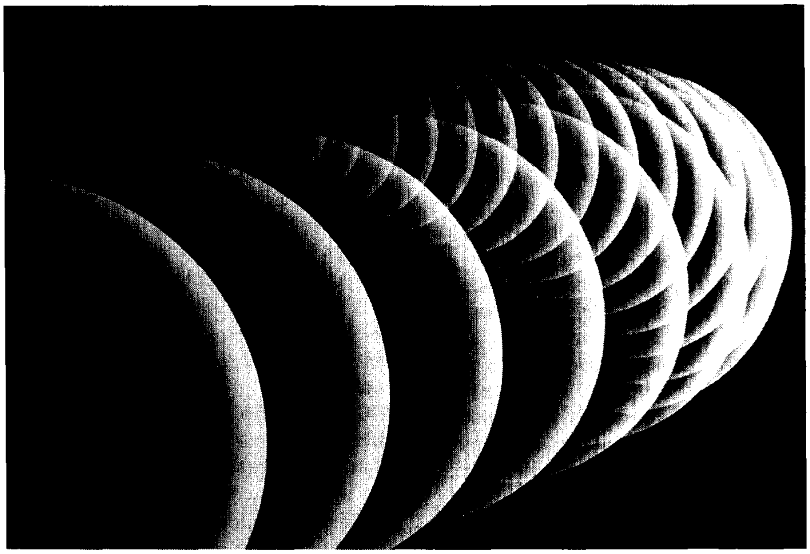
Adaptation and Learning Using Multiple Models, Switching, and Tuning

Kumpati S. Narendra, Jeyendran Balakrishnan, and M. Kemal Ciliz



© Photo Disc, 1993

This article presents a general methodology for the design of adaptive control systems which can learn to operate efficiently in dynamical environments possessing a high degree of uncertainty. Multiple models are used to describe the different environments and the control is effected by switching to an appropriate controller followed by tuning or adaptation. The study of linear systems provides the theoretical foundation for the approach and is described first. The manner in which such concepts can be extended to the control of non-linear systems using neural networks is considered next. Towards the end of the article, the applications of the above methodology to practical robotic manipulator control is described.

Introduction

The work reported in this article represents the present status of a rapidly growing area which has been investigated in detail since 1991 at Yale University under the direction of the first author. What started in 1991 as an initial effort on the part of the first two authors to improve the transient performance of linear adaptive control systems with large parametric uncertainties, has gradually evolved to a general methodology for the design of adaptive systems that can learn from experience about different environments, and in course of time act swiftly and accurately even as the environments vary more rapidly within the learned set. This methodology is based on the use of multiple models, switching, and tuning.

From the very beginning, work has been in progress concurrently in both linear and non-linear systems. In linear adaptive control, the aim was to develop stable and efficient switching and tuning schemes which would improve the transient response [1,2,3]. In non-linear systems, the effort was initially directed toward identifying and controlling dynamical systems using neural networks, when only inputs and outputs are accessible [4,5,6]. It was soon realized that the approach based on multiple models, developed for linear systems, could attain its full potential only when the system was operating in the non-linear domain. While stability analysis with linear models was initiated in 1991 [1], the first attempt on the control of aircraft using multiple neural network controllers and pattern recognition was carried

The authors are with the Center for Systems Science, Yale University, P.O.Box 208267, New Haven, CT 06520-8267. *The research of the first two authors was supported by NSF Grants ECS-9203928 and IRI-9216823. The third author was on leave from Bogazici University in Istanbul, Turkey, and was supported by a NATO-Turkish Scientific and Technological Research Council Fellowship.*

 0272-1708/95/$04.00©1995IEEE

out around the same time [7]. It is the confluence of the efforts in the two areas that has evolved into the general methodology reported in this article.

In 1993, the third author, who was visiting the Center for Systems Science at Yale, proposed robotic manipulator control as a test bed for evaluating the efficacy of the methodology in practical situations. The theoretical results derived earlier for general linear systems were extended to the specific problem of robot manipulator control. Both computer simulations and real-time control in the laboratory followed. The results obtained once again confirmed those predicted by theory, and revealed that improved performance could be achieved in practical systems by using multiple models, switching, and tuning.

The use of multiple models, tuning, or switching is not new in control theory. In fact, multiple Kalman filters were proposed in the 1970s by Lainiotis [8] and Athans et al. [9] to improve the accuracy of the state estimate in control problems. Many interesting practical applications of this approach have appeared in the following two decades. Moose and co-workers used these ideas in maneuverable target tracking, while Kaufman and colleagues applied it to medical control. Fault detection and control in aircraft was proposed by Maybeck and Pogoda, and Bar Shalom and colleagues adapted it to problems in air traffic control. All these cases do not involve switching, and only a linear combination of the control determined by different models is used. Further, no stability results for these systems have been reported.

In recent years, switching has assumed importance in the context of adaptive stabilization. This new trend was started in 1986 by Martennson [10] to relax the conditions under which unknown linear systems can be adaptively stabilized. Both direct and indirect switching schemes have appeared in the literature. In the former, switching to a new controller is based on the output of the plant, but such schemes were soon realized to have very little practical utility. In the latter, multiple models are used to determine both when and to which controller the switching should take place. Using such an approach, non-minimum phase systems [11] and minimum phase systems with unknown relative degree [12] have been adaptively controlled in a stable fashion.

In contrast to the above approaches, the research effort reported here has concentrated on improving performance in both linear and non-linear systems operating in rapidly time-varying environments, while assuring stability. Theoretical analysis of abstract switching and tuning systems, computer simulations, and applications of the approach to practical problems (e.g., distillation columns) are currently underway. On the basis of the above, the authors believe that the approach has enormous potential both as a source of interesting theoretical problems as well as a methodology of great practical relevance.

The article is written with the following objectives: The first is to present the basic concepts involved and provide the reader with an appreciation of the scope of the problem. The second is to demonstrate how the problem can be posed mathematically and indicate the theoretical results that are currently available in the linear case. The third objective is to point out the essential differences between the linear and non-linear problems and describe how neural networks can be used in the latter case. The last objective is to present experimental results in the control of robotic manipulators and to motivate the readers to extend the approach to practical problems of specific interest to them.

General Methodology

In this section, the basic definitions and concepts used throughout the article are presented, and the structure of the overall system and its operation are described briefly.

Model: A model can be considered as the representation of the essential parts of the system in a convenient form. Depending upon its purpose, a model may take many forms. The simplest and most useful method of representing the behavior of a system is by a mathematical model. Different mathematical models are described in the following sections for general linear and non-linear systems as well as specific robotic manipulators. In truly complex systems, multiple models (both heuristic and mathematical) based on different assumptions may have to be used to achieve improved performance.

Environment: The behavior (or equivalently the input-output characteristics) of a system may change in different environments. Assuming that the system is described by a set of differential or difference equations of the form

$$\begin{aligned} \dot{x}(t) &= f[x(t), u(t), p] & x(k+1) &= f[x(k), u(k), p] \\ y(t) &= h[x(t), p] & y(k) &= h[x(k), p], \end{aligned} \tag{1}$$

the equations obviously contain both the plant and the external environment. Different environments can be expressed by different values of the constant parameter vector p, if f and h are assumed to be fixed, or alternately by different functions (f_i, h_i), $i = 1, 2, \cdots$. Expressed in this fashion, faults in the system, sensor and actuator failures, external disturbances, and variations of parameters can be considered as corresponding to different environments.

Multiple Models: Any complex system is required to operate in different environments. When the environment changes, the input-output characteristics of the system will generally change rapidly or even discontinuously. If a single identification model is used, it will have to adapt itself to the new environment before appropriate control action can be taken. In linear systems, such adaptation may be possible, but the slowness of adaptation may result in a large transient error. In non-linear systems, where different environments are described by different functions f and h in Equation (1), a single model may not be adequate to identify the changes in the system (i.e., a model may not exist in the assumed framework to match the environment). Hence, multiple models are required both to identify the different environments as well as to control them rapidly. The need for multiple models goes even beyond this. In some environments different models may be available whose validity (or accuracy) depends upon the region in the state space where the system trajectories lie. All the above considerations suggest that multiple models may be preferable to a single model in many different situations and that the reasons for using multiple models may themselves not be simple.

Parameterization of Models and Controllers: For mathematical convenience, as well as for a precise definition of the control problem, we shall assume that the plant and all the identification models can be parametrized in the same fashion. If S is a closed and bounded set in a finite dimensional parameter space, it is assumed that the plant parameter vector p and the model parameter vectors $\hat{p}_i$ belong to S. Hence, the plant parameter vector can assume an infinite number of values and the objective is to improve the performance using a finite number of models (i.e., $\{\hat{p}_i\}_{i=1}^{N}$).

Corresponding to each parameter vector $\hat{p}_i$ there exists a neighborhood $S_i \subset S$ (also called the ith environment), which is characterized as follows: for all $p \in S_i$, the controller C_i yields a tracking error (between the output of the plant and the desired output) which, according to some criterion function J_c, is smaller than a constant ε_2. ε_2 can be considered as the maximum error that will result using a fixed controller C_i, when $p \in S_i$.

If $p \in S_i$, the error criterion J_c can be reduced further by tuning the controller. Toward this end the fixed controller C_i can be considered as providing the initial condition for an adaptive controller. The objective of adaptation is to determine a controller C_i^* such that the tracking error is smaller than a constant $\varepsilon_1 << \varepsilon_2$. If the above procedure is used, ε_2 can be considered as the maximum transient error, and ε_1 the maximum steady state error, when it is known *a priori* that $p \in S_i$.

