Coordinates Transformation and Learning Control for Visually-Guided Voluntary Movement with Iteration: A Newton-Like Method in a Function Space

M. Kawato*, M. Isobe, Y. Maeda, and R. Suzuki**

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

Abstract. In order to control visually-guided voluntary movements, the central nervous system (CNS) must solve the following three computational problems at different levels: (1) determination of a desired trajectory in the visual coordinates, (2) transformation of the coordinates of the desired trajectory to the body coordinates and (3) generation of motor command. In this paper, the second and the third problems are treated at computational, representational and hardware levels of Marr. We first study the problems at the computational level, and then propose an iterative learning scheme as a possible algorithm. This is a trial and error type learning such as repetitive training of golf swing. The amount of motor command needed to coordinate activities of many muscles is not determined at once, but in a step-wise, trial and error fashion in the course of a set of repetitions. Actually, the motor command in the \((n+1)\)-th iteration is a sum of the motor command in the \(n\)-th iteration plus two modification terms which are, respectively, proportional to acceleration and speed errors between the desired trajectory and the realized trajectory in the \(n\)-th iteration. We mathematically formulate this iterative learning control as a Newton-like method in functional spaces and prove its convergence under appropriate mathematical conditions with use of dynamical system theory and functional analysis. Computer simulations of this iterative learning control of a robotic manipulator in the body or visual coordinates are shown. Finally, we propose that areas 2, 5, and 7 of the sensory association cortex are possible sites of this learning control. Further we propose neural network model which acquires transformation matrices from acceleration or velocity to motor command, which are used in these schemes.

1 Introduction

Marr (1982) pointed out that an information processing device (brain) must be understood at the following different levels before one can be said to have understood it completely. (i) Computational theory, (ii) Representation and algorithm, (iii) Hardware implementation. We proposed a computational model of voluntary movement in Fig. 1 (Kawato et al. 1987a; Kawato et al. 1987b; Kawato 1988), which accounts for Marr's first level. Consider a thirsty person reaching for a glass of water on a table. The goal of the movement is moving the arm toward the glass to reduce thirst. First, one desirable trajectory in the task-oriented coordinates must be selected from out of an infinite number of possible trajectories which lead to the glass, whose spatial coordinates are provided by the visual system (determination of trajectory). Second, the spatial coordinates of the desired trajectory must be reinterpreted in terms of a corresponding set of body coordinates, such as joint angles or muscle lengths (transformation of coordinates). Finally, motor commands (e.g. torque) must be generated to coordinate the activity of many muscles so that the desired trajectory is realized (generation of motor command).

Kawato et al. (1987a) summarized several lines of experimental evidence which suggest that information in Fig. 1 is internally represented in the brain. Several works have been done regarding the step-by-step information processing in the central nervous system (CNS) shown by the three straight arrows in Fig. 1. For example, Flash and Hogan (1985) proposed a minimum-jerk model which explains computation of a desired trajectory in the task-oriented coordinates based on some performance index (goal of movement). Grossberg and Kuperstein (1986) and Psaltis et al.
(1987) proposed neural networks which execute coordinates transformation. However, we do not adhere to the hypothesis of the step-by-step information processing. Rather, Uno et al. (1987) proposed a learning algorithm which calculates the motor command directly from a goal of the movement represented as minimization of integral of square of torques' change (broken and curved arrow in Fig. 1). Further, as shown by a solid curved arrow in Fig. 1, in this paper we will show that motor command can be obtained directly from the desired trajectory represented in the task-oriented coordinates by an iterative learning algorithm. In this respect, our model differs from the three-level hierarchical movement plan proposed by Hollerbach (1982).

In this paper, we study the problems of coordinates transformation and motor command generation mainly at the computational and representational levels of Marr (1982). We propose an iterative learning control as an algorithm for simultaneously solving these two problems, and further provide mathematical proof which guarantees its convergence. Examples of simulation experiments which demonstrate efficiency of the proposed iterative learning scheme are presented. Finally, we propose that areas 2, 5, and 7 of the cerebral cortex are possible sites of this computation.

2 Computational Theory of Iterative Learning Control

Let us consider the problem of motor control in a computational framework. There exists causal relation between the motor command and the resulting movement pattern. Let $x(t)$ denote the time course of the motor command (torque) and $y(t)$ denote the time course of the movement trajectory in a finite time interval $t \in [0, T]$. The causal relation between $x$ and $y$ can be written as $G(x(\cdot)) = y(\cdot)$ using a functional $G$. If a desired movement pattern is denoted by $y_d(t)$, the movement error is defined as $F(x) = G(x) - y_d = y - y_d$.

The problem to generate a motor command $x_g$, which realizes the desired movement pattern $y_d$, is equivalent to find a root of the functional $F$. In other words, it is equivalent to find an inverse of the functional $G$.

Because controlled objects in motor control have many degrees of freedom and nonlinear dynamics, the functionals $G$ and $F$ are also nonlinear. Furthermore, since the exact dynamics of the controlled objects (arms, limbs, robotic manipulators) are generally unknown, we do not know exact forms of $G$ or $F$. Consequently, it is practically very difficult to calculate the inverse of $G$ or the root of $F$. We proposed a neural network model and the feedback-error learning rule, by which an internal neural model of inverse dynamics of the motor system (i.e. $G^{-1}$) is acquired during execution of movement (Kawato et al. 1987a; Kawato et al. 1987b; Kawato 1988). In this neural-network model, real time learning with hetero-synaptic plasticity takes place in the cerebellum and the red nucleus with using the feedback motor command as the error signal.

In this paper, we propose an iterative method to obtain the root of the functional $F$ even if the dynamics of the controlled system is unknown. This is a mathematical modification of the well known Newton method and is called Newton-like method. Let the space of the motor command be denoted by $X$ and that of the movement pattern by $Y$. The functional $F$ determines an error associated with a specific motor command.

$$F: X \rightarrow Y.$$  \hspace{1cm} (2.1)

The Newton method to find a root of $F$ is given as follows.

$$x^{n+1} = x^n - F(x^n)^{-1} F(x^n).$$  \hspace{1cm} (2.2)

However, this scheme cannot actually be used since we do not know the dynamics of the controlled system and hence we do not know the derivative of the functional: $F$. Instead, we can utilize the following Newton-like method, in which an approximation $M \in L(Y, X)$ of $F^{-1}$ is used. Here, $L(Y, X)$ is a space of linear operator from $Y$ to $X$ and $M$ can be somehow computed.

$$x^{n+1} = x^n - M(x^n) F(x^n).$$  \hspace{1cm} (2.3)

In succeeding sections we will give several practical iterative schemes of this type for motor control of multi-degrees of freedom objects in body and visual spaces. When the desired trajectory is represented in body coordinates, the scheme solves the problem of control. When it is represented in the visual task-oriented coordinates, the scheme solves both the problems of coordinates transformation and control.

The Newton-like method can be applied to various dynamical problems other than motor control. For
example, we proposed an iterative scheme for single electrode voltage clamp of neurons in the brain (Park et al. 1981; Kawato et al. 1985), and have successfully applied it to measurement of synaptic currents at dendrites of red nucleus neurons (Murakami et al. 1986).

