多層型ニューラルネットワークを用いた運動量変化 逆モデル学習によるマニピュレータ軌道制御*

堀 勝博**・陰山 正樹***・土谷 武士**

Manipulator Trajectory Control by Momentum Change Inverse Models Using Multilayer Neural Networks*

Katsuhiko HORI**, Masaki KAGEYAMA*** and Takeshi TSUCHIYA**

This paper proposes a learning method of inverse manipulator dynamics model using only position and velocity. The direct inverse modeling method that was proposed as a learning method using neural network requires sensing manipulator position, velocity, and acceleration, because this method is formularized on the basis of manipulator motion equation. However, since it is difficult at present to sense accurately manipulator acceleration, we could hardly implement this method by original formula. In the momentum change inverse modeling; the learning method that we proposed in this paper, manipulator motion causality is modeled not on the basis of manipulator motion equation but on the manipulator momentum change equation. With this formulation, sensing acceleration becomes unnecessary, inverse manipulator dynamics model can be learned using sensible position and velocity.

1. はじめに

マニピュレータの軌道制御では、非線形性や関節間の干渉が強く現れるので、簡単なフィードバック制御だけで精度良く高速に動作させることは一般に難しい。マニピュレータの正確なダイナミクスモデルが既知であるならば、モデルに基づいてフィードフォワード補償することが有効であるが、通常は、正確なモデルが事前に得られていないので、何らかの方法でマニピュレータのモデルを獲得することが必要となる。

そのためのアプローチとして、マニピュレータのように非線形関数で表わされる制御対象のモデルをニューラルネットワークを用いて学習する方法が様々な観点から提案されている^{1)~4)}。その中の一つに、CMACという機能的には多層型ニューラルネットワークと等価な神経

回路モデルを用いてマニピュレータの逆ダイナミクスモデルを学習によって獲得する Albus らの方法がある¹⁾。

これは、入力をトルク、出力を位置、速度、加速度とする順ダイナミクスモデルの入出力関係をそのまま逆にしたモデルを学習により獲得しようとする方法であり、制御対象のモデルをニューラルネットワークを用いて学習する方法の中では直接逆モデリング法という名称で分類されている^{4),5)}。この方法はマニピュレータの運動方程式という物理的性質に基づいているので、学習時にはマニピュレータの位置信号、速度信号とともに加速度信号が必要となる。しかし、現在は位置信号、速度信号に比べて加速度信号を正確に計測することは一般に難しく、この方法に実現性の点で原理的な限界があると考えられる。

本稿では、マニピュレータの逆モデル学習を位置信号と速度信号だけを用いて行うための一手法を提案する。本方法の基本的な考え方は、マニピュレータのダイナミクスを運動方程式ではなくその積分形である加速度に依存しない運動量変化関係式に基づいてモデル化することである。まず 2. でマニピュレータの運動量変化関係式に着目した逆モデル化(運動量変化逆モデル学習)の考え方について説明する。3. でその考え方を多層型ニューラルネットワークの学習則として定式化し、それを用い

* 原稿受付 1993 年 6 月 25 日

** 北海道大学 工学部 Faculty of Engineering, Hokkaido University; Kita 13-Jyo Nishi 8-chome, Kita ward, Sapporo-city, Hokkaido 060, JAPAN

*** ㈱ 東芝 研究開発センター Research & Development Center, Toshiba Co. Ltd.; 70 Yanagi-machi, Saiwai ward, Kawasaki-city, Kanagawa 210, JAPAN

Key Words : direct inverse modeling, momentum change inverse modeling, multilayer neural network, manipulator, trajectory control.