Since the plant parameter vector p can lie anywhere in S, it is necessary to determine which of the controllers C_i must be used at any instant, and when switching from one controller to another should take place. In any control problem, multiple identification models can be used at any instant but only one control input can be chosen. Hence, switching and tuning have to be carried out on the basis of identification errors rather than control errors.

Structure of the Control System: The structure of a system which can, at least in theory, estimate which environment is currently in existence and service it appropriately is shown in Fig. 1. This was originally introduced in [1]. There are N identification models denoted by $\{I_i\}_{i=1}^N$ with corresponding outputs $\{y_i\}_{i=1}^N$. At every instant, some measure of the identification errors $e_i \overset{\Delta}{=} \hat{y}_i - y$ is determined as $J_i(t)$, $i = 1, 2, ..., N$, and the model corresponding to $\min_i \{J_i(t)\}$ is chosen to determine the control input at that instant to the plant.

Adaptive and Learning Control: If $\hat{p}_i$ is a known parameter vector, and $S_i \subset S$ is a subset which satisfies the conditions described earlier, we call S_i an "anticipated environment." If $\bigcup_{i=1}^N S_i = S$, then every possible plant belongs to some anticipated environment. This corresponds to the case when learning concerning the plant is complete. In practice, this is rarely the case, and all the parameters $\hat{p}_i$, ε_1, and ε_2 (and hence S_i) have to be learned on-line. These give rise to a host of theoretical questions, many of which are, as yet, unanswered.

The adaptive and learning control problems are best illustrated by considering the case where a finite number of models with parameter vectors $\{\hat{p}_i\}_{i=1}^N$, corresponding to N anticipated environments $\{S_i\}_{i=1}^N$, have already been learned. If at a particular instant the controller C_i is in use and the plant switches to an environment $S_j \neq S_i$, $1 \le j \le N$, the performance index J_j of the jth model will be the minimum of the set $\{J_i\}_{i=1}^N$ and hence the controller will switch from C_i to C_j (Fig. 2). Adaptation of $\hat{p}_j$ then takes place within the set S_j. If, however, the new plant parameter vector $p \notin \bigcup_{i=1}^N S_i$, this corresponds to an unanticipated situation. In such a case, a new identification model I_{N+1} (described by $\hat{p}_{N+1}$) and the corresponding environment S_{N+1} have to be learned on-line. This is accomplished by continuing the adaptation process until the steady state error is smaller than ε_1. Once $\hat{p}_{N+1}$ is determined, S_{N+1} becomes an "anticipated environment" for all future performance of the system. As might be expected, the transient error while learning the appropriate control for unanticipated situations may be substantially greater than ε_2.

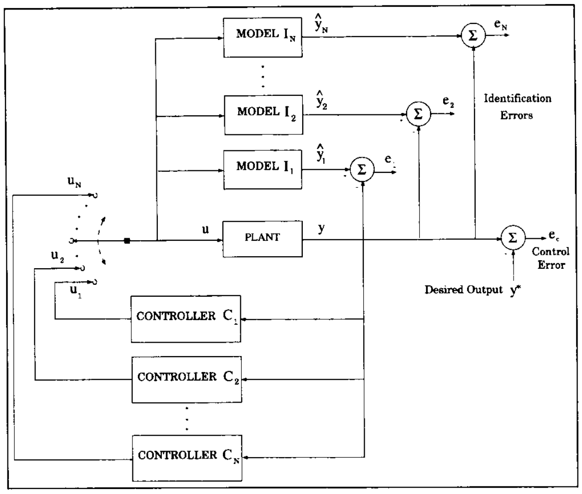


Fig. 1. Architecture of the switching and tuning system with N models and controllers.

Linear Systems

The previous section described a general methodology for the control of dynamical systems using adaptation and learning. In this section, we explain how the methodology is applied when the plant to be controlled is linear. The study of the linear case is important in many respects. The most important reason is that the stability of the resulting switching and tuning systems can be proved. This induces confidence that the same methodology when used in more realistic non-linear contexts will at least provide stability in a domain around the equilibrium state. Secondly, many control problems in industry are still being solved using linear techniques, and hence an understanding of the details involved in the linear case allows the direct extension of the method to such problems. Finally, most of the concepts and ideas comprising the general methodology find their simplest form in—and in many cases were developed in the context of—the linear case. This enables one to obtain a thorough understanding of all aspects of such concepts before testing them in the non-linear domain.

Problem Statement: The plant to be controlled is linear and of order n, with control input u and output y. Its transfer function is $W_p(s) = k_p \frac{Z_p(s)}{R_p(s)}$. The constant k_p and the $2n$ - 1 coefficients of the monic polynomials $Z_p(s)$ and $R_p(s)$ constitute the elements of the unknown parameter vector of the plant. It is assumed that the latter belongs to a closed and bounded set $S \subset \mathbb{R}^{2n}$. We further assume that every plant in S satisfies the standard assumptions of adaptive control [13, i.e., the order n, the relative degree n^*, and the sign of k_p are known, and the plant is minimum phase.

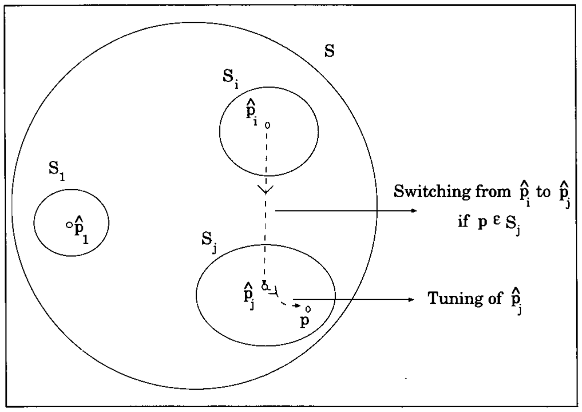


Fig. 2. Environments S_i in parameter space.

The control objective is to follow the output y_m of an asymptotically stable, linear reference model with transfer function $W_m(s)$, for any arbitrary bounded reference command input r. Our interest is in achieving this control objective for any plant in S with much better transient performance than can be achieved with conventional adaptive control. This enables the system to respond faster and more accurately when the plant parameters vary rapidly in the set S.

Structure of the Models: To construct models suitable for adaptive control, an alternate parametrization is chosen in which the new parameters p^* appear linearly [13]. In this parametrization, the plant output $y = W_p(s)\{u\}$ is given by

$$y = p^{*T}\overline{\omega}. \tag{2}$$

Here p^* is uniquely determined by the coefficients of $W_p(s)$, and the elements of the regression vector $\overline{\omega} \in \mathbb{R}^{2n}$ are generated by filtering the input u and output y [2]. In the discrete-time case, $\overline{\omega}$ would consist of the present and past values of u and y respectively. Based on equation (2), the output estimate $\hat{y}_j$ of each model I_j is chosen as

$$\hat{y}_j = \hat{p}_j^T\overline{\omega},$$

where $\hat{p}_j$ is the estimate of p^* as given by I_j. The regression vector $\overline{\omega}$ is shared by all the models (Fig. 3(a)), implying that no additional dynamics is needed for constructing the different models. The identification error of I_j, defined as $e_j = \hat{y}_j - y$, is a measure of the accuracy of the estimate $\hat{p}_j$, and is consequently used in the switching criterion to estimate the most accurate model at any instant.

Structure of the Controllers: From adaptive control theory [13], it is known that for perfect tracking $y(t) \equiv y_m(t)$ the control input is of the form $u^* = \theta^{*T}\underline{\omega}$. Here $\underline{\omega} \in \mathbb{R}^{2n}$ is another regression vector containing the reference input r as one of its elements [2]. θ^* is the ideal control parameter vector obtained by a unique transformation of p^*. The output of each controller C_j is therefore chosen as

$$u_j = \theta_j^T\underline{\omega},$$

where θ_j is C_j's estimate of θ^*, and the vector $\underline{\omega}$ is shared by all the controllers (Fig. 3(b)). For fixed (I_j, C_j), θ_j is the ideal control parameter vector corresponding to $\hat{p}_j$. If (I_j, C_j) is adaptive, $\hat{p}_j(t)$ is tuned using a suitable algorithm (e.g., gradient descent or least squares), and $\theta_j(t)$ is computed at every instant assuming that $\hat{p}_j(t) = p^*$. The switching scheme then determines which of the available candidate control inputs $\{u_1(t), u_2(t), ..., u_N(t)\}$ is to be used to control the plant.

Switching Scheme: The switching scheme consists of monitoring a performance index $J_j(t)$ based on the identification errors e_j for each model I_j, and switching to the controller corresponding to the model with the smallest value for $J_j(t)$. The rationale for this strategy is that a small identification error leads to a small tracking error [2,3]. The choice of the index is motivated by empirical observations which reveal that both instantaneous and long-term measures are needed to reliably estimate identifier accuracy [2]. A performance index incorporating this feature has the form

$$J_j(t) = \alpha e_j^2(t) + \beta\int_0^t e^{-\lambda(t-\tau)}e_j^2(\tau)d\tau. \tag{3}$$

Here $\alpha \geq 0$, $\beta > 0$, and $\lambda > 0$ are free design parameters. α and β determine the relative importance given to the instantaneous and long-term measures, and λ determines the long-term memory of the index. Simulation results in [2,3] have shown dramatic improvement in transient response with the above switching scheme, where α, β, and λ were chosen for the given set S.

