TR－A－0135

Recognition of Manipulated Objects by Motor Learning with Modular Architecture Networks

Hiroaki Gomi　　Mitsuo Kawato

1992. 3. 4

ATR視聴覚機構研究所

〒619-02 京都府相楽郡精華町光台2-2　　☎07749-5-1411

ATR Auditory and Visual Perception Research Laboratories

2-2, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan

Telephone: +81-7749-5-1411
Facsimile: +81-7749-5-1408

Recognition of Manipulated Objects by Motor Learning with Modular Architecture Networks

Hiroaki Gomi Mitsuo Kawato

ATR Auditory and Visual Perception Research Laboratories,
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-02, Japan
phone: +81-7749-5-1453 (GOMI), +81-7749-5-1452 (KAWATO)
Fax: +81-7749-5-1411

Abstract

For recognition and control of multiple manipulated objects, we present two learning schemes for neural-network controllers based on *feedback-error-learning* and modular architecture. In both schemes, the network consists of a recognition network and modular control networks. In the first scheme, a Gating Network is trained to acquire object-specific representations for recognition of a number of objects (or sets of objects). In the second scheme, an Estimation Network is trained to acquire function-specific, rather than object-specific, representations which directly estimate physical parameters. Both recognition networks are trained to identify manipulated objects using somatic and/or visual information. After learning, appropriate motor commands for manipulation of each object are issued by the control networks which have a modular structure. By simulation of simple examples, the potential advantages and disadvantages of the two schemes are examined.

Keywords

Modular architecture, Object manipulation, Feedback-error-learning, Gaussian mixture, Multi-modal control, Somatosensory information

Acknowledgment :

We would like to thank Drs. E. Yodogawa and K. Nakane of ATR Auditory and Visual Perception Research Laboratories for their continuing encouragement. This work was supported by HFSP Grant to M.K.

Contents ..... page

1 INTRODUCTION 4
2 RECOGNITION OF MANIPULATED OBJECTS 5
3 MODULAR ARCHITECTURE USING A GATING NETWORK 6
4 SIMULATION OF OBJECT MANIPULATION BY MODULAR ARCHITECTURE WITH A GATING NETWORK 8
4.1 SOMATIC INFORMATION FOR GATING NETWORK 9
4.2 VISUAL INFORMATION FOR GATING NETWORK 14
4.3 SOMATIC & VISUAL INFORMATION FOR THE GATING NETWORK 16
4.4 UNKNOWN OBJECT RECOGNITION BY USING SOMATIC INFORMATION 18
5 MODULAR ARCHITECTURE USING ESTIMATION NETWORK 20
6 DISCUSSION 24
6.1 ACQUISITION OF TASK-BASED REPRESENTATION 24
6.2 CONVERGENCE RATE BY FEEDBACK-ERROR-LEARNING 25
6.3 LIMITATION OF THE NUMBER OF MANIPULATED OBJECTS AND THE NUMBER OF ESTIMATION PARAMETERS 25
6.4 MODULAR ARCHITECTURE AS A STRUCTURAL CONSTRAINT FOR NETWORK DESIGN 26
7 CONCLUSION 26

1 INTRODUCTION

In previous studies of the adaptive/learning motor control using a neural network model, Barto et al. (1983), Jordan (1988) and Psaltis et al. (1987) addressed the problem of how to obtain the error signal for a neural-network-feedforward controller. In supervised learning (Barto, 1989), the difference between the desired response and the actual response cannot directly be used as the error for controller adaptation. The error for controller adaptation should not be trajectory error (plant performance error) but command error.

As one possible solution to this problem, Kawato et al. (1987) proposed a learning method to acquire a feedforward controller, which uses the output of a feedback controller as the error for training a neural network model. They called this learning method *feedback-error-learning*. Using this method, the neural network model for feedforward control acquires an inverse dynamics model of a controlled object. They successfully applied the *feedback-error-learning* scheme to several objects (Miyamoto et al., 1988; Kawato, 1990; Katayama & Kawato, 1991). Additionally, Kawato (1990) made clear the difference between several methods to train the neural network feedforward controller.

However, conventional feedforward neural-network controllers (Barto et al., 1983; Psaltis et al., 1987; Kawato et al., 1987, 1990; Jordan, 1988; Katayama & Kawato, 1991) can not cope with multiple manipulated objects or disturbances because they cannot immediately change the control law corresponding to different objects. In interaction with manipulated objects or, in more general terms, in interaction with an environment which contains unpredictable factors, feedback information is essential for control and object recognition. From these considerations, Gomi & Kawato (1990) have examined the adaptive-feedback-controller learning schemes using *feedback-error-learning*, from which *impedance control* (Hogan, 1985) can be obtained automatically. However, in that scheme, some higher system needs to supervise the setting of the appropriate mechanical impedance for each manipulated object or environment.

In this paper, we introduce semi-feedforward control schemes using neural networks which receive feedback and/or feedforward information for recognition of multiple manipulated objects based on both *feedback-error-learning* and modular

network architecture. These schemes have two advantages over previous ones as follows. (1) Learning is achieved without the exact target motor command vector, which is unavailable during supervised motor learning. (2) Although somatic information alone was found to be sufficient to recognize objects, object identification is predictive and more reliable when both somatic and visual information are used.

2 RECOGNITION OF MANIPULATED OBJECTS

The most important issues in object manipulation are (1) how to recognize the manipulated object and (2) how to achieve uniform performance for different objects. For the first issue, there might be several ways to acquire helpful information for recognizing manipulated objects. Visual information and somatic information (i.e. signals obtained from motor commands and performance outcomes during execution of movements) are most informative for object recognition for manipulation. For the second issue, the motor commands should be changed for different objects when the same performance is required. For example, when I lift a cup from a table to my mouth, the motor commands for the muscles depend on the quantity of the contents. If appropriate motor commands are not sent, I am likely to spill some of the contents. In manipulation of a particular object, the manipulated object should be clearly recognized by integrating several kinds of information, and the appropriate motor command should be generated based on the characteristics of the recognized object (i.e. dynamical properties and kinematical properties, etc.) to achieve good performance.

How should these two abilities be obtained in motor learning for object manipulation? The physical characteristics useful for object manipulation such as mass, softness, slipperiness and center of gravity, can not be predicted without the experience of manipulating similar objects. It is also unknown what kind of information is important for a particular manipulation task for which there is no experience. In this respect, object recognition for manipulation should be learned through object manipulation. Additionally, the ability to generate proficient motor commands to achieve uniform performance for different objects, should also be simultaneously acquired through object manipulation.

Manipulated object recognition has been also investigated in robotics research.

In many cases, the object recognition problem is examined as the object's 3D-shape reconstruction by using vision and/or tactile sensing (Allen, 1987). 3D-shape representation is heuristically adopted as the internal representation for manipulation planning. Of course, 3D-shape representation of objects is sufficiently available for manipulation planning, but it is still an open question whether this internal representation is effective for human-like manipulation or not. Furthermore, object recognition problems are discussed separately from the control problems (i.e. motor command generation problems) in which recognized properties should be utilized. In these cases, straightforward calculations of some parameters such as grasping points, hand shape and input forces for each grasping point, etc. were made based on dynamic and kinematic properties taking into consideration the object shape, center of gravity and surface condition etc.. Thus it was difficult to effectively change the internal representations of objects by taking account of the final performance error.

