복합구조물의 선형반복학습제어 정밀도 연구

이수철*

Precision of Iterative Learning Control for the Multiple Dynamic Subsystems

Soo Cheol Lee*

ABSTRACT

다양한 산업체에서 반복적인 특정업무를 수행하는 경우가 흔히 발생한다. 반복되는 오차의 경험치를 근거로 주어진 작업을 추진하는 과정에서 이들 업무의 정밀도제고를 추구함으로써 갖는 성능개선은 사업장의 품질관리와 직결된다. 학습제어의 본래 적용동기는 생산조립라인에 투입되어 반복적인 일을 수행하는 산업용로봇의 정밀도 제고이다. 본 논문에서 분산이산시형시스템에서 출발하였으며, 이를 산업용로봇에 적용하기 위하여 수학적으로 모델링한 모의실험을 통하여 알고리즘의 안정성과 반복오차를 줄여가는 과정을 보여 주었다. 입출력정보가 상호간섭 하는 산업용로봇과 같은 복합구조물에서도 모든 시스템(링크)의 정밀도를 만족함을 보여 줌으로써 복합구조물에서 선형반복학습제어의 안정성을 증명하였다.

Key Words : learning(학습), adaptive(적응), decentralized(분산), control(제어), multiple subsysyems(복합구조물)

1. Introduction

When a control system is required to execute the same command repeatedly, the error in following the command will be repeated (except for random disturbances). It seems a bit primitive to produce the same errors every time the command is given. The new field of learning control refers to controllers that can learn from previous experience executing a command in order to improve their performance. They learn what command input should be given to the system in order to have the response be the desired response. They eliminate the deterministic errors of the control system in executing the command, and they eliminate errors due to disturbances that repeat each time the command is given. Learning controllers aim to accomplish this with minimal knowledge of the system being controlled, and base their adjustments to the command on previous experience performing the command without relying on an a priori model of the system dynamics. There has been considerable research activity in this field in the last few years, some examples of which are given in the references[1-10].

The question arises, what happens if a separate learning controller is used with each of the separate feedback controllers of the robot arm. Such an application represents use of a decentralized learning control. A serious issue is whether the dynamic interactions in the dynamics of the systems governed by the separate learning controllers could cause the learning processes to fail to converge. In a previous paper[9], this question was addressed for the most basic form of

* 대구대학교 자동차산업기계공학부

learning control that is based on use of integral control like concepts applied in the repetition domain. There are various more sophisticated learning control approaches including one that makes use of indirect adaptive control ideas applied to learning in the repetition domain[4]. This approach has an important advantage over the simpler learning control law related to integral control concepts, because it can guarantee convergence of the learning process to produce zero tracking error. In a previous work[4], the authors presented a theory of indirect learning control based on use of indirect adaptive control concepts employing simultaneous identification and control[11]. In Ref. [10], they develop the indirect learning control algorithms, and study the use of such controllers in decentralized systems. It is the purpose of this paper to develop modified forms of the indirect decentralized learning control law in order to improve the control properties more. Some numerical results verify the theory in polar robot.

2. Mathematical Formulation

We first consider a time-varying or time-invariant discrete time system of the following form

$$x_{o,i}(k+1) = A_{o,ii}(k)x_{o,i}(k) + \sum_{\substack{j=1 \\ j\neq i}}^{s} A_{o,ij}(k)x_{o,j}(k) + B_{o,i}(k+1)v_i(k) + w_{o,i}(k)$$
$$y_i(k) = C_{o,i}(k)x_{o,i}(k) \quad ; \quad i = 1,2,3,\cdots,s \tag{1}$$

This represents s subsystems. The input and output matrices for the different subsystems are uncoupled, but there is dynamic coupling between the subsystems represented by the coupling matrices $A_{o,ij}$. The control input to subsystem i is v_i, its state is $x_{o,i}$, and its measured output is y_i. The $w_{o,i}$ represents disturbances that repeat with each repetition of the task. The subscript o refers to the open loop system model. The paper will later consider differential equation models with the same structure.

Now consider that each subsystem has its own decentralized feedback controller with feedback of only that subsystem's measured output. These controllers could be simple proportional controllers with rate feedback which is a common approach in robotics, or they can be more complex controllers including controller dynamics and a controller state variable. We include all such possibilities in the following formulation:

$$v_i(k) = v_{FB,i}(k) + u_i(k)$$
$$v_{FB,i}(k) = C_{FB,i}(k)x_{FB,i}(k) + K_i(k)[y_i(k) - y_i^*(k)]$$
$$x_{FB,i}(k+1) = A_{FB,ii}(k)x_{FB,i}(k) + B_{FB,i}(k)[y_i(k) - y_i^*(k)] \tag{2}$$

Here, the input v_i is the sum of the feedback control $v_{FB,i}$, and the learning control signal u_i. The desired output of the system is

$$y_i^*(k);\ k = 1,2,3,\cdots,p \tag{3}$$

and it is the task of the learning control signal

$$u_i(k)\ ;\ k = 0,1,2,\cdots,p-1 \tag{4}$$

to converge on an altered input command to the feedback system that causes the actual measurements $y_i(k)$ to correspond with these desired outputs. When dynamic controllers are used, the feedback control signal for each system i is determined as the output of this controller's dynamic state variable equation in equation (2). When output feedback is employed, then the dimension of the controller state reduces to zero in equations (2), leaving only the second term on the right in the middle equation.