To prevent arbitrarily fast switching, a hysteresis algorithm [11,12] is used in the minimization of the index, as follows: If the pair (I_j, C_j) is being used at instant t, and $J_k(t) = \min_i\{J_i(t)\}$, then (I_j, C_j) will be retained if $J_j(t) \leq J_k(t) + \delta$, and switched to (I_k, C_k) otherwise. Here $\delta > 0$ is the hysteresis constant.

Fixed Models or Adaptive Models?

In the architecture described thus far, there are many degrees of freedom available to the designer. These include whether to use fixed or adaptive models, whether to combine them in some fashion, how many of these models to use, and where to locate them in the set S. We address the first two questions in this subsection, and discuss the last two in the third subsection.

The use of only fixed models is computationally more efficient, since only the outputs of the models have to be computed. Adaptive models entail a significant overhead in computation because each parameter vector $\hat{p}_j(t)$ of dimension $2n$ has to be updated at every instant. Further, if only adaptive models are used, their parameters $\{\hat{p}_j(t)\}$ will eventually converge to a neighborhood of p^*. When p^* changes abruptly, the parameters $\{\hat{p}_j(t)\}$ have to be re-initialized to their original locations in order to quickly identify the changed value. This problem does not arise with fixed models. However, while even a single adaptive model ensures perfect accuracy asymptotically, a very large number of fixed models is generally needed to achieve the desired steady state accuracy.

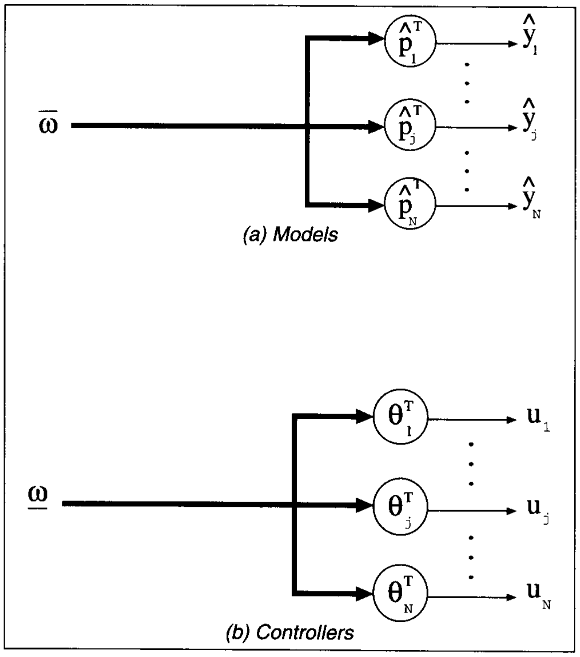


Fig. 3. Structure of the models and controllers. The output of model I_j is $\hat{y}_j = \hat{p}_j^T\overline{\omega}$, and the output of controller C_j is $u_j = \theta_j^T\underline{\omega}$. The parameter vectors $\hat{p}_j$ and θj are constant if the pair (I_j, C_j) is fixed, and are functions of time if (I_j, C_j) is adaptive.

An Efficient Combination of Fixed and Adaptive Models [3]: By using a single free running adaptive model I_A along with $N >> 1$ fixed models, small transient errors and a zero steady state error are obtained. Since fixed models are now needed only for improved transient response, a smaller number would suffice. However, the settling time of the error depends upon the convergence of the adaptive model I_A. Faster convergence can be obtained by including a re-initialized adaptive model I_R as follows: If there is a switch to a fixed model I_j at instant t, the parameter vector $\hat{p}_R(t)$ of I_R is reset to $\hat{p}_j$; I_R is left to adapt from this value until a different fixed model is chosen. A well known fact about adaptive systems is that convergence is fast when the initial parametric error is small. This is precisely what is accomplished by using switching between the fixed models to determine good initial conditions for the adaptation of I_R. With this strategy, a further reduction in the number of fixed models is obtained for the same level of performance. In the authors' opinion, it is this combination of N fixed models and two adaptive models—which uses switching to rapidly obtain a rough initial estimate, followed by tuning to improve accuracy—that yields the most efficient architecture for control in the presence of sudden changes in the environment.

Stability Results

It is well known that efficient design principles can be developed for LTI and linear adaptive control systems, only when their stability properties are well understood. We adopt the same viewpoint towards control systems based on switching and tuning, and believe that their stability must be guaranteed before design can be attempted with confidence. The stability of the multiple model switching and tuning systems described in this section have been proved, and the results are stated below. Due to lack of space, the reader is referred to the source papers [2,3] for details.

(i) If all the models are adaptive, the overall system is stable for any arbitrary switching scheme between the models, provided any fixed but arbitrarily small interval is allowed between switches. This completely decouples stability from performance and leaves the choice of the switching scheme to the designer [2].

(ii) If all the models are fixed, and the hysteresis switching scheme based on the index (3) is used, stability is assured provided at least one of the fixed models is "sufficiently close" to the plant. Precisely how close this is depends upon the plant parameters p^*, the set S, and the parameters α, β, and λ of the switching scheme [3].

(iii) With one free running adaptive model and any arbitrary (but finite) number of fixed models, the overall system is stable with zero steady state error, independent of any of the above factors. However, a sufficiently large number of fixed models are still required for satisfactory transient response [3].

(iv) With the addition of a re-initialized adaptive model, the stability properties enjoyed by the previous configuration are retained. The transient response, however, is substantially improved [3].

Comment 1: The stability results given above can be extended to the case where the sign of k_p is unknown, and only upper bounds on the order n and relative degree n^* of the plant are known.

Choice of the Models

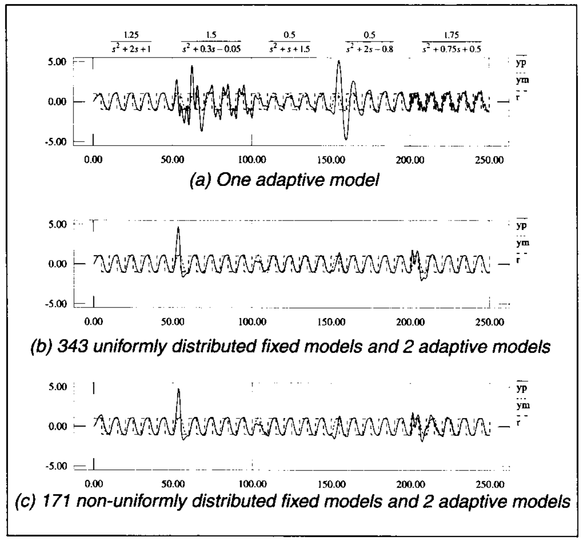
Having discussed specific architectures with fixed and adaptive models, and guaranteed their stability, we address the remaining design issues concerning the choice of the number of models, and their location in S.

These choices can be made off-line when there is sufficient prior information, i.e., the knowledge of the set S to which the plant parameters belong. A variety of means, including physical reasoning, actual measurements, and prior experience, may be used to gain knowledge of S. Ideally, there should be at least one fixed model close enough to each plant in S to meet performance specifications. However, computational considerations impose a bound N on the number of models. With N fixed, the simplest method of locating the N models is to distribute them uniformly over S. By taking advantage of the fact that the sensitivity of the transient response to mismatches in parameters varies over S, a more efficient method distributes the models non-uniformly, with more models in the sensitive regions. These ideas are illustrated in Simulation 1. A last and perhaps most efficient method of locating models in practice is by on-line learning of new models (ref. Simulation 2).

Simulation 1: The plant to be controlled is piecewise time-invariant, and is described by the equations

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -a_0(t)x_1 - a_1(t)x_2 + k_p(t)u \\ y &= x_1, \end{aligned}$$

where the unknown coefficients ($k_p(t)$, $a_1(t)$, $a_0(t)$) are piecewise constant. Equivalently, the transfer function $W_p(s) = \frac{k_p}{s^2 + a_1 s + a_0}$ remains constant over intervals. It is known that the parameters ($k_p(t)$, $a_1(t)$, $a_0(t)$) only vary within the compact set $S = [0.5,2] \times [0.25,2] \times [-1,2]$. The control objective is to follow the output y_m of a reference model $W_m(s) = \frac{1}{s^2 + 1.4s + 1}$ with input r, which is a square wave with amplitude 1, and period 10 units of time. In the simulation, the plant transfer function changes every 50 units of time, in the sequence $\frac{1.25}{s^2+2s+1} \to \frac{1.5}{s^2+0.3s-0.05} \to \frac{0.5}{s^2+s+1.5} \to \frac{0.5}{s^2+2s-0.8} \to \frac{1.75}{s^2+0.75s+0.5}$. The adaptive models were tuned using least squares with covariance resetting. The results are shown in Fig. 4.



(a) One adaptive model

(b) 343 uniformly distributed fixed models and 2 adaptive models

(c) 171 non-uniformly distributed fixed models and 2 adaptive models

Fig. 4. Response of a linear second order plant with transfer function changing every 50 units of time, using multiple models, switching, and tuning. The performance index used for switching is $J_j(t) = e_j^2(t) + 6\int_0^t e^{-0.65(t-\tau)} e_j^2(\tau)d\tau.$

The response obtained with a single adaptive model is seen from Fig. 4(a) to be unsatisfactory, especially when the system switches from a stable to an unstable mode. To improve this, the method described in the first subsection was used with a configuration of 343 fixed models distributed uniformly over S, 1 free running adaptive model and 1 re-initialized adaptive model. The resulting response shown in Fig. 4(b) is seen to be far superior, particularly the settling time after parameter changes. A non-uniform distribution of fixed models was then tried. It was found that the number of fixed models could be halved (to 171), by retaining all the models (=147) in the unstable region $[0.5,2] \times [0.25,2] \times [-1,0]$, distributing only 24 models in the stable region $[0.5,1.5] \times [0.25,1.5] \times [0,2]$, and none elsewhere. The resulting response is seen from Fig. 4(c) to be as good as that in Fig. 4(b).

