TR－H－001

A Neural Network Model for Arm Trajectory Formation Using Forward and Inverse Dynamics Models

Yasuhiro WADA　　Mitsuo KAWATO

1992. 5.21

ATR人間情報通信研究所

〒619-02 京都府相楽郡精華町光台2-2　　☎07749-5-1011

ATR Human Information Processing Research Laboratories

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

Telephone: +81-7749-5-1011
Facsimile: +81-7749-5-1008

A Neural Network Model for Arm Trajectory Formation Using Forward and Inverse Dynamics Models

Yasuhiro WADA **Mitsuo KAWATO**

ATR Human Information Processing Research Laboratories,
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02, Japan
Tel +81-7749-5-1044

Abstract

The minimum torque-change model predicts and reproduces human multi-joint movement data quite well. However, there are three criticisms of the current neural network models for trajectory formation based on the minimum torque-change criterion: (1) their spatial representation of time, (2) backpropagation is essential, and (3) they require too many iterations. Accordingly, we propose a new neural network model for trajectory formation based on the minimum torque-change criterion. Our neural network model basically uses a forward dynamics model, an inverse dynamics model and a trajectory formation mechanism which generates an approximate minimum torque-change trajectory. It does not require spatial representation of time or backpropagation. Furthermore, there are less iterations required to obtain an approximate optimal solution. Finally, our neural network model can be broadly applied to the engineering field because it is a new method for solving optimization problems with boundary conditions.

Keywords

Minimum Torque-change Model, Forward Dynamics Model, Inverse Dynamics Model, Optimization Problem, Trajectory Formation

Acknowledgment :

A part of this work was done in ATR Auditory and Visual Perception Research Laboratories. We would like to thank Dr. Y.Uno for discussing this work. We wish to express our thanks to Dr. Y.Tohkura of ATR Human Information Processing Research Laboratories for his encouragements. This work was supported by a Human Frontier Science Project Grant to M.K.

1. Introduction

To control voluntary movements, one must solve the following three computational ill-posed problems: (1) desired trajectory determination. (2) transformation of the task-space coordinates of the desired trajectory to intrinsic body coordinates and (3) motor command generation to achieve the desired trajectory. One way to solve these problems is to introduce a smoothness performance index. Two experimentally confirmed objective functions for voluntary movement were proposed. Flash and Hogan (1985) proposed a mathematical model, the *minimum-jerk model.* They proposed that the trajectory followed by the subject's arms tended to minimize the integral of the square of the jerk (rate of change of acceleration) of the hand position in the Cartesian coordinates space, integrated over the entire movement. A unique trajectory which yields the best performance is easily computed by applying the Euler-Lagrange equation because their model is based solely on the kinematics of movement, independent of the dynamics of the musculoskeletal system. Several hardware models which can compute minimum-jerk trajectories have been proposed using recurrent neural networks (Jordan, 1989; Massone and Bizzi, 1989; Hoff and Arbib, 1992).

Based on the idea that the objective function must be related to dynamics, Uno, Kawato and Suzuki (1989) proposed a *minimum torque-change model* which accounts for desired trajectory determination. The model is based on the theory that the trajectory of the human arm is determined so as to minimize the time integral of the square of the rate of torque change. Since the dynamics of the human arm or a robotic manipulator are nonlinear, finding the unique trajectory based on the minimum torque-change model is a nonlinear optimization problem. This is a rather difficult optimization problem since the smoothness criterion is represented in the motor command space on the one hand. On the other hand, movement conditions such as target point locations, via-points and obstacles are represented in the task-oriented coordinates. Thus, the optimization problem is computationally very intensive to be solved using the Euler-Lagrange equation. Hitherto, two kinds of hardware models have been proposed. Kawato, Maeda, Uno and Suzuki

(1990) proposed the cascade neural network, which is a cascade structure of the forward dynamics model (FDM), to generate a trajectory based on the minimum torque-change model. Conversely, a neural network model for the minimum torque-change criterion that uses the inverse dynamics model (IDM) was proposed by Nakamura, Uno, Suzuki and Kawato (1990). There are three criticisms of these neural networks: (1) they use a spatial representation of time, (2) backpropagation is essential, (3) they require too many iterations to obtain an optimal trajectory. In this paper, we propose a new model for trajectory formation that uses both the FDM and IDM. This model solves the three shortcomings above, and can be implemented as a biologically plausible neural network. The proposed network model can be used in a broad engineering field because it is a new method for solving general optimization problems with boundary conditions.

2. A neural network for optimal arm trajectory formation

2.1 Minimum torque-change criterion

This section briefly explains the minimum torque-change model. Trajectory formation is an ill-posed problem because there are an infinite number of possible trajectories that move the hand from the start to the target point. Therefore, a unique trajectory can not be determined. However, humans can move the arm between two targets, selecting one trajectory among an infinite number of trajectories. Therefore, the brain should be able to compute a unique solution by attaching an appropriate criterion to the ill-posed problem.

Flash and Hogan (1985) proposed the *minimum-jerk model* which is based on the kinematics of movement, independent of the dynamics of the musculoskeletal system. Their proposed performance index is the following quadratic measure:

$$C_J = \int_0^{t_f} \left\{ \left(\frac{d^3X}{dt^3} \right)^2 + \left(\frac{d^3Y}{dt^3} \right)^2 \right\} dt \tag{1}$$

Here, (X,Y) are Cartesian coordinates of the hand, and t_f is the movement time. Flash and Hogan (1985) showed that the unique trajectory yielding the best performance agreed with the experimental data on movement within the region just in front of the body. Their analysis was based solely on the kinematics of movement, independent of the dynamics of the musculoskeletal system, and was successful only when formulated in terms of hand motion in the extracorporeal space.

Uno, Kawato and Suzuki (1989) proposed the following alternative quadratic measure of performance. The objective function of the model is related to arm dynamics:

$$C_T = \int_0^{t_f} \sum_{j=1}^{M} \left(\frac{d\tau^j}{dt} \right)^2 dt \qquad (2)$$

where τ^j is the torque generated by the j-th actuator of M actuators, and t_f is the movement time. The objective function is the sum of the square of the rate of change of the torque, integrated over the entire movement. The minimum torque-change model can predict and produce human arm trajectories quite well. The optimization problem is to find the torque that minimizes the criterion C_T. However, it is difficult to get an optimal trajectory based on minimum torque-change because torque should be determined using complex nonlinear dynamics. That is, a nonlinear optimization problem with boundary conditions must be solved.

For movements between a pair of targets just in front of the body, predictions of both the models were close to the experimental data. However, the trajectories predicted by the minimum torque-change model were quite different from the minimum-jerk model in four behavioral situations. It was found that the minimum torque-change model predicted the real data better than the minimum-jerk model (Uno et al., 1989). The four situations were as follows: (1) discrete point-to-point movement: the starting point is an outstretched arm to the side and the end point is in front of the body, (2) movements between two points while resisting a spring, one end of which is attached to the hand while the other is fixed, and (3) vertical movements affected by gravity. In these three cases, the minimum-jerk model always predicts a straight path regardless of external forces or gravity. On the other hand, the minimum torque-change model predicts a curved path and these predictions are close to the experimental data. Finally, the most compelling evidence was examined: (4) a pair of via-point movements: with identical start and end points, but with dictated mirror-image via-points. Because the objective function C_J of the minimum-jerk model does not vary under translation, rotation and rolling, the minimum-jerk model predicts an identical path for rolling as well as identical speed profiles for the two subcases. On the other hand, the minimum torque-change model predicts two different trajectories. For the concave path, the speed profile has two peaks. However, for the convex path, the speed profile has only one peak. These predictions are close to the human data (Uno et al., 1989).

However, the two objective functions, C_J and C_T, are closely related because the rate of torque change is locally proportional to the jerk. If the arm dynamics are approximated by a point mass system, the two performance indexes are identical .

2.2 A neural network for optimal arm trajectory formation using forward and inverse models

In this section, a new neural network model for trajectory formation is proposed. A performance index for the movement between two targets is defined as the sum of a smoothness constraint energy multiplied by a regularization parameter λ and three hard constraints:

$$E = \frac{1}{2}\lambda\sum_{i=1}^{N}\sum_{j=1}^{M}(\tau_i^j - \tau_{i-1}^j)^2 + \frac{1}{2}\sum_{j=1}^{M}(\theta_d^j - \theta_N^j)^2 + \frac{1}{2}\sum_{j=1}^{M}(\dot{\theta}_d^j - \dot{\theta}_N^j)^2 + \frac{1}{2}\sum_{j=1}^{M}(\ddot{\theta}_d^j - \ddot{\theta}_N^j)^2 \qquad (3)$$

where τ_i^j is the torque generated by the j-th actuator of M actuators at the time i. The performance index is formulated here using discrete time i, and N shows the final time. Let θ, $\dot{\theta}$, and $\ddot{\theta}$ denote the position, velocity, and acceleration of the joint angle respectively; θ_d^j, $\dot{\theta}_d^j$ and $\ddot{\theta}_d^j$ represent the desired position, desired velocity, and desired acceleration of the j-th joint angle respectively. θ_N^j, $\dot{\theta}_N^j$ and $\ddot{\theta}_N^j$ are the position, velocity and acceleration of the j-th joint angle at the end time as predicted by the neural network model. The first term of Equation (3) is simply a discrete version of the minimum torque-change criterion (2). The second, third and fourth terms of Equation (3) are hard constraints regarding the movement conditions, that is, desirable position, velocity and acceleration at the end of the movement.