The system of importance to the learning controller relates the learning control signal u_i to the measured system response. This is accomplished by combining equations (1) and (2) to form the closed loop system dynamic equations

$$x_i(k+1) = A_{ii}(k)x_i(k) + \sum_{\substack{j=1 \\ j\neq i}}^{s} A_{ij}(k)x_j(k) + B_i(k+1)u_i(k) + w_i(k)$$
$$y_i(k) = C_i(k)x_i(k) \quad ; \quad i = 1,2,3,\cdots,s \tag{5}$$

The closed loop system matrices in this equation are

$$A_{ii}(k)=\begin{bmatrix} A_{o,ii}(k)+B_{o,i}(k)K_i(k)C_{o,i}(k) & B_{o,i}(k)C_{FB,i}(k) \\ B_{FB,i}(k)C_{o,i}(k) & A_{FB,ii}(k) \end{bmatrix} \quad (6)$$
$$A_{ij}(k)=\begin{bmatrix} A_{o,ij}(k) & 0 \\ 0 & 0 \end{bmatrix}$$
$$B_i(k)=\begin{bmatrix} B^T_{o,i}(k) & 0 \end{bmatrix}^T$$
$$C_i(k)=\begin{bmatrix} C_{o,i}(k) & 0 \end{bmatrix}$$

and the state vector for system i has been augmented to include the controller state as

$$x_i(k)=\begin{bmatrix} x^T_{o,i}(k) & x^T_{FB,i}(k) \end{bmatrix}^T \quad (7)$$

The exogenous term w_i is still an input that repeats every time the command is given to the system, but now it contains the repetitive command as well as the repetitive disturbance

$$w_i(k)=\begin{bmatrix} w_{o,i}(k)-B_{o,i}(k)K_i(k)y_i^*(k) \\ -B_{FB,i}(k)y_i^*(k) \end{bmatrix} \quad (8)$$

The s coupled subsystems of equation (5) can be written as one large state equation in an obvious manner

$$\begin{aligned} x(k+1) &= A(k)x(k)+B(k)u(k)+w(k) \\ y(k) &= C(k)x(k) \end{aligned} \quad (9)$$

Let the difference operator δ_r operating on any quantity represent the value of that quantity at repetition r minus the value at repetition r -1. Since $w(k)$ repeats each repetition, and since in the learning control problem it is assumed that the initial condition is the same every repetition, we can rewrite (9) as

$$\begin{aligned} &\delta_r \underline{y} = P\delta_r \underline{u} \quad (10) \\ &\underline{y}=[y^T(1) \;\; y^T(2) \;\; \ldots y^T(p)]^T \\ &\underline{u}=[u^T(0) \;\; u^T(1) \;\; \ldots u^T(p-1)]^T \end{aligned}$$

where

$$P=\begin{bmatrix} C(1)B(0) & 0 & \cdots & 0 \\ C(2)A(1)B(0) & C(2)B(1) & \cdots & 0 \\ C(p)(\Pi_{k=1}^{p-1}A(k))B(0) & C(p)(\Pi_{k=2}^{p-1}A(k))B(1) & \cdots & C(p)B(p-1) \end{bmatrix} \quad (11)$$

The product notation represents a matrix product going from larger arguments on the left to smaller arguments on the right.

Now consider how the coupling between subsystems appears in these equations. By reordering the elements of the matrices in (10) to separate elements into those that apply to each subsystem and those that couple the subsystems one can write (10) in the form

$$\delta_r \underline{y}_i = P_{ii}\delta_r \underline{u}_i + \sum_{\substack{j=1 \\ j\neq 1}}^{s} P_{ij}\delta_r \underline{u}_j \quad (12)$$

Here the P_{ii} and P_{ij} are lower block triangular matrices for the ith subsystem. The first represents the pulse responses of the ith subsystem to its own inputs and the second gives the pulse responses of the ith system to inputs in other subsystems. This equation serves as the basic equation for the development of all of our decentralized learning control strategies.

For purposes of illustration, suppose that the original system was time invariant, contained two subsystems (s=2), and that the desired trajectory is three time steps long (p=3). Then equation (12) for system one is

$$\begin{bmatrix} \delta_r y_1(1) \\ \delta_r y_1(2) \\ \delta_r y_1(3) \end{bmatrix} = \begin{bmatrix} C_1B_1 & 0 & 0 \\ C_1A_{11}B_1 & C_1B_1 & 0 \\ C_1(A_{11}^2+A_{12}A_{21})B_1 & C_1A_{11}B_1 & C_1B_1 \end{bmatrix} \begin{bmatrix} \delta_r u_1(0) \\ \delta_r u_1(1) \\ \delta_r u_1(2) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ C_1A_{12}B_2 & 0 & 0 \\ C_1(A_{11}A_{12}+A_{12}A_{22})B_2 & C_1A_{12}B_2 & 0 \end{bmatrix} \begin{bmatrix} \delta_r u_2(0) \\ \delta_r u_2(1) \\ \delta_r u_2(2) \end{bmatrix} \quad (13)$$

Note that due to causality, the P_{ii}, and P_{ij} matrices are lower block triangular, and that in addition the matrices coupling the subsystems, P_{ij} , have zero diagonal block elements.

We will assume that the number of output variables at each time step is the same as the number of input variables. Hence, the product $C(k)B(k-1)$ is square, and we furthermore require that it be full rank. This is required for the existence of a solution. If there are more outputs than inputs in the original description of the problem, one must limit the number of output variables which one wishes to force to have zero tracking error. In this case, one has the option of choosing a different set of outputs each time step so that zero tracking error is obtained for all desired output variables at some but not at every time step. Note that making such changes from one time step to the next will create a time-varying system from a time-invariant one.