The computational models of manipulation learning introduced in the following sections are examined as combined recognition and control problems.

3 MODULAR ARCHITECTURE USING A GATING NETWORK

Jacobs et al. (Jacobs et al., 1990; Jacobs, Jordan, 1991) and Nowlan (Nowlan, 1990; Nowlan & Hinton, 1991) have proposed a competitive modular network architecture which was applied to the task decomposition problem or classification problems. Jacobs & Jordan (1991) applied this network architecture to the multi-payload robotics task in which each expert network controller was trained for each category of manipulated objects in terms of the object's mass. In their scheme, the payload's identity was fed to the gating network to select a suitable expert network which acts as a feedforward controller.

We examined modular network architecture using *feedback-error-learning* for simultaneous learning of object recognition and the control task as shown in Figure 1. In our cases, several physical properties, not only mass but also viscosity and stiffness, are different in each object. In this learning scheme, the quasi-target vector for combined output of expert networks is employed instead of the exact target vector. This is because it is unlikely that the exact target motor command

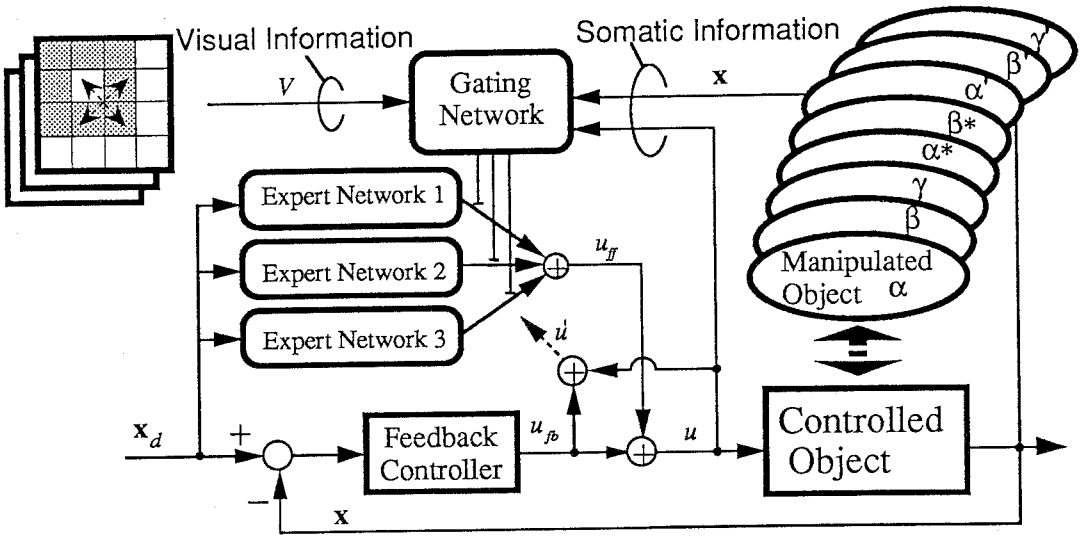


Figure 1 Configuration of the modular architecture using a Gating Network for object manipulation based on feedback-error-learning.

vector can be provided in supervised motor learning (Kawato, 1990; Jordan, 1988). The quasi-target vector of feedforward motor command, u' is produced by :

$$u' = u + u_{fb}. \tag{1}$$

Here, u denotes the motor command fed to the controlled object and u_{fb} denotes the feedback motor command. Using this quasi-target vector, the gating and expert networks are trained to maximize the log-likelihood function, $\ln L$, by using backpropagation.

$$\ln L = \ln \sum_{i=1}^{n} g_i e^{-\|u'-u_i\|^2/2\sigma_i^2} \tag{2}$$

Here, u_i is the i th expert network output, σ_i is a variance scaling parameter of the i th expert network and g_i, the i th output of the gating network, is calculated by

$$g_i = \frac{e^{s_i}}{\sum_{j=1}^{n} e^{s_j}}, \tag{3}$$

where s_i denotes the weighted input received by the i th output unit. The total output of the modular network, u_{ff}, is

$$u_{ff} = \sum_{i=1}^{n} g_i u_i . \tag{4}$$

The adaptation rules of the weights in each network are derived from the partial derivative of Equation 2 with respect to each weight, and are represented as follows.

$$\left.\begin{aligned} \frac{dw_{\text{gate}}}{dt} &= \eta_{\text{gate}} \sum_{i=1}^{n} \frac{\partial s_i}{\partial w_{\text{gate}}} \left(g(i|X,u') - g_i \right) \\ \frac{dw_{\text{expert}\,i}}{dt} &= \eta_{\text{expert}\,i} \frac{\partial u_i}{\partial w_{\text{expert}\,i}} g(i|X,u') \frac{(u'-u_i)}{\sigma_i^2} \end{aligned}\right\} \tag{5}$$

where w_{gate} denotes the gating network weight, $w_{\text{expert}\,i}$ denotes the i th expert network weight, $\eta_{\text{gate}}, \eta_{\text{expert}\,i}$ determine the learning rate in each network and $g(i|X,u')$ is the following equation corresponding to posterior probability.

$$g(i|X,u') = \frac{g_i e^{-\|u'-u_i\|^2/2\sigma_i^2}}{\sum_{j=1}^{n} g_j e^{-\|u'-u_j\|^2/2\sigma_j^2}} \tag{6}$$

where X denotes the input vector of the gating network. By maximizing Equation 2 using the steepest ascent method (i.e. adaptation rule Equation 5), the gating network learns to choose the expert network whose output is closest to the quasi-target command, and each expert network is tuned correctly when it is chosen by the gating network. The desired trajectory is fed to the expert networks so as to make them work as feedforward controllers.

4 SIMULATION OF OBJECT MANIPULATION BY MODULAR ARCHITECTURE WITH A GATING NETWORK

We show advantages of the learning schemes presented above by the following simulation results. The configuration of the controlled object and manipulated object is shown in Figure 2 in which M, B, K respectively denote the mass, viscosity and

stiffness of the coupled object (controlled- and manipulated-object). The manipulated object is changed every epoch (1 [sec]) while the coupled object is controlled to track the desired trajectory. Figure 3 shows the selected object in each epoch (a), the feedforward and feedback motor commands (b), and the desired and actual trajectories (c) before learning.

The desired trajectory vector, $\mathbf{x}_d$,which consists of acceleration, velocity and position, was produced by the Ornstein-Uhlenbeck random process. As shown in Figure 3, the error between the desired trajectory and the actual trajectory remained because the feedback controller in which the gains were fixed, was employed in this condition. Physical characteristics, *M B K*, of the objects used are listed in Figure 6.

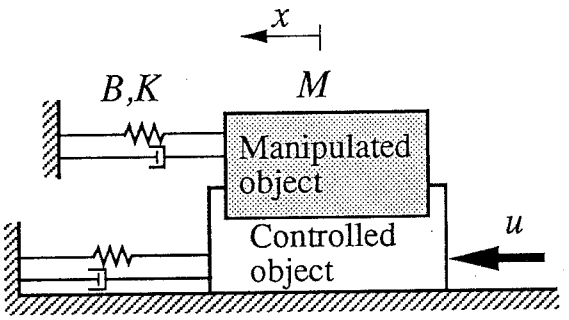


Figure 2 Configuration of the controlled object and the manipulated object.