Here, the gradient descent of energy E (Equation (3)) is calculated as follows:

$$\frac{d\tau_i^j}{ds} = -\frac{\partial E}{\partial \tau_i^j}$$
$$= \lambda(\tau_{i+1}^j+\tau_{i-1}^j-2\tau_i^j) + \sum_{k=1}^{M}(\theta_d^k-\theta_N^k)\frac{\partial \theta_N^k}{\partial \tau_i^j} + \sum_{k=1}^{M}(\dot{\theta}_d^k-\dot{\theta}_N^k)\frac{\partial \dot{\theta}_N^k}{\partial \tau_i^j}$$
$$+ \sum_{k=1}^{M}(\ddot{\theta}_d^k-\ddot{\theta}_N^k)\frac{\partial \ddot{\theta}_N^k}{\partial \tau_i^j} \qquad (4)$$

where s represents a relaxation time that is independent of movement time. The torque τ_i^j that minimizes the performance index E can be searched according to Equation (4). The second, the third and the fourth terms of Equation (4) show that a time-backward calculation (error back-propagation) (Rumelhart et al., 1986) is needed to satisfy the desired conditions. In optimal control, this calculation is equivalent to solving the adjoint equation of the Euler-Lagrange equation. However, if the terminal conditions are satisfied, the time-backward calculation (error back-propagation) is not needed, because the second, third, and fourth term all vanish. Thus, if the terminal-condition errors can be made equal to 0, the trajectory can be generated by just smoothing the torque (the first term of Equation (4)). Assume, for simplicity, that a trajectory at some moment satisfies the terminal conditions. In this case, the relaxation rule of Equation (4), that is, the gradient descent of energy E, is expressed as follows:

$$\frac{d\tau_i^j}{ds} = \lambda\left(\tau_{i+1}^j+\tau_{i-1}^j-2\tau_i^j\right) \qquad (5)$$

Equation (5) shows the operation used to smooth the torque. The trajectory (θ, $\dot{\theta}$, $\ddot{\theta}$) generated using the new torque updated by Equation (5), does not usually satisfy the desired boundary conditions. Thus, an incremental, compensatory trajectory is generated to cancel the error in the terminal conditions. That is, $\Delta\Theta$ is generated according to the following method, and the trajectory $\Theta + \Delta\Theta$ is obtained, which satisfies the terminal conditions. The trajectory $\Delta\Theta$ is computed by addressing a linear optimization problem instead of a nonlinear optimization problem. When generating $\Delta\Theta$, the control object dynamics are approximated by a simple linear system. Furthermore, the trajectory is determined by solving the linear optimization that minimizes criterion C_T, whose terminal conditions are the terminal condition errors induced by Equation (5). The trajectory $\Theta+\Delta\Theta$ then satisfies the terminal conditions. However, the torque τ and trajectory $\Theta+\Delta\Theta$ do not satisfy the original nonlinear arm dynamics (neither FDM or IDM). Here, the FDM and IDM are defined as the following equations:

FDM : $\Theta_{i+1}=F(\Theta_i,\tau_i)$

IDM : $\tau_i=I(\Theta_i)$

where $\Theta_i=(\theta_i^1,\theta_i^2,\cdots,\theta_i^M,\dot\theta_i^1,\dot\theta_i^2,\cdots,\dot\theta_i^M,\ddot\theta_i^1,\ddot\theta_i^2,\cdots,\ddot\theta_i^M)$, $\tau_i=(\tau_i^1,\tau_i^2,\cdots\tau_i^M)$

Accordingly, the torque that satisfies both terminal conditions and nonlinear arm dynamics is obtained by using the IDM:

$$\tau_i=I(\Theta_i+\Delta\Theta_i) \tag{6}$$

In this framework, the gradient descent of energy E is easy to compute as it is simply expressed by the torque smoothing term (Equation (5)) and error backpropagation and spatial representation of time can be avoided.

The algorithm described above is shown in Figure 1. Step1 : The torque and joint angle that satisfy the terminal condition and dynamics are calculated using the IDM, where $\Theta+\Delta\Theta$ satisfies the terminal conditions. Step 2 : The torque is smoothed. Step 3 : The joint angle trajectory is generated from the torque smoothed in Step 2 through FDM. The trajectory does not satisfy the terminal conditions. The terminal-condition errors are calculated. Step 4 : By finding a solution to the linear optimization problem, the compensatory trajectory $\Delta\Theta$ which cancels the terminal-condition errors is obtained.

The optimal trajectory based on minimum torque-change is obtained by repeating Step 1 to Step 4.

Any initial trajectory can be chosen as the starting point of the calculation. However, if it is a good approximation to the minimum torque-change trajectory, faster convergence is expected. Therefore, two kinds of reasonable initial trajectories are used; one is a trajectory based on the minimum-jerk criterion (Flash et al., 1985). The other is 0. When 0 is used as the initial trajectory, the IDM in the first iteration of the proposed schema outputs the torque 0 over time. Therefore, the hand position computed by the

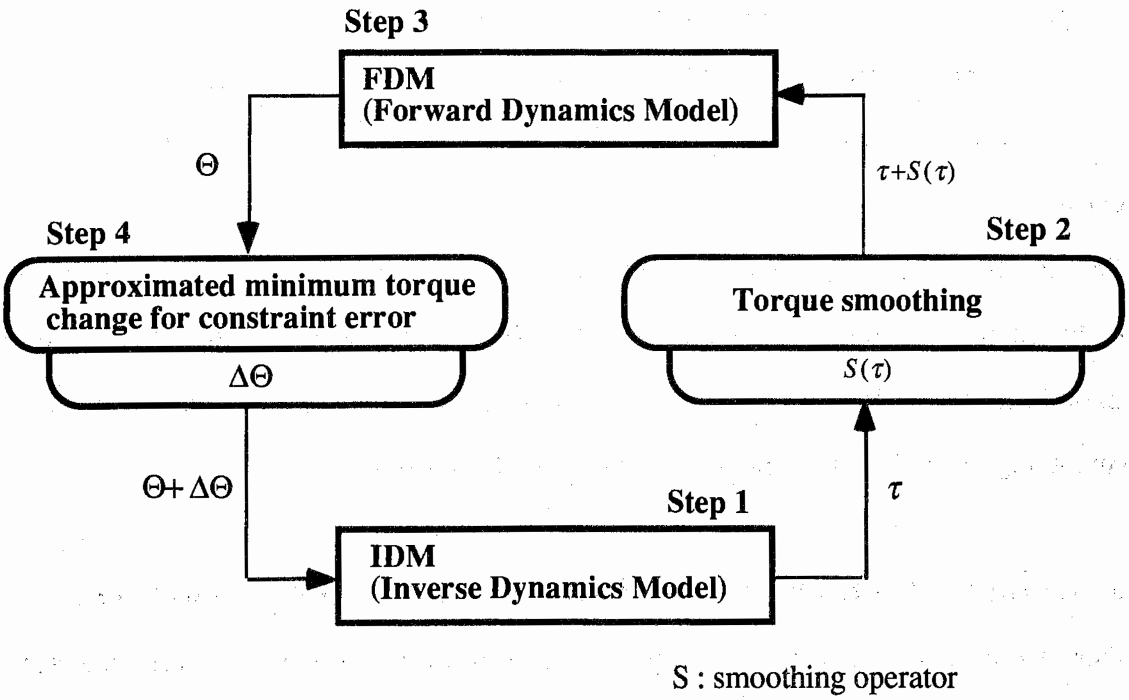

Figure 1
Neural network schema for arm trajectory formation using the forward dynamics model and inverse dynamics model. Step 1 shows the IDM whose input is a trajectory which satisfies the boundary conditions, and output is a torque series which satisfies the nonlinear arm dynamics. Torque is smoothed in Step 2. The terminal-condition errors are computed using the FDM (Step 3). In Step 4, the minimum torque-change trajectory for a linear approximated dynamics model of the arm is derived. The output of Step 4 is the trajectory which satisfies the boundary conditions of the original nonlinear optimization problem.

FDM in this iteration remains the initial position during movement time. Then, in the box " *approximated minimum-torque-change for constraint error* " in Figure 1, a trajectory is computed by solving a linear optimization problem with exactly the same boundary conditions as those of the nonlinear problem.

The smoothed torque is computed according to the next equation which is a discrete version of Equation (5).

$$\begin{aligned} \tau_i^j(s+1) &= \tau_i^j(s) + \frac{d\tau_i^j}{ds}\Delta s \\ &= \tau_i^j(s) + \lambda\left(\tau_{i+1}^j(s) + \tau_{i-1}^j(s) - 2\tau_i^j(s)\right), \end{aligned} \tag{7}$$

where Δs is a time step of discrete time and assumed equal to 1.

In the following section, iteration of Equation (7) is used as an example of the smoothing operation. The reason for this choice will be clarified in section 5. Let k be a iterative computation index, the smoothed torque is represented as follows:

$$\tau_i^j(s,k+1) = \tau_i^j(s,k) + \lambda\left(\tau_{i+1}^j(s,k) + \tau_{i-1}^j(s,k) - 2\tau_i^j(s,k)\right)$$

$$\tau_i^j(s+1) = \tau_i^j(s,n+1) \tag{8}$$

where $k = 1, 2, \cdots, n$

Thus, as the number of iterative computations n increases, the torque becomes smoother. If that number is quite large, the torque approaches 0 over time. Although the above smoothing operation was used in the computer simulation, we note that a variety of smoothing methods can be applied to the proposed algorithm.

3. Computer simulation of discrete point-to-point movement

This section presents the results of applying the proposed method to 2-joint arm trajectory formation. In this simulation, the following mathematical model of FDM and IDM was used. It has already been demonstrated that both the FDM and IDM can be achieved using neural networks (Kawato et al., 1987; Kawato et al., 1990; Kawato, 1990; Jordan et al., 1990).

$$\begin{aligned}\tau_1 &= (I_1 + I_2 + 2M_2L_1S_2\cos\theta_2 + M_2(L_1)^2)\ddot{\theta}_1 \\ &\quad + (I_2 + M_2L_1S_2\cos\theta_2)\ddot{\theta}_2 \\ &\quad - M_2L_1S_2(2\dot{\theta}_1 + \dot{\theta}_2)\dot{\theta}_2\sin\theta_2 + b_1\dot{\theta}_1\end{aligned} \tag{9}$$

$$\begin{aligned}\tau_2 &= (I_2 + M_2L_1S_2\cos\theta_2)\ddot{\theta}_1 \\ &\quad + M_2L_1S_2(\dot{\theta}_1)^2\sin\theta_2 + b_2\dot{\theta}_2\end{aligned} \tag{10}$$

M_i, L_i, S_i and I_i represent the mass, length, distance from the mass center to the joint, and the rotary inertia of the link i around the joint, respectively. Here, the same physical parameter values as those in Uno's paper are used (Uno et al., 1989). b_i and τ_i represent the coefficients of viscosity and the actuated torque of the joint i. Joints 1 and 2 correspond to the shoulder and the elbow. Joint 1 is located at the origin of the X-Y coordinates.