3. The Decentralized Application of Indirect Learning Control

The indirect learning control of reference[4] and the indirect decentralized learning control[10] are designed to apply to time varying linear systems, and to apply to time invariant linear systems as a degenerate case. Equation (4) can be thought of as a system representation in modern control form with the state vector being the history of the outputs for a repetition, with the identity matrix as the system matrix, and with the changes in the inputs from one repetition to the next as the control variables. Reference[4] shows how to apply indirect adaptive control in a centralized manner to such a modern control representation operating in the repetition domain. In Reference[10], various possible ways were considered to apply indirect adaptive control ideas in a decentralized manner, and they approved the stability and the convergence to zero tracking error for each subsystems of multiple subsystems. Here, the basic law will be introduced to apply the above indirect decentralized learning control to the polar coordinate robot.

Since system i does not know what inputs are being used in other systems, in order to allow each system to accomplish its goal of learning, the first decentralized learning process considered here requires that at each repetition only one subsystem learns, and the remaining subsystems keep their learning control signals frozen. Hence, if at repetition r it is subsystem i's turn to learn, then equation (12) becomes

$$\delta_r \underline{y}_i = P_{ii}\delta_r \underline{u}_i \tag{14}$$

Such input-output pairs obtained each time it is i's turn to learn, allow the decentralized learning controller for system i to estimate the matrix P_{ii}, call it $\hat{P}_{ii,r}$ Using this estimated matrix, the learning control law generates the change $\delta_r \underline{u}_i$ required to make a change $\delta_r \underline{y}_i$ that will cancel the error according to

$$\delta_r \underline{u}_i = \hat{P}_{ii,r}^{-1}(\underline{y}_i^* - \underline{y}_{i,r}) \tag{15}$$

There are various choices for the estimation of this matrix, including the projection algorithm, the orthogonalized projection algorithm, and the recursive least squares algorithm. Ideally, each of these can be computed in real time from one time step to the next, so that at the end of a repetition the information is available for immediate use whenever the next repetition starts. An important freedom in the learning control problem is that there is no requirement that the computation be made in real time. There is no requirement that learning take place at every repetition, so that one can skip learning for a repetition while one waits for the needed computation to be completed.

Note that one can make use of the lower block triangular nature of the estimated matrix in order to obtain the inverse in a recursive manner. The centralized indirect learning control results in Reference[4] guarantee zero tracking error without any requirement that the identified matrix $\hat{P}_{ii,r}$ converges to the true matrix. This result is analogous to standard results in adaptive control theory. We will not address such issues here. Instead we simply agree to introduce an independent $\delta_r \underline{u}_i$ if at some repetition, (15) produces a change in the leaning control input which is not independent of previous changes.

Here we consider the recursive least squares algorithm because it is relatively insensitive to noise, and because it can guarantee convergence in a finite number of steps when the data is noise-free and independent. The equations appropriate for (15) are given in Reference[3].Consider the computations made by the ith subsystem, and for the sake of simplicity of notation, we temporarily drop explicit indication of dependence on i in the symbols used. Let $\hat{p}_{l,r}$ represent the column vector which is the transpose of the lth row of $\hat{P}_{ii,r}$, but with the zero elements to the right of the block diagonal deleted from the column vector. Let $\delta_r^l \underline{u}$ represent the quantity $\delta_r \underline{u}$ but with the elements deleted that are multiplied by these zero elements in the product $P_{ii}\delta_r \underline{u}_i$.

And let $\delta_r^l \underline{y}$ represent the lth row of $\delta_r \underline{y}$. Then the recursive least squares update is

$$\hat{P}_{l,r} = \hat{P}_{l,r-1} + M_{l,r-2}\delta_r^l \underline{u}\left[\frac{\delta_r^l \underline{y} - (\delta_r^l \underline{u})^T \hat{P}_{l,r-1}}{1+(\delta_r^l \underline{u})^T M_{l,r-2}\delta_r^l \underline{u}}\right] \quad (16)$$

$$M_{l,r-1} = M_{l,r-2} - \frac{M_{l,r-2}\delta_r^l \underline{u}(\delta_r^l \underline{u})^T M_{l,r-2}}{1+(\delta_r^l \underline{u})^T M_{l,r-2}\delta_r^l \underline{u}} \quad ; \quad r \geq 2$$

The initial value $M_{l,0}$ is chosen as the identity matrix of the same dimension as the $\hat{P}_{l,r}$ corresponding $\hat{P}_{l,r}$. Note that this matrix need only be updated when the dimension of increases when l is increased, and the same $M_{l,r}$ can be used for all rows corresponding to the same time step in the multiple output case.

The decentralized indirect learning control algorithm based on the centralized indirect learning control algorithm of Reference[4] can be summarized as follows. Only one subsystem learns during each repetition, while the other subsystems keep their learning control signals unaltered. As the repetitions progress, each subsystem gets its opportunity to learn in an order that is pre-chosen and known to each of the learning controllers. Then at the repetition for which the ith subsystem learns, the learning control law for subsystem i is: equation (16), together with equation (15) with a recursive computation of $\hat{P}_{ii,r}^{-1}$, and together with the requirement for independent changes of the learning control signal for this subsystem. In a later subsection we will study the convergence behavior of this decentralized learning control scheme. We will also develop a modified version of the algorithm requiring less computation, and producing faster convergence.

