Inversion-based Adaptive Learning Control for a One-Link Flexible Manipulator

Xuezhen Wang and Degang Chen

Iowa State University
Department of Electrical and Computer Engineering
Ames, IA, 50011, U. S. A.
E-mail:xzwang@iastate. edu, djchen@iastate.edu

Abstract— **In this paper, an adaptive learning algorithm is applied to one-link flexible manipulator. After each repetitive trial, Least-Squares method is used to estimate the system parameters. The output tracking error and the identified system model are used through stable inversion to find the feed forward input, together with the desired state trajectories, for the next trial. An adaptive backstepping based tracking controller is used in each trial to ensure the regulation of the desired state trajectories. Simulation results demonstrate that the proposed learning control scheme is very effective in tip trajectory tracking for a flexible link robotic manipulator.**

Key Words: Stable Inversion, One-Link Flexible Manipulator

I. Introduction

Many practical robot systems operate the same task repeatedly. Due to this feature, it produces the repetitiveerr ors. Iterativelear ning control (ILC) is a feed forward control approach aimed at achieving high performance output tracking control by "learning" from past experience so as to eliminate the repetitive errors from future execution [3]. The concept of iterative learning for generating the optimal input to a system was first introduced by Uchiyama [9]. Arimoto *et al.*[4] then developed the idea and first proposed a learning control method for linear time-varying, continuous-time systems. Moore [3] modified the Arimoto method and extended it to systems with relative degree larger than one. Furuta and Yamakita [12] presented a modification of Moore's method. Their algorithm provided convergence in the sense of the L_2 norm but required the complete knowledge of the adjoints ystem, which is equivalent to needing the complete knowledge of the system dynamics.

Hauser presented a nonlinear version of Arimoto's method for a class of nonlinear systems [14] and provided sufficient conditions for its uniform convergence. Hauser's method is more general than Arimoto's method. For a more specific structure, Sugie and Ono [8] provided the learning controller given by a linear time-varying system and showed its convergence under some conditions. Kuc *et al.* [6] presented an ILC scheme for a class of nonlinear dynamic systems. Saab [7] presented sufficient conditions for the convergence of P-type learning algorithm for a class of time-varying, nonlinear systems.

Although existing learning algorithms have been theoretically proven to provide output error convergence with successful applications, many such algorithms have practical difficulties with nonminimum phase systems. Amann and Owens [10] showed that a zero of the plant in the RHP caused very slow convergence of the input sequence and resulted in a nonzero error for some iterative control algorithms . To remove the minimum phase requirement, Gao and Chen [13] developed a new adaptive learning algorithm for stable linear systems based on "stable inversion". Wang and Chen [16] presented an adaptive learning control algorithm for unstable nonminimum phase systems.

Flexible manipulators have many advantages over rigid links: They are lighter in weight, consume less power, and respond faster [15]. Due to the flexible nature of the system, the dynamics are highly nonlinear and complex. Many flexible manipulator systems are nonminimum phase. In this paper, an adaptive learning control algorithm[16] is applied to a one-link flexible robot manipulator system.

The remainder of the paper is organized as follows. Section II presents the adaptive learning control law. Section III describes the system dynamics of a one-link flexible manipulator and the adaptive learning controller is applied to design a tip trajectory tracking control of the manupulator. Finally, some conclusions are given in Section IV.

II. Adaptive Learning Algorithm

Consider a nonlinear time varying plant model in the k^{th} trial:

$$y_k(t) = \Phi(x_k(t), \theta, u_k(t)) \tag{1}$$

where, for all $t \in [0, T]$, $x_k(t) \in \Re^n, u_k(t) \in \Re^m, y_k(t) \in \Re^p$. And θ is a parameter vector.

In addition, we make the following assumptions:

(A1) The system has a well-defined relative degree $r = (r_1, \cdots, r_m)^T$ that is known. The linearization of the system about an equilibrium point, which is assumed to be the origin WLOG, is completely controllable.

(A2) The order of the system, n, is known.

(A3) The system parameter vector θ is unknown or known incompletely.

(A4) A desired output trajectory is given and is a sufficiently smooth function of t satisfying $y_d(t) = 0$ for

any $t \in (-\infty, 0] \cup [T, \infty)$ and finite for any $t \in (0, T)$, where $T > 0$.