4.1 Somatic information for gating network

We call the actual trajectory vector, **x**, (this vector consists of acceleration, velocity and position) and the final motor command, u, "somatic information." Somatic information should be most useful for on-line (feedback) recognition of the dynamical characteristics of manipulated objects. The latest four discrete time data (sampling period is 2 [msec]) of the somatic information were used as the gating network inputs for identification of the coupled object in this simulation. Then, s of Equation 3 is expressed as:

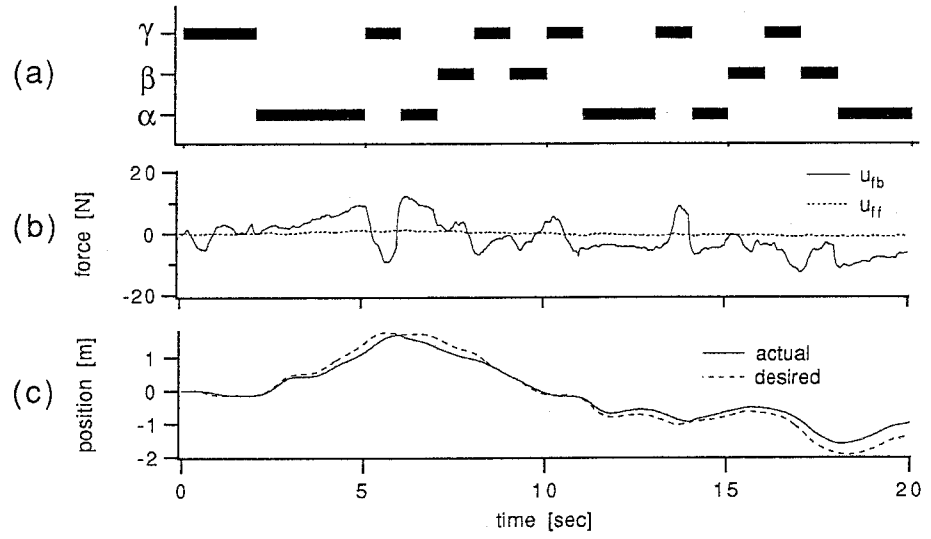


Figure 3 Temporal patterns of the objects, the motor commands, the desired and actual trajectories before learning.

$$s(t) = \psi_1(\mathbf{x}(t), \mathbf{x}(t-1), \mathbf{x}(t-2), \mathbf{x}(t-3), u(t), u(t-1), u(t-2), u(t-3)). \quad (7)$$

The task of the gating network is to divide the 16-dimensional input space into three sub-sets for each object. The physical characteristics, M, B, K, of coupled objects are listed in Figure 6. The object was changed every epoch (1 [sec]). The variance scaling parameter was $\sigma_i = 0.8$ and the learning rates were $\eta_{\text{gate}} = 1.0 \times 10^{-3}$ and $\eta_{\text{expert}i} = 1.0 \times 10^{-5}$. The three-layered feedforward neural network (input 16, hidden 30, output 3) was employed for the gating network and the two-layered linear networks (input 3, output 1) were used for the expert networks.

Figure 4 shows the time courses of the moving average of u_{fb} and u_{ff} squared, and Figure 5a-c shows the time courses of weights in each expert network during the learning phase. The feedback motor command, u_{fb}, gradually decreased during

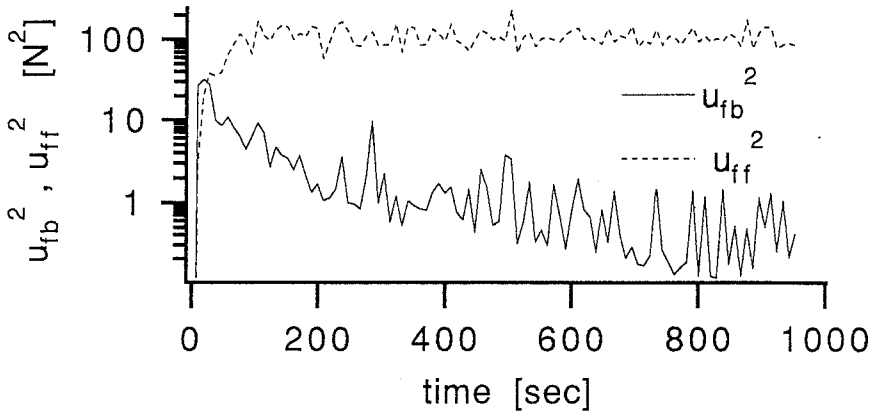


Figure 4 The time courses of the moving average of squared motor commands.

learning and the expert network weights (i.e. gains for each input) approach asymptotic values. Comparing the weights of expert networks for each input after learning and the actual physical characteristics of coupled objects listed in Figure 6, we realize that expert networks No.1, No.2, No.3 respectively obtained the inverse dynamics of coupled objects γ, β, α, which were employed during the learning phase. Figure 7 shows the time variation of the objects (a), the gating network outputs (b), motor commands (c) and trajectories (d) after learning. The gating network outputs for the objects responded correctly through out most of the test phase (compare Figure 7a with 7b). For more systematic analysis, the statistical results of correspondence are shown in Figure 6. Each bar height in Figure 6 denotes the averaged value of each gating network output while each object was selected during the test phase. As a consequence of adaptation, the feedback motor command, u_{fb}, was almost zero and the actual trajectory almost perfectly corresponded with the desired trajectory. In other words, the gating network successfully executed the 'recognition task' and each expert network acquired the 'internal model of an object for manipulation'.

Moreover, learning generalization in the state space was ascertained because the test simulation was done for the random desired trajectory (O-U process).