Two kinds of an approximated dynamic model of the arm are used to calculate the compensatory trajectory $\Delta\Theta$ in the computer simulation. The first model is a linear approximated model along the previous iteration trajectory. Thus the approximated dynamics is described by a linear differential equation with time-varying coefficients. In this case, the optimal trajectory is found by applying the Riccati equation (Bryson and Ho, 1975). The second model is a simple point-mass model with time-invariant parameters and no interaction between the joints. The minimum torque-change trajectory for this model is equivalent to the minimum-jerk trajectory in the joint-angle-coordinate space. The second model is a much poorer approximation than the first.

3.1 The numerical experiment using the linear approximated model

In this section, optimality and convergence of the new method are examined using the linear approximated model. First, the analytical computation method of the compensatory trajectory based on minimum torque-change, is described using the linear approximated model. Equations (9) and (10) are generally represented by Equation (11).

$$\frac{d}{dt}\dot{\theta}_i = f_i(\theta_1,\theta_2,\dot{\theta}_1,\dot{\theta}_2,\tau_1,\tau_2) \qquad (i=1,2) \tag{11}$$

The linear approximated equation around the trajectory $\tilde{\Theta}(t) = (\tilde{\theta}_1,\tilde{\theta}_2,\dot{\tilde{\theta}}_1,\dot{\tilde{\theta}}_2,\tilde{\tau}_1,\tilde{\tau}_2)$, which is generated by the smoothed torque $\tau + S(\tau) = (\tilde{\tau}_1,\ \tilde{\tau}_2)$, is described by Equation (12).

$$\frac{d}{dt}X(t) = A(t)X(t) + B(t)U(t) \tag{12}$$

$$\mathbf{X}(t) = (\xi_1(t)\ \ \xi_2(t)\ \ \dot{\xi}_1(t)\ \ \dot{\xi}_2(t)\ \ \eta_1(t)\ \ \eta_2(t))^T$$

$$\mathbf{U}(t) = (\dot{\eta}_1(t)\ \ \dot{\eta}_2(t))^T$$

$$A(t) = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{\partial f_1(\tilde{\Theta})}{\partial \theta_1} & \frac{\partial f_1(\tilde{\Theta})}{\partial \theta_2} & \frac{\partial f_1(\tilde{\Theta})}{\partial \dot{\theta}_1} & \frac{\partial f_1(\tilde{\Theta})}{\partial \dot{\theta}_2} & \frac{\partial f_1(\tilde{\Theta})}{\partial \tau_1} & \frac{\partial f_1(\tilde{\Theta})}{\partial \tau_2} \\ \frac{\partial f_2(\tilde{\Theta})}{\partial \theta_1} & \frac{\partial f_2(\tilde{\Theta})}{\partial \theta_2} & \frac{\partial f_2(\tilde{\Theta})}{\partial \dot{\theta}_1} & \frac{\partial f_2(\tilde{\Theta})}{\partial \dot{\theta}_2} & \frac{\partial f_2(\tilde{\Theta})}{\partial \tau_1} & \frac{\partial f_2(\tilde{\Theta})}{\partial \tau_2} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{B}(t) = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^T$$

Here, let ξ_1, ξ_2 denote positions, $\dot{\xi}_1$, $\dot{\xi}_2$ denote velocities , and η_1, η_2 represent torque respectively. The subscript denotes joint number. The position $\tilde{\theta}_i+\xi_i$, the velocity $\dot{\tilde{\theta}}_i+\dot{\xi}_i$ and the torque $\tilde{\tau}_i+\eta_i$, $(i = 1, 2)$ satisfy the boundary condition of discrete point-to-point movement.

Therefore, $\mathbf{X}(t)$ satisfy the following boundary conditions.

$$
\begin{aligned}
&\xi_1(0)=0 \quad \xi_1(t_f)=\Delta\theta_1 \\
&\dot{\xi}_1(0)=0 \quad \dot{\xi}_1(t_f)=\Delta\dot{\theta}_1 \\
&\xi_2(0)=0 \quad \xi_2(t_f)=\Delta\theta_2 \\
&\dot{\xi}_2(0)=0 \quad \dot{\xi}_2(t_f)=\Delta\dot{\theta}_2 \\
&\eta_1(0)=0 \quad \eta_1(t_f)=\Delta\tau_1 \\
&\eta_2(0)=0 \quad \eta_2(t_f)=\Delta\tau_2
\end{aligned} \tag{13}
$$

$\Delta\theta_1$, $\Delta\theta_2$ represent the position errors at the end point, $\Delta\dot{\theta}_1$, $\Delta\dot{\theta}_2$ represent the velocity errors at the end point, and $\Delta\tau_1$, $\Delta\tau_2$ are the torque errors at the end point. These terminal condition errors are induced by the smoothing operator S.

The minimum torque-change model is formulated as the following optimization problem:

$$
J = \frac{1}{2}\int (U^T Q U)dt \quad \rightarrow \quad Min \qquad Q = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{14}
$$

The optimization problem of the linear system can be solved by applying the Riccati equation. As a result, a compensatory trajectory, which is an approximated minimum torque-change trajectory, is generated.

The simulation conditions were as follows: (1) Movement time: 0.75(sec) (2) Sample time: 0.01(sec) (3) Trajectory: from T2 to T6 (The x-y Cartesian coordinates of initial and target points are shown in Table 1). (4) The iteration number of smoothing operations: $n = 100$. The convergence of the minimum torque-change criterion is shown in Figure 2. The x-axis is the number of iterative calculations, the y-axis is the criterion function value (Equation (2)). Two initial values were used in this simulation. One was a trajectory based on minimum-jerk (Flash and Hogan, 1985). The other was equal to 0. The optimal value of this problem can be calculated using a Newton-like method, that is, an iterative scheme to solve the two-point boundary-value problem (Uno et al., 1989). It is assumed that a nearly optimal solution is obtained by the Newton-like method. The criterion function value for the proposed neural network model converged near the optimal value, and the converged value of the performance index was obtained in less than 10 iterations. The converged values for the two initial values were almost the same. Thus, the proposed method can produce a trajectory close to the minimum torque-change trajectory for the original nonlinear arm dynamics in quite a small number of iterations when the minimum torque-change solution of the linear approximated model is used as the compensatory trajectory

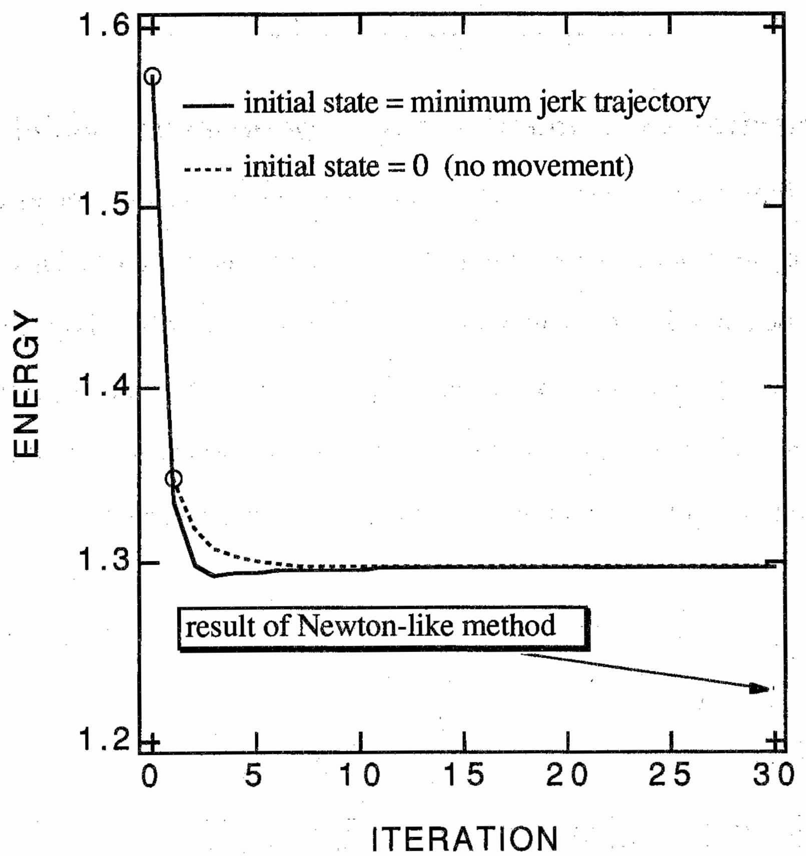

Figure 2
Convergence of the value of the minimum torque-change criterion when the new proposed schema is applied to a T2 - T6 movement in front of the body using a 2-joint arm. In this simulation, the linear approximated model with time-variant parameters is used as a model to compute the compensatory trajectory for satisfying the boundary conditions. Two kinds of initial trajectory are examined. One is the minimum-jerk trajectory. The other corresponds to 0 (no movement). Both results are almost the same. Satisfactory solutions are obtained which approximates the optimal solution computed using the Newton-like method.

Table 1
X-Y coordinates of initial target and intermediate points.

