A Neural Network Based Learning Controller for Robot Manipulators *

Chiang-Ju Chien[1] *and Li-Chen Fu*[2]

[1]Department of Electronic Engineering
Huafan University, 223, Shihtin, Taipei Hsien, Taiwan,
E-mail : cjc@huafan.hfu.edu.tw

[2]Department of Computer Science & Information Engineering
National Taiwan University, Taipei, Taiwan
E-mail : lichen@ccms.ntu.edu.tw

Abstract

In this paper, an iterative learning control using neural network design is presented for robot manipulators with input disturbance and re-initialization uncertainty. A sampled-data feedforward learning algorithm is designed under a feedback configuration and a rigorous proof via a discrete approach is given to study the learning performance. It is shown that under a sufficient condition on the learning gain, convergence and robustness of tracking error in the iteration domain can be guaranteed at each sampling instant if sampling period is small enough. Since the implementation of learning gain depends on the information of input-output coupling matrix of robot manipulator, a neural network is proposed to solve the implementation problem. A training procedure is applied to estimate the robot manipulator by using only input-output data. The neurons, equivalent to the premise and consequent parameters of a fuzzy system, are tuned by gradient descent and least squares estimate. This will give an initial setting of the neural-network based iterative learning controller. During the control iterations, the neural network can still be tuned for each iteration in order to improve the approximation accuracy and increase the tracking speed.

1 Introduction

Many adaptive approaches [1],[2],[3],[4] have been studied as means to control the robot systems with parametric uncertainties for which the traditional PID-type controller are not adequate. However, most of these works are developed under some restrictive assumptions. In considering the parametric or nonparametric uncertainties of robot manipulators, an iterative learning control (ILC) technique has become an interesting topic in the robotic literature recently. The basic strategy of iterative learning control is to repeat the same task over a finite time interval so that the control performance can be improved as iterations are large enough. This approach is an advantage when controlling systems can not be modeled accurately. Since the repeatability of operation is one of the main technical features for present industrial robots, it would be suitable to apply the iterative learning control technique to overcome the control problem of parametric or nonparametric uncertainties of robot manipulators.

Among the publications in the area of iterative learning control for robot manipulator, the basic structure of the learning controller for generating the current torque $\tau_{i+1}(t)$ at $(i+1)$th trial takes the following form of

$$\tau_{i+1}(t) = F_i(\tau_i(t), e_i(t)) \quad 0 \le t \le T \tag{1.1}$$

where $\tau_i(t)$ and $e_i(t)$ are the control torque and the tracking error at ith trial respectively and $F_i(\cdot)$ is the updated learning function. In the early works, the so-called D-type ILCs using acceleration error of joint variables to construct the learning controller were proposed for the control of robot manipulators [5],[6],[7],[8],[9]. Only a simple assumption of Lipschitz condition on the plant's nonlinearity is needed for the convergence analysis of iterative learning system. However, the acceleration measurement or estimation becomes a main disadvantage for this type of ILC algorithm. The so called P-type ILC using velocity feedback is then developed for the consideration of practical realization [10],[11],[12],[13]. Although the requirement of acceleration measurement is removed, more restrictive assumptions on the robot manipulator [10],[11],[12] or more complicated control structures [13] are now needed for the design and analysis of the learning systems. For example, besides the basic assumptions discussed in D-type ILC, the passivity properties in [10] or the boundedness of the derivative of input-output coupling matrix in [11],[12] are required to ensure the convergence of the learning systems.

*This work is supported by the National Science Council, R.O.C., under Grant NSC89-2213-E-211-012

For real implementation of iterative learning controller, it is necessary to store the sampled-data of desired output, plant output and control input in memory. Therefore, it is more practical to design and analyze the ILC system in discrete-time domain. However, it is noted that all the iterative learning controller discussed above are implemented and analyzed in the continuous-time domain. In order to design an iterative learning controller without acceleration measurement or estimation and implement the algorithm in a discrete-time domain, a sampled-data control strategy is proposed in this paper to solve the control objective. We first transform the sampled-data iterative learning control problem into a discrete-time version and then give a rigorous proof via a discrete approach to study the convergence and robustness. The feedforward discrete learning algorithm is designed under a feedback controller such that the closed-loop output tracking error in the absence of feedforward learning is within a reasonable bound. The feedforward iterative learning controller is then updated to meet the performance requirement. It is shown that under a sufficient condition on the feedforward learning gain, the convergence and robustness of the learning system can be guaranteed if the sampling period is small enough. Unlike the learning controller designed under a feedback configuration [7],[13], the proposed learning system can work without any high gain feedback controller and convergence condition is independent of the design of feedback controller. However, the learning rate can be improved if a stabilizing one is used.