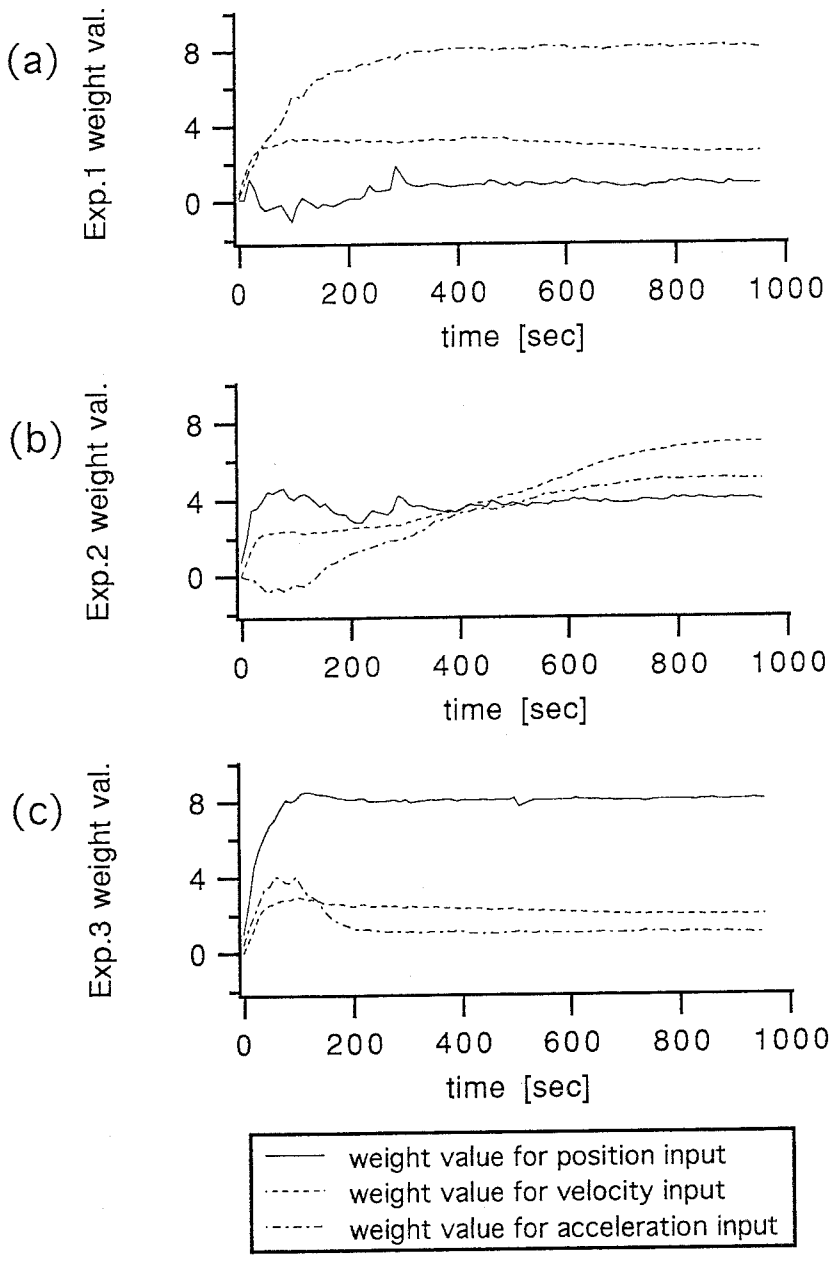


Figure 5 The time courses of Expert network weights when somatic information was used as the gating network input.

	Object physical characteristics M B K	retinal image	Expert Net. Weight values for each input, $\ddot{x}_d$ $\dot{x}_d$ x_d, after learning		
			No.1 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.2 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.3 $\ddot{x}_d$ $\dot{x}_d$ x_d
			8.1 2.5 0.87	5.0 6.9 4.0	0.97 1.9 8.0
α	1.0 2.0 8.0	none			
β	5.0 7.0 4.0	none			
γ	8.0 3.0 1.0	none			

Figure 6 Gating Network outputs vs. objects using Somatic information. (Averaged gating outputs in the test phase after learning.)

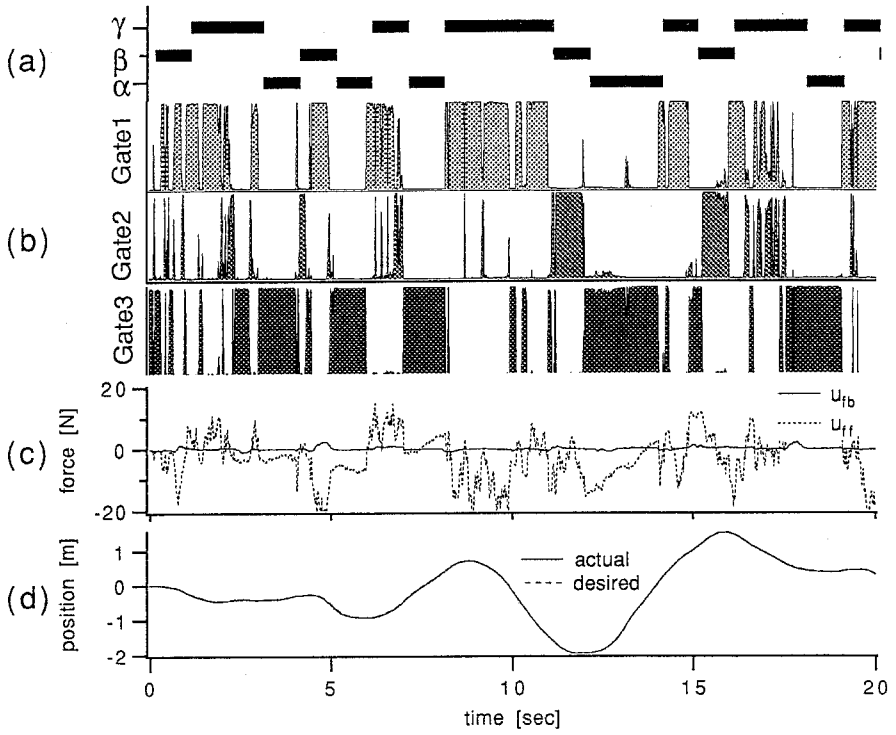


Figure 7 Temporal patterns of objects, gating outputs, motor commands and trajectories after learning using Somatic information.

4.2 Visual information for gating network

Consciously or unconsciously, we usually assume the manipulated object's characteristics by using visual information before the actual manipulation task. Visual information might be quite helpful for feedforward recognition. Object discrimination tasks from visual images using modular networks were investigated by Jacobs (1991) in which teacher signals (i.e. target object identities) were employed in training.

When visual information is available in the proposed scheme, s of Equation 3 is expressed as:

$$s(t) = \psi_2(V(t)) \,. \tag{8}$$

We used three simple visual cues corresponding to each coupled object in this simulation as shown in Figure 8. At each epoch in this simulation, one of three visual cues of the size 3×3 selected randomly is randomly placed at one of four possible locations on a 4×4 retinal matrix. Each black pixel of these cues takes the value one and each white pixel on the retinal matrix takes the value zero. The visual cues of each object are different, but objects α and α^* have the same physical characteristics, M, B, K, as shown in Figure 8. The gating network should identify the object and select a suitable expert network for feedforward control by using this visual information. The learning coefficients were $\sigma_i = 0.7$, $\eta_{\text{gate}} = 1.0\times10^{-3}$, $\eta_{\text{expert}\,i} = 1.0\times10^{-5}$. The same networks used in the above experiment were used in this simulation.

After learning, the expert network No.2 acquired the inverse dynamics of objects α and α^* (which have the same physical characteristics), and expert network No.3 accomplished this for object γ (compare object physical characteristics with expert network weights for each input listed in Figure 9). Figure 8 summarized the statistical analysis of the correspondence between objects and gating network output. The gating network almost perfectly selected expert network No.2 for object α and α^*, and almost perfectly selected expert network No.3 for object γ as shown in Figure 8 and Figure 9. Expert network No.1, which did not acquire inverse dynamics corresponding to any of the three objects, was not selected in the test phase after

learning. This is to say, the redundant expert network did not contribute to control tasks by learning as mentioned by Jacobs (1990). The actual trajectory in the test phase corresponded almost perfectly to the desired trajectory.