(A5) The system can be represented in terms of control input $u_k(\cdot)$ and output $y_k(\cdot)$ in the k^{th} trial by means of a nonlinear time-varying operator Φ as follows:

$$y_k(\cdot) = \Phi\{u_k(\cdot)\} \tag{2}$$

And the operator $\Phi\{\cdot\}$ is uniformly globally Lipschitz in u_k on the interval $[0, T]$. That is, $\|\Phi u_k - \Phi u_{k+1}\| \leq L\|u_k(t) - u_{k+1}(t)\|$, $\forall t \in [0, T]$ with a Lipschitz constant $0 \leq L < \infty$.

Iterative Learning Control Problems :
Given a desired output trajectory $y_d(t)$ and a tolerance error bound ϵ for a class of system (1) and (2), starting from an arbitrary continuous initial control input $u_0^d(\cdot)$ and initial state $x_0^d(\cdot)$, iterative learning control will try to find a sequence of desired state trajectories $x_k^d(\cdot)$ and desired control inputs $u_k^d(\cdot)$, which when applied to the system, produces an output sequence $y_k(\cdot)$ such that

(1) $\|y_d(\cdot) - y_k(\cdot)\|_\infty \leq \epsilon$, as $k \to \infty$, where k is the trial number and $\|f\|_\infty = \sup_{t\in[0,T]}\|f(t)\|$.

(2) $\|u_k^d(t)\| \leq \epsilon, \|x_k^d(t)\| \leq \epsilon, \forall t \in (-\infty, 0] \cup [T, +\infty)$.

(3) $u_k^d(t)$, $x_k^d(t)$, $u_k(t)$, and $x_k(t)$ are uniformly bounded.

In this dynamic process, the functions have two arguments: continuous time t and the trial number k. In the sequel, it is assumed that the variation of the operator over two consecutive trials are slow and can be neglected. Then the operator obtained by the identification performed in the k^{th} trial can be used to determine the input for the $(k+1)^{th}$ trial. This general description of the problem allows a simultaneous description of linear or nonlinear dynamics, continuous or discrete plant, and time-invariant or time-varying systems.

When applying a linear ILC, however, the plant must fulfill the following conditions: (1) The desired trajectory $y_d(t)$ is identical for every trial and satisfies Assumption (A4). (2) Each trial has the fixed period T. (3) The system parameters are fixed or very slowly time-varying.

At any trial k, define a tracking error to be $e_k = y_d - y_k$. Learning control convergence means that $\|e_k\| \to 0$ as $k \to \infty$. The λ-norm defined in Arimoto *et al.* [4] has been adopted in many papers [3] as the topological measure in the proof of the convergence property for a newly proposed ILC. The formal definition [4] of the λ-norm for a function $f : [0, T] \to \Re^n$ is given by

$$\|f(\cdot)\|_\lambda \triangleq \sup_{t\in[0,T]} e^{-\lambda t}\|f(t)\| \tag{3}$$

It is easily observed that $\|f\|_\lambda \leq \|f\|_\infty \leq e^{\lambda T}\|f\|_\lambda$ for $\lambda > 0$, where $\|f\|_\infty \triangleq \sup\|f(t)\|_\infty$, implying the $\lambda-$norm is equivalent to the sup norm.

The block diagram of the adaptive learning system is shown in Figure 1 [16].

The proposed adaptive learning control strategy has three components: a parameter estimator, a stable inverse system, and an adaptive backstepping feedback controller. The parameter estimator is in charge of "learning" the parameterized model of the system. During each trial, the input and output trajectories are recorded. Then off-line Least-Squares method is applied to obtain the optimal estimate of parameters. Also obtained during each trial is the output tracking error signal. This error signal and the estimated model are used by the stable inverse system to learn the optimal input signal for the next trial. Although the estimated model may be nonminimum phase which normally leads to unbounded inverse solutions, stable inversion guarantees a unique and bounded inverse solution. This "learning" action is done "off-line" between two consecutive trials. Afterwards, the new feed forward input is used by the adaptive backstepping feedback controller to stabilize the system and to ensure regulation of the tracking error. The controller is designed following a recursive backstepping procedure and it takes advantage of the parametric strict-feedback structure of the system. The controller parameters are also continuously updated in real-time using an adaptive control law. The same feedback control algorithm is used during every trial. In the following, we give the implementation of the adaptive learning algorithm for continuous-time systems.