3 Newton-Like Method in Body Space

3.1 Newton-Like Method with Constant Transformation Matrices

The dynamics of the musculoskeletal system or the robotic manipulator is described by the following differential equation:

\[ \frac{dy}{dt} = h(y, z) + R^{-1}(z)x, \]

\[ \frac{dz}{dt} = y + \omega, \]

\[ y(0) = 0, z(0) = 0. \]  

(3.1)

Here, \( x \) is torque input (i.e. motor command), \( y \) is velocity, \( z \) is position (e.g. joint angle, muscle length), and \( \omega \) is the initial (angular) velocity. \( R \) is an inertia matrix, which is positive definite and always invertible. The product \( Rh \) corresponds to a summed torque of centripetal, Coriolis, frictional and gravitational forces. \( h \) and \( R^{-1} \) are assumed continuously differentiable. As explained in the previous section, the problem to find the desired torque \( x_d \) which realizes the desirable velocity \( y_d \) and the desirable trajectory \( z_d \), within a finite time interval \([0, T]\), is equivalent to obtaining the root of the nonlinear functional \( F(x) = y - y_d \). It can be solved by the following Newton method.

\[ x^{n+1} = x^n + R(z^n)[d(y_d - y^n)]/dt \]

\[ -D_h(y^n, z^n) \times (y_d - y^n) \]

\[ -D_z[h(y^n, z^n) + R^{-1}(z^n)x^n] \times (z_d - z^n). \]  

(3.2)

Here, \( x^n, y^n, z^n \) are torque, velocity and trajectory during the \( n \)-th iteration. In this scheme, the motor command in the \( (n + 1) \)-th iteration is a sum of the motor command in the \( n \)-th iteration plus the three modification terms which are, respectively, proportional to acceleration, speed and position errors between the desired trajectory and the realized trajectory in the \( n \)-th iteration.

It is not so difficult to see that (3.2) is the Newton method 2.2 in function spaces (see Sect. 5). However, this scheme cannot be realized since the dynamics of the controlled object is unknown, and hence the matrices \( R, D_z, h, D_z(h + R^{-1}z^n) \) in (3.2) are unknown. On the other hand, the following Newton-like method can be actually implemented.

\[ x^{n+1} = x^n + R(0)[d(y_d - y^n)]/dt \]

\[ -D_h(0, 0) \times (y_d - y^n) \]

\[ -D_z(0, 0) \times (z_d - z^n). \]  

(3.3)

Here, \( R(0), R(0)D_zh(0, 0), R(0)D_zh(0, 0) \) are matrices estimated at the initial state. They are transformations, respectively, from acceleration, velocity and position to torque. This kind of Newton-like method can be formally formulated as (2.3).

The Newton-like method 3.3 can be practically utilized since the transformation matrices can be easily estimated from the following step response experiment. Let \( y(t) \) denote the step response of velocity when a unit-step torque is applied only to the \( i \)-th actuator with the initial condition \( (y, z) = (0, z_0) \) at time \( t = 0 \). Then, the transformation matrices can be approximately estimated as follows.

\[ R(z_0) \sim (y(0), y(0), ..., y(t_0))^{-1} \]

\[ R(z_0)D_zh(0, z_0) \sim -(y(\infty), y(\infty), ..., y(\infty))^{-1}. \]  

(3.4)

Here, \( n \) is the number of actuators. We assume that gravitational force is compensated beforehand.

In this paper, for simplicity, we use a model of an industrial robotic manipulator with three degrees of freedom and three actuators shown in Fig. 2 as a controlled object. Although it is much simpler than musculo-skeletal systems such as a human arm, they both have several essential features (nonlinear dynamics, multiple degrees of freedom, and interactions.}

Fig. 2. A robotic manipulator with three degrees of freedom. \( \theta_1, \theta_2, \theta_3 \) show joint angles of the first, second and third joints, respectively. \( l_0, S_0, M_0, l_1, l_2, l_3 \) and \( l_4 \) represent the length, the position of the center of mass, the mass, the three inertial moments of the \( k \)-th link, respectively. See Table 1 for values of these physical parameters.
between different freedoms) in common. Its physical parameters are shown in Table 1. According to Arimoto (1984) and Arimoto and Miyazaki (1984), we incorporated the dynamics of the electro-mechanical coupling in the motor into the dynamics equation of the manipulator. Exact shapes of the dynamics equation of the manipulator and motors can be found as (33) of Arimoto (1984). Because of the electro-mechanical coupling of the motor, the overall dynamics equation contains the frictional term proportional to the velocity, which dissipates energy. Further, its inertia matrix \( R \) contains a constant and diagonal part, which is proportional to the square of the reduction ratio of gear between the motor and the joint. In the following simulations, the reduction ratios were chosen low (\( \leq 1 \)), so this constant part of \( R \) is negligible. Consequently, overall dynamics of the manipulator used in our simulations had strong nonlinearity.

Figure 3 shows simulation of the step response experiment for estimation of transformation matrices. The left column of Fig. 3 shows angular velocities of the three joints and the right column shows voltage inputs to the three actuators. In this case, only the input to the third actuator was non zero. It is easy to see that the above matrices correspond to transformations from acceleration to torque and from velocity to torque, respectively. Although these estimations are of course not exact, it does not matter both from mathematical and practical standpoints as seen in the next theorem.

The following theorem can be proved regarding the convergence of the scheme 3.3 (see Sect. 5 for proof).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>First link</th>
<th>Second link</th>
<th>Third link</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_a (m) )</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>( S_a (m) )</td>
<td>5.0</td>
<td>30</td>
<td>2.0</td>
</tr>
<tr>
<td>( M_a (kg) )</td>
<td>0.1166</td>
<td>0.0644</td>
<td>0.0429</td>
</tr>
<tr>
<td>( I_a (kgm^2) )</td>
<td>0.1166</td>
<td>0.0644</td>
<td>0.0429</td>
</tr>
<tr>
<td>( I_a (kgm^2) )</td>
<td>0.0250</td>
<td>0.0038</td>
<td>0.0025</td>
</tr>
</tbody>
</table>

**Fig. 3. Simulation of the step response experiment for estimation of transformation matrices.** The left column shows the velocity of the three joint angles of the manipulator. The right column shows time courses of input voltages \( V_1, V_2, V_3 \) to the three motors of the manipulator. \( \phi_k \) and \( \dot{\phi}_k \) represent the \( k \)-th element of the input torque and the velocity, when a unit-step torque is applied only to the \( i \)-th actuator. It implies that \( \phi_k = 0 \) if \( i \neq k \).
Theorem 1. If the dynamics Eq. (3.1) is dissipative, input $x$ is bounded, and $|I - R(0)R^{-1}(0)| < 1/3$ holds, then the Newton-like method 3.3 converges exponentially, regardless of the starting point $x_0$.

The condition of the theorem is satisfied for usual industrial robotic manipulators with high reduction-ratio gears from motors to joints, because the high reduction-ratio dramatically weakens the nonlinearity of the manipulator dynamics. Consequently, the inertia matrix $R$ of usual industrial robotic manipulators does not change much for various postures $z$. But the assumption of the theorem is not satisfied for direct-drive manipulators or human arms. So, for these controlled objects, the scheme 3.3 might become unstable in the course of repetitions.