4. Convergence of the Decentralized Indirect Learning Control Algorithm

For convenience, consider the case of two subsystems, s=2 . Later we will consider how the results generalize to more subsystems. Define the error of the ith subsystem at repetition r as

$$\underline{e}_{i,r} = \underline{y}_i^* - \underline{y}_{i,r} \quad (17)$$

During repetition r, let it be subsystem 1's turn to learn while subsystem 2 does not change its learning control signal:

$$\begin{aligned} \underline{u}_{i,r+1} &= \underline{u}_{i,r} + \delta_{r+1_i}\underline{u}_i \\ \delta_r \underline{u}_1 &= \hat{P}_{11,r-2}^{-1}\,\underline{e}_{1,r-1} \\ \delta_r \underline{u}_2 &= 0 \end{aligned} \quad (18)$$

Note that the repetition number used on the estimate of the P_{11} matrix reflects the last repetition for which new information was obtained for this subsystem. The error propagation equations for this repetition are:

$$\begin{aligned} \underline{e}_{1,r} &= [I - P_{11}\hat{P}_{11,r-2}^{-1}]\underline{e}_{1,r-1} \\ \underline{e}_{2,r} &= \underline{e}_{2,r-1} - P_{21}\hat{P}_{11,r-2}^{-1}\underline{e}_{1,r-1} \end{aligned} \quad (19)$$

Then at repetition r+1, the situation is reversed and gives the following error propagation

$$\begin{aligned} \underline{e}_{1,r+1} &= [I - P_{11}\hat{P}_{11,r-2}^{-1}]\underline{e}_{1,r-1} + [P_{12}\hat{P}_{22,r-1}^{-1}P_{21}\hat{P}_{11,r-2}^{-1}]\underline{e}_{1,r-1} \\ &\quad - P_{12}\hat{P}_{22,r-1}^{-1}\underline{e}_{2,r-1} \\ \underline{e}_{2,r+1} &= [I - P_{22}\hat{P}_{22,r-1}^{-1}][\underline{e}_{2,r-1} - P_{21}\hat{P}_{11,r-2}^{-1}\underline{e}_{1,r-1}] \end{aligned} \quad (20)$$

By construction, the estimate of P_{ii} must converge to the true values when there is no noise in the data. Let us suppose that at repetition r-1 this identification has been achieved by both subsystems, but that the errors $\underline{e}_{1,r-1}$ and $\underline{e}_{2,r-1}$ contain all nonzero components. Then $\underline{e}_{1,r}$ will be identically zero, as it will be at r+2, r+4,... The error $\underline{e}_{2,r}$ will still have all nonzero elements, but at r+1 , r+3 , r+5 , ... it will have all zero elements. Note that a product such as $P_{21}\hat{P}_{11,r-2}^{-1}$ is lower block triangular, and furthermore the diagonal blocks are all zero. When two such products are multiplied together, then the subdiagonal blocks become zero as well. Hence, $\underline{e}_{1,r+1}$ has one zero element appearing according to equation (20), and of course $\underline{e}_{2,r+1}$ is identically zero. Now replace r by r+2 in (19) and (20), and repeat.

The above reasoning establishes that once the pattern has been set up, the number of nonzero elements of the error that is not automatically zero at a given repetition, increases by two every time the repetition number increases by two. Therefore, in a finite number of repetitions the decentralized indirect learning controllers converge to zero tracking error for the desired trajectories of both subsystems.

If the learning control was accomplished in a centralized manner, convergence in the absence of noise is reached on or before the repetition for which the full P matrix is identified. Here we have not identified the whole *P* matrix, but rather P_{11} and P_{22} only, and it is necessary to add additional repetitions once these are determined in order to account for the dynamic coupling between the subsystems.

It is of interest to see how the convergence behavior is affected by increasing the number of subsystems. Suppose that there are three subsystems, and that the P_{ii} have all converged to the correct values by repetition r. Following the same reasoning as above, we find that at repetition r, *r*+3, *r*+6, *r*+9, *r*+12, the error in underline $\underline{e}_1$ is identically zero, and the errors $\underline{e}_2$ and $\underline{e}_3$ have a number of definitely zero elements given by 1,2; 3,3; 4,5; 6,6; respectively. The "irregularity" in this sequence is perhaps surprising, but one notes that as *r* increases by 6, the number of definitely zero elements increases by three. This same property is observed for the errors at times *r*+1, *r*+4, *r*+7, ..., and at times *r*+2, *r*+5, *r*+8, ...

Theoretically, this control law starts from the deterministic system with periodic disturbances. The periodic disturbaces become deleted mathematically and physically in this learning control problem. And, noise problem will be solved in the next paper.

5. Improved Decentralized Learning Control Algorithms

The approach taken to obtain the above algorithms is the application of indirect adaptive control ideas to the system equations in the repetition domain. In order to implement the decentralized versions for these indirect learning control algorithms it was necessary to have the subsystems cooperate in taking turns learning. In this section we develop another approach with a different type of agreement between the learning controllers.

For simplicity, consider a system containing only two subsystems, and refer to equation (13). During the first run of the system, subsystem 1 applies $\underline{u}_{1,0}$ (which could be zero) and observes response $\underline{y}_{1,0}$, while subsystem 2 applies $\underline{u}_{2,0}$ and observes $\underline{y}_{2,0}$. In the first repetition, each subsystem can change its control at the first time step, and observes

$$\begin{aligned}\delta_1 y_1(1) &= C_1 B_1 \delta_1 u_1(0) \\ \delta_1 y_2(1) &= C_2 B_2 \delta_1 u_2(0)\end{aligned} \tag{21}$$

These equations are decoupled, so that each system can learn without concern for disturbances from the dynamic coupling with the other subsystem. We presume that the number of inputs and outputs of each subsystem is known, so that the number of repetitions needed for each subsystem to identify its $C_i B_i$ is known in the deterministic case. One can start with an a priori estimate of each, and use a recursive formulation to update the estimate after each repetition. During these repetitions, each controller uses its estimate $E_r(C_i B_i)$ to determine its control action for this time step in the next repetition according to

$$\delta_r u_i(0) = [E_r(C_i B_i)]^{-1}[y_i^*(1) - y_{i,r-1}(1)] \tag{22}$$

As before, if this learning control rule fails to give a linearly independent change in the control, such a change is substituted. If desired the learning control signals at later time steps can be adjusted as well, according to any desired rule, but any such adjustment has no influence on the convergence of the method being discussed here.