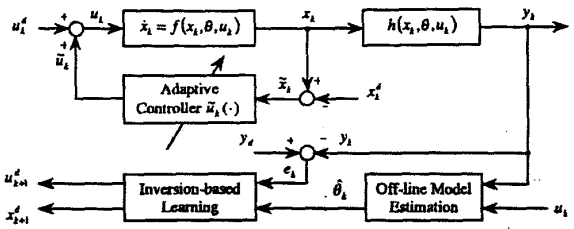

Fig. 1. Block Diagram of Adaptive Learning Control System

A. *Solution to Stable Inversion of nonminimum Phase Systems*

Consider a LTI system in the form:

$$\sum\nolimits_1 : \begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

If $\hat{G}$ is nonminimum phase, this will lead to unbounded solutions. The stable inversion theory [5] provides an avenue to overcome this difficulty. The procedure to obtain a unique stable inverse solution $u^d = \hat{H} y_d$ is illustrated below. There are four steps:

(1) Find the time-domain state-space model of $\hat{G}$:

Since $\hat{G}u^d = \hat{G}\hat{H}y_d = y_d$, a state-space representation of $\hat{G}$ yields

$$\dot{x}^d(t) = \hat{A}x^d(t) + \hat{B}u^d(t) \tag{4}$$

$$y_d(t) = \hat{C}x^d(t) + \hat{D}u^d(t) \tag{5}$$

where x^d is the state and $\hat{A}, \hat{B}, \hat{C}$, and $\hat{D}$are matrices with suitable sizes.

(2) Find its inverse in state space:

Differentiate $y_d(t)$ until u^d appears explicitly in the right hand side. Solvefor u^d and substitute into (4)

and (5) to obtain

$$\dot{x}^d(t) = \bar{A}x^d(t) + \bar{B}y_d^{(r)}(t) \quad (6)$$
$$u^d(t) = \bar{C}x^d(t) + \bar{D}y_d^{(r)}(t) \quad (7)$$

where $\bar{A}, \bar{B}, \bar{C}$, and $\bar{D}$ are defined according to the substitution.

(3) Decompose the inverse system into center, stable, and unstable subsystems:

Perform a change of variables so that

$$x^d = Pz = P[z^c, z^s, z^u]^T \quad (8)$$

which leads to

$$\dot{z}^c = A^c z^c + B^c y_d^{(r)} \quad (9)$$
$$\dot{z}^s = A^s z^s + B^s y_d^{(r)} \quad (10)$$
$$\dot{z}^u = A^u z^u + B^u y_d^{(r)} \quad (11)$$
$$u^d = [C^c \ C^s \ C^u][z^c \ z^s \ z^u]^T + \bar{D}y_d^{(r)} \quad (12)$$

where A^c, A^s, and A^u are real Jordan matrices of suitable dimensions; A^c has r eigenvalues at zero; A^s has all eigenvalues in the open left-half plane; A^u has all eigenvalues in the open right-half plane.

(4) Obtain the stable inverse system:

Pick the transformation matrix P so that the center subsystem is a simple chain of r integrators. Solve that and impose two boundary conditions on the stable and unstable subsystems to yield

$$z^c = [y_d, \dot{y}_d, \cdots, y_d^{(r-1)}]^T \quad (13)$$
$$\dot{z}^s = A^s z^s + B^s y_d^{(r)}, t \geq 0; z^s(t) = 0, \ \forall\, t \leq 0 \quad (14)$$
$$\dot{z}^u = A^u z^u + B^u y_d^{(r)}, t \leq T; z^u(t) = 0, \ \forall\, t \geq T \quad (15)$$

These together with (8) and (12) define the desired stable inverse system.

The stable inverse system always yields bounded solutions for bounded and smooth y_d. This can be clearly seen from (13, 14, 15) since the center solution z^c is clearly bounded, the stable subsystem is in the forward time, and the unstable subsystem is in the reverse time, all leading to bounded solutions.

In this paper, we only consider LTI systems in this form:

$$\sum_2 : \begin{cases} \dot{x}_{i-1,k} = x_{i,k}; \ i = 2, 3, \cdots, n \\ \dot{x}_{n,k} = x_k^T a + u_k \\ y_k = Cx_k \end{cases}$$

Based on the stable solutions outline presented above, to facilitate iterative learning, we modify the inversion process slightly as follows. Referring to Figure 1, let G denote the system operator from u_k to y_k. Let u_{k+1} be the new input for the next trial. Then the new tracking error signal will be

$$\begin{aligned} e_{k+1} &= y_d - Gu_{k+1} \\ &= y_d - G(u_{k+1}^d + \tilde{u}_{k+1}) \\ &= e_k - G(u_{k+1}^d - u_k^d) - G(\tilde{u}_{k+1} - \tilde{u}_k) \end{aligned}$$

Then we get

$$e_{k+1} + G(\tilde{u}_{k+1} - \tilde{u}_k) = e_k - Gu_{k+1}^e$$

where $u_{k+1}^e = u_{k+1}^d - u_k^d$.