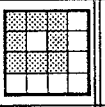
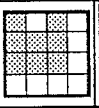
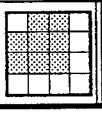
	Object physical characteristics M B K	retinal image (randomly moved)	Expert Net. Weight values for each input, $\ddot{x}_d$ $\dot{x}_d$ x_d, after learning		
			No.1 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.2 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.3 $\ddot{x}_d$ $\dot{x}_d$ x_d
			4.3 3.0 -0.34	1.2 2.0 8.0	8.0 3.0 0.99
α	1.0 2.0 8.0				
α^*	1.0 2.0 8.0				
γ	8.0 3.0 1.0				

Figure 8 Gating Network outputs vs. objects using Visual information. (Averaged gating outputs in the test phase after learning.)

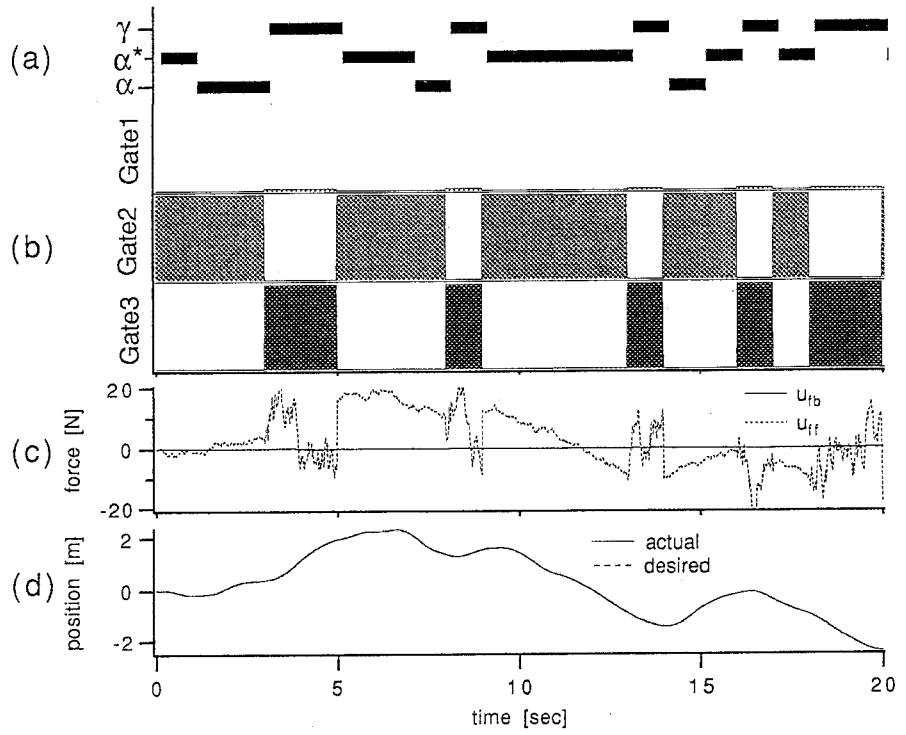


Figure 9 Temporal patterns of objects, gating outputs, motor commands and trajectories after learning using Visual information.

4.3 Somatic & visual information for the gating network

We show here the simulation results by using both somatic and visual information as the gating network inputs. In this case, s of Equation 3 is represented as:

$$s(t) = \psi_3(\mathbf{x}(t),\cdots,\mathbf{x}(t-3),u(t),\cdots,u(t-3),V(t)). \quad (9)$$

In this simulation, the objects α and β* had different physical characteristics, but shared the same visual cue as listed in Figure 10. Thus, to identify the coupled objects one by one, it is necessary for the gating network to utilize not only visual

information but also somatic information. The learning coefficients were $\sigma_i = 1.0$, $\eta_{gate} = 1.0 \times 10^{-3}$ and $\eta_{expert\,i} = 1.0 \times 10^{-5}$. The gating network had 32 input units, 50 hidden units, and 3 output units, and the expert networks were the same as in the above experiment.

After learning, expert networks No.1, No.2, No.3 acquired the inverse dynamics of objects γ, β*, α, respectively, as listed in Figure 10. As shown in Figure 11, the gating network identified the object almost correctly. As shown in the statistical analysis in Figure 10, the recognition performance (the averaged value of the gating network output to each expert network for each object) is better than that in Section 4.1. The recognition performance for object γ, however, was not as good as the result in Section 4.2, even though visual information was available for recognizing object γ. This might be because the sensory signals, visual and somatic, were not normalized or slanted properly in this simulation.

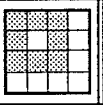
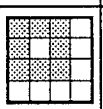
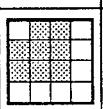
	Object physical characteristics M B K	retinal image (randomly moved)	Expert Net. Weight values for each input, $\ddot{x}_d$ $\dot{x}_d$ x_d, after learning		
			No.1 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.2 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.3 $\ddot{x}_d$ $\dot{x}_d$ x_d
			8.1 2.4 0.8	5.1 6.9 4.0	1.0 1.9 8.0
α	1.0 2.0 8.0				
$\beta*$	5.0 7.0 4.0				
γ	8.0 3.0 1.0				

Figure 10 Gating Network outputs vs. objects using Somatic & Visual information. (Averaged gating outputs in the test phase after learning.)

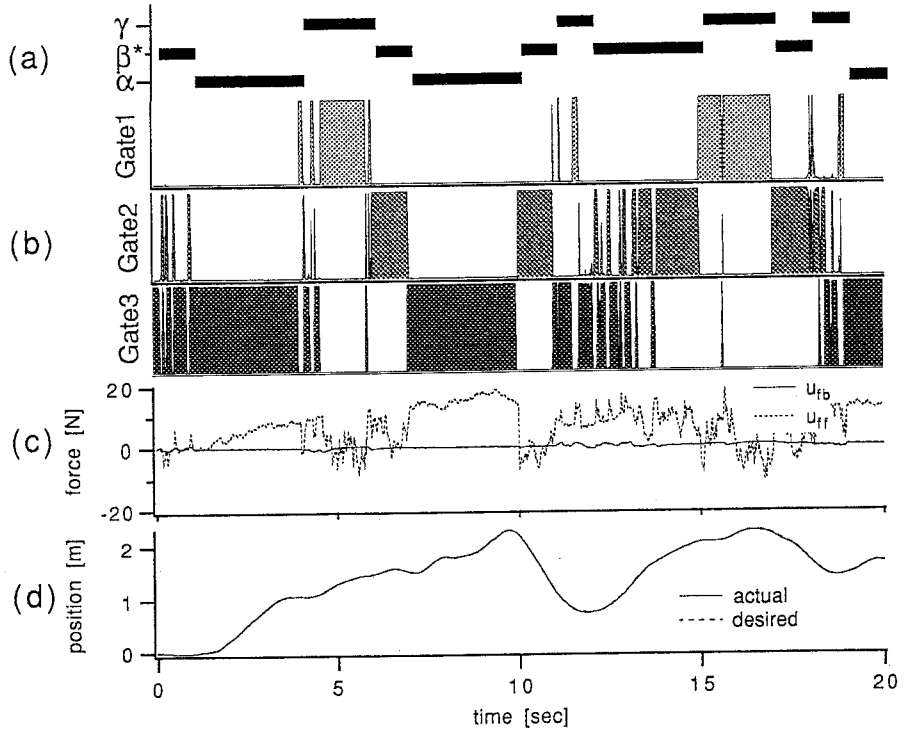


Figure 11 Temporal patterns of objects, gating outputs, motor commands and trajectories after learning using Somatic & Visual information.