Once enough repetitions have been made so that each subsystem has obtained knowledge of its $C_i B_i$, then starting with the next repetition no change is made in the leaning control signal for time step zero, and adjustments are made for the learning control at time step 1. According to (13)

$$\delta_r y_1(2) = C_1 A_{11} B_1 \delta_r u_1(0) + C_1 B_1 \delta_r u_1(1) + C_1 A_{12} B_2 \delta_r u_2(0) \tag{23}$$

The agreement not to change the learning control signals at the zeroth time step, reduces this equation to

$$\delta_r y_1(2) = C_1 B_1 \delta_r u_1(1) \tag{24}$$

Several situations can apply here. In the case of a time-invariant system without noise in the measurements, each subsystem has already determined the values in $C_i B_i$, and hence each subsystem knows how to get zero tracking error for this time step in the next repetition. When the measurements are noisy, this equation is one more equation to use in a recursive least squares approximation of $C_i B_i$. In the case of time-varying system equations, the $C_i B_i$ in (24) has different time arguments than in (21), and repetitions must be allocated to accomplish this identification in the same manner as was done for (21).

This wave of learning is continued until all time steps have been treated. In the absence of noise in the measurements, this learning control approach produces zero tracking error in a finite number of repetitions (provided the desired trajectory is feasible).

Note that by using this wave of learning, zero tracking error is achieved without having identified all of the elements of the P or of the P_{ii} matrices. This is possible because the control law involved is a one-step ahead control.

Note also that no penalty is paid for accomplishing the learning control in a decentralized manner. Stated in other terms:

Result: In a noise free environment, if the maximum number of repetitions needed for any subsystem to accomplish zero tracking error provided there were no dynamic coupling between subsystems is R, then the decentralized indirect learning control can accomplish zero tracking error in this same number of repetitions.

This result follows from the fact that at each time step of learning as the repetitions progress, each subsystem can learn without disturbance from other subsystems.

The learning control law proposed here is a decentralized version of a combination of learning control laws suggested and studied in [5], one involving learning in a wave and the other involving identification of one Markov parameter. The number of repetitions used here to obtain zero tracking error is related to the number of discrete time steps in the repetitive process, and this can be a large number. It was shown in [5] that if one accepts an asymptotic approach to zero tracking error rather than insisting on zero error in a finite number of repetitions, it is possible to set the learning control gains so that essentially zero error is obtained in many fewer repetitions. This result was for centralized learning control, but it suggests that one might observe faster convergence in the decentralized indirect learning control if one uses a properly chosen learning gain ϕ in the control law used in future time steps in advance of the wave of learning

$$\delta_r u_i(k-1) = \phi[E_r(C_i B_i)]^{-1}[y_i^*(k) - y_{i,r-1}(k)] \tag{25}$$

In the presence of noise, once the cooperative wave of learning is completed (i.e. it has progressed from the first time step to the last time step of the repetitive process), one will not have zero tracking error. One must decide what actions to take to continue learning once this repetition is completed. One can of course repeat the wave, using the new information gained to improve one's estimates of the $C_i B_i$, whether time-varying or time-invariant. On the other hand, one can use the deadbeat control law of the form of (22) for each time step, and provided the estimates of $C_i B_i$ are sufficiently good that the eigenvalues of

$$I - (C_i B_i)[E_r(C_i B_i)]^{-1} \tag{26}$$

are inside the unit circle, convergence is guaranteed. Such a condition should surely be satisfied. For the purposes of the next section, we will assume that it is this rule that is followed after the first wave is completed.

6. Numerical Examples

This paper presented the concept for application of indirect learning control in a decentralized manner, and showed that it can lead to guaranteed convergence to zero tracking error. The concept still leave us with a number of choices for implementation. We can have each subsystem take turns learning, and we can choose how often they alternate. Or we can have the subsystems learn simultaneously, but do so in a wave, and in this case we can choose how fast to make the wave progress. In the identification process, we also have choices. One can use the recursive least squares method as was suggested above, or one can simply solve the simultaneous equations for the unknowns, which can be more reasonable when the number of unknowns is small, as is the case when learning in a wave. In this section we examine these options by studying several examples.

6.1. Dynamic Model for the polar coordinate robot

The nonlinear equations for motion of the polar coordinate robot in Fig. 8 are given as

$$(m_B + m_L)\ddot{\rho}(t) - [m_B\rho(t) + m_L(\rho(t)+L)]\dot{\theta}_1(t)^2 = F(t)$$

$$[I_3 + m_B\rho(t)^2 + m_L(\rho(t)+L)^2]\ddot{\theta}_1(t) + 2[m_B\rho(t) + m_L(\rho(t)+L)]\dot{\rho}(t)\dot{\theta}_1(t) = M_{o,1}(t) \tag{27}$$

where $\rho(t)$ is the radial extension of the prismatic joint measured from the center of the support point to the center of mass of the prismatic beam (without load), and $\theta_1(t)$ is the angle of rotation of the beam about the vertical axis. The beam mass is $m_B = 39.28kg$, its half length is $l = 0.6$, and its moment of inertia about the vertical axis is $I_3 = 1.93kgm^2$. The mass of the point mass load located at the end of the beam is $m_L = 10kg$. The force and moments applied to each joint are supplied by proportional plus derivative feedback controllers given by

$$F(t) = K_1[\rho(t) - \rho^*(t)] + K_2[\dot{\rho}(t) - \dot{\rho}^*(t)] + u_1(t)$$
$$M_1(t) = K_3[\theta_1(t) - \theta_1^*(t)] + K_4[\dot{\theta}_1(t) - \dot{\theta}_1^*(t)] + u_2(t) \tag{28}$$

where, K_1, K_2, K_3, K_4 are the feedback gains with values 98.6, 443.5, 450.9, 182.2 respectively, and $u_1(t)$ and $u_2(t)$ are the learning control signals.