The goal of designing learning control law is to make $e_{k+1}(\cdot)$ gradually decrease as k increases. For any remaining errors, the feedback control action will try to reduce them. Therefore, wes imply set $e_{k+1} = 0$ and $\tilde{u}_{k+1} - \tilde{u}_k = 0$ to design u_{k+1}^e. That is, we want

$$e_k = Gu_{k+1}^e \quad (16)$$

Since G is unknown and $\hat{G}_k$ is the best estimate model after the k^{th} trial, we will use

$$u_{k+1}^e = \hat{H}_k e_k \quad (17)$$
$$x_{k+1}^e = P_k z = P_k[z^c, z^s, z^u]^T \quad (18)$$

so that $e_k = \hat{G}_k u_{k+1}^e$. Then our learning algorithm would be

$$u_{k+1}^d = u_k^d + u_{k+1}^e \quad (19)$$
$$x_{k+1}^d = x_k^d + x_{k+1}^e \quad (20)$$

As a further modification, one may introduce a forgetting factor α $(0 < \alpha < 1)$ and use:

$$u_{k+1}^d = u_k^d + \alpha u_{k+1}^e \quad (21)$$
$$x_{k+1}^d = x_k^d + \alpha x_{k+1}^e \quad (22)$$

where u_{k+1}^e and x_{k+1}^e are given by (17) and (18), which are the stable inverse solutions from e_k and $\hat{G}_k$. (Note that: in the rest of the paper, u^d, x^d, and u^e represent u_{k+1}^d, x_{k+1}^d, and u_{k+1}^e respectively for notational convenience).

However, the controller design in the next subsection will still assume $\alpha = 1$ so that u^d, x^d, and y_d satisfy the dynamics of $\hat{G}_k$, that is :

$$\sum_3 : \begin{cases} \dot{x}_{i-1}^d = x_i^d; \ i = 2, \cdots, n \\ \dot{x}_n^d = \hat{a}_k^T x^d + u^d \\ y_d = \hat{C}_k x^d \end{cases}$$

B. Adaptive Backstepping Controller Design

There are a lot of methods to design a controller. Since the parameters of the systems are unknown, we need to design an adaptive controller. Here we follow a popular approach of adaptive backstepping design [2].

Define $\tilde{x}_k = x_k - x^d$ and $\tilde{u}_k = u_k - u^d$. For clarity, we will drop the subscript k in sections 3.2 and 3.3 if it does not cause confusion. One can easily verify:

$$\sum_4 : \begin{cases} \dot{\tilde{x}}_{i-1} = \tilde{x}_i; \ i = 2, \cdots, n \\ \dot{\tilde{x}}_n = \tilde{x}^T a + [a - \hat{a}]^T x^d + \tilde{u} \end{cases}$$

The goal is to design $\tilde{u}$ to guarantee the regulation of $\tilde{x}$. Since a is unknown, let $\vartheta(t)$ be the on-line estimate of a and rewrite the last equation as follows:

$$\begin{aligned} \dot{\tilde{x}}_n &= \tilde{x}^T\vartheta + [\vartheta - \hat{a}]^T x^d + \tilde{u} + [a - \vartheta]^T(\tilde{x} + x^d) \\ &= \psi^T(\tilde{x})\tilde{a} + \tilde{u}_2 \end{aligned}$$

where $\tilde{a} = a - \vartheta$, $\psi^T(\tilde{x}) = x^T$. Then $\sum_4$ is in a standard form with matching condition, then the goal becomes to design $\tilde{u}_2$ to guarantee the regulation of $\tilde{x}$. Details of the derivations are skipped here, but the final controller is given by:

$$u = u^d + \tilde{u}_2 - \tilde{x}^T\vartheta - [\vartheta - \hat{a}]^T x^d \quad (23)$$