	X (cm)	Y (cm)
T1	- 0.92	30.36
T2	-24.33	30.89
T3	-19.94	47.11
T4	0.15	58.92
T5	21.09	49.33
T6	21.24	32.63
P1	0.15	58.93
P2	1.31	36.96

3.2 The numerical experiment using a point-mass model

In this section, a simple point-mass model is used. This model is expressed as Equation (15). Here, the minimum torque-change trajectory of a point-mass model (Equation (15)) is represented as the 5th order polynomial in time (Equation (16)), which is easily derived using the Euler-Lagrange equation. Because, in this case, the change in torque is exactly proportional to the jerk of the joint angle, Equation (16) is equivalent to the optimal solution of the minimum-jerk criterion in the joint-angle-coordinate space.

$$\eta_j = I_j \ddot{\xi}_j \tag{15}$$

$$\xi_j(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{16}$$

$$a_0 = \xi_j(0) \quad a_1 = 0, \quad a_2 = 0$$

$$a_3 = \left\{10(\xi_j(t_f) - \xi_j(0)) - 4\dot{\xi}_j(t_f)\cdot t_f + \frac{1}{2}\ddot{\xi}_j(t_f)\cdot t_f^2\right\} \Big/ t_f^3$$

$$a_4 = \left\{-15(\xi_j(t_f) - \xi_j(0)) + 7\dot{\xi}_j(t_f)\cdot t_f - \ddot{\xi}_j(t_f)\cdot t_f^2\right\} \Big/ t_f^4$$

$$a_5 = \left\{6(\xi_j(t_f) - \xi_j(0)) - 3\dot{\xi}_j(t_f)\cdot t_f + \frac{1}{2}\ddot{\xi}_j(t_f)\cdot t_f^2\right\} \Big/ t_f^5$$

where η_j, ξ_j, $\dot{\xi}_j$ and $\ddot{\xi}_j$ represent the torque, position, velocity and the acceleration at the j-th joint, respectively and I is the inertia of the link. a_i ($i=0,\ldots,5$) are parameters that are determined by movement time and boundary conditions (position, velocity and acceleration at the start and end point). The boundary conditions are the same as

Equation (13); however, the boundary condition for acceleration is given instead of the torque conditions. In this simulation, mathematical equations are used to obtain the compensatory trajectory. However, it is known that a recurrent neural network can learn approximate optimal trajectories (Massone et al., 1989; Jordan, 1989; Hoff et al., 1992). Furthermore Hoff and Arbib (1992) have shown that the rigorous minimum-jerk trajectory can be generated using a recurrent neural network.

The simulation conditions were as follows: (1) Movement time: 0.75(sec) (2) Sample time: 0.01(sec) (3) Trajectory: five kinds of movement in front of the body (The start and end points are shown in Table 1). The parameters used in this simulation, and the number of iterations to obtain a minimum for the objective function, are shown in Table 2. The trajectories of five movements, the speed profile of the T4-T1 movement, and the torque profile of the T4-T1 movement are shown in Figures 3, 4 and 5, respectively. Each trajectory was obtained in less than 5 iterations. The value of the criterion function attained at the first minimum point during the iterative computation is shown in Table 3. Each objective function value of the trajectory was close to the optimal value calculated using the Newton-like method. The number of iteration to obtain the minimum objective function value is almost same in the case using the linear approximated model, and the objective function value calculated using the linear approximated model was closer to the optimal value. However, the values calculated using the point-mass model and the linear approximated model were not so different. Each trajectory generated by the proposed method agreed with the hand paths calculated using the Newton-like method. For horizontal movement between two targets located approximately in front of the body, the minimum torque-change criterion predicts hand paths that are not completely straight, that is, slightly convex. On the other hand, the hand paths based on the minimum-jerk model are completely straight. Both models can also predict a single-peaked, bell-shaped speed profile of the hand. Accordingly, the hand paths generated by the proposed method were slightly convex (Figure 3). This method also produced a single-peaked, bell-shaped speed profile (Figure 4). The torque produced by the proposed method was smoother than that of the minimum-jerk model (Figure 5).

Table 2

Computer simulation parameters and number of iterations required to calculate a minimum for the objective function for each movement

movement	λ	number of smoothing	number of iterations
T2-T6	0.3	30	5
T3-T6	0.3	60	1
T1-T3	0.3	30	2
T4-T1	0.3	30	3
T4-T6	0.3	30	1

λ is the smoothing operator parameter. Number of smoothing shows the number of iterative computations in the smoothing operation. Number of iterations shows the number of iterative computations required to obtain a minimum for the objective function point in the proposed model for trajectory formation.

Table 3

Value of the minimum torque-change criterion.

movement	proposed method	Newton-like	minimum jerk
T2-T6	1.374	1.229	1.573
T3-T6	1.183	1.131	1.184
T1-T3	3.164×10^{-1}	3.051×10^{-1}	3.227×10^{-1}
T4-T1	1.920×10^{-1}	1.589×10^{-1}	2.968×10^{-1}
T4-T6	7.515×10^{-1}	7.156×10^{-1}	7.814×10^{-1}

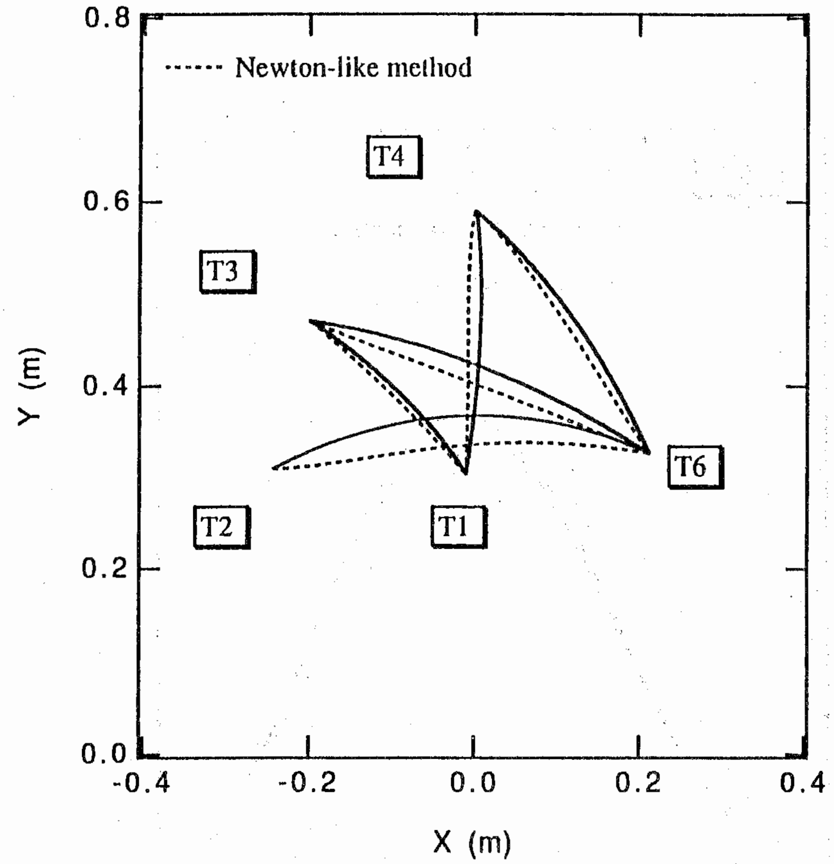

Figure 3
Trajectory of discrete point-to-point movements. Five kinds of trajectory in front of the body using a 2-joint arm are shown. The origin of the X-Y coordinates represents the location of joint 1 (shoulder). The trajectory generated by the proposed method based on the minimum torque-change criterion, and that of the Newton-like method are compared. Both trajectories are not straight, unlike minimum-jerk trajectories, but are slightly convex.

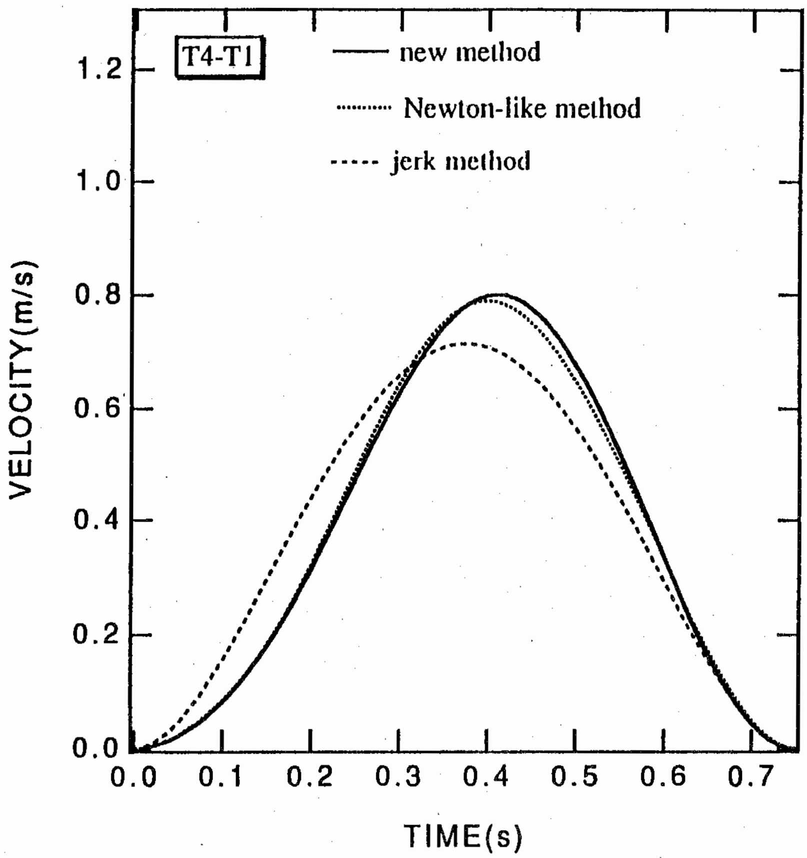

Figure 4
Speed profile of a T4 - T1 movement in front of the body. The speed profiles using the proposed model, the Newton-like method which is an optimal profile with respect to the minimum torque-change model, and the minimum-jerk model, show a single-peaked, bell-shaped speed profile which agrees with the speed profile observed in human arm movement. The profile of the proposed method is most similar to that of the Newton-like method. Thus, this method can generate a trajectory based on minimum torque-change.