The desired trajectory is again given by Fig. 7, where the subsystem 1 graph is $\rho^*(t)$ in meters, and the subsystem 2 graph is $\theta^*(t)$ in radians.

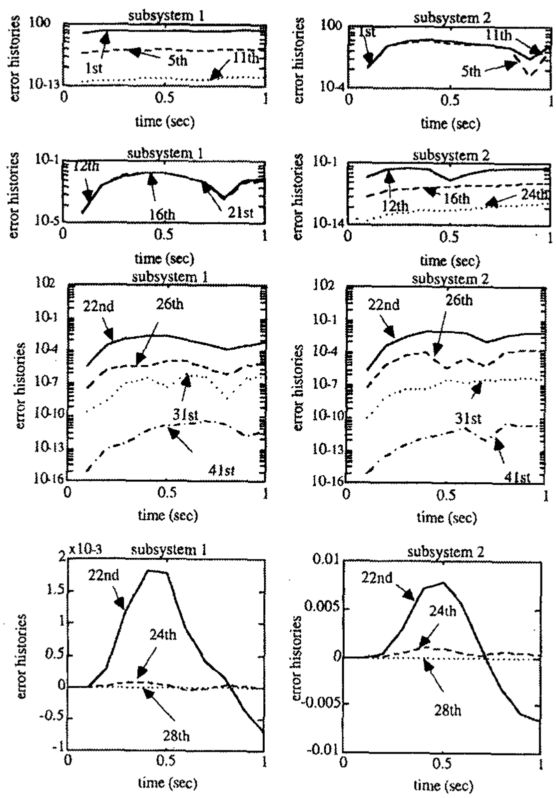

Fig. 1 Error histories for the linearized polar coordinate robot using alternate learning after initial identification of P_{ii} by each subsystem.

6.2. Subsystems Alternate Learning the Complete Trajectory

Figure 1 presents results when this decentralized learning process is used on the linearized model of the polar coordinate robot example with 10 time steps during the 1 second maneuver. In this figure and those that follow, subsystem 1 refers to control of radial displacement, measured in meters, subsystem 2 controls the rotation angle, given in radians. The repetition 1 corresponds to using the feedback controller alone, and then repetitions 2 through 11 correspond to having subsystem 1 doing the learning. Without noise this number of repetitions is sufficient for system 1 to learn all elements of P_{11}. However, in this example we use

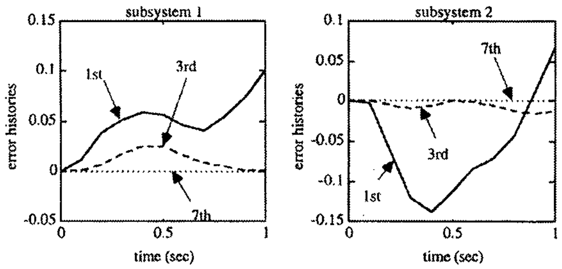

Fig. 2 Error histories for decentralized learning in the linearized polar coordinate model using alternate learning every repetition starting with repetition 2.

the usual recursive least squares equation (16), which has a 1 in the denominator that is introduced to avoid the possibility of singularity. This means that the resulting identification is not exact in the noise free case except asymptotically. The initial condition for the recursive least squares is an a priori estimate of P_{ii} that is in error -- every element being 10% too high. The next two parts of the figure present the corresponding repetitions when subsystem 2 learns P_{22}, repetitions 12 through 21. Then the subsystems alternate learning every repetition, with subsystem 1 learning in repetition 22. Both linear plots and logarithmic plots are presented for these repetitions. By the 28th repetition the error is zero to the plotting accuracy of the linear plot.

Figure 2 simulates the same system, but the learning is alternated between subsystems starting from the first repetition, without separate repetitions allocated for each system to learn its own P_{ii}. The first repetition corresponds to feedback control only, and then in the first repetition subsystem 1 learns. The theory presented guarantees convergence for this case as well (in both cases a sufficiently small sample time must be used). On the linear plots, essentially zero tracking error is reached after 7 repetitions, which is much faster than the 28th repetition in the previous case. This difference would be even more extreme if there were more time steps in the trajectory. We conclude that it is best to start alternating from the first repetition.

6.3. Subsystems Learn Progressive Time-Steps Simultaneously

In the decentralized learning method of the above examples, each subsystem eventually needs to identify all elements of its P_{ii} matrix. When the subsystems learn simultaneously, one time step at a time, the identification is limited to having each subsystem find the instantaneous value of its own input-output matrix product. Rather than use the recursive least squares approach of (16), here we compute this product as follows:

$$u_{i,r}(k) = u_{i,r-1}(k) + \delta_r u_i(k)$$

where

$$\delta_r u_i(k) = [E_r(C_i(k+1)B_i(k))]^{-1}.(y_i^*(k+1) - y_{i,r-1}(k+1)) \tag{29}$$

The estimate $E_{r+1}(C_i(k+1)B_i(k))$ of $C_i(k+1)B_i(k)$ of subsystem i is chosen as the latest value according to

$$E_{r+1}(C_i(k+1)B_i(k)) = \frac{\delta_r y_i(k+1)}{\delta_r u_i(k)} \tag{30}$$

Care must be taken to avoid singularity problems in performing this division, when the learning control signal approaches convergence. We will vary the number of repetitions used for learning at each time step. If the system were truly a linear time-varying discrete time system with no coupling between subsystems in the input and output matrices, one would prefer to average the set of numbers obtained from (30) for this time step, rather than use the latest value, in order to average the effects of noise in the data. However, noise may not be the issue. Other considerations suggest using the latest value. In the process of discretizing the linearized differential equations, coupling was introduced between the subsystems, which for small sample times is small but not zero. Also, the actual system is nonlinear, and the linearized differential equation model considered here to model the system is linearized about the desired trajectory. Hence, it is only as the system approaches the desired trajectory that the estimate of the