However, the analysis shows that implementation of learning gain will depend on integration of robot's input-output coupling matrix between two successive sampling instants. Using the property of universal approximation to nonlinear functions, a neural network is proposed for solving the problem of learning gain implementation. According to the affine structure of robot dynamics, a neural network structure similar to a Takagi and Sugeno's type fuzzy system [14] is presented to provide the information of input-output coupling matrix even only input and output control data is available. The weights of neurons, equivalent to the premise and consequent parameters of a Takagi and Sugeno's fuzzy system, are tuned by gradient descent and least squares estimate [15]. A training procedure is first applied to estimate the robot dynamics by using only input and output training data. After training, the neural network can provide the integration of coupling matrix for an initial setting of learning gain. Based on this neural network, the iterative learning controller will show a better learning performance when compared with the traditional constant learning gain approaches. A numerical simulation is then given to demonstrate the effect of proposed sampled-data ILC using fuzzy network design.

2 Problem Formulation

Consider a non-redundant robot manipulator with the equation of motion given in the joint-space formulation as follows :

$$M(\theta(t))\ddot{\theta}(t)+V(\theta(t),\dot{\theta}(t))\dot{\theta}(t)+G(\theta(t),\dot{\theta}(t))+\tau_w(t)=\tau(t) \tag{2.1}$$

where $\theta(t) \in R^{m\times m}$ is the vector of generalized joint angular positions, $M(\theta(t)) \in R^{m\times m}$ is the symmetric inertia matrix, $V(\theta(t),\dot{\theta}(t)) \in R^{m\times 1}$ is the Coriolis plus centrifugal force vector, $G(\theta(t),\dot{\theta}(t)) \in R^{m\times 1}$ is the gravitational plus frictional forces, $\tau_w(t) \in R^{m\times 1}$ is the input disturbance vector, and $\tau(t) \in R^{m\times 1}$ is the generalized torque acting on the joint shafts. It is well known that the inertia matrix $M(\theta(t))$ can be shown to be positive definite and bounded as

$$0 < m_1 I \le M(\theta(t)) \le m_2 I \tag{2.2}$$

for all $\theta(t) \in R^{m\times 1}$ with some $m_1, m_2 > 0$ where I is the $m \times m$ identity matrix. Since the inverse of inertia matrix exists for all values of joint variables, the dynamic formulation of the robot manipulator can be rewritten as follows :

$$\ddot{\theta}(t) = f(\theta(t),\dot{\theta}(t)) - M^{-1}(\theta(t))\tau_w(t) + M^{-1}(\theta(t))\tau(t) \tag{2.3}$$

where $f(\theta(t),\dot{\theta}(t)) = -M^{-1}(\theta(t))[V(\theta(t),\ \dot{\theta}(t))\ \dot{\theta}(t) + G(\theta(t),\dot{\theta}(t))]$. Let the state variable at the ith iteration to be $x_i(t) = [\theta_i(t), \dot{\theta}_i(t)]^\top$ and choose the output variable as $y_i(t) = \dot{\theta}_i(t)$, then we have

$$\begin{aligned} \dot{x}_i(t) &= F(x_i(t)) + B(x_i(t))\tau_i(t) + w_i(t) \\ y_i(t) &= Cx_i(t) \end{aligned} \tag{2.4}$$

where $F(x_i(t)) = [\dot{\theta}_i(t), f(\theta_i(t),\ \dot{\theta}_i(t))]^\top$, $B(x_i(t)) = [0,\ M^{-1}(\theta_i(t))]^\top$, $w_i(t) = [0,\ -M^{-1}(\theta_i(t))\tau_w(t)]^\top$ and $C = [0, I]$. Here, 0 denotes the zero matrix or vector for some suitable dimension. It is noted that $\|C\| = 1$, $\|B(x_i(t))\| \le \frac{1}{m_1} \triangleq \bar{b}$ and $CB(x_i(t)) = M^{-1}(\theta_i(t))$.