4.4 Unknown object recognition by using somatic information

It is possible that the scheme shown in Figure 1 might also be applied to unknown objects because the gating network switches between the expert networks not crisply but fuzzily. This is because the desired output of the gating network has Gaussian distribution as shown in Equation 6. To examine this, we applied the modular network trained in the experiment described in section 4.1 to unknown objects. Figure 13 shows the temporal responses for unknown objects whose physical characteristics were slightly different from known objects (compare the parameters in Figure 6 and Figure 12) in the case using somatic information as the gating network inputs. Even though each tested object was not the same as any of the known (learned) objects, the gating network selected the expert network whose inverse dynamics model is the closest to the unknown object's (compare Figure 12

with Figure 6). For object α', expert network No.3 was chosen with high probability and for object β', expert network No.2 was selected in higher probability than the other two expert networks. For object γ', expert network No.1 was selected most often. As shown in Figure 13c, during some period in the test phase, the feedback command increased because of an inappropriate feedforward command.

	Object physical characteristics M B K	retinal image	Expert Net. Weight values for each input, $\ddot{x}_d$ $\dot{x}_d$ x_d, after learning		
			No.1 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.2 $\ddot{x}_d$ $\dot{x}_d$ x_d	No.3 $\ddot{x}_d$ $\dot{x}_d$ x_d
			8.1 2.5 0.87	5.0 6.9 4.0	0.97 1.9 8.0
α'	2.0 3.0 7.0	none			
β'	4.0 6.0 5.0	none			
γ'	9.0 2.0 2.0	none			

Figure 12 Gating Network outputs vs. objects during the unknown object recognition task using Somatic information. (Averaged gating outputs.)

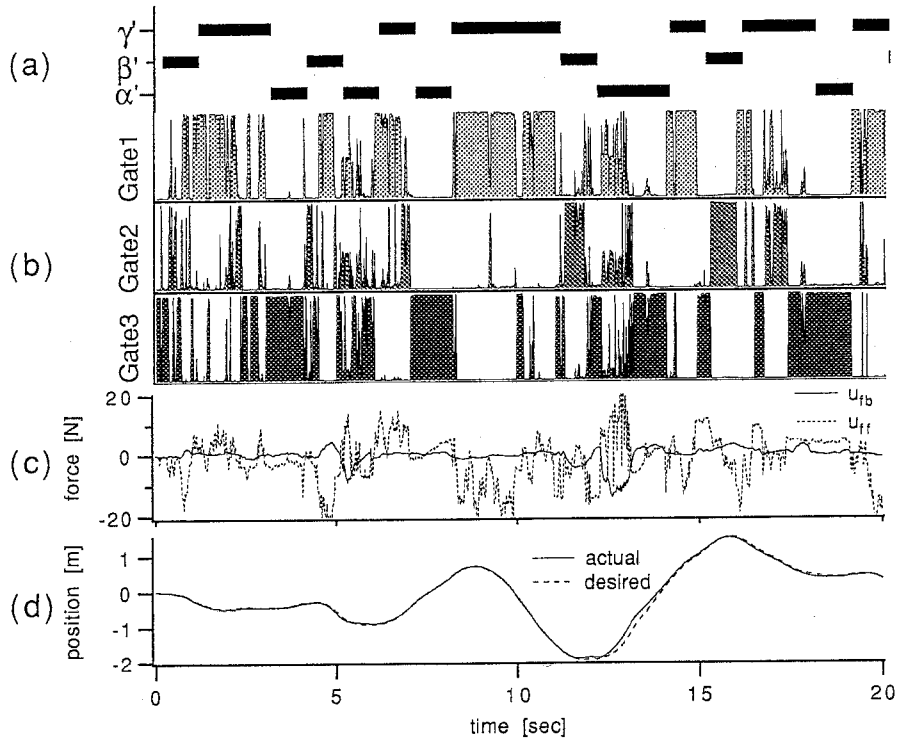


Figure 13 Temporal patterns of objects, gating outputs, motor commands and trajectories during the unknown object recognition task using Somatic information.

5 MODULAR ARCHITECTURE USING ESTIMATION NETWORK

The modular architecture shown in the above section is competitive in the sense that expert networks compete with each other to occupy a niche in the input space. We here propose a new cooperative modular architecture where expert networks specialized for different functions cooperate to produce the required output. In this scheme, estimation networks are trained to recognize physical parameters of manipulated objects by using feedback information. Using this method, an infinite

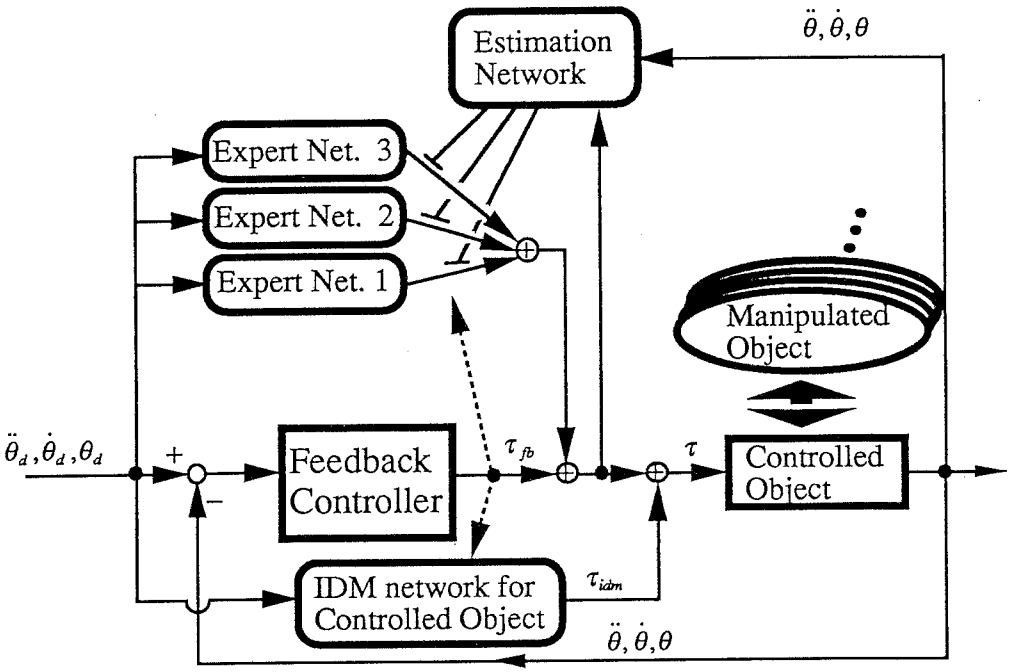


Figure 14 Configuration of the modular architecture using Estimation Network for object manipulation by feedback-error-learning.