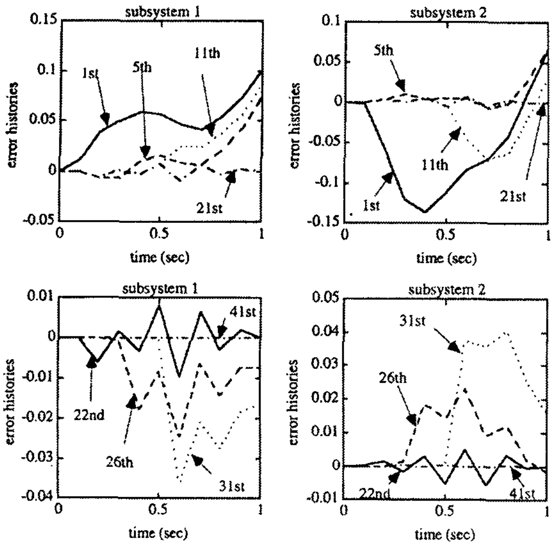

Fig. 3 Error histories for learning in a wave in the linearized polar coordinate model, progressing one time step every two repetitions, and performing two waves of learning.

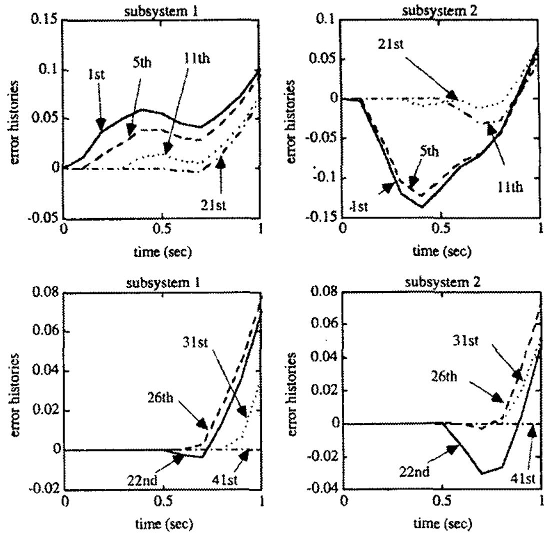

Fig. 4 Error histories for learning in a wave, progressing one time step every four repetitions, and performing one wave of learning.

$C_i(k+1)B_i(k)$ product approaches the true value.

Figure 3 presents results of learning in the same linearized polar coordinate model as above, with the same sample time. The first repetition is with feedback control only, and then the wave of learning starts, using two repetitions for each time step. After the wave finishes the final time step at repetition 21, a second wave of learning is performed for repetitions 22 through 41. The computations use noise free data computed from the time varying difference equation. Due to the effects mentioned above, the error at the end of the first wave is not zero, although the error is much improved over feedback alone. During the second wave, the error behind the wave is made very small, although the error in front of the wave is somewhat accentuated temporarily during the learning process. Figure 4 shows the corresponding results when four repetitions are used at each time step, and only one wave of learning is used, for the same total of 41 repetitions. During repetitions 1 through 21 in Fig. 3, one has somewhat better and more uniform error histories than in Fig. 4, but during the second wave of learning for repetitions 22 through 41 in Fig. 3 the errors do not decay monotonically and can be worse than in Fig. 4. So, the double wave of learning with two repetitions per time step has better initial behavior, but one pays some price later with worse transient behavior after the transients have decayed significantly.

The concept of learning in a wave was first introduced in Ref. [8] for centralized linear learning control as a method to improve the learning control transients. It was introduced here for a different reason, as a way to decouple the subsystems when the subsystems are learning simultaneously. It may have advantages in producing better transients as well. This may be true when one has only poor a priori knowledge of the system, but the fact that the learning transients are better in Figs. 2 than in 3 indicates that when one knows the P_{ii} matrices to within 10%, there is no need to learn in a wave for purposes of improving transients.

All of the above results used the time varying linearized difference equations for horizontal motion of the polar coordinate robot model. Figures 5 and 6 apply

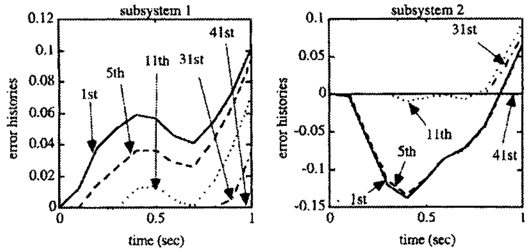

Fig. 5 Error histories for one wave of learning in the nonlinear polar robot model, using for learning at each time step, with sample time T=0.1 sec.

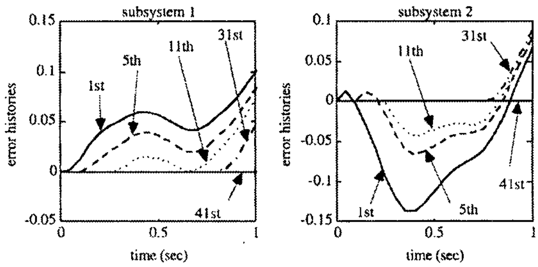

Fig. 6 Error histories for one wave of learning in the nonlinear polar robot model, using two repetitions for learning at each time step, with sample time T=0.05 sec.