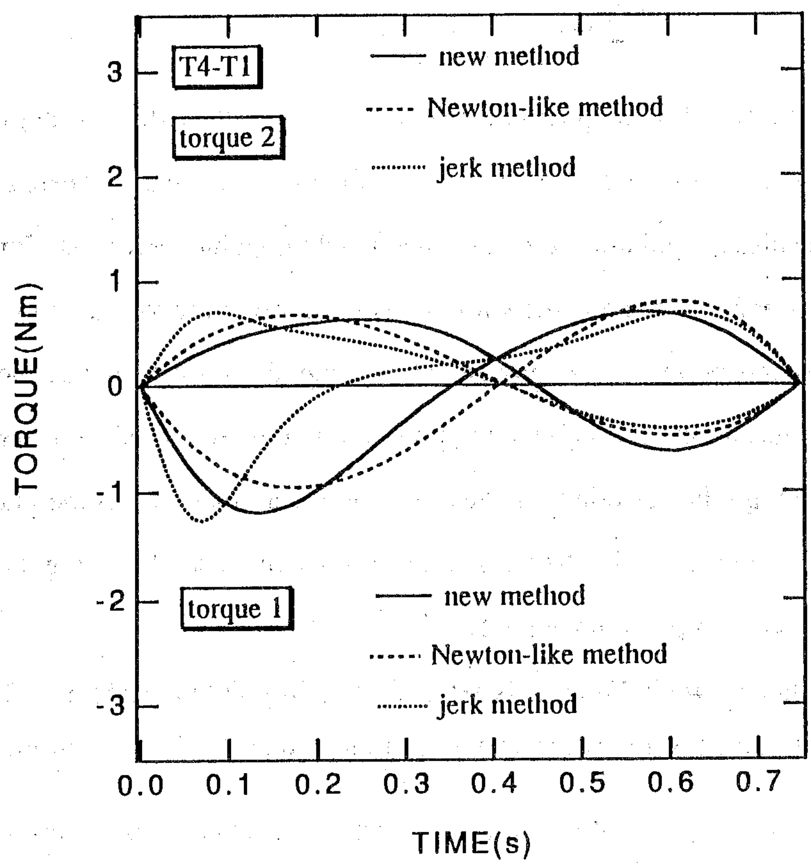

Figure 5
Torque profile of a T4 - T1 movement in front of the body. The torque profiles of the proposed model, the Newton-like method which is an optimal profile with respect to the minimum torque-change model, and the minimum-jerk model are compared. Torque 1 and torque 2 represent shoulder and elbow torque respectively. The profiles of the proposed model and the Newton-like method are smoother than that of the minimum-jerk model.

4. An Extension to via-point movement

4.1 A point-mass model for via-point movement

The proposed method is extended to generate a trajectory with via-points. In this case, position constraints at the via-points are additional to the problem of discrete point-to-point movement. The performance index of a via-point movement is defined as follows, instead of Equation (3). Trajectory formation using only one via-point is explained, however, it is easy to extend this to a movement with several via-points.

$$E = \frac{1}{2}\lambda\sum_{i=1}^{N}\sum_{j=1}^{M}(\tau_i^j - \tau_{i-1}^j)^2 + \frac{1}{2}\sum_{j=1}^{M}(\theta_d^j - \theta_N^j)^2 + \frac{1}{2}\sum_{j=1}^{M}(\dot{\theta}_d^j - \dot{\theta}_N^j)^2$$
$$+ \frac{1}{2}\sum_{j=1}^{M}(\ddot{\theta}_d^j - \ddot{\theta}_N^j)^2 + \frac{1}{2}\sum_{j=1}^{M}(\theta_{via}^j - \theta_V^j)^2 , \qquad (17)$$

where θ_{via}^j represents the desired position at a via-point of the j-th joint angle. θ_V^j represents the position at V time ($1 \leq V \leq N-1$) of the j-th joint angle. There are two differences between discrete point-to-point movement and via-point movement. First, the movement time between the start and the via-point is not given, however, the movement time is given for discrete point-to-point movement. Second, the velocity and acceleration constraints at the time passing through the via-point are not given. Thus, conversely, if the time passing through the via-point, the velocity and the acceleration at the via-point can be determined, a trajectory with the via-point can be easily generated using the same method for discrete point-to-point movements.

Here, the algorithm shown in Figure 1 is extended for via-point movements. The via-point time V for the compensatory trajectory is chosen so that $\sum_{j=1}^{M}(\theta_{via}^j - \theta_V^j)^2$ in Equation (17) is minimized after Step 3 (FDM) in Figure 1. The point-mass model is used as the approximate dynamics model for generating the compensatory trajectory. The minimum torque-change trajectory of the point-mass model is expressed as Equation (16). Thus, the trajectory from the starting point to a via-point could be calculated if the movement time, position, velocity and acceleration at the start and the via-point were to be given. The torque-change criterion value from time 0 to t_{via} when passing through the via-point is expressed as follows:

$$J = I^2\int_0^{t_{via}}\left(\frac{d\ddot{\xi}}{dt}\right)^2 dt \quad \rightarrow \quad Min \qquad (18)$$

where

$$\xi(0) = \Delta\theta_0 \qquad \xi(t_{via}) = \Delta\theta_{via}$$
$$\dot{\xi}(0) = \Delta\dot{\theta}_0 \qquad \dot{\xi}(t_{via}) = \Delta\dot{\theta}_{via}$$
$$\ddot{\xi}(0) = \Delta\ddot{\theta}_0 \qquad \ddot{\xi}(t_{via}) = \Delta\ddot{\theta}_{via}$$

$\Delta\theta_0$, $\Delta\dot{\theta}_0$ and $\Delta\ddot{\theta}_0$ represent the position, velocity and the acceleration errors for the boundary conditions, respectively, at time 0. $\Delta\theta_{via}$, $\Delta\dot{\theta}_{via}$ and $\Delta\ddot{\theta}_{via}$ represent those errors at time V. However, the values of $\Delta\dot{\theta}_{via}$ and $\Delta\ddot{\theta}_{via}$ are not specified yet at this point and will be given below. A solution of Equation (18) is given by Equation (19) similar to Equation (16).

$$\xi(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 \qquad (19)$$

where b_i $(i=0,\ldots,5)$ could be determined if the above six constraints were to be given. Here, the torque-change criterion function of the point-mass model is expressed as Equation (20). Since the values $\Delta\dot{\theta}_{via}$ and $\Delta\ddot{\theta}_{via}$ are not given, Equation (20) is a function of the velocity and acceleration at via-time.

$$\begin{aligned} J(\Delta\dot{\theta}_{via}, \Delta\ddot{\theta}_{via}) &= I^2 \int_0^{t_{via}} \left(\frac{d\ddot{\xi}}{dt} \right)^2 dt \\ &= I^2 \int_0^{t_{via}} \left(6b_3 + 24b_4 t + 60 b_5 t^2 \right)^2 dt \qquad (20) \end{aligned}$$

For $\Delta\dot{\theta}_{via}$ and $\Delta\ddot{\theta}_{via}$ to minimize Equation(20), the following conditions are necessary:

$$\frac{\partial J}{\partial \Delta\dot{\theta}_{via}} = 0 \qquad (21)$$

$$\frac{\partial J}{\partial \Delta\ddot{\theta}_{via}} = 0 \qquad (22)$$

Therefore, the velocity and acceleration at via-time are obtained as follows:

$$\Delta\dot{\theta}_{via} = \frac{1}{4t_{via}} \left(10\Delta\theta_{via} - 10\Delta\theta_0 - 6\Delta\dot{\theta}_0 t_{via} - \Delta\ddot{\theta}_0 t_{via}^2 \right) \qquad (23)$$

$$\Delta\ddot{\theta}_{via} = \frac{1}{3t_{via}^2} \left(10\Delta\theta_{via} - 10\Delta\theta_0 - 10\Delta\dot{\theta}_0 t_{via} - 2\Delta\ddot{\theta}_0 t_{via}^2 \right) \qquad (24)$$

That is, the via-point time can be chosen and the velocity and acceleration computed, so as to minimize the torque-change criterion function for the approximate model. Furthermore, the algorithm described above can be applied to trajectory formation with more than one via-point. Because the algorithm can determine the velocity and acceleration errors at the first via-point using only the error of the position, velocity, acceleration at the start point, the velocity and acceleration errors at the second via-point can be similarly determined from errors at the first via-point. Equations like (23) and (24) can be derived straightforwardly. The reason of this straightforward extension to multiple via-point cases is that only the objective function from the start point to the first via-point is considered in Equation (20) and the latter half of the integral is not taken into account.

4.2 Numerical experiments

The simulation conditions were as follows: (1) Movement time: 1.0(sec) (2) Sample time: 0.01(sec) (3) Start point T3, End point T5, and Via point P1 or P2 (The x-y Cartesian coordinates of each point are shown in Table 1). The iteration number for smoothings, the smoothing parameter λ, and the number of iterations needed to reach the first minimum for the objective function, are shown in Table 4. The minimum-jerk trajectory in the Cartesian coordinates was chosen as the initial trajectory. The trajectories of the two movements, and the speed profiles of T3-P2-T5 are shown in Figures 6 and 7 respectively. Minimal values of the criterion function for the three schemes (proposed method, Newton-like method, and minimum-jerk model) are shown in Table 5. Each trajectory was obtained in less than 5 iterations, and the minimum value of the criterion function was close to the optimal value obtained using the Newton-like method. The hand paths generated using the proposed method were almost the same as the hand paths of the Newton-like method (minimum torque-change model), and the minimum-jerk model. These two models can predict a curved hand path with a single-peaked or double-peaked speed profile, and this depends on the location of the via-point. In this case, the two via-points P1 and P2 were located symmetrically with respect to the line connecting the common start and end points. The minimum-jerk model predicted identical speed profiles for both cases; however, the minimum torque-change model predicted two different profiles: that for via-point P1 had only one peak; however, that for P2 had two peaks (Uno et al., 1989). The speed profile for via-point P2 predicted by the proposed method had two peaks, as shown in Figure 7. These simulation results show that the proposed method for via-point movement can generate approximately a trajectory passing through via-points based on minimum torque-change criterion in only several iterations. We emphasize that the objective functions obtained by the proposed method are much smaller than those of the minimum-jerk model (Table 5). If one compares Table 3 and Table 5, it is suggested that the new method is more efficient for complex movements rather than simple movements.