When we used the scheme 3.3 for control of a relatively small movement (all the three joint angles must change 2.0 rad during 1 s), the repetitions diverged as shown in Fig. 4. The left column shows the time courses of the three joint angles and the middle column shows angular velocities of the three joints. Desired time courses are shown by broken curves with $d$ and the realized time courses are shown by solid curves. The right column represents the input voltage to the three motors. The number attached to the curves represents the iteration number. The velocity time course of the first joint angle oscillated wildly and amplitudes of the input voltages were very large (scale of the ordinate of the right column are very large). That is, iterative control diverged in the course of repetitions. Simulation results of succeeding control experiments are shown in a similar format.
initial posture and they were used throughout the entire movement. However, for controlled objects with strong nonlinearity, the transformation matrices change substantially along the desired trajectory. To solve this problem we can estimate the transformation matrices at various postures and can use their interpolations so that the Newton-like method gets closer to the original Newton method 3.2. Interpolated approximations of matrices $R, RD, h$ are denoted by $\bar{R}, \bar{RD}, \bar{h}$. The Newton-like method which is closer to the original Newton method can be derived using these interpolated matrices:

$$x^{n+1} = x^n + \bar{R}(z^n)[d(y_d - y^n)/dt - D_0\bar{h}(0, z^n) \times (y_d - y^n)].$$

(3.5)

Sufficient conditions for convergence of this modified Newton-like method are given in Theorem 2 in Sect. 4.1: If $\|I - R^{-1}(z)R(z)\| < 1$ and the starting point $x^0$ is sufficiently close to $x_d$, then the scheme 3.5 exponentially converges. We must note that the condition about the closeness of the interpolated matrix $\bar{R}(z)$ to the real inertia matrix $R(z)$ is rather mild. That is, the estimation of the interpolated approximation $\bar{R}(z)$ needs not be very precise. This point is very important both from the engineering and physiological viewpoints.

The modified Newton-like method was certain to converge for manipulators with low reduction-ratio gears. Figure 5 shows results of iterative control of the same movement as Fig. 4 while using the scheme 3.5 with interpolated transformations. As can be seen, the desired trajectory and the velocity time course were realized almost perfectly after about 4-th iteration.

From physical symmetry of the dynamics of the robotic manipulator, we can easily assert that the transformation matrices $R, RD, h$ do not depend on the first joint angle $\theta_1$, and they are periodic regarding the second and the third joint angles $\theta_2, \theta_3$ with the period $2\pi$. Based on this a priori knowledge about the controlled system, the interpolated transformation matrices $\bar{R}, \bar{RD}, \bar{h}$ were represented as the following double trigonometric Fourier series in $(\theta_2, \theta_3)$. Only
expression for $\dot{R}$ is given:

$$\dot{R}(\theta_2, \theta_3) = \sum_{k=1}^{2} \sum_{i=1}^{2} a_{ki} \sin(2\pi k \theta_2) \sin(2\pi i \theta_3)$$

$$+ \sum_{k=0}^{2} \sum_{i=1}^{2} b_{ki} \sin(2\pi k \theta_2) \cos(2\pi i \theta_3)$$

$$+ \sum_{k=0}^{2} \sum_{i=1}^{2} c_{ki} \cos(2\pi k \theta_2) \sin(2\pi i \theta_3)$$

$$+ \sum_{k=0}^{2} \sum_{i=0}^{2} d_{ki} \cos(2\pi k \theta_2) \cos(2\pi i \theta_3).$$

This expression has 25 unknown coefficients $a_{ki}$, $b_{ki}$, $c_{ki}$, $d_{ki}$. In order to estimate these coefficients, the transformation matrices $R$, $RD_\theta h$ were estimated at 25 different postures (5 different $\theta_2$ x 5 different $\theta_3$) as described by (3.4) and the coefficients were determined by solving the resulting set of linear algebraic equations. We found that the estimated interpolated matrices were fairly close to the real transformation matrices even with these simple procedures.

Later in discussion we will propose neural network models which automatically estimate these interpolated transformation matrices based on synaptic plasticity in the CNS.

4 Iterative Learning Control in Visual Space

4.1 Derivation of Newton-Like Methods

Let us consider the iterative learning control in the task-oriented (visual) coordinates $(p, q)$. $p$ is velocity and $q$ is position. $z$ and $y$ represent position and velocity in the body coordinates as in the previous section. The coordinates transformation and the associated Jacobian $J$ are denoted as follows:

$$q = f(z),$$

$$p = df(z)/dz \times y = J(z)y.$$  \hspace{1cm} (4.1)

In the task oriented coordinates, the dynamics equation of the controlled system is given as follows:

$$dp/dt = g(p, q) + Q^{-1}(q)x,$$

$$= J'(z)(J^{-1}(z)p)^2 + J(z)[h(J^{-1}(z)p, z) + R^{-1}(z)x]$$

$$dq/dt = p + v_i,$$

$$p(0) = 0, q(0) = 0.$$  \hspace{1cm} (4.2)

Here, $v_i$ is the initial velocity in the task-oriented coordinates. Although (4.2) is superficially similar to (3.1), $Q = RJ^{-1}$ is no longer positive definite and changes dramatically for various postures $q$. Actually, (4.2) can not be defined at a singular posture, where the Jacobian $J$ is not invertible. Due to this fact, the simple scheme 3.3 can be by no means be utilized in the visual space. The Newton-like method which determines the necessary torque for a desired trajectory $(p_d, q_d)$ represented in the task-oriented coordinates are classified into the following two types. Both of them can be easily derived from the corresponding Newton methods [i.e. from a linearized Eq. of (4.2)].

The first is the method in which everything is done in the visual coordinates $(p, q)$. In this method, the transformation matrices $Q$, $D_p \dot{g}$ are estimated from time courses of $(p, q)$ in response to a step input to each actuator, similarly to the estimation in the body space. The iterative scheme is given as follows using these transformation matrices.

$$x^{n+1} = x^n + \dot{Q}(q^n)[(dp_d - p^n)/dt - D_p \dot{g}(0, q^n)(p_d - p^n)].$$  \hspace{1cm} (4.3)

Here, $\dot{Q}$ and $D_p \dot{g}$ are approximations of $Q$ and $D_p g$. $\dot{Q}$ and $D_p \dot{g}$ can be constructed by interpolating estimated values of matrices $Q$, $D_p g$ at several different postures within the visual coordinates similarly to estimation in the body coordinates described in Sect. 3.2.

The other Newton-like method utilizes both visual and somatosensory informations.

$$x^{n+1} = x^n + \dot{R}(z^n)[J^{-1}(z^n)p_d - p^n]/dt$$

$$- D_\theta \dot{h}(0, z^n)J^{-1}(z^n)(p_d - p^n)].$$  \hspace{1cm} (4.4)

Here, $R$ and $D_\theta \dot{h}$ are estimated and interpolated in step response experiments (3.4) within the body space as explained in Sect. 3.2. It must be noted that the proposed scheme (4.4) in the task-oriented coordinates is reduced to the scheme (3.5) in the body coordinates, if the task-oriented coordinates coincides with the body coordinates. For, in this case, $J = J^{-1} = I$ and $p = y$, $p_d = y_d$ hold. This point is important since the following theorem can also be used to guarantee convergence of the simulation experiments described in Sect. 3.2.

The following theorem can be proved about the convergence of the method 4.4 (see Sect. 5 for proof).