$$\tilde{u}_2 = \alpha_n(\tilde{x}, \vartheta) \quad (24)$$

$$\dot{\vartheta} = \Gamma\psi z_n \quad (25)$$

where Γ is an adaptation gain matrix. The variables z_i and the stabilizing functions $\alpha_i, i = 1, \ldots, n$, are defined by the following recursive expressions:

$$z_i = \tilde{x}_i - \alpha_{i-1}(\tilde{x}_1, \ldots, \tilde{x}_{i-1}) \quad (26)$$

$$\alpha_i = c_i z_i - z_{i-1} + \sum_{j=1}^{i-1} \frac{\partial \alpha_{i-1}}{\partial \tilde{x}_j}\tilde{x}_{j+1}, \quad i = 1, \cdots, n-1 \quad (27)$$

$$\alpha_n = c_n z_n - z_{n-1} - \psi^T\vartheta + \sum_{j=1}^{n-1} \frac{\partial \alpha_{n-1}}{\partial \tilde{x}_j}\tilde{x}_{j+1} \quad (28)$$

This adaptive controller guarantees global boundedness of $\tilde{x}(t), \vartheta(t)$, and regulation of $\tilde{x}_i(t), i = 1, \ldots, n$, i.e., $\tilde{x}_i(t) \to 0$, as $t \to \infty$.

C. Parameter Estimator

Off-line Least-Squares method is used to estimate the parameters. To get $\hat{a}_{k+1}$, we use the method as follows:

$$\hat{a}_{k+1} = d_k\hat{a}_k + (1 - d_k)\bar{a}_k \quad (29)$$

where $d_k \in [0, 1)$ is a memory factor and $\bar{a}_k$ is determined by using off-line Least-Squares method [1] using data from the k^{th} trial. First using filter for the last equation of $\sum_3$, we get

$$\frac{1}{s+\lambda}\dot{x}_n = \frac{1}{s+\lambda}x^T a + \frac{1}{s+\lambda}u \quad (30)$$

$$x_n - \frac{1}{s+\lambda}u = \frac{1}{s+\lambda}x^T \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} + \lambda \end{pmatrix} \quad (31)$$

where $\lambda > 0$. Then weha ve $Z = W^T\hat{a}_\lambda$, where $Z = (x_n - \frac{1}{s+\lambda}u)$, and $W^T = \frac{1}{s+\lambda}x^T$. By solving the ordinary differential equations (ODE), we getaZd W. Now collect all the data of Z and W. Suppose there are totally M samples for the k^{th} trial. Let $\Phi = [W_1^T, \cdots, W_M^T]^T$ and the regressor vector be $\Psi = [Z_1, \cdots, Z_M]^T$. The Lease-Squares solution is

$$\hat{a}_\lambda = (\Phi^T\Phi)^{-1}\Phi^T\Psi \quad (32)$$

From this we get $\bar{a}_k$. Then $\hat{a}_{k+1}$ is obtained by (31). Similarly, for the linear model,

$$y = Cx \quad (33)$$

We can use the same argument to get the estimates of C, except that no filtering is needed.

D. Adaptive Learning Algorithm

The process of the algorithm is as follows:

Step 0 : Given ϵ, the initial conditions $\hat{a}_0$, $\hat{C}_0$, the initial input $u_0^d(t) = 0$, and initial state trajectory $x_0^d(t) = 0$ on $t \in [0, T]$. Set k=0.

Step 1: Let $e_k(t) = y_d(t) - y_k(t)$. Get $\hat{G}_k$ from $\hat{a}_k$ and $\hat{C}_k$. Use stable inversion to get $u_{k+1}^e = \hat{H}_k e_k$ and x_{k+1}^e. Use equation (21) and (22) to get u_{k+1}^d and x_{k+1}^d.

Step 2: $u_{k+1}^d(t)$ is used as feed forward by the adaptive backstepping feedbackcon troller to stabilize the system and to ensure regulation of $x_{k+1}^d(t)$, i.e., $x_k(t) \to x_{k+1}^d(t)$, as $t \to \infty$. The input and output trajectories are recorded respectively.

Step 3 : Then the tracking error signal $e_{k+1}(t)$ is calculated. If $\|e_{k+1}(\cdot)\|_\infty \leq \epsilon$, stops. Otherwise, set $k = k + 1$, and go to Step 4 .

Step 4: Use off-line Least-Squares method to obtain the parameter estimates $\hat{a}_k$ and $\hat{C}_k$, and go back to Step 1.

III. A One-Link Flexible Manipulator

The proposed adaptive learning control algorithm is applied to a one-link flexible manipulator.