てマニピュレータの制御則を定式化する。4.では2リンクマニピュレータ軌道制御を例題としたシミュレーションによって、直接逆モデリング法との比較検討を加えて、本方法の有効性について検証する。最後に5.で今後の課題について述べる。

2. 運動量変化逆モデル学習の考え方

2.1 直接逆モデリング法

最も基本的なマニピュレータの逆モデリング法である直接逆モデリング法^{1),4),5)}は、マニピュレータの入力(トルク τ) から出力(位置 q ・速度 $\dot{q}$ ・加速度 $\ddot{q}$) への因果関係(運動方程式)を直接逆にした Fig. 1 のようなモデルを学習により獲得して、その因果関係の逆関係を利用してマニピュレータの動作を制御する方法である。この方法では、マニピュレータの運動の因果関係を、「各時刻における運動方程式」に基づいてモデル化するので、逆モデルを学習する時に運動方程式に現れる位置、速度信号とともに加速度信号の測定が必要となる。

Fig. 1 Direct inverse modeling

2.2 運動量変化に着目した逆モデル学習

これに対して、マニピュレータの運動の因果関係を、位置信号と速度信号で表わされる「二時刻間における運動量変化関係式」に基づいてモデル化するというのが、本稿で提案する運動量変化に着目した逆モデル学習(運動量変化逆モデル学習)の基本的な考え方である。

マニピュレータの運動量変化関係式は、運動方程式を時間積分して以下のように導出される。マニピュレータの運動方程式は、

$$M(q)\ddot{q} + \dot{M}(q)\dot{q} + \frac{\partial}{\partial q} \left\{ U(q) - \frac{1}{2} \dot{q}^T M(q) \dot{q} \right\} = \tau \quad (1)$$

で表わされる。ここで、

$q, \dot{q}, \ddot{q}$: 位置, 速度, 加速度ベクトル ($n \times 1$)

$M(q)$: 慣性マトリクス ($n \times n$)

$U(q)$: 重力によるポテンシャルエネルギー

τ : 入力トルクベクトル ($n \times 1$)

である。この式の時刻 kT から時刻 $(k+1)T$ 間の時間積分をとると、

$$\begin{aligned} & \int_{kT}^{(k+1)T} \left[M(q)\dot{q} + \dot{M}(q)\dot{q} \right. \\ & \quad \left. + \frac{\partial}{\partial q} \left\{ U(q) - \frac{1}{2} \dot{q}^T M(q) \dot{q} \right\} \right] dt \\ &= \int_{kT}^{(k+1)T} \tau dt \end{aligned} \quad (2)$$

左辺を変形して、

$$\begin{aligned} & [M(q)\dot{q}]_{kT}^{(k+1)T} \\ & + \int_{kT}^{(k+1)T} \left[\frac{\partial}{\partial q} \left\{ U(q) - \frac{1}{2} \dot{q}^T M(q) \dot{q} \right\} \right] dt \\ &= \int_{kT}^{(k+1)T} \tau dt \end{aligned} \quad (3)$$

が得られる。左辺第1項がマニピュレータの運動量変化分である。さらに、左辺第2項被積分関数の原始関数を $V(q, \dot{q})$ とおくと、

$$[M(q)\dot{q} + V(q, \dot{q})]_{kT}^{(k+1)T} = \int_{kT}^{(k+1)T} \tau dt \quad (4)$$

となる。

ここで、0次ホールドした制御入力を用いたデジタル制御によってマニピュレータを動作させることを前提とすると、入力 τ はサンプリング間で一定値 τ_k であるから、上式は、

$$[M(q)\dot{q} + V(q, \dot{q})]_{kT}^{(k+1)T} = \tau_k T \quad (5)$$

となる。この(5)式が本稿で着目するマニピュレータの運動量変化関係式である。

この式に基づいて、本稿では次のような手順でマニピュレータの逆モデル学習および軌道制御を行った。まずモデルを学習する段階では、Fig. 2のように、入力を時刻 $kT, (k+1)T$ の位置 q_k, q_{k+1} と速度 $\dot{q}_k, \dot{q}_{k+1}$ とし、出力(教師信号)を時刻 kT に加えたトルク τ_k として、運動量変化逆モデルの学習を行う。つづいて軌道制御の段階では、学習した Fig. 2 のモデルの入力に時刻 kT

