the minimum m.s.e. are obtained by setting $\nabla=0$.

$$W^*=Q^{-1}P,$$
$$\text{m.s.e.}_{\text{min}}=E[z^2]-PQ^{-1}P.$$

Consequently, we can conclude that the synaptic weights converge in mean to the optimal values for which the mean square output error is minimum.

It is worthwhile to note that the $\text{m.s.e.}_{\text{min}}$ critically depends on the choice of subsystems. For example if $X(t)=(z(t), 0, \ldots, 0)^T$ then the $\text{m.s.e.}_{\text{min}}$ is zero. On the other hand, if there are no correlations between $X(t)$ and $z(t)$, then P equals zero and the $\text{m.s.e.}_{\text{min}}$ equals $E[z^2]$.

If we feed a delayed input $u(t-\Delta t)$ to the subsystems and the desired output $z(t)$ to the modifiable synapses, then the learning rule (3) attains optimal synaptic weights $W^*_{\Delta t}$, with which $y(t)=W^TX(t-\Delta t)$ best approximates $z(t)$. If we feed input $u(t)$ to the subsystems after $W^*_{\Delta t}$ is learned and fixed, $y(t)$ clearly approximates $z(t+\Delta t)$. Consequently, in this case, the internal neural model of Fig. 3b acts as a predictor. These two separate steps (identifier mode and prediction mode) can be simultaneously realized by the following modified learning equation of (2)

$$\tau dW(t)/dt=X(t-\Delta t)[z(t)-y(t-\Delta t)]$$
$$=X(t-\Delta t)[z(t)-X(t-\Delta t)^TW(t-\Delta t)].$$

In this modified learning rule, the rate of change of the synaptic weights is proportional to the product of delayed input signal $X(t-\Delta t)$ and the error signal between desired output $z(t)$ and delayed output signal $y(t-\Delta t)$. It is worthwhile to note that the heterosynaptic modification in the red nucleus is most prominent when the conditioned stimulus ($u(t)$) precedes the unconditioned stimulus ($z(t)$) by about 100 ms (Tsukahara et al. 1981).

References

Albus JS (1971) A theory of cerebellar functions. Math Biosci 10:25–61

Albus JS (1975) A new approach to manipulator control: the cerebellar model articulation controller (CMAC). J Dyn Syst Meas Control 97:270–277

Allen GI, Tsukahara N (1974) Cerebrocerebellar communication systems. Physiol Rev 54:957–1006

Amari S (1977) Neural theory of association and concept-formation. Biol Cybern 26:175–185

Arbib MA (1981) Perceptual structures and distributed motor control. In: Brooks VB (ed) Handbook of physiology, sect 1: vol 11, part 2. American Physiol Soc, Bethesda, pp 1449–1480

Arimoto S, Kawamura S, Miyazaki F (1984a) Bettering operation of dynamic systems by learning: a new control theory for sevomechanism or mechatronics systems. 23rd IEEE Conf Des Control 2:1064–1069

Arimoto S, Kawamura S, Miyazaki F (1984b) Can mechanical robots learn by themselves; Proceedings of 2nd International Symposium on Robotics Research, Kyoto, Japan

Cheney PD, Fetz EE (1980) Functional classes of primate corticomotoneuronal cells and their relation to active force. J Neurophysiol 44:773–791

Dubowsky S, DesForges DT (1979) The application of model reference adaptive control to robotic manipulators. J Dyn Syst Meas Control 101:193–200

Eccles JC: Introductory remarks. In: Massion J, Sasaki K (eds) Cerebro-cerebellar interactions, pp 1–18. North-Holland Elsevier, Amsterdam Oxford New York, pp 1–18

Evarts EV (1981) Role of motor cortex in voluntary movements in primates. In: Brooks VB (ed) Handbook of physiology, sect 1: vol 11, part 2. American Physiol Soc, Bethesda, pp 1083–1120

Geman S (1979) Some averaging and stability results for random differential equations. SIAM J Appl Math 36:86–105

Ghez C, Fahn S (1985) The cerebellum. In: Kandel ER, Schwartz JH (eds) Principles of neural science. Elsevier, New York, pp 502–522

Gilbert PFC, Thach WT (1977) Purkinje cell activity during motor learning. Brain Res 128:309–328

Flash T, Hogan N (1985) The coordination of arm movements; an experimentally confirmed mathematical model. J Neurosci 5:1688–1703