A. Dynamics Model

A nonlinear one-link flexible manipulator model is obtained from [11]. A simple modeling technique divides the flexible link into rigid segments that are connected by elastic springs, where link deformation is concentrated. The following treatment will be limited to the case of two equal segments of uniform mass, moving along the horizontal plane. Let m and l denote the total link mass and length, k the spring elasticity, and u the input torque. With reference to Figure 2, θ_1 is the angular position of the link base, while θ_2 is the flexible variable. The dynamic equations are

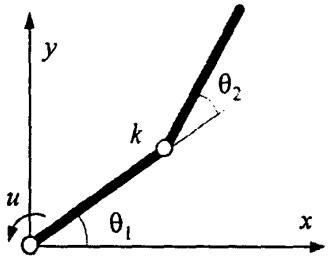

Fig. 2.A Simple one-link flexible manipulator

$$\begin{bmatrix} b_{11}(\theta_2) & b_{12}(\theta_2) \\ b_{12}(\theta_2) & b_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} c_1(\theta_2, \dot{\theta}_1, \dot{\theta}_1) \\ c_2(\theta_1, \dot{\theta}_1) + k\theta_2 + d_2\dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} u \quad (34)$$

with the elements of the inertia matrix $B(\theta_2)$ given by

$$b_{11}(\theta_2) = a + 2c\cos(\theta_2)$$
$$b_{12}(\theta_2) = b + c\cos(\theta_2)$$
$$b_{22} = b$$

and Coriolis and centrifugal terms

$$c_1(\theta_2, \dot{\theta}_1, \dot{\theta}_2) = -c(\dot{\theta}_2^2 + 2\dot{\theta}_1\dot{\theta}_2)\sin\theta_2$$
$$c_2(\theta_2, \dot{\theta}_1) = c\dot{\theta}_1^2\sin\theta_2$$

where

$$a = 5m\ell^2/24, \quad b = m\ell^2/24, \quad c = m\ell^2/16$$

In (34), d_1 and d_2 are damping coefficients representing viscous friction at the joint and link structural (passive) dissipation, respectively. The linearized expression of the end-effector angular position, as seen from the base,

$$y = \theta_1 + \frac{1}{2}\theta_2 \tag{35}$$

will be taken as controlled output for the system.

The parameters for the one-link flexible manipulator were chosen the same as in De Luca [11] $l = 1$ m, $m = 0.2$ kg, $k = 5$ Nm/rad, and $d_1 = d_2 = 0.01$ $Nm{\cdot}sec/rad$.

The state-space of the linearized model is described by:

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

where $x = (\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) \in \Re^4$,

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -4114 & -2744 & -6.171 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$C = [41140 \;\; 82.29 \;\; -17.14 \;\; 0]$$

B. Simulation Results

Let the desired output trajectory be defined as follows:

$$y_d = \begin{cases} \pi^2\left(\frac{1}{2\pi t} - \frac{1}{(2\pi)^2\sin(2\pi t)}\right), & 0 \le t \le t_f \\ \frac{\pi}{2}, & t > t_f \end{cases}$$

as shown by the solid curve in Figure 3.

For the given trajectory, the following data were used: $y_0 = 0^o, y_f = 90^o$. The initial conditions are $\theta_1 = \theta_2 = \dot{\theta}_1 = \dot{\theta}_2 = 0$.

The parameters $a = [0 \; -4114 \; -2744 \; -6.171]^T$, $C = [41140 \;\; 82.29 \;\; -17.14]$. This system has two poles at $-2.3339 \pm 52.2641i$, one pole at -1.5031, and one pole at 0. Two zero are at 51.4515 and -46.6504 respectively. Take the initial conditions $\hat{a}_0 = [0.5 \; -4110 \; -2740 \; -6.1]^T$, $\hat{C}_0 = [41138 \;\; 81.5 \;\; -16.8]$. Take an initial input

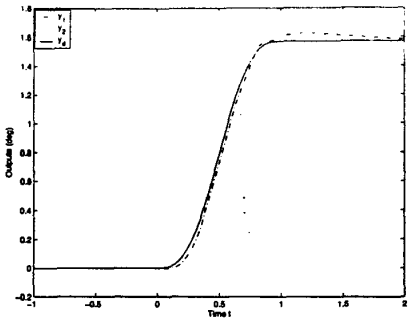

Fig. 3. Tip trajectory tracking for one-link flexible robot manipulator

$u_0^d(t) = 0$ and initial state trajectory $x_0^d(t) = 0$. Following the learning controller design strategy described in Section II, the simulation results for the trial $k = 1$ and $k = 2$ are shown in Figure 3.