number of manipulated objects in a limited domain can be treated by using a small number of estimation networks. This is because the value of each object's parameter is interpolated and estimated directly by the estimation networks, and the number of estimation networks is limited to the number of effective parameters necessary for controlling the target objects. Figure 14 shows the configuration of manipulation control employing this idea. In this figure, the IDM network is prepared for the controlled object (i.e. manipulator etc.) and expert networks No.1, 2, 3 are trained to represent different functional forces which correspond to different physical characteristics of manipulated objects. Each output of the expert networks is multiplied by each output of the estimation network to produce the motor commands. Each expert network works cooperatively rather than competitively. The number of outputs of the estimation network is restricted to the number of effective parameters of the manipulated objects. For example, to properly control the manipulated object which

is attached at the top of the 2-link manipulator as shown in Figure 15, the feedforward motor command, τ_{ff}, should be expressed as follows.

$$\tau_{ff} = \mathbf{J}^T\left(\mathbf{M}\ddot{x}_d + \mathbf{B}\dot{x}_d + \mathbf{K}x_d\right) + \tau_{idm} \tag{10}$$

Here,

$$\mathbf{M} = \begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix}, \mathbf{B} = \begin{bmatrix} B & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{K} = \begin{bmatrix} K & 0 \\ 0 & 0 \end{bmatrix},$$

$\mathbf{J}^T$ denotes a transposed Jacobian matrix, and x_d is the desired trajectory position vector in Cartesian space. If the IDM network perfectly compensates only for the controlled object, the appropriate feedforward motor command for a manipulated object (i.e. one that is produced by expert networks No.1, 2, 3 and the estimation network shown in Figure 14) is expressed as:

$$\tau_{mi} = M\Psi_{1i}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d) + B\Psi_{2i}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d) + K\Psi_{3i}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d). \tag{11}$$

where τ_{mi} denotes the feedforward motor command for the i th link and M, B, K, respectively denote mass, viscosity, stiffness of the manipulated object and $\ddot{\theta}_d, \dot{\theta}_d, \theta_d$, respectively denote the angular acceleration, angular velocity and angular position vector of the manipulator. Ψ_{ji} is a preprocessing nonlinear function realized by each expert network according to following equations for producing the appropriate feedforward motor command for the manipulated object.

$$\begin{aligned}&\Psi_{11}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d) \\ &= \ddot{\theta}_{d1}\left(l_1^2 + l_2^2 + 2l_1l_2\cos\theta_{d2}\right) + \ddot{\theta}_{d2}\left(l_2^2 + l_1l_2\cos\theta_{d2}\right) - \left(2\dot{\theta}_{d1} + \dot{\theta}_{d2}\right)\dot{\theta}_{d2}l_1l_2\sin\theta_{d2}.\end{aligned} \tag{12}$$

$$\begin{aligned}&\Psi_{21}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d) \\ &= \left[\dot{\theta}_{d1}\left(l_1\sin\theta_{d1} + l_2\sin(\theta_{d1} + \theta_{d2})\right) + \dot{\theta}_{d2}l_2\sin(\theta_{d1} + \theta_{d2})\right]\left(l_1\sin\theta_{d1} + l_2\sin(\theta_{d1} + \theta_{d2})\right).\end{aligned} \tag{13}$$

$$\begin{aligned}&\Psi_{31}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d) \\ &= \left(l_1\cos\theta_{d1} + l_2\cos(\theta_{d1} + \theta_{d2})\right)\left(l_1\sin\theta_{d1} + l_2\sin(\theta_{d1} + \theta_{d2})\right).\end{aligned} \tag{14}$$

$$\begin{aligned}&\Psi_{12}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d) \\ &= \ddot{\theta}_{d1}\left(l_2^2 + 2l_1l_2\cos\theta_{d2}\right) + \ddot{\theta}_{d2}l_2^2 - \dot{\theta}_{d1}^2l_1l_2\sin\theta_{d2}.\end{aligned} \tag{15}$$

$$\Psi_{22}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d)$$
$$= \dot{\theta}_{d1} l_2 (l_1 \sin\theta_{d1} + l_2) \sin(\theta_{d1} + \theta_{d2}) + \dot{\theta}_{d2} l_2^2 \sin(\theta_{d1} + \theta_{d2}). \quad (16)$$

$$\Psi_{32}(\ddot{\theta}_d, \dot{\theta}_d, \theta_d)$$
$$= l_2 (l_1 \cos\theta_{d1} + l_2 \cos(\theta_{d1} + \theta_{d2})) \sin(\theta_{d1} + \theta_{d2}). \quad (17)$$

Here, l_1, l_2, θ_{d1} and θ_{d2} respectively denote the first link length, second link length, first link desired angle and second link desired angle of the manipulator. We expect that these preprocessings are obtained in each expert network and the parameters, M, B, K, will be compellingly represented as the estimation network outputs by using the constraint of the limited number of estimation-network outputs. This expectation comes from previous studies in which the efficient and compressed representations were obtained in the hidden layer of the network when the number of hidden units was kept small (Cottrell et al. 1987; Bourlard & Kamp, 1988; Irie & Kawato, 1990).

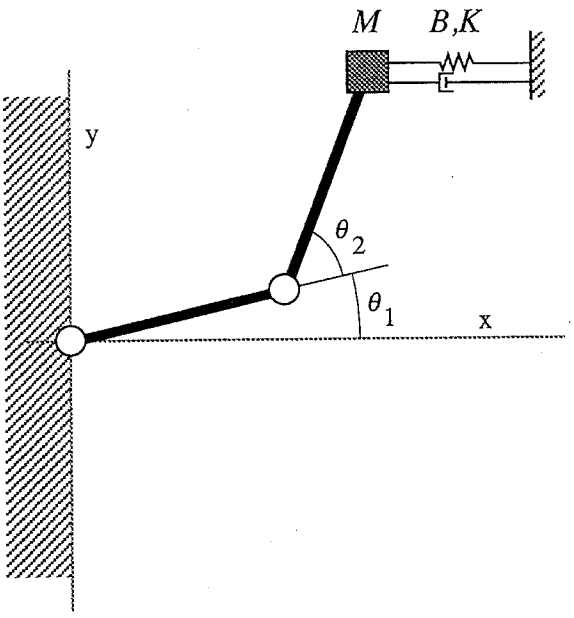


Figure 15 Configuration of the 2-link manipulator and a manipulated object.

We applied this idea to recognizing the mass of the manipulated objects in one-dimensional movement. Figure 16a shows the output of the estimation network compared to actual masses. The realized trajectory almost coincided with the desired trajectory as shown in Figure 16b. This learning scheme can be applied not only to estimating mass but also to other physical characteristics such as softness or slipperiness.

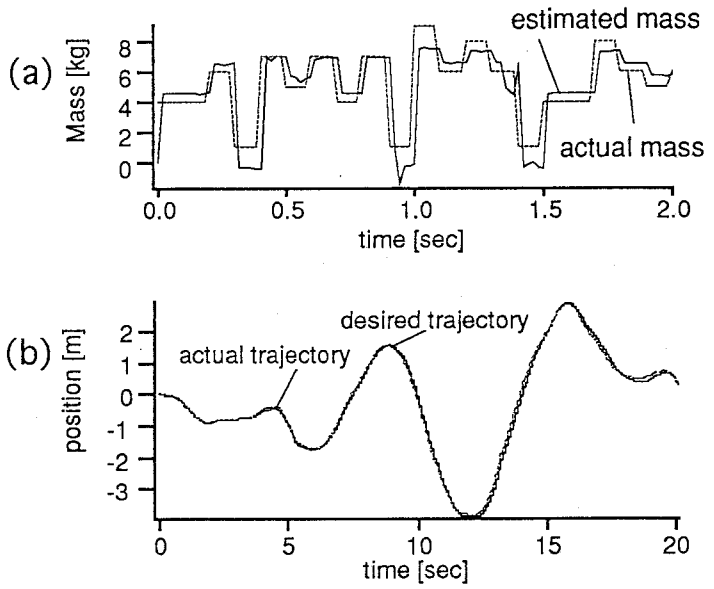


Figure 16 a. Comparison of the actual and the estimated mass. b. The desired and the actual trajectories.