Fig. 2 Momentum change inverse modeling

の位置、速度と時刻 $(k+1)T$ の制御目標位置、速度を入れ、その制御目標を達成するために時刻 kT に加えるべきトルクをモデルの出力から得て制御を行う。実際の制御時には、マニピュレータのパラメータ変動や外乱等によって目標軌道から外れる場合に対処するために、この運動量変化逆モデルによる制御器とPD型フィードバック制御器を組み合わせる。以上の手順のより具体的な説明と定式化については次節で説明する。

このように、運動量変化に着目した逆モデル学習では、ディジタル制御への適用を前提として、マニピュレータの運動の因果関係を加速度項が現れない「二時刻間における運動量変化関係式」に基づいてモデル化することで、加速度信号を使わずに位置と速度信号だけを利用して逆モデルを獲得することが可能となる。

3. 運動量変化逆モデル学習によるマニピュレータ軌道制御

3.1 運動量変化逆モデル学習則

前節(5)式に示した運動量変化に着目したマニピュレータの逆モデルにおいて、関数 $M(q)$ と $V(q, \dot{q})$ の構造、パラメータが既知ではないので、これをマニピュレータの動作試行データから学習することを考える。

逆モデルを表現するために、任意の関数を近似的に表わすことが可能な Fig. 3 のような m 層の多層型ニューラルネットワークを用いる。多層型ニューラルネットワークの第 k 層の第 j ユニットの入力を i_j^k 、出力を o_j^k 、第 $k-1$ 層の第 i ユニットの出力から第 k 層の第 j ユニットのシナプス荷重を $w_{ij}^{k-1,k}$ 、各ユニットの入出力関数を f とおくと、それらの変数の関係は、

$$o_j^k = f(i_j^k) \quad (6)$$

$$i_j^k = \sum_i w_{ij}^{k-1,k} o_i^{k-1} \quad (7)$$

のように表わされる⁷⁾。各ユニットの入出力関数 $f(x)$ は正規型シグモイド関数

Fig. 3 Multilayer neural network

$$f(x) = \frac{1 - \exp(-x)}{1 + \exp(-x)} \quad (8)$$

とする。このとき、第 m 層すなわち出力層の第 j ユニットの出力に対する教師信号を t_j としたバックプロパゲーションアルゴリズムによる教師付き学習⁶⁾における l 回目のシナプス荷重の変化分 $\Delta w_{ij}^{k-1,k}(l)$ は、 α, β を正値の学習定数として、

$$\Delta w_{ij}^{k-1,k}(l) = -\alpha d_j^k o_i^{k-1} + \beta \Delta w_{ij}^{k-1,k}(l-1) \quad (9)$$

ただし、

$$d_j^k = \begin{cases} \frac{1}{2} \{1 - f(i_j^k)^2\} (o_j^m - t_j), & k = m \\ \frac{1}{2} \{1 - f(i_j^k)^2\} \sum_i w_{ji}^{k+1} d_i^{k+1}, & k = 1, \dots, m-1 \end{cases} \quad (10)$$

と表わされ、 l 回更新された後のシナプス荷重 $w_{ij}^{k-1,k}$ は(9)式の l 回目の変化分 $\Delta w_{ij}^{k-1,k}(l)$ を用いて

$$w_{ij}^{k-1,k}(l+1) = w_{ij}^{k-1,k}(l) + \Delta w_{ij}^{k-1,k}(l) \quad (11)$$

と表わすことができる⁷⁾。

この多層型ニューラルネットワークによってマニピュレータの運動量変化逆モデルを表わすために、入力層に二サンプル時刻でのマニピュレータの位置と速度を、出力層にトルク信号を対応させる。各ユニットの入出力関数が正規型シグモイド関数であるので、多層型ニューラルネットワーク自体から出力される $[-1, 1]$ の信号を K_M 倍に増幅してマニピュレータへのトルク信号とする。