Theorem 2. If $|I - J(z)R^{-1}(z)\dot{R}(z)J^{-1}(z)| < 1$ and the starting point $x^0$ is sufficiently close to $x_d$, then the scheme 4.4 exponentially converges.

The first condition of the theorem is satisfied if $\dot{R}$ is not a so bad approximation of $R$, and $z$ does not approach the singular postures. By comparing Theorem 2 with Theorem 1, we note that the extra condition on the closeness of the starting point to the true value is required for iteration in the visual space.

It is not so difficult to see that a similar theorem for convergence of the scheme (4.3) can be obtained with the corresponding condition $|I - J(z)R^{-1}(z)\dot{Q}(z)| < 1$ on the estimated transformation matrix. This corollary of Theorem 2 can be used to guarantee convergence of simulation experiments presented in the next subsection.
4.2 Simulation of Iterative Control Within the Visual Space

We succeeded in computer simulation of iterative learning control of a hand position of the manipulator observed by a camera in the polar coordinates with use of the scheme 4.3. Arrangement of the camera and the manipulator and the definition of the polar coordinates $(\theta_1, \psi, r)$ are illustrated in Fig. 6.

The interpolated transformation matrices $\mathbf{Q}$ and $\mathbf{D}_q$ were represented as an expansion in $\psi, r$ similarly to the double Fourier series in (3.6). Based on the periodicity of the matrices in $\psi$, they were expanded as the trigonometric Fourier series in $\psi$ up to the second order and polynomial series in $r$ up to the fourth order. Consequently, this expansion again contains 25 unknown coefficients. They were estimated at 25 different postures (5 different $\psi \times 5$ different $r$) using the step response experiment. It must be noted that any coordinates transformation or its Jacobian was not used at all in this simulation experiment. In some sense, they were all contained in the estimated, interpolated transformation matrices.

Figure 7 shows results of the computer simulation. The left column shows desired trajectory and realized trajectory in the first and the sixth iteration in the visual polar coordinates. The middle column shows velocity time courses of the desired and the realized

![Figure 6. Arrangement of the camera and the manipulator. The definition of the polar coordinates $(\theta_1, \psi, r)$ is also illustrated](image)

![Figure 7. Simulation results of iterative control in the visual polar coordinates with the scheme 4.3. The left column shows trajectories in the visual polar coordinates. The middle column shows velocity time courses. The right column represents input voltage waveforms to the three actuators](image)
movements. The right column represents input voltage waveforms to the three actuators. After six iteration, the desired trajectories in $\theta_i$ and $\psi$ were almost realized, but the error was still observed in the $r$ axis. The realized trajectory got closer to the desired trajectory after several more iteration.

In this method, although no coordinates transformation from the visual to the body space was done, the two problems of coordinates transformation and the generation of motor command were simultaneously solved, as shown by a curved arrow in Fig. 1.

4.3 Simulation of Iterative Control Using Visual and Somatosensory Informations

We used the algorithm 4.4 for learning control of an arm in binocular-retinal coordinates by iteration. As shown in Fig. 8(a), the hand position of the manipulator was measured by two cameras (eyes). The Cartesian coordinates $(X, Y)$ of the hand image on the receptive surface of the left eye and the right eye were designated as $(X_1, Y_1)$ and $(X_2, Y_2)$, respectively. The desired trajectory was given in this four-dimensional binocular coordinates $(2 \times 2)$. The two cameras were set at the height of the shoulder of the manipulator. Their distances from the manipulator were equal. Directions of the two cameras were parallel.

Since the dimension 4 of the observable quantities $(X_1, Y_1, X_2, Y_2)$ is larger than the dimension 3 of the controllable quantities (input voltages to the three motors $V_1, V_2, V_3$), the Jacobian $J$ is not a square matrix and its inverse $J^{-1}$ cannot be defined in the usual sense. We adopted the "singularity low-sensitive motion resolution matrix" $J^*$ proposed by Nakamura and Hanafusa (1984) as a pseudo-inverse of the Jacobian. This matrix is actually the pseudo-inverse of $J$ which minimizes $[(I - JJ^*)p]^2 + k\|J^*p\|^2$. The first term of this performance index requires that the inverse transformation is mathematically exact, and the second term requires that the joint angles does not move too much. $k$ is a weighting ratio of these two requirements. The concrete formula of $J^*$ is given as follows.

$$J^* = (J^TJ + kI)^{-1}J^T.$$  \hspace{1cm} (4.5)

In the simulation, the Jacobian $J^*$ was calculated as follows. First, a model equation of coordinates transformation (4.1) was analytically derived. This equation contains several unknown parameters such as distance between the camera and the manipulator. Then, for several configurations, the hand position was measured both in the body coordinates and the visual coordinates. Based on these data, the parameters of coordinates transformation were determined on the least square error criterion. Finally, the Jacobian $J$ was calculated analytically from the corresponding coordinates transformation. $J^*$ was calculated from (4.5) using the above estimated parameters. These procedures are explained in Appendix.
The transformation matrices $R, RD, h$ were estimated and interpolated exactly the same as in the body space described in Sect. 3.2.

Figure 8b shows the results of iterative learning control. After about five iterations, the desired trajectories and the velocity time courses in the binocular retinal coordinates were almost perfectly realized. Since the four entities of the retinal coordinates are not independent, the desired trajectory which is given on the two eyes independently might not be able to be realized in the three dimensional space. In the simulation shown in Fig. 8, the desired trajectory was realizable. We did various simulations including unrealizable desired trajectories. In summary, first, the desired trajectory which can be realized in the three dimensional space was achieved by the iterative control. Second, even if the desired trajectory given in the binocular-retinal coordinates is unrealizable, the iteration soon settled down to some trajectory, which compromised two contradictory requirements (desired trajectories) on the two coordinates to some extent. Third, the desirable trajectory passing through the singular point can also be accomplished by this scheme (Isebo et al. 1987), whereas Theorem 2 does not guarantee the convergence for a trajectory via the singular points.

We present another example of application of the scheme 4.4 to trajectory control in the task-oriented coordinates. Simulations were done about iterative learning control of a hand position of a three degrees of freedom manipulator within a plane (Fig. 9). Physical parameters of the manipulator are shown in Table 2. As shown in Fig. 9, this manipulator has three degrees of freedom (shoulder, elbow and wrist) and corresponding three motors. The position of the shoulder is fixed at the origin $O$ of the Cartesian coordinates $(X, Y)$ of the work plane. If only the position of the hand is controlled, this manipulator has one dimensional redundancy ($3 - 2 = 1$). The three link structure in Fig. 9 shows an initial configuration of the manipulator. The chain line is a desired trajectory of the hand position. The solid curves show realized trajectories in the 1, 3, 4, and 5-th iterations. The movement time was 1 s. After about 8 iterations, the desired trajectory was almost perfectly realized.

As in the previous simulation shown in Fig. 8, we adopted the "singularity low-sensitive motion resolution matrix" as a pseudo-inverse of the Jacobian. The Jacobian which is necessary for calculating the pseudo-inverse matrix from (4.5) was calculated analytically from the manipulator model kinematics. The transformation matrices $R, RD, h$ were estimated and interpolated similarly in the body space as described in Sect. 3.2.