Given a finite initial state $x_i(0)$ and a finite time interval $[0, T]$, the control objective is to design a sampled-data iterative learning controller $\tau_i(t)$ in $[0, T]$ such that the output tracking error at each sampling instant is within a given error bound ϵ^*, i.e., $\|e_i(n\Delta)\| = \|y_d(n\Delta) - y_i(n\Delta)\| \le \epsilon^*, \forall n \in \{0, 1, \cdots, N\}$ as the iteration $i \to \infty$. In order to achieve this control objective, we shall assume the following properties for the robot manipulator (2.1):

(A1) For any realizable output trajectory $y_d(t)$ and an appropriate initial condition x_0, there is a unique control torque $\tau_d(t)$ generating the trajectory for the nominal plant. In other words, the following differential equation is satisfied when disturbance $w_i(t) = 0$,

$$\begin{aligned} \dot{x}_d(t) &= F(x_d(t)) + B(x_d(t))\tau_d(t) \\ y_d(t) &= Cx_d(t) \end{aligned} \tag{2.5}$$

where $\tau_d(t)$ is uniformly bounded for all $t \in [0, T]$ with the bound $\bar{\tau}_d = \sup_{t\in[0,T]} \|\tau_d(t)\|$.

(A2) $F(x_i(t))$ and $B(x_i(t))$ are uniformly globally Lipschitz in $x_i(t)$ on the interval $[0, T]$ (i.e., $\|h(x_1(t)) - h(x_2(t))\| \le \ell_h \|x_1(t) - x_2(t)\|, \forall t \in [0, T]$ and for some positive constant $\ell_h < \infty, h \in \{F, B\}$)

(A3) The initial state error $\|x_d(0) - x_i(0)\| \le \epsilon_1,\ \forall\ i$ and the disturbance $\|w_i(t)\| \le \epsilon_2\ \forall\ i$ and $\forall\ t \in [0, T]$ for some positive constants ϵ_1, ϵ_2.

3 Design and Analysis of the Sampled-Data Iterative Learning Controller

In order to achieve the control objective and implement the learning algorithm in a discrete time domain for practical realization, we propose a sampled-data iterative learning strategy which combines a velocity feedback controller and a feedforward iterative learning controller as follows. Let Δ be the sampling period of the sampler and $n \in \{0, 1, 2, \cdots, N-1\}$ with $N\Delta = T$.

- *the velocity feedback controller* :

$$\tau_{i+1}^b(n\Delta) = L_b(n\Delta)e_{i+1}(n\Delta) \tag{3.1}$$

  Here the $L_b(n\Delta)$ is a bounded feedback gain which is chosen to stabilize the system and with the bound $\bar{\ell}_b = \sup_{i\in[0,\infty]} \sup_{n\in[0,N]} L_b(n\Delta)$.

- *the feedforward iterative learning controller* :

$$\tau_{i+1}^f(n\Delta) = \tau_i(n\Delta) + L_f(n\Delta)e_i((n+1)\Delta) \tag{3.2}$$

  Here the $L_f(n\Delta)$ is a bounded learning gain to be designed for convergence and with the bound $\bar{\ell}_f = \sup_{i\in[0,\infty]} \sup_{n\in[0,N]} L_f(n\Delta)$.

- *the control torque at $i+1$th iteration, $i = 0, 1, 2, \cdots$* :

$$\tau_{i+1}(t) = \tau_{i+1}^b(n\Delta) + \tau_{i+1}^f(n\Delta),\ t \in [n\Delta, (n+1)\Delta) \tag{3.3}$$