このとき、次の手順で学習を行う。

- (i) まず、マニピュレータの動作試行データをとる。あるサンプリング時刻 kT において、マニピュレータの位置、速度信号が $q_k, \dot{q}_k$ で、マニピュレータへ τ_k のトルクを T 時間入力したとき、次のサンプリング時刻 $(k+1)T$ で $q_{k+1}, \dot{q}_{k+1}$ になったとする。このときの動作試行データの組 $\{q_k, \dot{q}_k, q_{k+1}, \dot{q}_{k+1}, \tau_k\}$ を獲得する。
- (ii) 多層型ニューラルネットワークの入力層に二時刻におけるマニピュレータの位置と速度 $q_k, \dot{q}_k, q_{k+1}, \dot{q}_{k+1}$ を入力し、出力層の教師信号を多層型ニューラルネットワークの増幅係数 K_M でスケーリングしたトルク τ_k/K_M として、多層型ニューラルネットワークの荷重を(11)式に従って変化させる。
- (iii) 上の二つの操作を多数のサンプル点に対して行う。

各サンプル間において、マニピュレータは前節(5)式の運動量変化関係式に従って動作しているの、上のような学習を行うことにより、多層型ニューラルネットワー

クは次の関数 $\mathcal{F}$ を近似的に実現することになる.

$$\frac{\tau_k}{K_M} = \mathcal{F}(q_k, \dot{q}_k, q_{k+1}, \dot{q}_{k+1}) \quad (12)$$

ただし,

$$\begin{aligned} \mathcal{F}(q_k, \dot{q}_k, q_{k+1}, \dot{q}_{k+1}) \\ = \frac{1}{T} [M(q) \dot{q} + V(q, \dot{q})]_{kT}^{(k+1)T} \end{aligned} \quad (13)$$

である.

3.2 運動量変化逆モデルを用いたマニピュレータ軌道制御則

つぎに, 学習により獲得した運動量変化逆モデルを用いたマニピュレータ軌道制御則について説明する.

マニピュレータの目標軌道は, 各サンプリング時刻における位置と速度として与えられるものとする. 時刻 kT の目標位置, 目標速度をそれぞれ $q_k^d, \dot{q}_k^d$ で表わす. いま, 時刻 kT でマニピュレータの入力を

$$\begin{aligned} \tau_k = \frac{1}{T} [M(q_{k+1}) \dot{q}_{k+1}^d - M(q_k) \dot{q}_k \\ + V(q_{k+1}, \dot{q}_{k+1}) - V(q_k, \dot{q}_k)] \end{aligned} \quad (14)$$

とすると, 時刻 $(k+1)T$ におけるマニピュレータの速度 $\dot{q}_{k+1}$ を目標速度 $\dot{q}_{k+1}^d$ と一致させることができる. これは, (14) 式を (5) 式に代入することによりわかる. 時刻 kT においてマニピュレータが目標位置にある場合, すなわち目標軌道から外れていない場合, (14) 式の制御入力によって常に目標とする軌道制御が達成できる.

ところが, (14) 式には未来時刻 $(k+1)T$ における位置と速度信号 $q_{k+1}, \dot{q}_{k+1}$ が現れているので, このままでは制御器として実現することができない. そこで, マニピュレータが目標軌道から大きく外れることなく動作しているということを仮定して, これら未来時刻 $(k+1)T$ 信号の代わりに時刻 $(k+1)T$ における目標位置と目標速度信号を用いて, 制御入力を

$$\begin{aligned} \tau_k = \frac{1}{T} [M(q_{k+1}^d) \dot{q}_{k+1}^d - M(q_k) \dot{q}_k \\ + V(q_{k+1}^d, \dot{q}_{k+1}^d) - V(q_k, \dot{q}_k)] \end{aligned} \quad (15)$$

とする. しかし, 何らかの原因でマニピュレータが目標軌道から大きく外れてしまった場合には, この制御則だけでは目標とする軌道制御が達成できないので, 現在時刻 kT におけるマニピュレータの位置および速度も制御して, 目標軌道に復帰する機能を持たせるための PD 型フィードバック制御器を (15) 式に加えた入力

$$\begin{aligned} \tau_k = \frac{1}{T} [M(q_{k+1}^d) \dot{q}_{k+1}^d - M(q_k) \dot{q}_k \\ + V(q_{k+1}^d, \dot{q}_{k+1}^d) - V(q_k, \dot{q}_k)] \\ + K_P(q_k^d - q_k) + K_D(\dot{q}_k^d - \dot{q}_k) \end{aligned} \quad (16)$$

を実際の制御入力とした.