Table 4

Computer simulation parameters and number of iterations required to calculate a minimum for the objective function for each movement

movement	λ	number of smoothing	number of iterations
T3-P1-T5	0.3	30	3
T3-P2-T5	0.3	30	3

λ is the smoothing operator parameter. Number of smoothing shows the number of iterative computations in the smoothing operation. Number of iterations shows the number of iterative computations required to obtain a minimum for the objective function in the proposed model for trajectory formation.

Table 5

Value of the minimum torque-change criterion.

movement	proposed method	Newton-like	minimum jerk
T3-P1-T5	3.850×10^{-1}	3.168×10^{-1}	6.709×10^{-1}
T3-P2-T5	4.348×10^{-1}	3.322×10^{-1}	6.323×10^{-1}

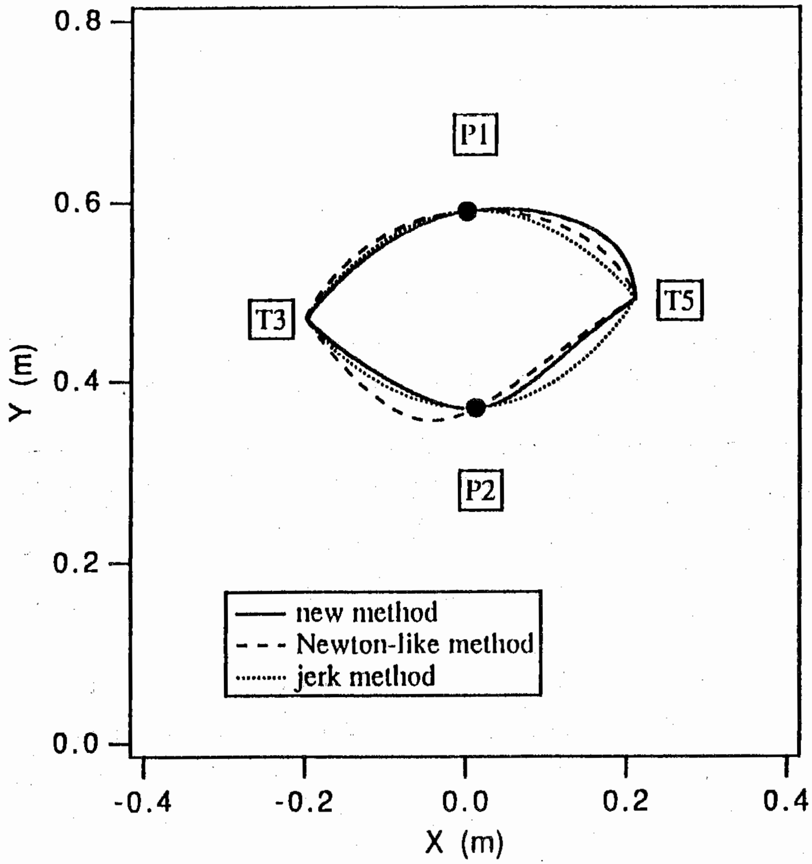

Figure 6
Trajectory of via-point movement in front of the body (T3 - P1 - T5 and T3 - P2 -T5). The trajectories of the proposed method, and the Newton-like method which are based on the minimum torque-change criterion are compared to that of the minimum-jerk model. The three trajectories are almost identical.

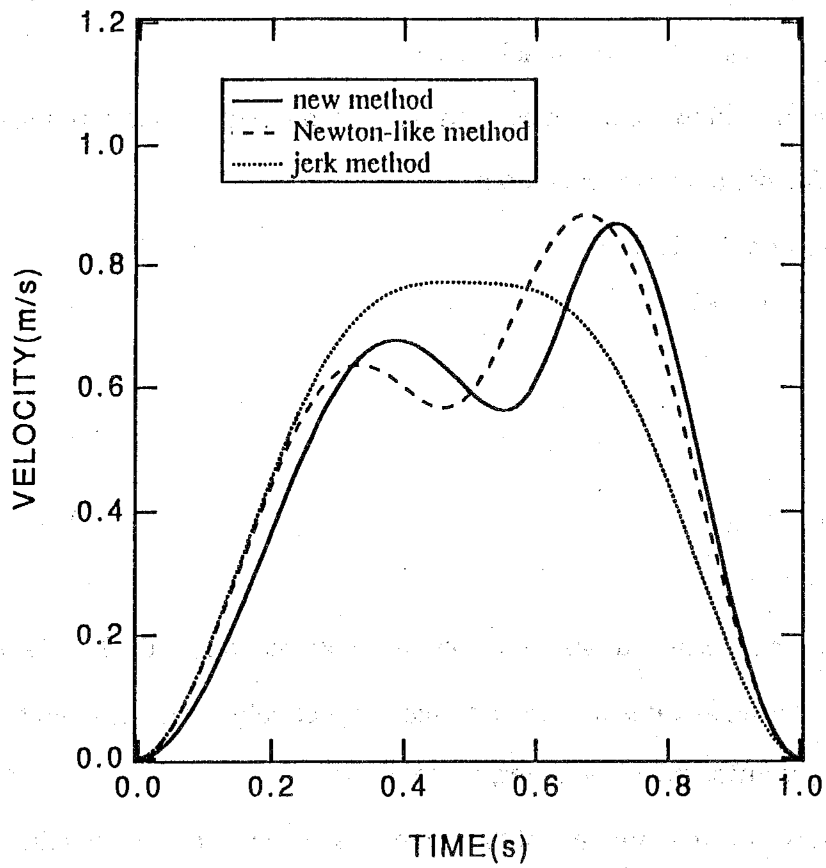

Figure 7
Speed profile of a T3-P2-T5 movement. The speed profile based on the minimum-jerk criterion has only one peak, and in a T3-P1-T5 movement it also has only one peak. The speed profile based on the minimum torque-change criterion has one peak in a T3-P1-T5 movement, however, in a T3-P2-T5 movement, it has two peaks which correspond to the hand trajectory observed in human arm movement. Thus, in a T3-P1-T5 movement, the speed profiles based on the two kinds of criteria are not so different, but in a T3-P2-T5 they are quite different, as shown in this figure.

5. Mathematical considerations of the proposed method for nonlinear optimization problems

In this section, the proposed network is formulated as a general optimal algorithm, and the theoretical framework for this method is described. The optimality and convergence of the new method applied to a nonlinear optimization problem is discussed mathematically in sections 5.1 and 5.2. The following discussion is for the scalar case but it can be easily extended to multi-variable cases.

First, we define the following nonlinear optimization problem which minimizes the criterion function J under boundary conditions:

[Nonlinear Optimization Problem: N]

$$J = \int \left(\frac{du}{dt} \right)^2 dt \quad \rightarrow \quad Min \tag{25}$$

Subject to

$$\frac{dx}{dt} = f(x, u) , \tag{26}$$

$$x(0) = 0 , x(t_f) = x_{df} , \tag{27}$$

$$u(0) = 0 , u(t_f) = 0 , \tag{28}$$

where x and u represent a state variable and a control variable, respectively. x_{df} and t_f represent a desired terminal value and an end time, respectively. It is assumed that the nonlinear function f is differentiable with respect to x and u.

Let us first illustrate the iteration rule of the control variable by generalizing the neural network model proposed in section 2 as follows. We define an inverse dynamics model G, which is an inverse function of the forward dynamics model F.

$$G(X) = u ,$$

where $(x, \dot{x}) = X = F(u)$

The control variable at the $j+1$-th iteration, as shown in Figure 1, is outputted by IDM, whose input is computed by adding the trajectory generated by the smoothed torque at the j-th iteration to the compensatory trajectory generated by an approximate linear dynamics model. Thus, the control variable at the $j+1$-th iteration is derived as follows:

$$u^{j+1} = G\left(F\left(u^j + S(u^j)\right) + \Xi\right) \tag{29}$$

where $u^j + S(u^j)$ is the smoothed torque, and $\Xi = (\xi, \dot{\xi})$ is the trajectory which compensates for the terminal-condition errors induced by $u^j + S(u^j)$ and is the solution for the linear optimization problem, defined below. The following equation is obtained using Taylor's expansion of function F in Equation (29) around u^j.

$$u^{j+1} = G\left(F(u^j) + \frac{\partial F(u^j)}{\partial u}S(u^j) + o\big(S(u^j)\big) + \Xi\right)$$

Taking Taylor's expansion of G around $X^j = F(u^j)$:

$$u^{j+1} = u^j + \frac{\partial G(X^j)}{\partial X}\left(\frac{\partial F(u^j)}{\partial u}S(u^j) + o\big(S(u^j)\big)\right) + \frac{\partial G(X^j)}{\partial X}\Xi$$

$$= u^j + S(u^j) + \frac{\partial G(X^j)}{\partial X}o\big(S(u^j)\big) + \frac{\partial G(X^j)}{\partial X}\Xi$$

where a higher order term than the second order term is ignored because the trajectory change between the j-th and the $j+1$-th iteration is usually quite small. The fourth term in above equation is expressed as $-\eta^j$. η^j is obtained as the optimal solution to the following linear problem. If $S(u^j)$ is small, the following equation is derived because $o\big(S(u^j)\big)$ becomes negligibly small.

$$u^{j+1} = u^j + S(u^j) - \eta^j$$

However, it is not assumed that $S(u^j)$ is samll in the computer simulation and mathematical consideration described below. Therefore, we define the iteration rule of the control variable as follows:

$$u^{j+1} = u^j + \tilde{S}(u^j) - \eta^j \tag{30}$$

where $\tilde{S}(u^j)$ is $S(u^j) + \frac{\partial G(X^j)}{\partial X}o\big(S(u^j)\big)$.

Note that the two following trajectories satisfy exactly the same boundary conditions.

$$\frac{\partial F(u^j)}{\partial u}S(u^j) + o\big(S(u^j)\big)$$

$$-\Xi$$

The linear optimization problem is defined to compute a trajectory which compensates for boundary conditions.