In the above simulation, many fixed models are needed to assure satisfactory response, even though the plant itself switched only between five locations in S. The latter is the case in many practical situations, where the parameters are generally clustered in smaller subsets in parameter space. If these clusters are known *a priori*, the number of models can be reduced substantially. Otherwise, the clusters must be learned on-line by determining new models, as outlined in Section 2. This approach can be used when the plant parameters remain constant at each value for sufficiently long intervals to enable adaptive identification and control. By learning models in this fashion, fast and accurate response is obtained when the parameters switch to the vicinity of the newly created models at any future time. Simulation 2 describes the procedure followed.

Simulation 2: The problem considered in Simulation 1 is attempted with control using autonomous learning of new models. The same sequence of plant parameter changes is presented in two cycles, with the parameters remaining constant for 500 time units in the first cycle, and only 50 units in the second cycle. The system is started with a single adaptive model I_A. The latter tunes itself to the new plant after every change, and its identification error $e_A(t)$ becomes smaller than 0.01 before the next change. The change is automatically detected by the increase in e_A, at which point the current estimates of I_A are saved as a new fixed model for use in the switching process. A re-initialized adaptive model I_R is also included as described earlier. By the end of the first cycle, the system had learned five fixed models corresponding to each plant. When the faster second cycle started at time 2500, the learning controller successfully switched in each case to the corresponding fixed model learned earlier, and then tuned from this model by switching to I_R. The resulting response during the second cycle, shown in Fig. 5, is comparable to that in Fig. 4(c) even though only 5 fixed models were used.

Non-Linear Systems

The discussion in the previous section revealed that the general approach based on multiple models, switching, and tuning can be used effectively to improve the performance of linear time-varying systems. In this section, we attempt to extend the approach to a class of non-linear systems.

It is well known that the problem of optimally controlling a non-linear dynamical system described by known equations is a very difficult one. When the equations describing the system have unknown parameters, we have a non-linear adaptive control problem which is generally intractable. The problem becomes truly formidable when even the functions in the describing equations are unknown and change rapidly with time. Such non-linear time-varying adaptive control problems are arising with increasing frequency in today's technology and controllers have to be designed which act rapidly, accurately and in a stable fashion. It is to this class of problems that we attempt to extend the ideas described in the previous section. Needless to say, strong assumptions concerning the unknown plant have to be made to realize the advantages of the approach proposed.

The structure of the overall system, as well as the methodology of design based on multiple models, switching, and tuning, are the same for both linear and non-linear systems. The choice of the models used for identification of the plant, the structure of the controllers, and the adaptive algorithms needed for parameter

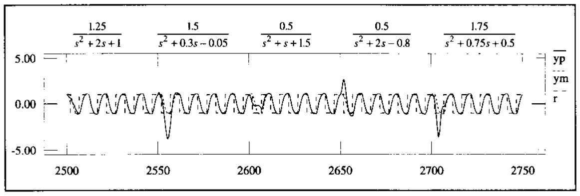


Fig. 5. Response of a linear second order plant with transfer function changing every 50 units of time, with an autonomous learning controller using five fixed models learned on-line, and two adaptive models. The performance index used for switching is $J_j(t) = e_j^2(t) + 6\int_0^t e^{-0.65(t-\tau)} e_j^2(\tau) d\tau$.

adjustment are, however, different. These are achieved by judiciously combining recent results in non-linear control theory with well-established architectures in linear adaptive control, and using the approximation capabilities of neural networks. For example, the representation of non-linear dynamical systems and their control, extensions to multivariable systems, and the use of multiple models, switching, and tuning have been carried out respectively in [5,6], [14], and [15,16]. In this section, we merely describe the highlights of these papers, suggest methods for using them effectively, and indicate some of the difficulties encountered.

Neural Networks: From a systems theoretic point of view, artificial neural networks can be considered as practically implementable parameterizations of non-linear maps from one finite dimensional space to another. Different artificial neural network architectures, motivated partly by different networks in biological systems, have been proposed in the literature. Among these are multilayer neural networks (MNN) and radial basis function networks (RBFN). We assume that the reader is familiar with the structure and parametrization of such networks. For the purpose of our paper, the following results concerning neural networks are important.

(i) Numerous authors have shown (e.g., [17]) that MNN with only one hidden layer can approximate any continuous function over a compact domain to any degree of accuracy. RBFN have also been shown to possess this property.

(ii) It was shown by Barron [18] that the L^2 approximation error of a multilayer neural network is independent of the dimension of the input space, if the function approximated satisfies a bound on the spectral norm. This makes MNN better than other approximation schemes based on linear combinations of basis functions (such as polynomials, RBFNs, and orthogonal functions), when the input space is of dimension greater than one.

(iii) Gradient methods are now well developed for adjusting the parameters of a neural network based on the output error of the network. Many other methods have also been proposed to improve the speed of convergence of the training algorithms.

From (i) it follows that if the existence of an input-output map of a dynamical system, or a non-linear map relating the measured variables to the control input, can be established using system theoretic principles, neural networks can be used to approximate them arbitrarily closely. In most problems of interest to us, the dimension of the input space is high (e.g., $2n$ if the order of the plant is n) and hence an MNN is a parsimonious representation according to (ii). The great interest in implementing neural networks in practical systems is due to (iii).

Before proceeding to control a non-linear plant using multiple models, switching, and tuning, the theoretical foundations for identifying and controlling it using a single model must be well understood. In the remainder of this subsection, the principal ideas involved are outlined.

System Representation: Let a dynamical system be represented by the state equations

$$x(k+1) = f[x(k), u(k), p]$$

$$y(k) = h[x(k), p]$$

where $x(k) \in \mathbb{R}^n$ and $u(k), y(k) \in \mathbb{R}$. Further let the function f, $h \in C^1$. We assume that f and h are unknown, and we have access only to the input u and output y. It has been shown [6] that if the linearized system around the equilibrium state is observable, an input-output representation exists which has the form

$$y(k+1) = F[y(k), y(k-1), ..., y(k-n+1), u(k), u(k-1), ..., u(k-n+1)], \tag{4}$$

i.e., a function F exists which maps $y(k)$ and $u(k)$, and their $n-1$ past values, into $y(k+1)$. In view of this, a neural network N_F can be trained, with $y(k), y(k-1), ..., y(k-n+1), u(k), u(k-1), ..., u(k-n+1)$ as the inputs, to approximate F over the domain of interest.

Relative Degree d: If the non-linear system has a relative degree (or delay) d so that $u(k)$ affects the output at time $k+d$ but not earlier, it can be shown that an input-output representation of the system exists which has the form

$$y(k+d) = \bar{F}[y(k), y(k-1), ..., y(k-n+1), u(k), ..., u(k-n+1)],$$

or equivalently

$$y(k) = \bar{F}[y(k-d), y(k-d-1), ..., y(k-n-d+1), u(k-d), ..., u(k-n-d+1)].$$

In this case $\bar{F}$ can be approximated by a neural network $N_{\bar{F}}$ to yield an estimate $\hat{y}(k)$ of the output $y(k)$. The error $e(k) \stackrel{\Delta}{=} y(k) - \hat{y}(k)$ is used in training the network.

Multivariable Systems [14]: The same concepts described thus far can be extended to represent and control multivariable non-linear systems. Defining $Y(k+d) \stackrel{\Delta}{=} [y_1(k+d_1), y_2(k+d_2), ..., y_n(k+d_n)]^T$ where $y(k) \stackrel{\Delta}{=} [y_1(k), ..., y_n(k)]^T$ is the output vector of the system, it can be shown that

$$Y(k+d) = \bar{F}[y(k), y(k-1), ..., y(k-n+1), u(k), u(k-1), ..., u(k-n+1)],$$

for some $\bar{F} : \mathbb{R}^{2n} \to \mathbb{R}^n$. In this case $\bar{F}$ is approximated by a neural network.

System Representation with Disturbance [19]: Let $v(k)$ be an additive disturbance at the input and let $v(k)$ be an output of an unforced dynamical system described by the equation

$$w(k+1) = f_D(w(k)), \quad w(0) = w_0$$

$$v(k) = h_D(w(k))$$

where $f_D : \mathbb{R}^p \to \mathbb{R}^p$. In this case the overall system can be described by a representation of the form

$$y(k+1) = F_D[y(k), y(k-1), ..., y(k-n-p+1), u(k), u(k-1), ..., u(k-n-p+1)],$$

i.e., the dimension of the system is increased from $2n$ to $2(n+p)$.

Adaptive Control of Non-Linear Systems

The structure of an indirect adaptive controller using a single identification model is shown in Fig. 6. This structure was originally proposed in [4] and is reproduced here for easy reference.