Here, $i+1$th and ith iterations denote the current and previous iterations, respectively. For the initial iteration, only feedback controller is available. The feedforward iterative learning controller will work after first iteration. In the proposed sampled-data iterative learning algorithm, the error history is sampled at time $n\Delta$, $n \in \{0, 1, 2 \cdots, N-1\}$ and stored in the memory. Only inputs at the sampling instants are updated in the next iteration. The learning system will be analyzed at each sampling instant via a discrete approach. In this learning system, a feedforward discrete learning algorithm is designed under a feedback controller and is updated by past control data in the previous trial. A stabilizing controller is designed such that the closed-loop output tracking error in the absence of feedforward learning is within a reasonable bound. The feedforward iterative learning controller is then updated to meet the performance requirement. It will be shown that under a learning condition on the feedforward learning gain, the convergence of the learning system in the iteration domain can be guaranteed and that condition will be independent of the feedback controller. This means that the feedforward learning can still work in the uncertain environment without the feedback controller. However, it will be shown by simulations that the learning rate can be greatly improved if a suitable stabilizing controller is used. The main result is now summarized in the following Theorem :

Theorem 1 : Consider the dynamic equation of robot manipulator (2.1) satisfying assumptions (A1) – (A3) and give a realizable output trajectory y_d on the time interval $[0, T]$ with τ_d satisfying assumption (A1). If the sampled-data iterative learning controller (3.1) – (3.3) is used with the sampling period Δ small enough and the learning gain $L_f(n\Delta)$ satisfying

$$\sup_{i\in[0,\infty]} \sup_{x_i\in R^n} \left\| I - L_f(n\Delta) \int_{n\Delta}^{(n+1)\Delta} CB(x_i(z))dz \right\| \leq \rho_f < 1 \tag{3.4}$$

then we can guarantee the boundedness of tracking errors in each iterate and

$$\lim_{i\to\infty} \mathbf{E}_i \leq \sigma$$

where $\mathbf{E}_i = [\|e_i(0)\|, \|e_i(\Delta)\|, \cdots, \|e_i(N\Delta)\|]^\top$ and $\sigma = [\epsilon_1, \sigma_1, \cdots, \sigma_N]^\top$ with $\sigma_j, j = 1, \cdots, N$ depending on the constants ϵ_1, ϵ_2 and

$$d^* = \sup_{n\in\{0,1,\cdots,N-1\}} \sup_{n\Delta\leq z<(n+1)\Delta} \|\tau_d(z) - \tau_d(n\Delta)\|$$

.

4 Implementation of the Learning Gain Using a Neural Network

As presented in theorem 1, if the sampled-data iterative learning controller is designed with condition (3.4) being satisfied, then convergence and robustness of the closed-loop learning system can be achieved with guaranteed performances. In this section, we will study how to implement the learning gain such that (3.4) can be achieved. Actually one can easily find that the best choice of L_f is to set

$$L_f(n\Delta) = \left(\int_{n\Delta}^{(n+1)\Delta} CB(x_i(z))dz \right)^{-1}$$

In order to simplify the discussion, iteration index i and disturbance $w_i(t)$ will be omitted in this section unless otherwise specified. Let $y(t) = [y_1(t), \cdots, y_m(t)]^\top, \tau(t) = [\tau_1(t), \cdots, \tau_m(t)]^\top$, we can derive the following equation from (2.4) that

$$\begin{aligned} \dot{y}(t) &= CF(x(t)) + CB(x(t))\tau(t) && (4.5)\\ &= CF(y_1(t), y_1^{(-1)}(t), \cdots, y_m(t), y_m^{(-1)}(t)) \\ &\quad + CB(y_1(t),, y_1^{(-1)}(t), \cdots, y_m(t), y_m^{(-1)}(t))\tau(t) && (4.6) \end{aligned}$$

since it can be easily found that there exists a diffeomorphism from $[y_1, y_1^{(-1)}, \cdots, y_m, y_m^{(-1)}]$ to the state x. If we use the following sampled-data transformations to approximate the operation of differentiation and integration :

$$\begin{aligned} \dot{y}(t) &= \frac{1}{\Delta}\left(y(t+\Delta) - y(t)\right) \\ \int_{t-\Delta}^{t} y(z)dz &= \frac{\Delta}{2}\left(y(t) + y(t-\Delta t)\right) \end{aligned}$$

then (4.6) can be expressed as the following sampled-data system if sampling period Δ is small enough :