ここで, 制御則 (16) 式の右辺第 1 項は, (12) 式において $q_{k+1} = q_{k+1}^d, \dot{q}_{k+1} = \dot{q}_{k+1}^d$ と代入したものと等しい. よって, この制御則は,

$$\begin{aligned} \tau_k = K_M \mathcal{F}(q_k, \dot{q}_k, q_{k+1}^d, \dot{q}_{k+1}^d) \\ + K_P(q_k^d - q_k) + K_D(\dot{q}_k^d - \dot{q}_k) \end{aligned} \quad (17)$$

のように学習によって獲得した関数 $\mathcal{F}$ を用いて表わすことができる. この制御則を用いたときの制御系全体の構造を Fig. 4 に示す.

Fig. 4 Control system structure using momentum change inverse model

4. シミュレーション結果

上記のアルゴリズムの有効性を, Fig. 5 のように y 軸下向きに重力が作用する 2 リンクマニピュレータ軌道制御を例題としたシミュレーションによって検証する. マニピュレータの物理定数は Table 1 の値を用いた. 各関節には位置センサと速度センサが取り付けられていて, 角度信号と角速度信号が測定できるものとする.

Fig. 5 Two-link manipulator

ここでの制御目的は, マニピュレータが一つの制御則によって複数の目標軌道をできるだけ誤差少なく描くことである. 目標軌道として, Fig. 6 に示した 9 種類の楕円軌道を使用した. これらの目標軌道を x, y 軸の位置, 速度目標値として書くと,

Table 1 Parameters of two-link manipulator

Length	link 1	L_1 [m]	0.30
	link 2	L_2 [m]	0.30
Mass	link 1	M_1 [kg]	0.665
	link 2	M_2 [kg]	0.350
	motor drive link 2	m_2 [kg]	1.750

Fig. 6 Desired trajectories

$$\begin{aligned} x_k^{di} &= R_x^i \cos(\omega k T) + R, & y_k^{di} &= R_y^i \sin(\omega k T) + R \\ \dot{x}_k^{di} &= -R_x^i \omega \sin(\omega k T), & \dot{y}_k^{di} &= R_y^i \omega \cos(\omega k T) \end{aligned}$$

となり、これらを逆運動学変換して角度、角速度目標値として用いた。ただし、 k はサンプリング番号、 R_x^i, R_y^i は i 番目の目標軌道の x, y 軸半径、 ω は角速度、 T はサンプリング周期、 (R, R) は目標軌道の中心座標であり、それぞれTable 2の値を用いた。

まず、マニピュレータの運動量変化逆モデルの学習を4層のニューラルネットワークを用いて行った。各層におけるユニット数が入力層から出力層に向かって8-20-20-2であるネットワークを使用した。学習のための試行データは、上記9種類の目標軌道に対してPD型フィードバック制御器で粗く制御したときのマニピュレータの入出力データを用いて、その中から任意のサンプル時刻 k での入力トルク τ_k と位置 q_k と速度 $\dot{q}_k$ 、および次の時刻 $k+1$ での位置 q_{k+1} と速度 $\dot{q}_{k+1}$ を取り出して一組の教師パターンとした。各軌道には1周につき500サンプル点あるので合計で4,500組の教師パターンがあり、これを一つの学習データセットと

Table 2 Parameters of desired trajectories, learning and control

Desired trajectories	$R_x^1, R_x^2, R_x^3, R_x^4, R_x^5, R_x^6, R_x^7, R_x^8, R_x^9$ [m]	R
	$R_y^1, R_y^2, R_y^3, R_y^4, R_y^5, R_y^6, R_y^7, R_y^8, R_y^9$ [m]	$0.7R$
Learning	R [m]	0.2
	ω [rad/s]	$2\pi/500T$
Control	α	0.2
	β	0.8
Control	T [s]	1.0×10^{-3}
	K_P	500
	K_D	10
	K_M	5