In the figure, TDL represents a tapped delay line which provides the present and past values of $u(k)$ and $y(k)$ as inputs to the neural networks for identification and control. The problem now is to demonstrate that the output of the unknown plant will track a given output. This has been demonstrated in [5]. If the plant has a delay d, a function $g : \mathbb{R}^{2n} \to \mathbb{R}$ exists such that the input

$$u(k) = g[y(k), y(k-1), ..., y(k-n+1), y^*(k+d), u(k-1), ..., u(k-n+1)]$$

results in $y(k)$ following a desired output $y^*(k)$ after exactly d steps. If $y^*(k+d)$ is known at time k, $g(\ ...)$ can be approximated by a neural network N_g with $2n$ inputs and one output.

Comment 2: In model reference adaptive control, $y^*(k)$ is given as the output of a reference model with a reference input $r(k)$. If the reference model is a pure delay of d units, it follows that $y^*(k+d) = r(k)$ which is known at time k. Hence, in the control problem, the inputs to N_g are the same as those to N_F in equation (4), with $u(k)$ replaced by $r(k)$.

Comment 3: Unlike the identification model in which the parameters are adjusted using backpropagation, the parameters of the controller have to be adjusted within a feedback loop. The computation of a gradient in this case is considerably more involved and computationally intensive.

Approximate Identification Models: From Comment 3 it is clear that even if an exact model of the plant is available, practical considerations may dictate that approximate static backpropagation methods be used to update the control parameters on-line. An alternate approach is to use approximate models which simplify the computation of the control input. Such a model has been reported in [20] and is described by the equation

$$y(k+1) = F_I[y(k), y(k-1), ..., y(k-n+1), u(k-1), ..., u(k-n+1)] + G_I[y(k), y(k-1), ..., y(k-n+1), u(k-1), ..., u(k-n+1)]\, u(k).$$

Once F_I and G_I are approximated as described earlier using neural networks N_{F_I} and N_{G_I}, the control can be computed at every instant using the formula

$$u(k) = \frac{y^*(k+d) - N_{F_I}[y(k), y(k-1), ..., y(k-n+1), u(k-1), ..., u(k-n+1)]}{N_{G_I}[y(k), y(k-1), ..., y(k-n+1), u(k-1), ..., u(k-n+1)]}.$$

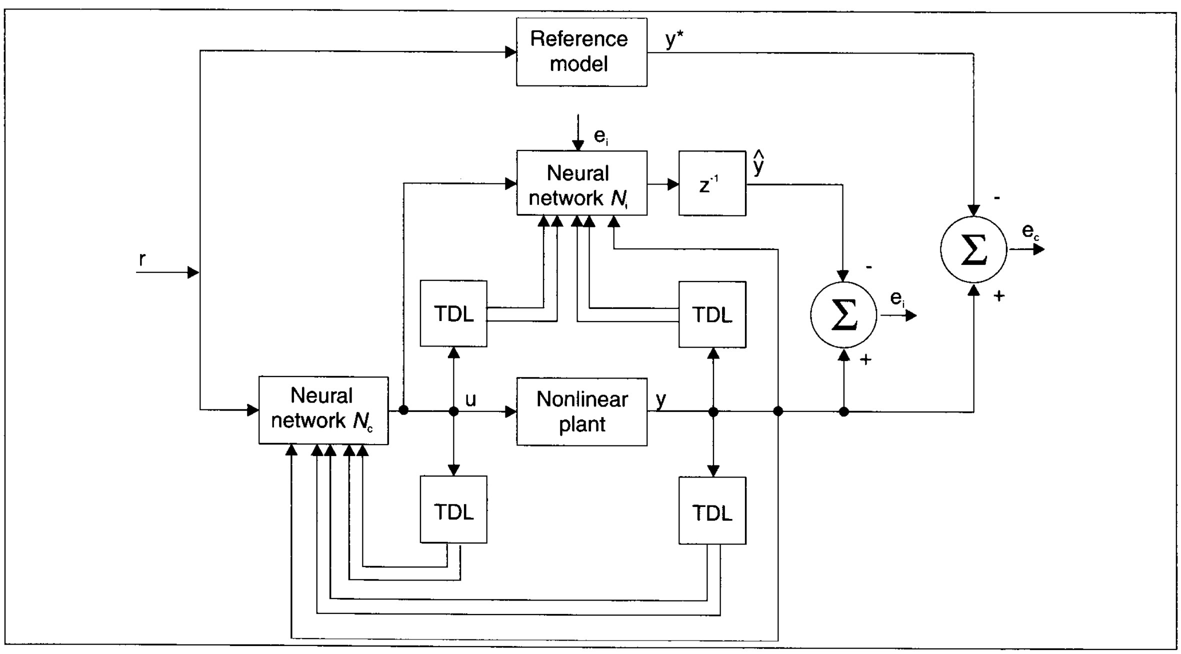


Fig. 6. A general architecture for indirect adaptive control using neural networks. N_i and N_c represent the neural networks used for identification and control respectively.

Numerous simulations have been carried out on a wide spectrum of non-linear problems and the approximate model performed at least as well as the exact model in most cases.

Adaptive Control of Non-Linear Systems Using Multiple Models

The method described in the previous subsection for adaptively controlling a non-linear system using a single model can be extended in a straightforward manner to the case where multiple models are present. Many of the comments made in the linear case are also applicable here. However, there are many features that are peculiar to identification and control using neural networks. These are best illustrated by considering specific examples. For further details concerning such studies the reader is referred to [15] and [16].

A two input, two output system of third order is described by the equations

$$
\begin{aligned}
&x_1(k+1) = \\
&0.9x_1(k)\sin(x_1(k)) + \left(2 + \frac{1.5x_1(k)u_1(k)}{1+x_1^2(k)u_1^2(k)}\right)u_1(k) + \left(x_1(k) + 2\frac{x_1(k)}{1+x_1^2(k)}\right)u_2(k) \\
&x_2(k+1) = x_3(k)(1+\sin(4x_3(k))) + \frac{x_3(k)}{1+x_3^2(k)} \\
&x_3(k+1) = (3+\sin(2x_1(k)))u_2(k) \\
&y_1(k) = x_1(k), \qquad y_2(k) = x_2(k).
\end{aligned}
\tag{5}
$$

It is seen that output $y_1(k)$ has a relative degree of one while $y_2(k)$ has a relative degree two (the relative degree of an output is the minimum relative degree with respect to each of the inputs). The objective is to determine control inputs $u_1(k)$ and $u_2(k)$ so that the outputs $y_1(k)$ and $y_2(k)$ follow respectively the signals

$$
r_1(k-1) = y_1^*(k) = 0.5\sin\left(\frac{2k\pi}{50}\right) + 0.5\sin\left(\frac{2k\pi}{25}\right)
$$

$$
r_2(k-2) = y_2^*(k) = 0.25\sin\left(\frac{2k\pi}{50}\right) + 0.75\sin\left(\frac{2k\pi}{25}\right)
$$

where $r_i(k)$ are the reference inputs and $y_i^*(k)$ are the desired outputs (i=1,2). The approximate model described earlier was used to identify the system as well as to generate the control inputs. After 500,000 steps, during which period backpropagation was used, the parameters of the neural networks converged. The inputs $u_1(k)$ and $u_2(k)$ and the corresponding control output errors $e_1(k)$ and $e_2(k)$ are shown in Fig. 7.

Comment 4: The process of convergence in neural networks is generally slow. It is particularly the case when backpropagation is used. By using more sophisticated methods the speed of convergence can be improved, but generally only by an order of magnitude. It must therefore be assumed that the plant remains in the same environment, denoted as E_1, for this period (in the linear case, this was denoted as $\hat{p}_1$). In more realistic cases, the plant encounters the same environment repeatedly and the learning process is continued.

Comment 5: Unlike the linear case, the methods described can be applied only to systems whose outputs are bounded. It is therefore assumed that the system has been stabilized using linear controllers. Consequently, the objective in using the above method is to improve the performance.

External Disturbance (E_2): We next consider the case where an additive external disturbance $d_1(k) = 0.1 + 0.2\sin\left(\frac{2k\pi}{5}\right)$ is present at the input u_1. The corresponding output errors are seen to be large (greater than a threshold of 0.3) and it is classified as belonging to a new environment E_2. The neural network N_1 corresponding to E_1 is stored and a second network is initiated at N_1 and adapted as before, to obtain the network N_2 which eliminates the disturbance at the output. The effect of using network N_1 on environment E_2, and N_2 on environment E_1, are shown in Fig. 8. The large errors obtained in the two cases shows that the identification models can be used to detect whether or not the disturbance is present.

Parameter Change (E_3): The same process is repeated when the parameters of the system change so that the terms 0.9 $x_1(k)$ sin ($x_1(k)$) and $\left(2 + \frac{1.5x_1(k)u_1(k)}{1+x_1^2(k)u_1^2(k)}\right)$ in Equation (5) change to $0.5x_1(k)$ sin ($x_1(k)$) and $\left(4 + \frac{1.5x_1(k)u_1(k)}{1+x_1^2(k)u_1^2(k)}\right)$, respectively. The output errors of both the models stored earlier are affected, with the error of N_2 being greater than that of N_1. Hence, adaptation commences at values given by N_1, and a network N_3 is determined.

Disturbance and Parameter Change (E_4): In the same manner as described earlier a neural network N_4 is initiated and adapted when the disturbance and parameter change occur simultaneously, and the final values of the parameters of the network N_4 are stored. It is clear that every time the environment changes, it has to be detected using the output errors of all the stored models, and a new network must be initiated. A reasonable choice for initiating the new network is from the network corresponding to $\min_i\{J_i\}$ where J_i is a performance index based on the identification error e_i of the ith model (e.g., the index (3)).