$$\begin{aligned} & y(t+\Delta) \\ &= \bar{F}(y_1(t), y_1(t-\Delta), \cdots, y_m(t), y_m(t-\Delta)) \\ &\quad + \bar{B}(y_1(t), y_1(t-\Delta), \cdots, y_m(t), y_m(t-\Delta))\tau(t) \end{aligned} \tag{4.7}$$

where $\bar{F} \in R^{m\times 1}$ and $\bar{J} \in R^{m\times m}$ can be explicitly expressed as

$$\begin{aligned} \bar{F} &= y(t) + \Delta C F\Big(y_1(t), \frac{\Delta}{2}(y_1(t)+y_1(t-\Delta)), \cdots, \\ &\quad y_m(t), \frac{\Delta}{2}(y_m(t)+y_m(t-\Delta)\Big) \\ \bar{B} &= \Delta C B\Big(y_1(t), \frac{\Delta}{2}(y_1(t)+y_1(t-\Delta)), \cdots, \\ &\quad y_m(t), \frac{\Delta}{2}(y_m(t)+y_m(t-\Delta)\Big) \end{aligned}$$

If we let $t = n\Delta$, the sampled-data representation of (4.6) can now take the following form from (4.7) as

$$\begin{aligned} & y((n+1)\Delta) \\ &= \bar{F}(y_1(n\Delta), y_1((n-1)\Delta), \cdots, \\ &\quad y_m(n\Delta), y_m((n-1)\Delta)) \\ &\quad + \bar{B}(y_1(n\Delta), y_1((n-1)\Delta), \cdots, \\ &\quad y_m(n\Delta), y_m((n-1)\Delta))\tau(n\Delta) \end{aligned} \tag{4.8}$$

On the other hand, we can directly integrate (4.5) and then replace t by $(n+1)\Delta$ as follows :

$$\begin{aligned} y((n+1)\Delta) &= y(n\Delta) + \int_{n\Delta}^{(n+1)\Delta} CF(x(z))dz \\ &\quad + \int_{n\Delta}^{(n+1)\Delta} CB(x(z))dz\tau(n\Delta) \end{aligned} \tag{4.9}$$

The choice of learning gain $L_f(n\Delta)$ can be achieved by comparing (4.8) and (4.9) as follows :

$$\begin{aligned} L_f(n\Delta) &= \left(\int_{n\Delta}^{(n+1)\Delta} CB(x(z))dz\right)^{-1} \\ &= \bar{B}^{-1}(y_1(n\Delta), y_1((n-1)\Delta), \cdots, \\ &\quad y_m(n\Delta), y_m((n-1)\Delta)) \end{aligned} \tag{4.10}$$

Obviously, if $\bar{B}$ is achieved, then the learning gain can be implemented. To implement the learning gain, we approximate the equivalent sampled-data system (4.8) by a neural network. Let $\hat{y}_k((n+1)\Delta)$ be the kth output of the neural network which approximates output $y_k((n+1)\Delta), k = 1, 2, \cdots, m$. The neural network for $\hat{y}_k((n+1)\Delta)$ is constructed as follows:

$$\begin{aligned} & \hat{y}_k((n+1)\Delta) \\ &= \sum_{\ell=1}^{\mathcal{L}} \bar{w}_k^\ell \cdot \Big(a_{k,1}^\ell \tau_1(n\Delta) + \cdots + a_{k,m}^\ell \tau_m(n\Delta) \\ &\quad + a_{k,m+1}^\ell y_1(n\Delta) + \cdots + a_{k,2m}^\ell y_m(n\Delta) \\ &\quad + a_{k,2m+1}^\ell y_1((n-1)\Delta) + \cdots \\ &\quad + a_{k,3m}^\ell y_m((n-1)\Delta) + \beta_k^\ell\Big) \end{aligned} \tag{4.11}$$