The one dimensional redundancy of the manipulator shown in Fig. 9 can be used to execute some extra tasks other than the trajectory control of the hand position. Based on the idea of Hanafusa et al. (1983), the inverse of the Jacobian in the scheme 4.4 was divided into two parts, each of which corresponds to the high priority task (trajectory control of hand position) and an extra task with low priority, respectively. For example, we found that the time course of the direction of the hand can be simultaneously controlled as well as the hand position. In Fig. 10 we show simulation experiment in which the extra task was avoidance of an obstacle (more specifically, the time course of the first joint angle $\theta_1$ was controlled with second priority). When this extra task was not incorporated into the iterative scheme, the manipulator collided with the obstacle. On the other hand, the iterative scheme 4.4 with the decomposition of the
inverse Jacobian realized the hand trajectory strictly while avoiding the obstacle after 12 iterations as shown in Fig. 10.

5 Proof of Convergence

In this section we give proofs of Theorem 1 and Theorem 2. That is, we will prove convergence of the schemes 3.3 and 4.4 under appropriate mathematical assumptions. Proof of Theorem 1 is similar to that given in Kawato et al. (1985), but is more complicated because of the presence of the factor \( R^{-1}(z) \) multiplied to the motor command \( x \) in the dynamics Eq. (3.1).

5.1 Proof of Theorem 1

As pointed out repeatedly, the scheme 3.3 can be regarded as a Newton-like method in function spaces. A function space of \( x \), motor command or input torque, is defined as \( X = C[0, T]^m \), a space of \( m \)-dimensional continuous functions within the time interval \([0, T]\). A function space of velocity \( y \) is defined as \( Y = C^1[0, T]^m \) a space of continuously differentiable functions within the interval \([0, T]\) with the zero initial condition \( y(0) = 0 \). Continuous differentiability of \( y \) is guaranteed from the same properties of \( h \) and \( R^{-1} \) of the dynamics Eq. (3.1). Let us define a nonlinear mapping \( F: X \to Y \) as follows.

\[
F(x) = y - y_x,
\]

where \( y_x \) is desired velocity time course, and \( y \) is a solution of (3.1) when the torque input \( x \) is applied. The problem of the control is equivalent to find the zero of \( F \). In the scheme 3.3, the approximation \( Mf^{-1} \) is expressed as follows.

\[
Mf = R(0) \left\{ \frac{dy}{dt} - D_hh(0, 0) \times y - D_xh(0, 0) \left\{ \int_0^t y(t) \right. \right\} \right\}.
\]

This operator clearly belongs to \( L(Y, X) \). Because the proof is almost identical, for simplicity of notation we will prove the convergence of the Newton-like method with the following approximation operator \( M \). That is, we omit the position error (integration) term.

\[
Mf = R(0) \{ \frac{dy}{dt} - D_hh(0, 0) \times y \}.
\]

We define norms of spaces \( X \) and \( Y \) as follows:

\[
\|x(t)\|_X = \|x(t)\|_X = \max_{1 \leq i \leq m, 0 \leq t \leq T} \|e^{-M}x(t)\|
\]

\[
\|y(t)\|_Y = \|My(t)\|_Y = \|My(t)\|_Y.
\]

For a finite \( h \), these norms are topologically equivalent to usual norms of \( C[0, T]^m \) and \( C^1[0, T]^m \) because of \( \gamma(0) = 0 \) and Gronwall's lemma. So, \( X \) and \( Y \) are Banach spaces with these norms. Note that the norm of \( M \) is equal to one with above definitions of norms of spaces \( X \) and \( Y \).

It is not difficult to see that the first derivative of \( F \) can be computed from a solution of the following variational system of the dynamics Eq. (3.1). This can be understood from the definition of \( F \), the dynamics Eq. (3.1), definition of norms of spaces \( X \) and \( Y \), and definition of derivative in functional spaces (cf. Hirsch and Smale 1974):

\[
F(x)^* = \xi,
\]

\[
\frac{dy}{dt} = h(y, z) + R^{-1}(z)x
\]

\[
\frac{dz}{dt} = y + \omega_t, \quad y(0) = 0, \quad z(0) = 0,
\]

\[
\frac{d\xi}{dt} = D_xh(y, z)\xi + R^{-1}(z)y + D_z\{h(y, z) + R^{-1}(z)x\}y,
\]

\[
\frac{d\eta}{dt} = \zeta, \quad \xi(0) = 0, \quad \eta(0) = 0,
\]

We prepare the following three lemmas for proof of Theorem 1.

Lemma 1. We can assume that \( |R^{-1}(z)| < K \) for \( z \in T^m \) because \( T^m \) is compact and \( R(z) \) is positive definite. Furthermore, we assume the following.

\[
|B| = \begin{pmatrix} D_xh(y, z) & D_z\{h(y, z) + R^{-1}(z)x\} \\ 0 & I \end{pmatrix} < N,
\]

for \( 0 \leq t \leq T \).
here matrix norm is defined as \(|A| = \max_{i,j} |a_{ij}|\). Then, the solutions of (5.4) satisfy
\[
\max(\|\xi\|_b, \|\eta\|_b) < \varepsilon \|y\|_b,
\]
and \(e \to 0\) as \(b \to \infty\).

Proof. Let us define \(v = (\xi, \eta)^T\). Multiplying both sides of (5.4) by \(e^{-bt}\), after integration, we get
\[
c^{-bt}v(t) = \left[ \int_0^t e^{-bs} e^{-s(i R^{-1}(z(s)) R^{-1}(z(0)))} ds \right] + \int_0^t e^{-bs} N e^{-s(i R^{-1}(z(s)))} w(s) ds.
\]

If we define \(w(t) = e^{-bt} \max|v(t)|\), then we have
\[
w(t) \leq K(1 - e^{-bt})/b \cdot \|y\|_b + \int_0^t N e^{-b(s)} w(s) ds,
\]
with \(0 \leq t \leq T\). (5.5)

Applying Gronwall's lemma to integral inequality 5.5 we get
\[
w(t) \leq (1 - e^{-bt})/b \cdot K \exp[N(1 - e^{-bt})/b] \|y\|_b,
\]
with \(0 \leq t \leq T\). (5.6)

Consequently,
\[
\max(\|\xi\|_b, \|\eta\|_b) \leq (1 - e^{-bt})/b \cdot K \exp[N(1 - e^{-bt})/b] \|y\|_b.
\]

Lemma 2. In addition to assumptions of Lemma 1, we assume that \(|R(0)| < L\) and \(|I - R(0) R^{-1}(z)| < 1/3 - \delta (\delta > 0)\) for all \(z \in T^m\). Then, for sufficiently large \(b\), the following estimation holds:
\[
\|I - F(x)M\| < 1/3.
\]

Proof. From definition of \(F\), we have \(F(x)My = \xi\) by substituting \(y = My\) into (5.4). Applying Lemma 1 to (5.4), the following estimation is obtained.

\[
\max(\|\xi\|_b, \|\eta\|_b) < \varepsilon \|y\|_b = \varepsilon \|My\|_b = \varepsilon \|y\|_y.
\]

From definition of operator norm,
\[
\|I - F(x)M\| = \sup_{y \in Y} \|y - F(x)My\|_Y.
\]