At the 2^{nd} trial, the output $y_2(t)$ converges to the desired $y_d(t)$ exactly by the dotted curve. Table 1 shows the parameter estimates. We can see the estimated parameters are very close to the true values at the 4^{rd} trial.

The above results demonstrate that the proposed learning control is very effective in reproducing the desired trajectories for one-link flexible manipulator. Notice that preloading is required in this case.

IV. Conclusion

An adaptive learning algorithm was proposed for one-link flexible robot manipulator. The adaptive backstepping feedback control law is employed to guarantee regulation of tracking error and a stable inverse system is used to update the feed-forward input for the next trial. Given a desired trajectory, the learning controller is able to learn and eventually drive the closed loop dynamics to track the desired trajectory. Simulation results demonstrate the effectiveness of the proposed method.

References

[1]R. Johansson, *System Modelling Identification*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[2]M. Krstić, I. Kanellakopoulos, and P. V. Kokotovic, *Nonlinear and Adaptive Control Design*, Wiley, New York, 1995.

[3]K. L. Moore, *Iterative learning control for deterministic systems*, Springer-Verlag, New York, 1993.

[4]S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *Journal of Robotic Systems*, Vol. 1, pp. 123-140, 1984.

[5]D. Chen and B. Paden, "Stable inversion of nonminimum phase nonlinear systems," *International Journal of Control*, Vol. 64, pp. 81-96, 1996.

[6]J. E. Kuc and M. B. Zaremba, "An iterative learning control theory for a class of nonlinear dynamics systems," *Automatica*, Vol. 28, pp. 1215-1221, 1992.

[7]S. Saab, "On the P-type learning control," *IEEE Transactions on Automatic Control*, Vol. 39, pp. 2298-2302, 1994.

[8]T. Sugie and T. Ono, "An iterative learning control law for dynamical systems," *Automatica*, Vol. 27, pp. 729-732, 1991.

[9]M. Uchiyama, "Formation of high speed motion pattern of mechanical arm by trial," *Transactions of the Society of Instrumentation and Control Engineers*, Vol. 19, pp. 706-712, 1978.

[10] N. Amann, D. H. Owens, and E. Rogers, "Non-minimum phase plant in iterative learning control," *Proceedings of the 2th Conference on Intelligent System Engineering*, pp. 107-112, 1994.

[11] A. De Luca, L. Lanari, and G. Ulivi, "Nonlinear regulation of end-effector motion for a flexible robot arm," *New Trends in Systems Theory*, Genova, Italy, July 9-11, 1990.

[12] K. Furuta and M. Yamakita, "The design of a learning control system for multivariable systems," *Proceedings of IEEE International Symposium on Intelligent Control*, pp. 371-376, 1987.

[13] J. Gao and D. J. Chen, "Iterative learning control for nonminimum phase systems," *American Control Conference*, 1998.

[14] J. Hauser, "Learning control for a class nonlinear systems," *Proceedings of the 26th Conference on Decision and Control*, pp. 859-860, 1987.

[15] P. Chaichanavong and D. Banjerdpongchai, "A case study of robust control experiment on one-link flexible robot arm," *American Control Conference*, pp. 4319-4324, 1999.

[16] X. Z. Wang and D. J. Chen, "Adaptive learning control for nonminimum phase systems," *2000 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 26-31, Vol. 1, 2000.

TABLE I

PARAMETER ESTIMATION OF ONE-LINK FLEXIBLE MANIPULATOR SYSTEMS.

k	$\hat{a}_{0,k}(0)$	$\hat{a}_{1,k}(-4114)$	$\hat{a}_{2,k}(-2744)$	$\hat{a}_{2,k}(-6.171)$	$\hat{C}_{0,k}(-41140)$	$\hat{C}_{1,k}(82.29)$	$\hat{C}_{2,k}(-17.14)$
1	0.101	-4110.4	-2733.6	-5.567	41135.67	80.34	-15.6
2	0.058	-4112.7	-2738.44	-5.89	41138.65	81.88	-16.87
3	0.012	-4113.9	-2740.32	-6.05	41138.75	82.11	-17.05
4	0.005	-4114.2	-2742.4	-6.128	41141.8	82.35	-17.2