The four network models N_1, N_2, N_3, N_4 discussed in this simple example can be considered as the networks resulting from the learning experience of the system. We now consider the case where three of the four environments occur in quick succession, i.e., the environments E_1, E_2, and E_4 are active over intervals [0,100), [100,200) and [200,300) respectively. In this case, the change in environment is detected almost instantaneously by one of the models and the corresponding control is used. Both the identifier and controller chosen are then tuned on-line. The effect of using a single identification model and controller, as well as the reduced error obtained using multiple models, switching, and tuning, are shown in Figure 9.

Comment 6: Many of the features common to more complex problems are clear from the simple example considered above.

(i) Storing the neural networks (i.e., the parameter values) corresponding to identification models in different environments permits the system to respond rapidly, when any one of them occurs at a later time.

(ii) Due to the high dimensionality of the parameter space, identification models cannot, in general, be chosen off-line as in the linear case, but have to be learned as the system is in operation. However, in critical cases (such as reconfigurable control of aircraft), the ability to detect what failure has occurred and take effective action may be essential for the survival of the system. Such cases preclude the use of on-line learning and the

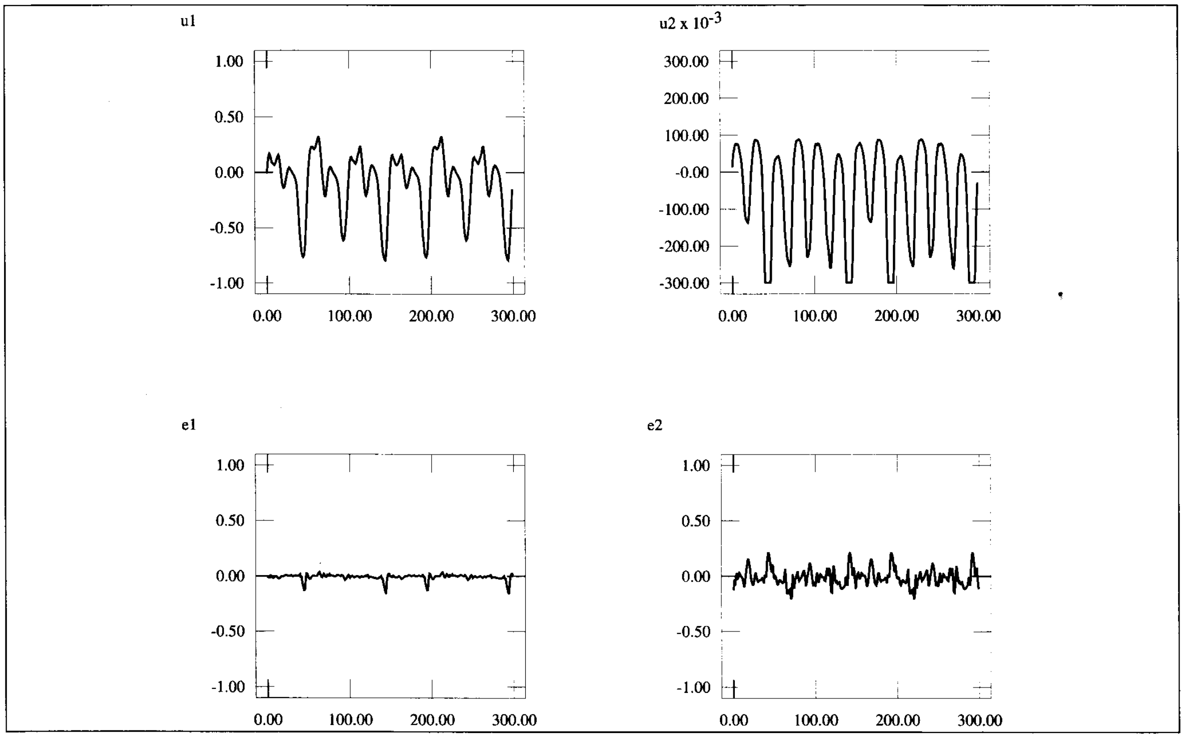


Fig. 7. Adaptive control of a two input two output non-linear plant using neural networks.

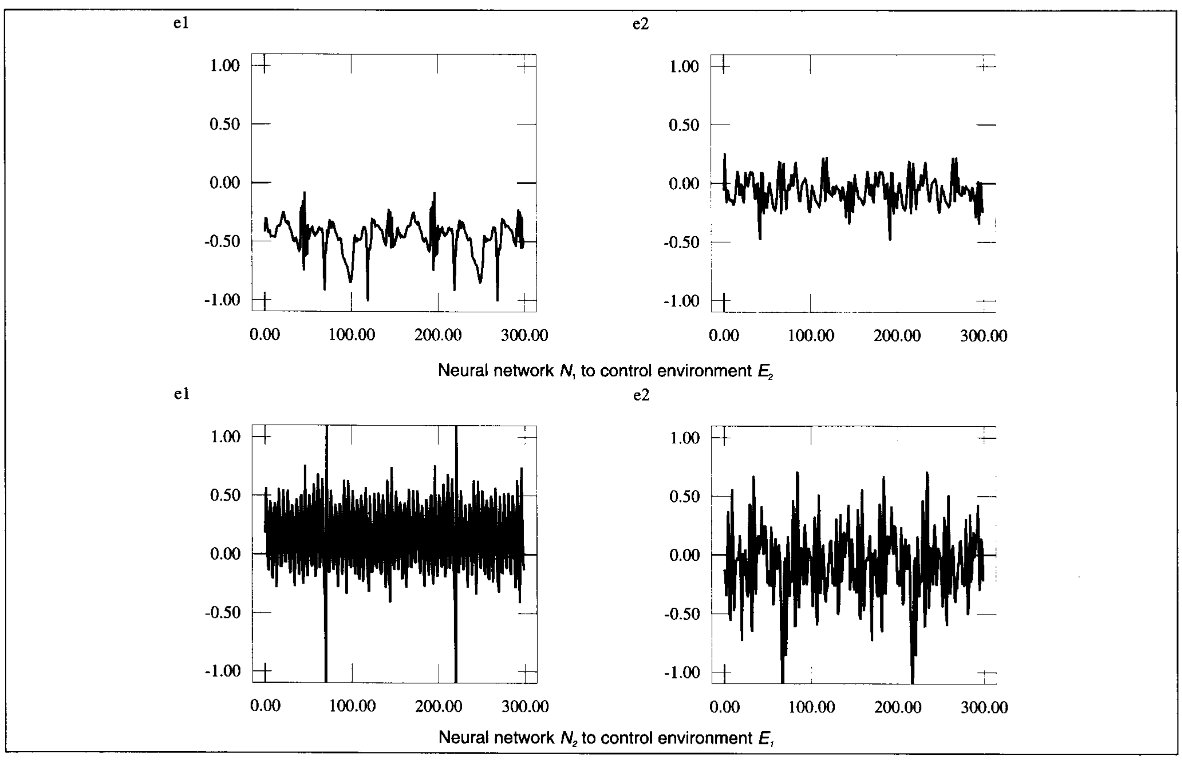


Fig. 8. The use of identification errors to detect the environment.

training and storing of networks may have to be carried out using computer simulations based on prior information concerning new environments.

(iii) The number of situations, which demand distinct control actions for fast and accurate response, can grow very rapidly, requiring hundreds or even thousands of networks to be stored in complex systems. In such cases, neural networks with a hierarchical structure may become necessary for the classification of environments. For example, at the first level, the outputs of the network may be used to detect whether a change has occurred, and if so whether it is due to noise, external disturbances, parameter variations, or changes in the reference inputs. The lower levels can then be used to categorize the classes more precisely.

Very little is currently known about the creation and modification of multiple models for non-linear systems, but the preliminary results for simple cases are very encouraging.

Robotic Manipulator Control

The methodology developed in the previous sections is currently being tried out in different industrial problems. In this section we describe some of our efforts to test the methods proposed on rigid robotic manipulators in the laboratory. As might be expected, while the general approach is the same as that described in the previous sections, appropriate modifications had to be made both in the theory as well as in the practical implementation of the controller. We believe that this will be typical of other applications also, where the methods have to be tailored to the specific problem.

Tracking control of robotic manipulators which have non-linear dynamics is a problem which has attracted a lot of attention in recent years. The changes in the inertial dynamics of the robot during the execution of various tasks make the control problem a difficult one. However, the fact that the dynamic model can be parametrized linearly in terms of the unknown (or partially known) inertial parameters has made the problem tractable in the past. Numerous researchers have proposed globally stable adaptive control algorithms and demonstrated experimentally that such methods result in performance that is superior to those achieved by their non-adaptive counterparts [21]. However, as mentioned earlier in this paper, adaptive methods while resulting in asymptotic convergence, may have large transient errors. It is for reducing the transient errors in practical applications that the multiple model approach is attractive.

In what follows, the reader is assumed to be familiar with adaptive control of robotic manipulators and only a brief discussion of the modified adaptive algorithm is given for a generic n-link robot arm. In particular, only the modifications that have to be made in existing methods are described. Following this, results obtained by implementing the algorithm on a SCARA type manipulator (UCB/NSK Direct Drive Arm in the Robotics Laboratory of the Mechanical Engineering Department at University of California, Berkeley) are presented. For a detailed mathematical treatment of the algorithm, the reader is referred to [22,23].