We estimate the numerator of this equation.
\[
\|y - F(x)My\|_Y = \|y - \xi\|_Y = \|M(y - \xi)\|_b
\]
\[
= \|R(0)\{dy/dt - d\xi/dt - D_d h(0,0)(y - \xi)\}\|_b
\]
\[
= \|R(0)\{dy/dt - D_d h(0,0)(y - \xi)\}\|_b
\]
\[
= \|R(0)\{dy/dt - D_d h(y,z,\xi) - R^{-1}(z) R(0) \{dy/dt - D_d h(0,0)(y - \xi)\}\}\|_b
\]
\[
= \|R(0)\{dy/dt - D_d h(y,z,\xi) - R^{-1}(z) R(0) \{dy/dt - D_d h(0,0)(y - \xi)\}\}\|_b
\]
\[
\leq \|R(0)\{I - R^{-1}(z) R(0)\}\{dy/dt - D_d h(0,0)(y)\}\|_b
\]
\[
+ \|R(0)\{-D_d h(y,z,\xi) - R^{-1}(z) x, \eta - D_d h(0,0)(y - \xi)\}\|_b
\]
\[
+ \|R(0)\{-D_d h(y,z,\xi) - D_d h(0,0)(\xi)\}\|_b
\]
\[
< \|1/3 - \delta\|_y + 2 L N e\|y\|_y
\]
\[
= (1/3 + 2 L N e - \delta)\|y\|_y < 1/3 \|y\|_y
\]
\([\varepsilon \to 0\) as \(b \to \infty\)].

Lemma 3. Under the assumptions of Lemma 2, the following estimation holds:
\[
\|F(x) - F(x*)\| < 2/3, \quad \text{for all } x, x* \in X.
\]

Proof. \(F(x)\gamma = \xi\) is defined by (5.3) and (5.4). Similarly, \(F(x*)\gamma = \xi*\) is defined by the same equations when replacing \(x, y, z, \xi, \eta\) by \(x*, y*, z*, \xi*, \eta*\) and preserving \(\gamma\). From Lemma 1, \(\max(\|\xi\|_b, \|\eta\|_b) < \varepsilon \|y\|_b\) and \(\max(\|\xi\|_b, \|\eta\|_b) < \varepsilon \|y\|_b\) hold. Then, the following estimation holds:
\[
\|F(x) - F(x*)\| = \|\xi - \xi*\|_b = \|M(\xi - \xi*)\|_b
\]
\[
= \|R(0)\{dy/dt - d\xi/dt - D_d h(0,0)(\xi - \xi*)\}\|_b
\]
\[
= \|R(0)\{D_d h(y,z,\xi) - R^{-1}(z) \gamma + D_d h(0,0)(\xi - \xi*)\}\|_b
\]
\[
+ \|R(0)\{D_d h(y,z,\xi) - R^{-1}(z) \gamma + D_d h(0,0)(\xi - \xi*)\}\|_b
\]
\[
\leq \|R(0)\{R^{-1}(z - R^{-1}(z) \gamma + D_d h(0,0)(\xi - \xi*)\}\|_b + 4 L N e\|y\|_b
\]
\[
< (1/3 - \delta + 4 L N e)\|y\|_b < 2/3 \|y\|_b
\]
\([\varepsilon \to 0\) as \(b \to \infty\)].

Consequently, we have:
\[
\|F(x) - F(x*)\| = \sup_{y \in Y} \|F(x)\gamma - F(x*)\gamma\|_Y < 2/3.
\]

(5.10)

Theorem 3. Let \(C\) denote a ball in \(X\) defined as follows.
\[C = \{x; \|x - x^0\|_X < \|F(x^0)\gamma(1 - \beta)\|_Y\} < 0 < \beta < 1\].

The subset of solutions of dynamics Eq. (5.3) to the input \(x \in C\) is denoted by \(D\). We assume that the following three conditions are satisfied for \(x \in C, y, z \in D\).
\[
|I - R(0) R^{-1}(z)| < 1/3 - \delta,
\]
\[
|B| < N,
\]
\[
\alpha = 1 + 6 L N e - 3 \delta < \beta < 1\] for \(b > 1\).

Then the Newton-like method \(x^{n+1} = x^n - M \cdot F(x^n)\) converges and the following holds.
\[
\|x^n - x_d\|_X < \|F(x^0)\gamma(1 - \alpha)\|_Y.
\]
Proof. \((y^*,z^*)\) denotes the solution of (5.3) with the input \(x^*\). Note that \((y^2, z^2) \in E_d\) and \((y^1, z^1) \in E_d\) since 
\[\|x^1 - x^0\|_x = \|M(x^2)\|_x = \|F(x^2)\|_y < \|F(x^2)\|_y/(1-\beta)\].
For the sake of induction we assume that \(x^k \in C\), 
\((y^k, z^k) \in D\) for \(k = 0, \ldots, n+1\). From the mean value theorem of derivative in norm space, we obtain:
\[
\|F(x^{n+1})\|_y \leqslant \|F(x^n) - F(x^n) F(x^{n+1} - x^n)\|_{y} \\
+ \|F(x^n) F(x^{n+1} - x^n)\|_{y} \\
+ K \|x^{n+1} - x_n\|_x + \|I - F(x^n) M\| F(x^n)\|_y,
\]
where \(K = \sup\{\|F(x) - F(x)\| : x \in S = \{(1-t)x^* + tz^{n+1} : 0 \leq t \leq 1\}, x \in C\) because \(x^*, x^{n+1} \in C\). Hence, we can bound \(K\) from Lemma 3. Since 
\[
\|F(x^{n+1})\|_y \leq (2/3 - 2\delta + 4LN\|x\|_y) \|F(x^n)\|_y \\
+ (1/3 + 2\delta + 2LN\|x\|_y) \|F(x^n)\|_y,
\]
holds, we can assume that \(x^{n+1} \in C\) and \((y^{n+2}, z^{n+2}) \in D\) from the following inequality.
\[
\|x^{n+2} - x^0\|_x \leq \sum_{i=0}^{n+1} \|M F(x^n)\|_y \\
+ \sum_{i=0}^{n+1} \|F(x^n)\|_y < \sum_{i=0}^{n+1} \|F(x^n)\|_y(1 - \alpha) \\
< \|F(x^n)\|_y/(1 - \beta).
\]
By a similar evaluation, we can show that \(\{x^n\}\) is a Cauchy sequence. Since \(X\) is a Banach space, \(\{x^n\}\) has a limit point \(x^*_n\). We can also show that \(\|x^*_n - x^0\|_x \leq \|F(x^n)\|_y/\alpha\) Finally, we obtain:
\[
\|F(x^n)\|_y \leq \|F(x^n)\|_y \|x^*_n - x^0\|_x - (n+\infty) \Rightarrow (5.11)
\]
Remark. As explained in Sect. 3.1, the dynamics equation 3.1 is dissipative because of the frictional force which is proportional to the velocity. Since the dynamics (3.1) is dissipative, \(D\) is bounded because of boundedness of \(C_M(x) \leq \delta\), and we can choose \(N\). Consequently Theorem 1 is derived from Theorem 3.