[Linear Optimization Problem: L]

$$J_\eta = \int\left(\frac{d\eta}{dt}\right)^2 dt \quad \rightarrow \quad Min \tag{31}$$

Subject to

$$\frac{d\xi}{dt} = \frac{\partial f(x^j, u^j)}{\partial x}\xi + \frac{\partial f(x^j, u^j)}{\partial u}\eta \tag{32}$$

$$\xi(0) = 0\,, \quad \xi(t_f) = \Delta x^j(t_f) \tag{33}$$

$$\eta(0) = 0, \quad \eta(t_f) = \tilde{S}(u^j)(t_f) \tag{34}$$

Here, x^j, u^j are the trajectory and the control variable at the j-th iteration, respectively. $\Delta x^j(t_f)$ shows the terminal-condition errors when the input $u^j + S(u^j)$ is fed to the nonlinear dynamics equation (26). Let (u^*, x^*) be an isolated optimal solution that minimizes the criterion function J, and let (η^{*j}, ξ^{*j}) be an isolated optimal solution that minimizes the linear optimal problem L. Thus, ξ^{*j} is the compensatory trajectory.

5.1 Optimality of the converged solution

In this section we will show that the convergence of the proposed algorithm is equivalent to the optimality of the solution. First we discuss the necessary condition: if u^j is equal to u^* $(u^j = u^*)$, then η^{*j} becomes equal to $\tilde{S}(u^*)$ $(\eta^{*j} = \tilde{S}(u^j))$, thus the iteration converges. That is, if the control variable is equal to the optimal value, the iterative algorithm Equation (30) converges. Conversely, we then discuss the sufficient condition: if $\tilde{S}(u^j)$ is equal to η^{*j} $(\eta^{*j} = \tilde{S}(u^j))$, then u^j becomes equal to u^* $(u^j = u^*)$. That is, if the control variable converges, the control variable u^j becomes the optimal solution.

A linear dynamic equation around the optimal solution is represented by the following equation:

$$\frac{d\xi}{dt} = \frac{\partial f(x^*, u^*)}{\partial x}\xi + \frac{\partial f(x^*, u^*)}{\partial u}\eta \tag{35}$$

where,

$$\xi(0) = 0, \quad \xi(t_f) = \Delta x_f^*$$
$$\eta(0) = 0, \quad \eta(t_f) = \tilde{S}(u^*)(t_f) \tag{36}$$

It is evident that the above necessary condition is equivalent to the following lemma 1.

[Lemma 1]

The optimal solution η^* for the linear optimization problem [L] at $x^j = x^*$ is equal to $\tilde{S}(u^*)$.

Proof

We use reductio ad absurdum. Assume that the optimal solution η^* for [L] is not equal to

$\tilde{S}(u^*)(\eta^* \neq \tilde{S}(u^*))$. The following shows that this assumption contradicts the assumption that u^* is the optimal solution for [N]. The following new control variable is constructed for [N].

$$\hat{u} = u^* + \varepsilon\left(\tilde{S}(u^*) - \eta^*\right) \quad (|\varepsilon|<<1) \tag{37}$$

If ε is small, $\hat{u} - u^*$ is approximated by the solution to the linear equation (35) with the input $\varepsilon\left(\tilde{S}(u^*) - \eta^*\right)$. Furthermore, the boundary conditions for $\tilde{S}(u^*)$ and η^* are exactly the same with that of [L]: according to Equation (36). Therefore, $\hat{u}$ satisfies the boundary conditions Equations (27) and (28), $x(0) = 0$, $x(t_f) = x_{df}$, $u(0) = 0$, and $u(t_f) = 0$. If the following inequality is shown, this proof is completed, as the inequality contradicts the assumption that u^* is the optimal solution for [N]

$$\int\left(\frac{d\hat{u}}{dt}\right)^2 dt < \int\left(\frac{du^*}{dt}\right)^2 dt$$

The next equation is obtained by ignoring the term ε^2.

$$\begin{aligned}\int\left(\frac{d\hat{u}}{dt}\right)^2 dt &= \int\left[\frac{d}{dt}\left\{u^* + \varepsilon\left(\tilde{S}(u^*)-\eta^*\right)\right\}\right]^2 dt \\ &\cong \int\left(\frac{du^*}{dt}\right)^2 dt + 2\varepsilon\int\frac{du^*}{dt}\left(\frac{d\tilde{S}(u^*)}{dt} - \frac{d\eta^*}{dt}\right)dt\end{aligned} \tag{38}$$

Let I be an integral of the second term on the right-hand side of Equation (38). ε's sign can be determined according to I's sign. That is,

if $I>0$ then $\varepsilon<0$

if $I<0$ then $\varepsilon>0$

Thus the required inequality can be derived.

Here, if $I = 0$, because $\frac{d\tilde{S}(u^*)}{dt} - \frac{d\eta^*}{dt}$ is not always equal to 0 over time $\left(\because\ \eta^* \neq \tilde{S}(u^*)\right)$, $\hat{u}$ is also optimal to the first approximation. Therefore, u^* does not become an isolated extreme value in problem [N], because the first variation of the criterion function corresponds to 0 in this case. (Q.E.D)

The following lemma 2 is equivalent to the sufficient condition.

[Lemma 2]

If $\tilde{S}(u^j)$ is equal η^{*j} ($\eta^{*j} = \tilde{S}(u^j)$), then u^j becomes equal to u^* ($u^j = u^*$).

Proof

We use reductio ad absurdum again. Let us assume that the control variable u^j is not the optimal solution for [N] ($u^j \neq u^*$), even when the iteration converges.

The following new control variable is constructed for [L].

$$\hat{\eta} = \tilde{S}(u^j) + \varepsilon\left(u^* - u^j\right) \tag{39}$$

First, it will be shown that $\hat{\eta}$ satisfies the boundary conditions of [L]. The next equation shows the dynamic equation at the optimal point.

$$\frac{dx^*}{dt} = f(x^*, u^*) \tag{40}$$

Equation (41) is the linear approximated equation of Equation (40) around u^j where $u^* \sim u^j$.

$$\frac{d\xi}{dt} = \frac{\partial f(x^j, u^j)}{\partial x}\xi + \frac{\partial f(x^j, u^j)}{\partial u}(u^* - u^j) \tag{41}$$

Then, both u^* and u^j satisfy the final conditions $x(t_f) = x_{df}$ of the dynamic equation (26), so ξ, that is, the difference between the trajectory generated by u^* and that by u^j, satisfies the condition $\xi(t_f) = 0$. Furthermore, $\hat{\eta}$ satisfies the boundary conditions of Equation (33) because $\tilde{S}(u^j)$ satisfies the boundary conditions of Equation (33). Moreover, $\hat{\eta}$ satisfies the boundary conditions of Equation (34) according to the following conditions:

$$u^*(0) = u^*(t_f) = u^j(0) = u^j(t_f) = 0$$

Equation (43) can be derived by using Equation (42) and ignoring the term ε^2.

$$\frac{d\hat{\eta}}{dt} = \frac{d\tilde{S}(u^j)}{dt} + \varepsilon\left(\frac{du^*}{dt} - \frac{du^j}{dt}\right) \tag{42}$$

$$\int\left(\frac{d\hat{\eta}}{dt}\right)^2 dt \cong \int\left(\frac{d\tilde{S}(u^j)}{dt}\right)^2 dt + 2\varepsilon\int\frac{d\tilde{S}(u^j)}{dt}\left(\frac{du^*}{dt} - \frac{du^j}{dt}\right)dt \tag{43}$$

Let I be an integral of the second term on the right-hand side of Equation (43). ε's sign can be determined according to I's sign in the same manner as the previous proof.

if $I > 0$ then $\varepsilon < 0$

if $I < 0$ then $\varepsilon > 0$

Therefore, the required inequality is obtained:

$$\int\left(\frac{d\hat{\eta}}{dt}\right)^2 dt < \int\left(\frac{d\tilde{S}(u^j)}{dt}\right)^2 dt$$

This contradicts the basic assumption that $\eta^{*j} = \tilde{S}(u^j)$ is optimal for [L].

If $I = 0$,

$$\frac{du^*}{dt} - \frac{du^j}{dt}$$

is not always equal to 0 over time. Therefore, η^{*j} does not become an isolated extreme value for [L]. (Q.E.D)

5.2 Convergence of the solution

This section discusses the monotone convergence of the criterion function. We ask whether the proposed algorithm decreases the criterion function for every iteration ($J(u^{j+1}) \le J(u^j)$).

The criterion function at the $j+1$-th iteration is calculated using Equation (30).

$$J(u^{j+1}) - J(u^j) = 2\int\left(\frac{du^j}{dt} + \frac{d\tilde{S}(u^j)}{dt}\right)\left(\frac{d\tilde{S}(u^j)}{dt} - \frac{d\eta^j}{dt}\right)dt$$
$$- \int\left(\frac{d\tilde{S}(u^j)}{dt} + \frac{d\eta^j}{dt}\right)\left(\frac{d\tilde{S}(u^j)}{dt} - \frac{d\eta^j}{dt}\right)dt \qquad (44)$$

Also, the smoothing operator S in sections 2, 3 and 4 is represented in continuous time as follows:

$$u^j_{(k+1)} = u^j_{(k)} + \lambda\frac{d^2u^j_{(k)}}{dt^2} \qquad (k = 1,2,\cdots,n \qquad \lambda < 1)$$

where n is the number of smoothings. For very large n, we assume that $u^j + \tilde{S}(u^j)$ is quite small. This is the most important condition for convergence. We note that this is not a reasonable assumption when the optimal solutions to [N] and [L] are very different. On the other hand when u^j has not converged yet, then

$$\tilde{S}(u^j) - \eta^j \neq 0 \qquad (\because \text{Equation (30)})$$

Furthermore $\tilde{S}(u^j) + \eta^j = u^j + \tilde{S}(u^j) - \left(u^j - \eta^j\right)$ is not small since $u^j - \eta^j$ is not small. Since the first term of the right-hand side of Equation (44) becomes negligibly small compared to the second term, it can be ignored.

$$J(u^{j+1}) - J(u^j) = \int\left(\frac{d\eta^j}{dt}\right)^2 dt - \int\left(\frac{d\tilde{S}(u^j)}{dt}\right)^2 dt$$

The following inequality holds because η^j is optimal for [L]

$$\int\left(\frac{d\eta^j}{dt}\right)^2 dt - \int\left(\frac{d\tilde{S}(u^j)}{dt}\right)^2 dt \le 0$$

Finally, the warranted inequality can be derived which shows the monotone convergence.

$$J(u^{j+1}) \le J(u^j)$$

Because $J(u)$ is lower bounded, it converges to the minimum value.