If we define $A_k^\ell = [a_{k,m+1}^\ell, \cdots, a_{k,2m}^\ell a_{k,2m+1}^\ell, \cdots, a_{k,3m}^\ell]^\top$ and $Y = [y_1(n\Delta), \cdots, y_m(n\Delta), y_1((n-1)\Delta), \cdots, y_m((n-1)\Delta)]^\top \equiv [Y_1, \cdots, Y_{2m}]^\top$, then (4.11) can be rewritten as

$$\begin{aligned} & \hat{y}_k((n+1)\Delta) \\ &= \sum_{\ell=1}^{\mathcal{L}} \bar{w}_k^\ell \cdot \left(A_k^{\ell\top} Y + \beta_k^\ell\right) \\ &\quad + \sum_{\ell=1}^{\mathcal{L}} [\bar{w}_k^\ell a_{k,1}^\ell, \cdots, \bar{w}_k^\ell a_{k,m}^\ell] \begin{bmatrix} \tau_1(n\Delta) \\ \tau_2(n\Delta) \\ \vdots \\ \tau_m(n\Delta) \end{bmatrix} \end{aligned} \tag{4.12}$$

Here $\bar{w}_k^\ell$ is the basis function of the neural network which can be represented as

$$\bar{w}_k^\ell = \prod_{j=1}^{2m} \mu_{k,j}^\ell(Y_j) \Big/ \sum_{\ell=1}^{\mathcal{L}} \prod_{j=1}^{2m} \mu_{k,j}^\ell(Y_j) \tag{4.13}$$

with

$$\mu_{k,j}^\ell(Y_j) = exp\left(-\frac{1}{2}\left(\frac{Y_j - c_{k,j}^\ell}{\sigma_{k,j}^\ell}\right)^2\right) \tag{4.14}$$

In this neural network structure, parameters $c_{k,j}^\ell, \sigma_{k,j}^\ell$ and $a_{k,j}^\ell, \beta_k^\ell$ are equivalent to premise parameters and consequent parameters of a Takagi and Sugeno's type fuzzy system respectively. So the network can be defined as a neural-fuzzy network since a fuzzy rule base can be included if necessary. The m-th input m-th output neural-fuzzy network $\hat{y}((n+1)\Delta) = \hat{F}(Y) + \hat{B}(Y)\tau(n\Delta)$ can now be explicitly represented as :

$$\begin{aligned} & \begin{bmatrix} \hat{y}_1((n+1)\Delta) \\ \hat{y}_2((n+1)\Delta) \\ \vdots \\ \hat{y}_m((n+1)\Delta) \end{bmatrix} \\ &= \begin{bmatrix} \sum_{\ell=1}^{\mathcal{L}} \bar{w}_1^\ell \cdot (A_1^{\ell\top} Y + \beta_1^\ell) \\ \sum_{\ell=1}^{\mathcal{L}} \bar{w}_2^\ell \cdot (A_2^{\ell\top} Y + \beta_2^\ell) \\ \vdots \\ \sum_{\ell=1}^{\mathcal{L}} \bar{w}_m^\ell \cdot (A_m^{\ell\top} Y + \beta_m^\ell) \end{bmatrix} \\ &\quad + \begin{bmatrix} \sum_{\ell=1}^{\mathcal{L}} [\bar{w}_1^\ell a_{1,1}^\ell, \cdots, \bar{w}_1^\ell a_{1,m}^\ell] \\ \sum_{\ell=1}^{\mathcal{L}} [\bar{w}_2^\ell a_{2,1}^\ell, \cdots, \bar{w}_2^\ell a_{2,m}^\ell] \\ \vdots \\ \sum_{\ell=1}^{\mathcal{L}} [\bar{w}_m^\ell a_{m,1}^\ell, \cdots, \bar{w}_m^\ell a_{m,m}^\ell] \end{bmatrix} \begin{bmatrix} \tau_1(n\Delta) \\ \tau_2(n\Delta) \\ \vdots \\ \tau_m(n\Delta) \end{bmatrix} \end{aligned}$$

Although only input-output data is available for learning of the neural network, the coupling matrix can now be obtained by setting $\bar{B}(Y) = \hat{B}(Y)$ if a suitable learning is completed. For the learning of the neural network parameters, we use the hybrid learning rule with batch learning similar to [15]. That is, the weights of premise neurons

and consequent neurons are tuned by gradient descent and least squares estimate, respectively. Some training trajectories are applied to the robot manipulator to get the input-output sampled training data. An off-line training procedure is then used to estimate the robot dynamic as well as the coupling matrix. This will give an initial setting of the neural network based iterative learning controller. During the control interval, the neural network can also be tuned for each iteration in order to improve the approximation accuracy. If the input-output coupling matrix can be well approximated, the learning gain $L_f(n\Delta)$ will satisfy condition (3.4) with ρ_f tending to zero and a rapid tracking speed can then be expected.