5.2 Proof of Theorem 2
We prove Theorem 2 by using Theorem 1 of Dennis (1968). The definition of the function space of \(x\) is the same as in the previous subsection. A function space of the velocity \(p\) in the task-oriented coordinates is also similarly defined as \(P = C([0, T]^n\). The nonlinear functional \(F : X \rightarrow P\) is defined as \(F(x) = p - p_k\) where \(p\) is the solution of dynamics Eq. (4.2) in the visual space.
We will prove that the Newton-like method 4.4 converges with the following approximation operator.
\[
M(x) p = \tilde{R}(z_k) [J^{-1}(z_k) dp/dt - D_p \tilde{R}(0, z_k) J^{-1}(z_k) p],
\]
(5.12)
here \(z_k\) is configuration trajectory in the body space when the motor command \(x\) is applied. Because of this dependence, the approximation operator \(M\) depends on \(x\) in this case. The norm of the space \(X\) is the same as in the previous subsection. The norm of the space \(P\) is defined as follows:
\[
\|p(t)\|_p = \|dp/dt\|_p.
\]

It is clear from dynamics Eq. (4.2) that there exists a unique \(x_N\) which realizes \(p_k\). We prove Theorem 2 by showing that conditions of Dennis's theorem are satisfied under above definitions of the spaces and the operator \(M\). If the starting point \(x^0\) is sufficiently close to the true root \(x_N\), then we only need to show that the following three conditions are satisfied.

(1) \(\forall x \in Q \subseteq X, \|M(x)\|_y \leq B,\)

(2) \(\forall x \in Q, \|F^*(x)\|_y \leq K,\)

(3) \(\forall x \in Q, \|I - F^*(x) M(x)\|_y \leq \delta < 1,\)

here \(\Omega\) is some closed subset of \(X\). \(\Omega\) can be chosen as a closed ball centered at the desired torque, which contains the starting torque. We can assume that the desired trajectory and the trajectory in the starting iteration are sufficiently apart from the singular postures.

(f) Because we can choose \(\Omega\) such that corresponding trajectories \(z_k\), \(x \in \Omega\) do not pass through the singular point, the following estimaion holds:
\[
\|M(x)\|_y = \sup_{p \in P} \|M(x) p\|_y \\
= \sup_{p \in P} \|\tilde{R}(z_k) [J^{-1}(z_k) dp/dt - D_p \tilde{R}(0, z_k) J^{-1}(z_k) p]\|_y \\
< L \max_{x \in \Omega} \|J^{-1}(z_k)\|_y = B.
\]
(5.13)

(2) Similarly to the calculation of the first derivative \(F^*\) by the first variational system 5.4 of (5.3), the second derivative \(F^{**}\) can be computed as a solution of the second-order variational system of the dynamics Eq. (4.2). By applying Lemma 1 to the first variational system and then to the second variational system repeatedly, we can bound the norm of the second derivative, \(\|F^*(x)\|\).

(3) We can show the third condition similarly to the proof of Lemma 2, by using the assumption \(\|I - J(z) R^{-1}(z) \tilde{R}(z) J^{-1}(z)\| < 1\).

6 Discussion
In this paper, we studied the problem of coordinates transformation and the problem of control from a
computational viewpoint and suggested that these problems can be solved by the Newton-like method in functional spaces. Based on this fundamental idea, several practical iterative algorithms were derived from dynamics equations of controlled objects in the body and visual spaces. By computer simulations, efficiency of these algorithms in coordinates transformation and control was demonstrated. Finally, we proved the convergence of these algorithms under appropriate mathematical assumptions.

Here we develop neural network model for this iterative learning control, which corresponds to the study at the hardware level of Marr. During visually guided voluntary movement, the parietal association cortex receives both the visual and somatosensory informations about controlled objects (arms and hands) (Mountcastle et al. 1975). We propose that some parts of sensory association cortex (areas 2, 5, and 7) solve the problems of coordinates transformation and generation of motor command simultaneously by an iterative learning algorithm, as shown by a curved arrow in Fig. 1. This is a trial and error type learning of a single motor pattern, such as repetitive training of golf swing. That is, the amount of motor command needed to coordinate activities of many muscles is not determined at once, but in a step-wise, trial and error fashion in the course of a set of repetitions. In this motor learning, short term memory of time courses of trajectory and torque are required. Because of this, we suppose that the iterative learning is conducted in sensory association cortex of the cerebrum, instead of cerebellum, red nucleus or hippocampus. For, Fukushima (1973) proposed a dynamic associative memory which can recall a time sequence of input patterns. A morphological structure of this neural network model has the strongest resemblance to the cerebral cortex. Another reason for proposition of the cerebral cortex as a possible site of the iterative learning control is that this kind of learning must be done consciously. The area 2 is supposed to be involved in motor learning in the body coordinates. The areas 5 and 7 are supposed to be involved in motor learning in the visual coordinates.

We now discuss how the CNS estimates the constant transformation matrices \( R(z_0) \) and \( R(z_0)D_\theta h(0,z_0) \) by an associative-memory neural network, and acquires the interpolated transformation matrices \( R(z) \) and \( R(z)D_\theta h(0,z) \) by a multi-layer, nonlinear-transformation neural network without the "step-response experiments" described in Sect. 3.1.

First, we propose an associative-memory network which learns \( R(z_0) \) based on cross-correlation between the motor command and acceleration. If gravitational force is compensated and the velocity is zero at a starting posture \( z_0 \), the dynamics Eq. (3.1) is simplified as follows:

\[
R(z)dy/dt = x. 
\]  

(6.1)

Amari (1977) showed that the \( i,j \)-th element of \( R(z_0) \) can be acquired as synaptic weights \( R_{ij} \) (i, j = 1, ..., n) by the following "orthogonal-learning rule", even in the case where various inputs patterns \( \{dy/dt\}_a \) are not orthogonal with each other:

\[
\tau dR_{ij}/dt = -R_{ij} + c\{(Rdy/dt)_a - x_i\}dy/dt, \]  

(6.2)

where \( \tau \) is a time constant of synaptic modification, and \( c \) is a positive constant. Noting the condition for derivation of (6.1), synaptic modification of \( R_{ij} \) must occur only just after the initiation of movement (i.e. \( y < 1 \)) and only around the posture \( z_0 \) (\( z \approx z_0 \)). In summary, the transformation matrix \( R(z_0) \) is acquired by the associative memory network with the input \( dy/dt \) and the teaching signal \( x \), which modifies the synaptic weights \( R_{ij} \) only just around the initial posture \( z_0 \) based on cross-correlation between the acceleration and the motor command.

Since the dynamics Eq. (3.1) is simplified as,

\[
R(z)D_\theta h(0,z)y = x \]  

(6.3)

at a constant velocity \( y \), the transformation matrix \( R(z_0)D_\theta h(0,z_0) \) is acquired in an associative memory network which modifies its synaptic weights based on cross-correlation between the motor command and the velocity during a constant, low velocity movement around a posture \( z_0 \).

The interpolated transformation matrices might be acquired as a set of these constant matrices. However, there is another possibility that the CNS learns directly the nonlinear input-output transformations defined by left sides of (6.1) and (6.3). This can be done by the nonlinear multi-layer neural network with heterosynaptic plasticity while choosing \( z, y, dy/dt \) as the inputs and \( x \) as the teaching signal (cf. Kawato et al. 1987a; Kawato 1988). Just the same as in learning of the constant matrices by the associative network, synaptic modification for \( R(z) \) occurs just after initiation of movements, and that for \( R(z)D_\theta h(0,z) \) takes place during constant speed movement. Similarly to the above schemes, the Jacobian \( J \), which is necessary for the iterative learning control in the visual space, is obtained by associative memory based on cross-correlation between visual and somatosensory displacement.