Fujita M (1982a) Adaptive filter model of the cerebellum. Biol Cybern 45:195–206

Fujita M (1982b) Simulation of adaptive modification of the vestibulo-ocular reflex with an adaptive filter model of the cerebellum. Biol Cybern 45:207–214

Furukawa K (1984) Identification of a robotic manipulator by a neural model. Osaka Univ, Bachelor's Thesis

Hollerbach JM (1980) A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity. IEEE Trans SMC-10:730–736

Hollerbach JM (1982) Computers, brains and the control of movement. Trends Neuro Sci 5:189–192

Ito M (1970) Neurophysiological aspects of the cerebellar motor control system. Int J Neurol 7:162–176

Ito M (1984) The cerebellum and neural control. Raven Press, New York

Ito M, Shiida T, Yagi N, Yamamoto M (1974) The cerebellar modification of rabbit's horizontal vestibulo-ocular reflex induced by sustained head rotation combined with visual stimulation. Proc Jpn Acad 50:85–89

Ito M, Jastreboff PJ, Miyashita Y (1982) Specific effects of unilateral lesions in the flocculus upon eye movements in albino rabbits. Exp Brain Res 45:233–242

Ito M, Sakurai M, Tongroach P (1982) Climbing fibre induced depression of both mossy fibre responsiveness and gultamate sensitivity of cerebellar Purkinje cells. J Physiol 324:113–134

Kawato M, Hamaguchi T, Murakami F, Tsukahara N (1984) Quantitative analysis of electrical properties of dendritic spines. Biol Cybern 50:447–454

Llinás R, Walton K, Hillman D, Sotelo C (1975) Inferior olive: its role in motor learning. Science 190:1230–1231

Luh JYS, Walker MW, Paul RPC (1980) On-line computational scheme for mechanical manipulations. J Dyn Syst Meas Control 102:69–76

Marr D (1969) A theory of cerebellar cortex. J Physiol 202:437–470

Marr D (1982) Vision. Freeman, New York

Miyamoto H (1985) A motor learning model based on synaptic plasticity. Osaka University, Bachelor's Thesis

Miyamoto H, Kawato M, Suzuki R (1987) Hierarchical learning control of an industrial manipulator using a model of the central nervous system, Japan IEICE Technical Report, MBE-86–81:25–32

Poggio T, Torre V (1981) A theory of synaptic interactions. In: Reichardt WE, Poggio T (eds) Theoretical approaches in neurobiology. MIT Press, Cambridge, pp 28–46

Poirier LJ, Bouvier G, Bédard P, Bouchard R, Larochelle L, Olivier A, Singh P (1969) Essai sur les cirvuits neuronaux mipliqués dans le tremblement postural et l'hypokinesie. Rev Neurol 120:15–40

Raibert MH (1978) A model for sensorimotor control and learning. Biol Cybern 29:29–36

Sasaki K, Gemba H (1982) Development and change of cortical field potentials during learning processes of visually initiated hand movements in the monkey. Exp Brain Res 48:429–437

Sasaki K, Gemba H, Mizuno N (1982) Cortical field potentials preceding visually initiated hand movements and cerebellar actions in the monkey. Exp Brain Res 46:29–36

Setoyama T (1987) Symbolic calculation of subsystems of the neural inverse-dynamics model for a 6 degrees of freedom manipulator using REDUCE. Osaka University, Bachelor's Thesis

Tsukahara N (1981) Synaptic plasticity in the mammalian central nervous system. Annu Rev Neurosci 4:351–379

Tsukahara N, Kawato M (1982) Dynamic and plastic properties of the brain stem neuronal networks as the possible neuronal basis of learning and memory. In: Amari S, Arbib MA (eds) Competition and cooperation in neural nets. Springer, Berlin Heidelberg New York, pp 430–441

Tsukahara N, Oda Y, Notsu T (1981) Classical conditioning mediated by the red nucleus in the cat. J Neurosci 1:72–79

Uno Y, Kawato M, Suzuki R (1987) Formation of optimum trajectory in control of arm movement – minimum torque-change model – Japan IEICE Technical Report MBE 86–79:9–16

Widrow B, McCool JM, Larimore MG, Johnson CR (1976) Stationary and nonstationary learning characteristics of the LMS adaptive filter. Proc IEEE 64:1151–1162

Wilson HR, Cowan JD (1972) Excitatory and inhibitory interactions in localized populations of model neurons. Biophys J 12:1–24

Received: April 3, 1987

Dr. M. Kawato
Department of Biophysical Engineering
Faculty of Engineering Science
Osaka University, Toyonaka
Osaka
560 Japan