5 Numerical Examples

In order to show the effectiveness of the proposed sampled-data iterative learning controller, a continuous path tracking of a two link SCARA-type robot manipulator is used for computer simulation. The dynamic equation of the robot manipulator is given as follows [12] :

$$\begin{bmatrix} \frac{1}{3}m_1\ell^2 + \frac{4}{3}m_2\ell^2 + m_2c_2\ell^2 & \frac{1}{3}m_2\ell^2 + \frac{1}{2}m_2c_2\ell^2 \\ \frac{1}{3}m_2\ell^2 + \frac{1}{2}m_2c_2\ell^2 & \frac{1}{3}m_2\ell^2 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix}$$

$$+ \begin{bmatrix} -\frac{1}{2}m_2s_2\ell^2\dot{\theta}_2^2 - m_2s_2\ell^2\dot{\theta}_1\dot{\theta}_2 \\ \frac{1}{2}m_2s_2\ell^2\dot{\theta}_1^2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

where c_i, s_i denote $\cos\theta_i, \sin\theta_i$, respectively. In this dynamic equation, θ_1, θ_2 are joint variables, τ_1, τ_2 are input torque. The link masses and length in this simulation are set to be $m_1 = m_2 = 2$kg, $\ell = 0.5$m. We choose the state variables $x = [x_1, x_2, x_3, x_4]^\top$ as $x_1 = \theta_1, x_2 = \dot{\theta}_1, x_3 = \theta_2, x_4 = \dot{\theta}_2$ and output variables $y = [y_1, y_2]^\top$ as $y_1 = \dot{\theta}_1, y_2 = \dot{\theta}_2$, so that

$$\begin{aligned} \dot{x}(t) &= F(x(t)) + B(x(t))\tau(t) \\ y(t) &= Cx(t) \end{aligned}$$

Let the desired trajectories for $t \in [0, 0.5]$ be given as

$$\begin{aligned} \theta_{d1}(t) &= -\frac{5}{3}t^3 + \frac{5}{2}t^2 \text{ (radian)} \\ \dot{\theta}_{d1}(t) &= -5t^2 + 5t \text{ (radian/sec)} \\ \theta_{d2}(t) &= -\frac{5}{3}t^3 + \frac{5}{2}t^2 \text{ (radian)} \\ \dot{\theta}_{d2}(t) &= -5t^2 + 5t \text{ (radian/sec)} \end{aligned}$$

The feedback gain is chosen as a positive definite matrix $5I$. The sampling period is set to be 0.001. For the computer simulation, we let the initial states $x_2(0) = x_4(0) = 0.01, x_1(0) = x_3(0) = 0$ so that there are initial state errors and the tolerance bound $\epsilon^* = 0.01$. After a suitable learning of the neural network, the neural network based ILC shows the learning tracking performances for joint velocities in the following figures.

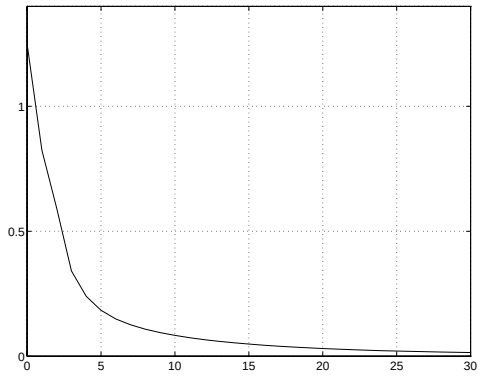

Figure 1 : Supremum velocity tracking error of joint 1 $\sup_{t\in[0,0.5]} |\dot{\theta}_{d1}(t) - \dot{\theta}_1(t)|$ versus iteration i