The iterative scheme 3.3 itself is only adaptation, since experiences obtained in the course of repetitions cannot be used for control of a different movement pattern. However, acquisition of the transformation matrices continues irrespectively of movement patterns, and it can be utilized for control of any movement trajectory. So, this corresponds to learning.
The iterative learning in the visual space solves the problems of coordinates transformation and control simultaneously, as shown by the solid curved arrow in Fig. 1. This computational scheme is superior to the step-by-step computations shown by straight arrows in Fig. 1, regarding accuracy of trajectory control. Coordinates transformation by neural networks either of mapping type (Grossberg and Kuperstein 1986) or input-output transformation type (Psaltis et al. 1987) inevitably contains some transformation errors. So, even if the desired trajectory in the body coordinates is perfectly realized, the step-by-step information processing can not precisely realize the desired trajectory in the visual task-oriented coordinates because of the error of coordinates transformation. On the other hand, in the iterative learning control, the degree of accuracy of control is not influenced by any errors in estimating transformation matrices or inverse Jacobian in the schemes 4.3 and 4.4. Consequently, arbitrary degree of accuracy of trajectory control in the visual space can be obtained within the measurement precision of the visual system. This merit is of course very important also in engineering application.

Let us compare the iterative learning control in the body space with the feedback-error-learning control proposed in Kawato et al. (1987a). Both learning schemes solve the problem of control (i.e. computation of necessary motor command which realizes the desired trajectory represented in the body space), but they are different in learning periods, precision of control and capability of generalization. The learning time required for iteration is usually much shorter (order of 10 s) than that for feedback-error-learning (at least 1 h). The control precision is much better in the iterative learning control than in the feedback-error-learning. The shortcoming of the iterative learning control is its complete lack of capability of generalization. When a new movement pattern is executed, the network must start the iteration from the very beginning even if it experienced a great number of other movement patterns. On the other hand, the network of feedback-error-learning can control any inexperienced movement pattern once it learned to control some movement pattern (Kawato et al. 1987).

In the brain, these two neural networks of iterative learning (sensory association cortex) and feedback-error-learning (cerebellum and red nucleus) are hierarchically arranged so that the two compensate their shortcomings with each other. Most of ordinary movements which does not require severe accuracy are controlled only by the feedback-error-learning network. When a high speed movement must be executed with high precision, the iterative learning network intervenes in the control. Even in this case, the presence of the feedback-error-learning network has good effects on overall control performance, by decreasing the iteration number and increasing the system stability. Actually Kawato et al. (1988) have recently succeeded in trajectory control of an industrial robotic manipulator (Unimate PUMA-260) by the hierarchical arrangement of the two neural networks, and found the above effects. They also succeeded in trajectory control of the manipulator in the visual space by using two position-sensor-head cameras as shown in Fig. 8.

Arimoto et al. (1984a, b) independently proposed the "betterment process" which is similar to the iterative scheme 3.3 with constant transformation matrices in the body space. They regarded their process as a contraction mapping and gave a different proof of convergence. Most important practical difference is that the transformation matrices are fixed in the betterment process and no way of estimating them is given. This might come from the fundamental conceptual difference of the two methods (Newton-like method vs. contraction mapping).

Atkeson and McIntyre (1986) proposed an iterative scheme in which an almost perfect nonlinear dynamics model of a controlled object is used in modification of motor command. First of all, an almost perfect model of a controlled object does not seem to reside in the brain. So, we used their algorithm for control of the PUMA-260 manipulator with the inverse dynamics model acquired by the feedback-error-learning. Its performance was worse than our Newton-like method which uses only linear information of the controlled system (Isobe, Kawato, Suzuki, unpublished observation). We do not understand exact reasons of this surprising result, but it might be possible that a linear method behaves better than a nonlinear method around the root x* of the functional F.

Appendix: Coordinates Transformation for Simulation Experiment in Sect. 4.3

In this appendix coordinates transformation from the three joint angles (θ1, θ2, θ3) to the visual task-oriented coordinates (X, Y, Z) for Fig. 8a is analytically derived.

First we calculate coordinates transformation from the joint angle coordinates to the manipulator standard coordinates. The hand position (x, y, z) of the manipulator is represented as follows:

\[ x = (L_2 \sin \theta_2 + L_3 \sin(\theta_2 + \theta_3)) \cos \theta_1 \]
\[ y = (L_2 \sin \theta_2 + L_3 \sin(\theta_2 + \theta_3)) \sin \theta_1 \]
\[ z = L_1 + L_2 \cos \theta_2 + L_3 \cos(\theta_2 + \theta_3) \]

Second we calculate coordinates transformation from the manipulator standard coordinates to the camera
standard coordinates of the hand position. Position of the origin $P_i$ of the camera standard coordinates measured in the manipulator standard coordinates is represented by $B_i = (B_{xi}, B_{yi}, B_{zi})$, and the rotation of the two coordinates system is represented by matrix $A_i$. Here, $i$ is 1 or 2 for the left and right camera, respectively. The rotation matrix $A_i$ is represented as follows if it rotates the coordinates system by $\xi_i$ around the z-axis, and then by $\eta_i$ around the y-axis, and by $\zeta_i$ again around the z-axis.

$$A = \begin{pmatrix}
\cos \xi \cos \eta \cos \zeta - \sin \xi \sin \zeta & \sin \xi \cos \eta & \cos \xi \\
-\cos \xi \cos \eta \cos \zeta - \sin \xi \sin \zeta & -\sin \xi \cos \eta & \cos \xi \\
\sin \xi \sin \eta & \cos \xi \sin \eta & \cos \xi
\end{pmatrix}
$$

Here, the index $i$ is suppressed. If the manipulator hand position is represented as $K = (x, y, z)$ in the manipulator standard coordinates, and represented as $C_i = (C_{xi}, C_{yi}, C_{zi})$ from the camera standard coordinates $P_i = Q_iR_iS_i$, the following relationship holds.

$$C_i = A (K - B_i)
$$

Finally we calculate the coordinates transformation from the camera standard coordinates to the camera visual coordinates. From a simple geometrical relationship, we obtain:

$$X_i = -k_i C_{zi} / C_{xi},
$$

$$Y_i = -k_i C_{yi} / C_{zi}.
$$

The parameter $k_i$ corresponds to the visual angle of the cameras.

Because we assume that the lengths of the three joints of the manipulator are known, there are 7 unknown parameters $(B_{xi}, B_{yi}, B_{zi}, \xi_i, \eta_i, \zeta_i, k_i)$ for each camera and 14 unknown parameters in the coordinates transformation from the joint angle coordinates to the binocular visual coordinates. These parameters were estimated by measuring the two coordinates values simultaneously at 27 different postures and so that these data were reproduced by the above equations on the least square error criterion.

The Jacobian which is used in the iterative scheme was first analytically derived from the above equations of coordinates transformation, then calculated with use of the estimated parameter values.

References


Received: November 18, 1987

Dr. Mitsuo Kawato
Cognitive Processes Department
ATR Auditory and Visual Perception Research Laboratories
Twin 21 Bldg. MID Tower
2-1-61 Shiromi Higashi-ku
Osaka
540 Japan