6 DISCUSSION

6.1 Acquisition of task-based representation

In the simulation of manipulation learning using visual information (section 4.2), the internal models for object manipulation (in this case, inverse dynamics) were represented not in terms of visual information but rather, of motor commands. That is to say, the same internal representation is acquired when the same motor command

is required for obtaining the final performance even if input visual cues are different. This is good not only for saving the internal representation but also for deriving the common features between objects in terms of motor control. Although the current simulation is preliminary, it indicates the very important issue that task-based internal representations of objects (or environments), rather than declarative ones, were automatically acquired by motor learning. This observation supports the 'functional representation' notion in Artificial Intelligence research as a promising strategy in high-level recognition processes. For example, the abstract idea of 'chair' can not simply be acquired from many image examples of 'chair', but task-based (i.e. functional) representation, i.e., a "chair is an object on which people can sit", makes the many different shapes of chairs comprehensible.

6.2 Convergence rate by *feedback-error-learning*

The quasi-target motor command in the first scheme and the motor command error in the second scheme are not exactly correct because the proposed learning schemes are based on the *feedback-error-learning* method. Thus, the learning rates in the proposed schemes should be slower than those schemes in which exact target commands are employed (Gomi, Kawato, 1990). In our preliminary simulations of proposed modular learning schemes, they were about five times slower. However, we emphasize that exact target motor commands are not available in supervised motor learning.

6.3 Limitation of the number of manipulated objects and the number of estimation parameters

The limited number of controlled objects which can be dealt with by the modular architecture with a gating network is a considerable problem (Jacobs, 1991; Nowlan, 1990, 1991). This problem depends on choosing an appropriate number of expert networks and an appropriate value for the variance scaling parameter, σ. Once this is done, the expert networks can interpolate the appropriate output for a number of unknown objects. The simulation results in Section 4.4 show a successful example for unknown objects. Our second scheme provides a more satisfactory solution to this problem.

On the other hand, one possible drawback of the second scheme is that it may be

difficult to estimate many physical parameters for complicated objects, even though the learning scheme which directly estimates the physical parameters can handle any number of objects.

6.4 Modular architecture as a structural constraint for network design

Modular architecture is one of the structural constraints in neural network design for a particular task. As Jacobs (1990) described, modular architecture might be a good constraint for many kinds of underconstrained learning problems. If a single neural network is used for the object recognition problem shown here, learning will not be successful because the network has too many degrees of freedom. In particular, the importance of modular architecture will increase for problems in which many kinds of tasks should be learned together.

7 CONCLUSION

We showed here basic examinations of two types of modular architecture with neural networks - a gating network and a direct estimation network. Both networks can use feedback and/or feedforward information for recognition of multiple manipulated objects. In order to examine the fundamental performance of modular networks, we did not take into account the geometrical properties of manipulated objects which are important in deciding the grasping points and forming the hand shape. In the future, we will attempt to integrate two proposed network architectures and to advance this combined architecture with sophisticated visual processing in order to model tasks involving skilled motor coordination and high level recognition.

References

Allen, P.K. (1987). *Robotic object recognition using vision and touch.* Kluwer Academic Publishers.

Barto, A.G., Sutton, R.S., Anderson, C.W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. on Sys. Man and Cybern.* SMC-13, pp.834-846.

Barto, A. G. (1989). Connectionist learning for control. In Miller, T., Sutton, R.S., Werbos, P.J.(Eds.), *Neural Networks for Control*, MA: The MIT Press, pp.5-58.

Bourlard, H., Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59, pp.291-294.

Cottrell, G. W., Munro, P., Zipser, D. (1987) Image compression by back propagation; An example of extensional programming. *UCSD Institute of Cognitive Science Technical Report* , 8702, pp.1-24

Gomi, H., Kawato, M. (1990). Learning control for a closed loop system using feedback-error-learning. *Proc. the 29th IEEE Conference on Decision and Control*, Hawaii, Dec., pp.3289-3294.

Hogan, N. (1985). Impedance control: An approach to manipulation: Part I - Theory, Part II - Implementation, Part III - Applications. *ASME Journal of Dynamic Systems, Measurement, and Control,* Vol.107, pp.1-24.

Irie, B., Kawato, M. (1990). Extraction of the nonlinear global coordinate system of a manifold by a five-layered hour-glass network. *ATR Technical Report*, TR-A-0094, pp.1-11.

Jacobs, R.A., Jordan, M.I., Barto, A.G. (1990). Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *COINS Technical Report* , 90-27, pp.1-49.

Jacobs, R.A., Jordan, M.I. (1991). A competitive modular connectionist architecture. In Lippmann, R.P. et al., (Eds.), *Advances in Neural Information Processing Systems 3*, San Mateo, CA: Morgan Kaufmann Publishers, pp.767-773.

Jordan, M.I. (1988). Supervised learning and systems with excess degrees of freedom. *COINS Technical Report* , 88-27, pp.1-41.

Kawato, M., Furukawa, K., Suzuki, R. (1987). A hierarchical neural-network model for control and learning of voluntary movement. *Biol. Cybern.*, 57, pp.169-185.

Kawato, M. (1990). Computational schemes and neural network models for formation and control of multijoint arm trajectory. In Miller, T., Sutton, R.S., Werbos, P.J.(Eds.), *Neural Networks for Control*, MA: The MIT Press, pp.197-228.

Katayama, M., Kawato, M. (1991). Learning trajectory and force control of an artificial muscle arm by parallel-hierarchical neural network model. In Lippmann, R.P. et al., (Eds.) *Advances in Neural Information Processing Systems 3*, San Mateo, CA: Morgan Kaufmann Publishers, pp.436-442.

Miyamoto, H., Kawato, M., Setoyama, T., Suzuki, R. (1988). Feedback-error-learning neural network for trajectory control of a robotic manipulator. *Neural Networks*, Vol.1, pp.251-265.

Nowlan, S.J. (1990). Competing experts: An experimental investigation of associative mixture models. *Univ. Toronto Tech. Rep.* CRG-TR-90-5, pp.1-77.

Nowlan, S.J., Hinton, G.E. (1991). Evaluation of adaptive mixtures of competing experts. In Lippmann, R.P. et al., (Eds.), *Advances in Neural Information Processing Systems 3*, San Mateo, CA: Morgan Kaufmann Publishers, pp.774-780.

Psaltis, D., Sideris, A., Yamamura, A. (1987). Neural controllers. *Proc. IEEE Int. Conf. Neural Networks*, Vol.4, pp.551-557.