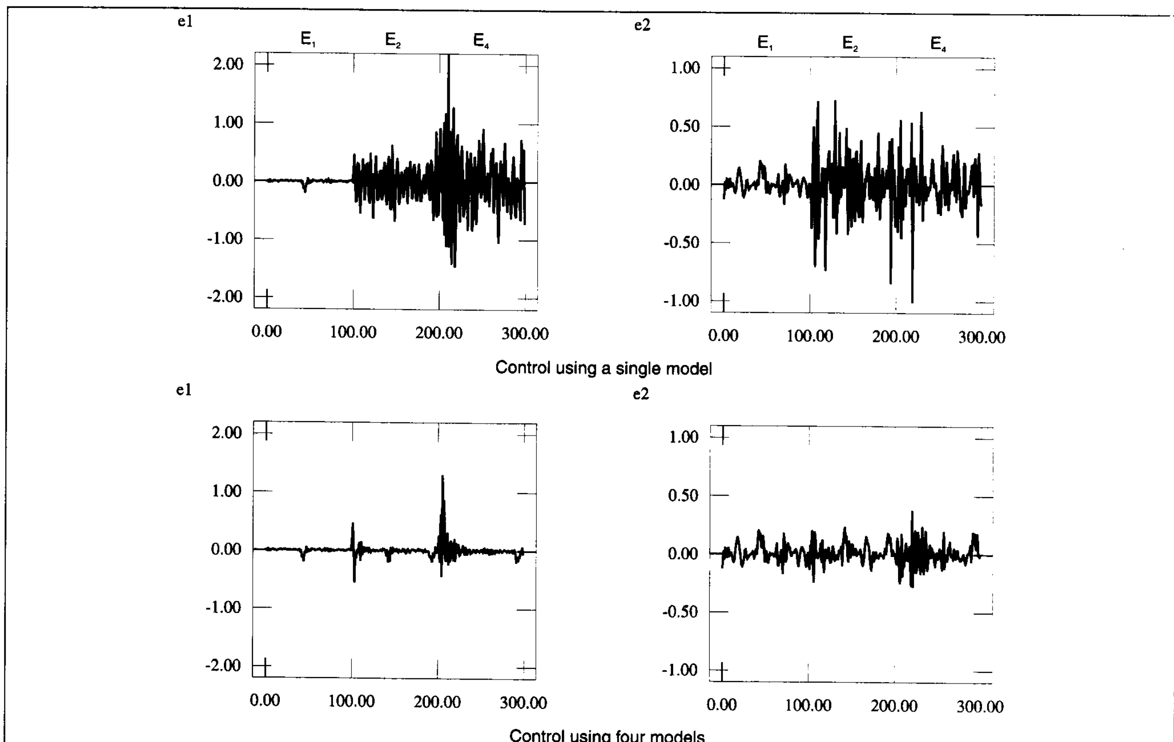


Fig. 9. Comparison of single and multiple model adaptive control.

Statement of the Problem

An n-link rigid robotic manipulator is described by the equation

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q)$$

where $\mathbf{q} \in R^n$ is the joint position vector, $\tau \in R^n$ is the joint torque vector, $M(\mathbf{q}) \in R^{n \times n}$ is the positive definite inertia matrix, $C(q, \dot{q}) \in R^{n \times n}$ is a matrix representing the Coriolis and centrifugal effects (cross dynamic terms) and $\mathbf{g}(\mathbf{q}) \in R^n$ represents the gravitational forces. Given a bounded prespecified trajectory $\mathbf{q}_d$ along with its smooth first and second derivatives $\dot{q}_d$ and $\ddot{q}_d$, the task is to design a controller such that the desired trajectory is followed as closely as possible with good transient characteristics.

The indirect adaptive control approach which has been proposed in the literature, using a single identification model, may be described as follows. The parameter vector $\mathbf{p}$ of the manipulator is estimated and in turn used to generate the joint torque vector. The identification model is of the input prediction type and is described by the equation

$$\hat{\tau} = Y(q, \dot{q}, \ddot{q})\,\hat{p}, \tag{7}$$

where $\hat{\tau}$ is the prediction of the torque input τ, $\hat{p}$ is the estimate of $\mathbf{p}$, and $Y(q, \dot{q}, \ddot{q})$ is the non-linear regressor matrix. The acceleration information is assumed to be available. The estimate $\hat{p}$ is updated using, say, a gradient algorithm of the form

$$\dot{\hat{p}} = -\Gamma Y^T(q, \dot{q}, \ddot{q})\tilde{\tau}, \tag{8}$$

where

$$\tilde{\tau} \overset{\Delta}{=} \tau - \hat{\tau} = Y(q, \dot{q}, \ddot{q})p - Y(q, \dot{q}, \ddot{q})\hat{p} = Y(q, \dot{q}, \ddot{q})\tilde{p}$$

is the torque prediction error, Γ is a positive definite adaptation gain matrix and $\tilde{p} = p - \hat{p}$ represents the parameter error vector.

Adaptive Control Using Multiple Models

As mentioned earlier, the overall structure of the robotic control system, as well as the schemes used for switching and tuning, are the same as in the linear case, with the difference being in the specific structure of the models and controllers. Fig. 10 shows the resulting architecture. The models $\{I_j\}_{j=1}^{N}$ have identical structure in that all of them share the same regressor matrix $Y(q, \dot{q}, \ddot{q})$, while the parameter vectors $\hat{p}_j$ (j=1, ..., N) have different initializations $\hat{p}_j(0)$ chosen from a given parameter set, as in the linear case. The selection of these initial estimates (i.e., learning) can be done off-line if approximate ranges of the masses of different loads are available. An on-line learning process is also possible as discussed in the section on linear systems. Since all the parameter vectors $\hat{p}_j$ are updated in the same manner as in the single model case, the equations describing the evolution of $\tilde{p}_j$ are exactly the same as in equations (7) and (8) with $\tilde{p}$ and $\tilde{\tau}$ replaced by $\hat{p}_j$ and $\tilde{\tau}_j$, respectively.

The N controllers $\{C_j\}_{j=1}^{N}$ also have identical structures, with torque input τ_j corresponding to C_j given by

$$\tau_j = \hat{M}_j(q)(\ddot{q}_d + K_v\dot{e} + K_p e) + \hat{C}_j(q, \dot{q})\dot{q} + \hat{g}_j(q),$$

where $\hat{M}_j$, $\hat{C}_j$, and $\hat{g}_j$ are estimates of M, C, and $\mathbf{g}$ respectively, evaluated using the estimate $\hat{p}_j$, $e \overset{\Delta}{=} q_d - q$ is the tracking error vector, and $\dot{e} \overset{\Delta}{=} \dot{q}_d - \dot{q}$.

As discussed in the second section, at any given instant all the identification errors of the N models are available, but only one of the torque vectors τ_j can be chosen as the input to the manipulator. This is done exactly as in the third section by monitoring a performance index $J_j(t)$ (j=1, 2, ..., N) of the form given in Equation (3), and choosing the torque τ_j corresponding to the minimum as the torque input τ to the manipulator. The resulting switching and tuning system was shown to be stable in [23].

The algorithm outlined above can also be implemented avoiding the use of the acceleration vector, by means of the filtering method proposed in [24]. In this case each identification model I_j is of the form

$$\hat{\tau}_j = Y_f(q, \dot{q})\hat{p}_j$$

where $Y_f(q, \dot{q}) = H(s)Y(q, \dot{q}, \ddot{q})$ is the filtered regressor matrix which is shared by all the models, with the parameter vector $\hat{p}_j$ having different initializations, and $H(s) = \dfrac{a}{s+a}$ is a first order filter with $a > 0$. The final torque vector τ can be generated in the same manner as before. The overall control scheme was shown to be stable in this case as well [23]. Extensive simulation tests for different scenarios were also carried out and reported in [23].

Experimental Results

The method described in the preceding subsection was implemented on the two link UCB/NSK Direct Drive arm shown in Fig. 11. For comparison purposes, a passivity based direct adaptive control algorithm utilizing only one model was also implemented with the same PD gains. The sampling time was set to 2 ms for the servo loop and the adaptation dynamics. A 7th order polynomial (with 0 initial and 135 degree end point) was used for both links as the desired trajectory. Eight different models were chosen arbitrarily around the initial estimates of the nominal parameter vector. The design parameters in the performance index $J_j(t)$ were chosen as $\alpha = 0.4$, $\beta = 0.8$, and $\lambda = 0$. The tracking errors of link 1 for the multiple model based controller and the single model adaptive controller are shown in Fig. 12(a). Note that there is a significant performance improvement with the use of multiple models, switching, and tuning. The switchings between these 8 models were also monitored and plotted in Fig. 12(b).