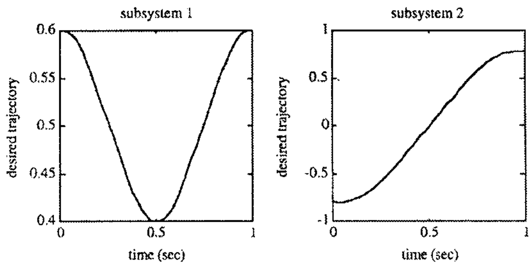

Fig. 7 Desired trajectories for two different subsystems

decentralized learning control to the nonlinear differential equation model in order to see the effects of nonlinearities. Figure 5 repeats Fig. 4 for these nonlinear differential equations, i.e., it uses learning in a wave with four repetitions for each time step, before letting the wave progress a time step. As before, the

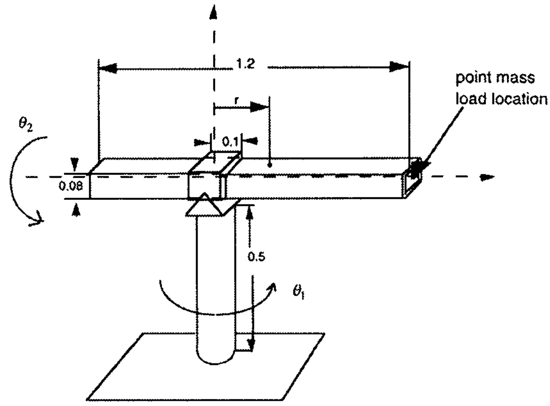

Fig. 8 Scheme of Polar Coordinate Robot

wave of learning is complete after 41 repetitions. The error histories are similar to those for the linearized time varying model in Fig. 4, although for subsystem 2 the results are somewhat worse throughout for repetition 5, and somewhat worse at the end in repetition 11. In the nonlinear case, the multiple repetitions for each time step try to correct for not only the coupling in the input matrices introduced in the time discretization, but also for system nonlinearities. Figure 6 cuts the sample time in half to 0.05 seconds, which decreases both the coupling in the input matrices and the influence of nonlinearities during one time step. The number of repetitions per time step is decreased to two, so that the wave of learning is again finished at the end of 41 repetitions. Thus, we study the trade-off between decreasing these coupling effects by decreasing the number of time steps, versus decreasing these coupling effects by repeated repetitions at the same time step.

Comparing Figs. 5 and 6, does not give a clear indication of which approach is best. For subsystem 2, using the smaller sample time results in a substantial improvement in performance at repetition 5, but a somewhat worse error in repetition 11.

7. Concluding Remarks

In this paper, two classes of methods were developed for decentralized indirect learning control based on different agreements between the subsystems as to when each subsystem learns. In the first, the subsystems agree to alternate the learning of the complete trajectory with the repetitions.

The second algorithm has the appeal of learning in a wave progressing from the start of the p-step process and progressing to the end, with all subsystems learning simultaneously the same time step. Fewer parameters need to be identified when learning in a wave than in the alternate learning approach, and this distinction is even more extreme in the case of time-invariant systems. In [8], learning in a wave similar to this was used with the integral control based learning control, as one technique to have control over the size of the transients in the learning process. Numerical results indicate that learning in a wave is preferable to the alternate learning method when one has very poor a priori knowledge of the system, but the reverse is true if one has a reasonable system model.

Examples also illustrate the trade-offs between how many repetitions are used for each time step when learning in a wave, how many waves of learning to use, and how small a sample time to use.

Acknowledgement

Supported by Taegu University Grant in 1999.

References

1. Arimoto, S., Kawamura, S., and Miyasaki, F., "Bettering Operation of Robots by Learning," Journal of Robotic Systems, Vol. 1, No. 2, pp. 123-140, 1984.
2. Middleton, R. H., Goodwin, G. C., and Longman, R. W., "A Method for Improving the Dynamic Accuracy of a Robot Performing a Repetitive Task," International Journal of Robotics Research, Vol. 8, No. 5, pp. 67-74, October 1989.
3. Phan, M. and Longman, R. W., "A Mathematical Theory of Learning Control for Linear Discrete Multivariable Systems," Proceedings of the AIAA/AAS Astrodynamics Conference, Minneapolis, Minnesota, pp. 740-746, August 1988.
4. Phan, M. and Longman, R. W., "Indirect Learning Control with Guaranteed Stability," Proceedings of the 1989 Conference on Information Sciences and Systems, Johns Hopkins University, Balti more, MD, pp. 125-131, 1989.
5. Phan, M. and Longman, R. W., "Learning System Identification," Modeling and Simulation, Instrument Society of America, Vol. 20, Part 5, pp. 1857-1864, 1989.
6. Phan, M. , Wei, Y., Horta, L. G., Juang, J. –N., and Longman, R. W., "Learning Control for Slewing of a Flexible Panel," Dynamics and Control of Large Structures, Proceedings of the Seventh VPI&SU Symposium, L. Meirovitch, Editor, pp. 39-52, May 1989.
7. Phan, M, Juang, J.-N., and Longman, R. W., "Recent Developments in Learning Control and System Identification for Robots and Structures," Dynamics of Flexible Structures in Space, Proceedings of the First International Conference, Cranfield, England, May 1990.
8. Chang, C.-K., Longman, R. W., and Phan, M., "Techniques for Improving Transients in Learning Control Systems," Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Durango, Colorado, Advances in the Astronautical Sciences, August 1991.
9. Lee, S. C., Longman, R. W., and Phan, M., "Linear Decentralized Learning Control," Paper No. 91-447, Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Durango, Colorado, Advances in the Astronautical Sciences, August 1991.
10. Longman, R. W., Lee, S. C., and Phan, M., "Indirect Decentralized Learning Control," Advances in the Astronautical Sciences, American Astronautical Society, Proceedings of the 1992 AAS/AIAA Spaceflight Mechanics Meeting, Colorado Springs, February 1992.
11. Goodwin, G. C. and Sin, K. S., "Adaptive Filtering Prediction and Control," Prentice Hall, Inc., 1984.