した。どの目標軌道に対しても偏りなく大域的な逆モデルを獲得するために、1学習データセット内において教師パターンをランダムに選択しながら学習を行った。ここで、PD型フィードバック制御器のゲイン値および学習定数は、Table 2の値を用いた。Fig. 7は運動量変化逆モデル学習を500学習データセットまで繰り返して行ったときの学習誤差の推移である。学習誤差は、各学習データセットに対する学習二乗誤差を未学習時のニューラルネットワークに対する学習二乗誤差で割った相対値である。学習二乗誤差は、各教師パターンに対するニューラルネットワークの出力信号と教師信号との差の二乗の1学習データセット内全教師パターンに対する総和をとったものである。ただし、シナプス荷重は $[-0.001, 0.001]$ の微小な範囲のランダム値で初期化し、未学習時の学習二乗誤差は、 $0.157902 \text{ [N}^2\text{m}^2]$ であった。運動量変化逆モデル学習は、上記の学習データに対してFig. 7のように400学習データセット程度の繰り返して達成できた。

このとき、目標軌道の1~4番目のパターンに対して、マニピュレータはFig. 8(a)~(d)のような軌道を描いた。一方、PD型フィードバック制御器だけで制御した結果をFig. 9(a)~(d)に示した。運動量変化逆モデルを学習した多層型ニューラルネットワークによるフィードフォワード制御器+PD型フィードバック制御

Fig. 7 Learning error transition of momentum change inverse modeling

Fig. 8 Control result with PD controller and multilayer neural network controller learned momentum change inverse model of manipulator

Fig. 9 Control result with PD controller

器による結果より、どの目標軌道に対してもPD型フィードバック制御器だけの場合に比べて誤差の少ない軌道制御が達成できることを確認した。

また、直接逆モデリング法と本方法を比較するために、直接逆モデリング法を適用して上記と同じことを行った結果について次に示す。まず、マニピュレータの(直接)逆モデルの学習を上と同じく4層のニューラルネットワークを用いて行った。各層におけるユニット数は6-20-20-2である。学習のための試行データとして、9種類の目標軌道に対してPD型フィードバック制御器で粗く制御したときのマニピュレータの入出力データを用いたことは上と同じであるが、ここではマニピュレータの関節に加速度センサが取り付けられていないことを条件としているので、その中から取り出してくる教師パターンとしては、任意のサンプル時刻 k での入力トルク τ_k と位置 q_k と速度 $\dot{q}_k$ 、さらに加速度信号の代わりに速度差分信号 $(\dot{q}_{k+1} - \dot{q}_k)/T$ を用いて一組とした。学習定数として上と同じ値 $\alpha = 0.2, \beta = 0.8$ を用いて、500 学習データセットまで繰り返して学習を行った結果、学習誤差の推移は Fig. 10 (a) のようになった。本来直接逆モデリング法ではマニピュレータの加速度信号を用いるべきであるが、ここでは加速度が測定できないという条件でそれを速度差分信号で代用して学習を行ったため、逆モデル学習には誤差が残存した。学習定数を $\alpha = 0.05, \beta = 0.2$ として学習速度を緩和した場合においても、Fig. 10 (b) に示すように学習の定常誤差は残存した。このときの学習が不十分なニューラルネットワークを用いて制御した場合、Fig. 11 (a)~(d) のように、PD型フィードバック制御器だけの場合よりも良いが制御誤差の残る結果となった。

つぎに、サンプリング周期を $T = 1.0 \times 10^{-4}, 1.0 \times 10^{-3}, 1.0 \times 10^{-2}$ と変えて上記と同様の学習を行ったときの本方法と、直接逆モデリング法に対する学習二乗誤差の定常値を Table 3 に示す。これ

Fig. 10 Learning error transition of direct inverse modeling without acceleration

Fig. 11 Control result with PD controller and multilayer neural network controller learned insufficiently direct inverse model