6. Discussion

A new neural network model for trajectory formation was developed which basically uses a forward dynamics model, an inverse dynamics model and an approximate minimum torque-change trajectory generation mechanism. This model can solve the three difficult criticisms relating to the cascade, and other neural networks. These are, (1) they use a spatial representation of time, (2) backpropagation is essential, and (3) too many iterations are required to obtain the optimal trajectory. However, this model does not use a spatial representation of time. It does not require backpropagation in iterative computation. It needs only an approximate linear model and an IDM that is an inverse function of the FDM to satisfy the boundary conditions. Furthermore, this model can reach an optimal solution in a few iterations as shown in the computer simulation. Accordingly, the new neural network model solves the three criticisms of the previous models.

In the mathematical proof and computer simulation, a smoothing operator that needs many iterative calculations was used. Such an iterative smoothing operator is not necessary, however, and smoothing filters of one-shot type can be easily designed.

We emphasize that the entire model can be implemented using neural networks. The model consists of five main parts; a FDM, an IDM, an approximated linear model, a smoother and a via-points seeker. As already mentioned, a FDM can be obtained using Jordan's recurrent neural network (Jordan et al., 1990). Kawato et al. (Kawato et al., 1987; Kawato, 1990) have already reported that an IDM can be obtained by neural network learning. The torque smoother and via-point seeker are simple enough to be implemented as neural networks. In the case of the point-mass model, the minimum torque-change trajectory can be expressed as the minimum-jerk trajectory in the joint angle space. Hoff and Arbib (1992) have already pointed out that a minimum-jerk trajectory can be obtained using a recurrent neural network when the final velocity and acceleration are equal to 0. It is easy to extend their model to general conditions, so that the velocity and acceleration at the start and end points are not necessarily 0. These conditions are needed in via-point trajectory formation. That is, the minimum-jerk trajectory under the above conditions can be defined using the following dynamics:

$$\frac{d}{dt}\Theta = A\Theta + B\Theta_v \tag{45}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -60/D^3 & -36/D^2 & -9/D \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 60/D^3 & -24/D^2 & 3/D \end{bmatrix}$$

$$\Theta = \left(\theta \quad \dot{\theta} \quad \ddot{\theta}\right)^T$$

$$\Theta_v = \left(\theta_v \quad \dot{\theta}_v \quad \ddot{\theta}_v\right)^T$$

where v shows the end point and θ_v, $\dot{\theta}_v$ and $\ddot{\theta}_v$ define the given position, velocity and acceleration at the end point. D represents the remaining movement time. Accordingly, when $\dot{\theta}_v = 0$ and $\ddot{\theta}_v = 0$, the above equation corresponds to the dynamic equation proposed by Hoff and Arbib (1992). Thus, the point-mass model can be obtained using a recurrent network in the same manner as Hoff and Arbib. The detailed model for trajectory formation and its five main parts obtained using neural networks, are shown in Figure 8.

The new method is a general method for nonlinear optimization problems with boundary conditions. It has several advantages in engineering applications. For example, it does not require an inversion of matrices in iteration, it reaches an optimal solution in a short time, and the output after passing through the IDM always satisfies the boundary conditions and dynamics equation. Therefore, it can be extended to many other engineering problems.

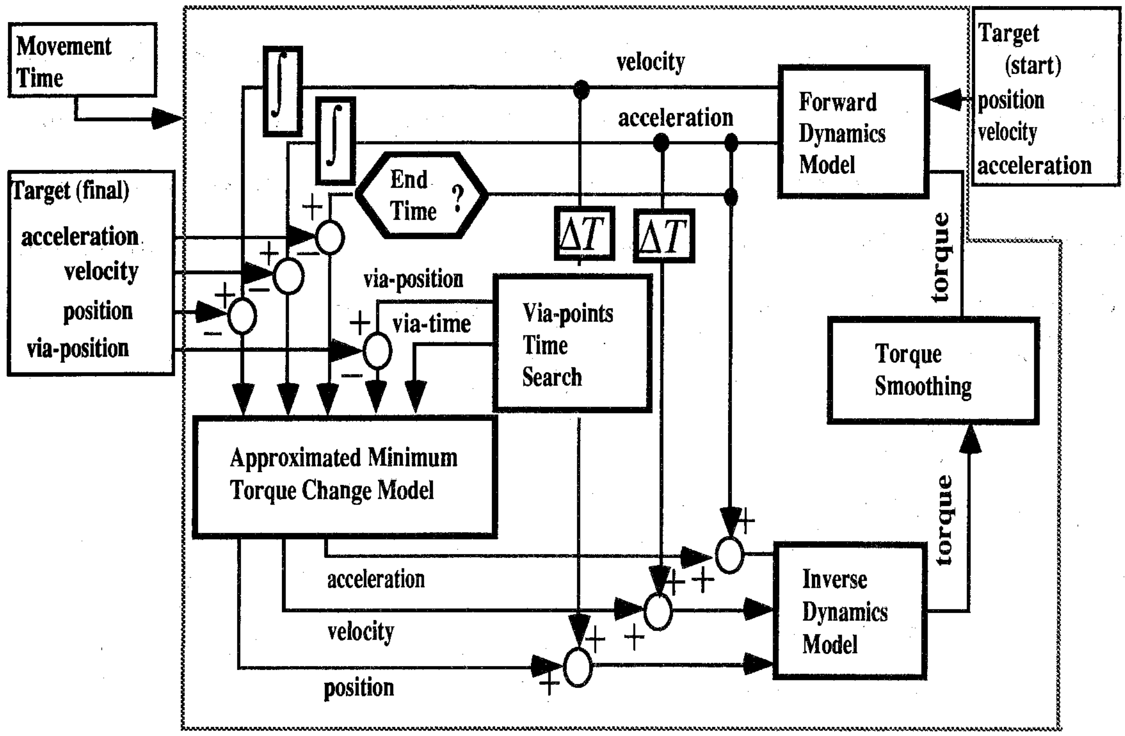

Figure 8
Neural network structure of arm trajectory formation. The five main parts, that is, the *Forward Dynamics Model, Inverse Dynamics Model, Approximated Minimum Torque Change Model, Torque Smoother* and *Via-points Time Search*, can all be implemented as neural networks. In this figure, the flow of the signals is illustrated in detail. The inputs of the proposed neural network are positions, velocities and accelerations at the start and the end times, movement time, and via-point positions. The position, velocity and acceleration time courses that satisfy the boundary conditions are fed into the *Inverse Dynamics Model* and it outputs the torque. The smoothed torque is fed into the *Forward Dynamics Model.* It outputs the time series of the velocity and acceleration, and then the errors of the position, velocity and acceleration at the end point are computed when compared with the desired values. The position and velocity at the end point are calculated using integral operations from the start time to the end time. The via-point positions and via-point passing times are solved in the *Via-points Time Search.* All the boundary condition errors are fed into *Approximated Minimum Torque Change Model.* It outputs the compensatory trajectory. Finally, the position, velocity and acceleration which satisfy the boundary conditions are computed by adding the compensatory trajectory to the output trajectory of the *Forward Dynamics Model.* ΔT shows the time step in computation.

In the future, the potential applications of this proposed model to a minimum muscle-tension-change model and a minimum motor-command-change model will be studied. Efficiency of the proposed model will be checked when the IDM is not a perfect inverse function of the FDM. Furthermore, it is expected that this trajectory formation model can be used as pattern recognition network as a kind of duality exists between pattern formation and recognition in this framework (Kawato, 1989).

REFERENCES

Bryson, AE. and Ho, YC. (1975). Applied optimal control, Wiley, New York

Flash, T., Hogan, N. (1985). The coordination of arm movements; An experimentally confirmed mathematical model, The Journal of Neuroscience, **5**,7, pp. 1688-1703

Hoff, B., Arbib, M.A. (1992). Models of trajectory formation and temporal interaction of reach and grasp, submitted to *Journal of Motor Behavior*

Jordan, M.I., Rumelhart, D.E. (1990). Forward models: Supervised learning with a distal teacher, *MIT Center for Cognitive Science Occasional Paper #40.*

Jordan, M.I. (1989). Indeterminate motor skill learning problems. In M. Jeannerod (Ed.), Attention and Performance, XIII. Cambridge, MA; MIT Press.

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

Kawato, M. (1989). Motor theory of speech perception revisited from minimum torque-change neural network model, 8th Symposium on future electron devices pp. 141-150

Kawato, M., Maeda, Y., Uno, Y., Suzuki, R. (1990). Trajectory formation of arm movement by cascade neural network model based on minimum torque-change criterion, *Biological Cybernetics,* **62**, pp. 275-288

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

Massone, L., Bizzi, E. (1989). A neural network model for limb trajectory formation. *Biological Cybernetics,* **61**, pp. 417-425

Nakamura, M., Uno, Y., Suzuki, R., Kawato, M. (1990). Formation of Optimal Trajectory in Arm Movement Using Inverse Dynamics Model, *Japan IEICE Technical Report,* NC89-63 (in Japanese)

Rumelhart, D.E., Hinton, G.E., Williams, R.J. (1986). Learning Representations by Back Propagating Errors, *Nature,* 323, 9, Oct. pp. 533-536

Uno, Y., Kawato, M., Suzuki, R. (1989). Formation and control of optimal trajectory in human arm movement -minimum torque-change model, *Biological Cybernetics,* **61**, pp. 89 - 101