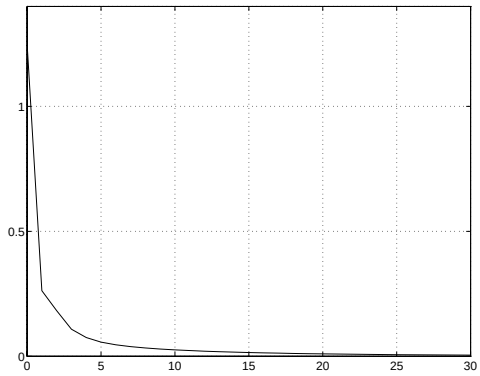

Figure 2 : Supremum velocity tracking error of joint 2 $\sup_{t\in[0,0.5]} |\dot{\theta}_{d2}(t) - \dot{\theta}_2(t)|$ versus iteration i

6 Conclusions

In this paper, an iterative learning control using neural network design is presented for robot manipulators with input disturbance and re-initialization uncertainty. It is shown that under a sufficient condition on the learning gain, convergence and robustness of tracking error in the iteration domain can be guaranteed at each sampling instant if sampling period is small enough. Since the implementation of learning gain depends on the information of input-output coupling matrix of robot manipulator, a neural network is proposed to solve the implementation problem. A training procedure is applied to estimate the robot manipulator by using only input-output data. The neurons, equivalent to the premise and consequent parameters of a fuzzy system, are tuned by gradient descent and least squares estimate.

References

[1] J.J. Craig, P. Hsu and S.S. Sastry, "Adaptive control of mechanical manipulators." Proceedings if the IEEE Conference on Robotics and Automation, San Francisco, California, U.S.A., pp. 190–195, 1986.

[2] R.H. Middleton and G.C. Goodwin, "Adaptive computed torque control for rigid link manipulations," *Systems & Control Letters*, vol. 10, pp. 9–16, 1988.

[3] J-J.E., Slotine and W. Lee, "Composite adaptive control of robot manipulators," *Automatica*, vol. 25, no. 4, pp. 509–519, 1989.

[4] M.W. Spong and R. Ortega, "On adaptive inverse dynamics control of rigid robots," *IEEE Trans. Automat. Contr.*, vol. 35, no. 1, pp. 92–95, 1990.

[5] S. Arimoto, S. Kawamura and F. Miyazaki, "Bettering operation of robots by learning," *J. of Robot. Syst.*, vol. 1, no. 2, pp. 123–140, 1984.

[6] J.J. Craig, "Adaptive control of robot manipulators through repeated trials," *Proceedings of the American Control Conference*, San Diego, California, U.S.A., pp. 1566–1573, 1984.

[7] P. Bondi, G. Casaline and L. Gambardella, "On the iterative learning control theory for robotic manipulators," *IEEE Trans. on Robotics and Automation*, vol. 4, no. 1, pp. 14-22, 1988.

[8] Z. Bien, D.H. Hwang and S.R. Oh, "A nonlinear iterative learning method for robot path control," *Robotica*, vol. 9, pp. 387–392, 1991.

[9] Z. Qu, J. Dorsey, D.M. Dawson and R.W. Johnson, "Linear learning control of robot motion," *Journal of Robotic Systems*, vol. 10, no. 1, pp. 123–140, 1993.

[10] S. Kawamura, F. Miyazaki and S. Arimoto, "Realization of robot motion based on a learning method," *IEEE Trans. Systems, Man and Cybernetics*, vol. 18, no. 1, pp. 126–134, 1988.

[11] S.S. Saab, "On the P-type learning control," *IEEE Trans. Automat. Contr.*, vol. 39, no. 11, pp. 2298–2302, 1994.

[12] C.J. Chien and J.S. Liu, "A P-type iterative learning controller for robust output tracking of nonlinear time-varying systems," *International Journal of Control*, Vol. 64, No. 2, pp. 319–334, 1996.

[13] B.H. Park, T.Y. Kuc and J.S. Lee, "Adaptive learning of uncertain robotic systems," *Int. J. Contr.*, vol. 65, no. 5, pp. 725–744, 1996.

[14] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control", *IEEE Trans. Systems, Man and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[15] J.S. Jang, "ANFIS : Adaptive-network-based fuzzy inference system", *IEEE Trans. Systems, Man and Cybernetics*, vol. 23, no. 3, pp. 665–684, 1993.

The following block diagram shows the control structure of the neural-network based learning system (n denotes the abbreviation of $n\Delta$).