より、同一のサンプリング周期に対して本方法のほうがより精度の高い逆モデルを得ることができること、本方法を用いることでサンプリング周期にかかわらず逆モデル学習が可能となることがわかる。このことから、マニピュレータの位置信号と速度だけしか測定できない場合には、運動量変化逆モデル学習は有効であると考えられる。

5. おわりに

本稿では、マニピュレータの運動量変化関係式という力学的性質を利用することによって、逆モデル学習時に加速度信号が不要となる運動量変化逆モデル学習法を提案し、例題として2リンクマニピュレータ軌道制御に適用してその有効性を検証した。

本稿で提案した運動量変化逆モデル学習では、逆モデルの入力次数が直接逆モデルよりも大きくなり、より多くの学習時間が必要となる問題点が例題では確認された。この点に関しては、今後より実的な適用により検討していくべき課題である。また、本稿ではモデル学習はオフラインで行う方式をとったが、マニピュレータのモデルが動作時に大きく変動する場合には、学習と制御を別々に行う本方法では高精度の制御ができなくなることが考えられる。これに対して、学習と制御をオンラインで同時に行う方式の学習的軌道制御法として知られている Betterment Process⁹⁾ や繰り返し制御法⁹⁾ では、動作時の周期的モデル変動に対しても精度の高い制御が実現可能であり有効である。しかし、これらの方法では学習結果が学習時の目標軌道に大きく依存するため、目標軌道を変更した場合には初めから学習を再開しなければならず、この点ではマニピュレータの大域的な逆モデルを得る本方式が有利であると考えられる。したがって、動作時にマニピュレータモデルが大きく変動する場合に対処するための方策としては、本方法によりオフラインで得た大域的モデルで制御したものを初期入力列として、オン

Table 3 Square of learning error

$T[s]$	Momentum change inverse modeling Square of learning error [$N^2 m^2$]	Direct inverse modeling Square of learning error [$N^2 m^2$]
1.0×10^{-4}	4.1054520×10^{-4}	8.4739656×10^{-3}
1.0×10^{-3}	5.4760414×10^{-4}	1.4689558×10^{-2}
1.0×10^{-2}	1.2680912×10^{-3}	2.0616122×10^{-2}

ラインでは繰り返し型の制御方式を併用することが考えられ、この点について検討することが今後の課題である。

参 考 文 献

- 1) J. S. Albus : A New Approach to Manipulator Control : The Cerebellar Model Articulation Controller (CMAC) ; Trans. of the ASME J. of Dyn. Sys. Meas. and Contr., Vol. 97, No. 3, pp. 220~227 (1975)
- 2) W. T. Miller, F. H. Glanz and L. G. Kraft : Application of a General Learning Algorithm to the Control of Robotic Manipulators ; International Journal of Robotics Research, Vol. 6, No. 2, pp. 84~98 (1987)
- 3) 川人, 宇野, 鈴木 : 随意運動制御における適応と学習 II ; 日本ロボット学会誌, Vol. 6, No. 3, pp. 222~230 (1988)
- 4) M. I. Jordan : Supervised Learning and Systems with Excess Degrees of Freedom ; COINS Technical Report, Vol. 88, No. 27, pp. 1~41 (1988)
- 5) (社) 日本工業技術振興協会ニューロコンピュータ研究部会 : ニューロコンピューティングの基礎理論, 海文堂 (1990)
- 6) D. E. Rumelhart, G. E. Hinton and R. J. Williams : Learning Internal Representations by Error Propagation, Parallel Distributed Processing, MIT Press, pp. 318~362 (1986)
- 7) 麻生 : ニューラルネットワーク情報処理, 産業図書 (1988)
- 8) 川村, 宮崎, 有本 : 動的システムの学習的制御法 (Betterment Process) の提案 ; 計測自動制御学会論文集, Vol. 22, No. 1, pp. 56~62 (1986)
- 9) 原 : 繰り返し制御 ; 計測と制御, Vol. 25, No. 12, pp. 1111~1119 (1986)