It was observed during the experimental tests that the tracking performance and the number of switchings were directly affected by the choice of the design parameters α, β, and λ in (3). The performance can be improved by experimentally tuning these parameters. Other experiments with different values for these

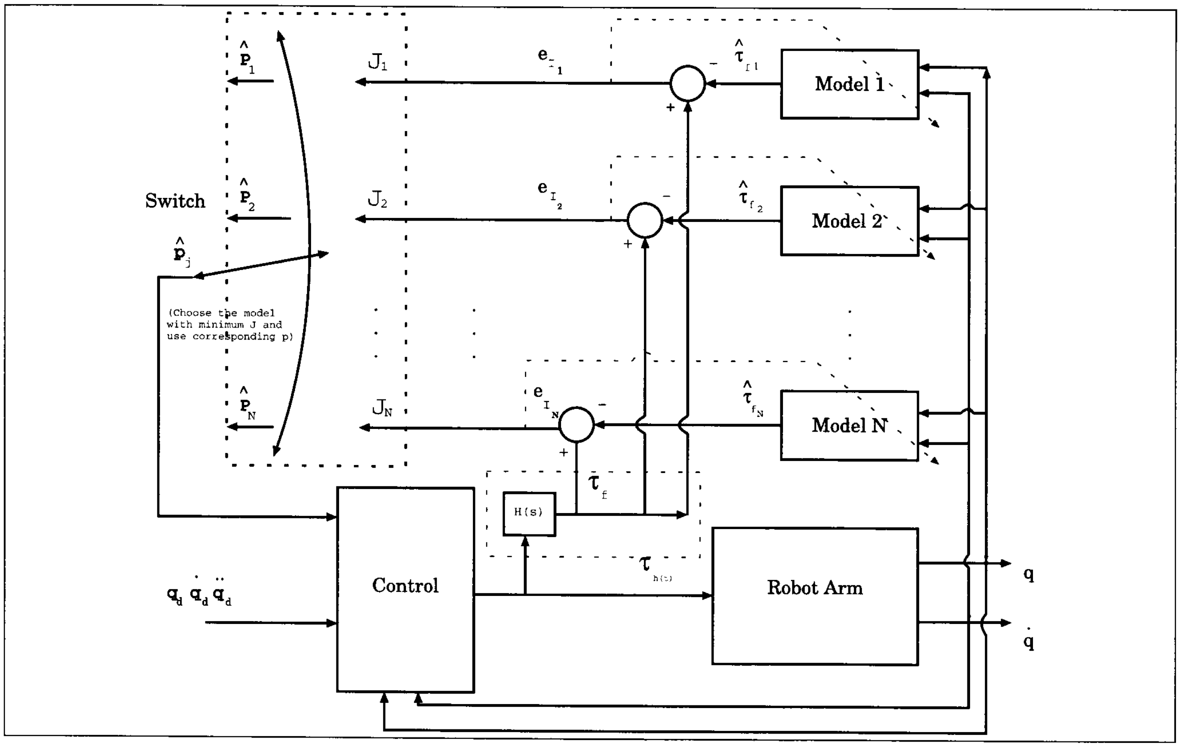


Fig. 10. Architecture of the control system for robotic manipulators with N models and controllers.

parameters and with the arm carrying a load were also carried out and will be reported elsewhere. An extension of the above algorithm based on a combined direct and indirect approach is also currently being investigated.

Conclusions

In this article a general methodology for the improvement of performance of dynamical systems operating in rapidly varying environments is introduced. Both linear and non-linear plants are considered and an indirect approach based on multiple models is used for control. When an environment is encountered for the first time, an adaptive approach is used to identify the system as well as to determine the control parameters. In such a case, large transient errors cannot be avoided. However, if the parameters of the identifiers and controllers are stored, the former can be used to recognize the environment if it occurs at a later stage, and the latter can be used to initiate the appropriate control immediately. Even assuming that the new environment is not identical to one which occurred earlier, but is close to it in some sense, tuning can be used to reduce the error.

A theoretical framework is provided by the linear case for which stability results have been derived. This provides the motivation for the non-linear case. Further, the location of the models in the linear case can be determined off-line using computer simulations. For specific applications, further modifications in their location can be made by on-line learning. The extension of the above approach to non-linear problems is conceptually straightforward, but the problems of identification and control are substantially more difficult. The methods developed for the linear case are then applied to a robot manipulator control problem. It is demonstrated experimentally that the use of multiple models reduces the transient error substantially in such a system.

The stability results derived in the linear case permit the proposed approach to be used even when the plant to be controlled is unstable. In contrast to this, the unknown plant is assumed to be bounded-input bounded-output stable in the non-linear case and the objective is primarily to improve performance. This, however, does not preclude the possibility of instability when switching and tuning are used. Work is currently in progress to determine sufficient conditions for the stability of such non-linear switching and tuning systems.

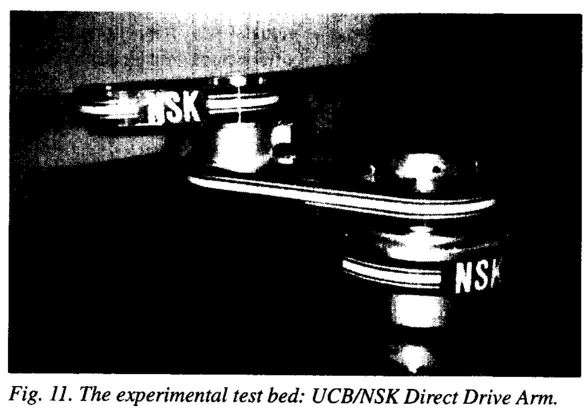
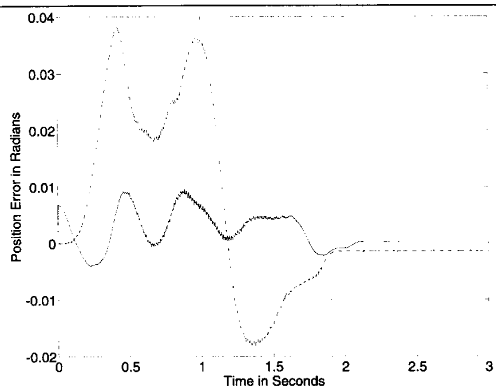
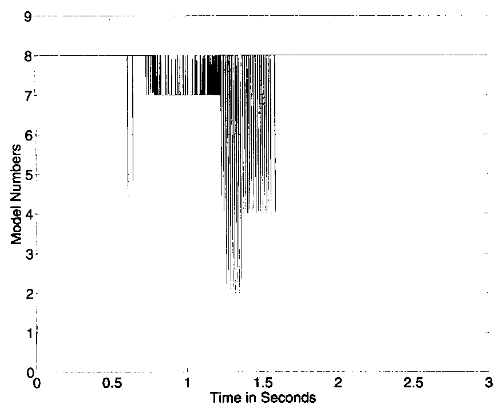


Fig. 11. The experimental test bed: UCB/NSK Direct Drive Arm.



(a) Tracking errors for link 1



(b) Switchings between the 8 models

Fig. 12. Experimental results for the control of the UCB/NSK Direct Drive Arm using eight adaptive models, switching, and tuning. (a) Multiple Model Based Adaptive Control: ____, Direct Adaptive Control with One Model: - —. (b) Switchings between eight models.

The creation, modification and pruning of models, the acquisition of their sensitivity characteristics, and the generation of equivalence classes of models have to be learned on-line if the systems are to be truly autonomous. All these raise theoretical questions for which answers are not known at present.

In the authors' opinion, the greatest payoff of the approach described will be in the control of complex non-linear systems in the presence of uncertainty. The simple examples and applications described in this paper epitomize a new and interesting class of problems which have practical utility, are analytically challenging but potentially tractable, and may therefore prove compelling to the control community.

Acknowledgment

The first author would like to thank his former graduate students Snehasis Mukhopadhyay and Asriel Levin who collaborated with him on the work reported in this article on non-linear systems, and Daniel Koditschek and Richard Wheeler for many interesting discussions on the control of complex systems. He would also like to thank his graduate student, Sai-Ming Li, who carried out the simulation results using neural networks.

The third author is grateful to Masayoshi Tomizuka for inviting him to the Mechanical Engineering Department at the University of California at Berkeley, to carry out the experimental work reported in this article.

References

[1] K.S. Narendra and J. Balakrishnan, "Performance Improvement in Adaptive Control Systems Using Multiple Models and Switching," *Proc. Seventh Yale Workshop on Adaptive and Learning Systems*, pp. 27-33, Center for Systems Science, Yale University, New Haven, CT, May 1992.

[2] K.S. Narendra and J. Balakrishnan, "Improving Transient Response of Adaptive Control Systems Using Multiple Models and Switching," *IEEE Transactions on Automatic Control*, AC-39(9): 1861-1866, September 1994. Also appeared in *Proc. IEEE 32nd CDC*, San Antonio, TX, December, 1993.

[3] K.S. Narendra and J. Balakrishnan, "Intelligent Control Using Fixed and Adaptive Models," Technical Report 9410, Center for Systems Science, Yale University, New Haven, CT, October 1994. Also appeared in condensed form in *Proc. IEEE 33rd CDC*, Lake Buena Vista, FL, December 1994.

[4] K.S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions Neural Networks*, 1(1):4-26, March 1990.

[5] A.U. Levin and K.S. Narendra, "Control of Nonlinear Dynamical Systems Using Neural Networks: Controllability and Stabilization," *IEEE Transactions on Neural Networks*, 4(2): 192-206, March 1993.

[6] A.U. Levin and K.S. Narendra, "Control of Nonlinear Dynamical Systems Using Neural Networks, Part II: Observability, identification and Control," *IEEE Transactions on Neural Networks*, 1995. To appear.

[7] K.S. Narendra and S. Mukhopadhyay, "Intelligent Control Using Neural Networks," *IEEE Control Systems Magazine*, 12(2): 11-19, April 1992. Also appeared in *Proc. ACC*, Boston, MA, 1991.

[8] D.G. Lainiotis, "Partitioning: A Unifying Framework for Adaptive Systems, I: Estimation, II: Control," *Proceedings of the IEEE*, 64: 1126-1143, 1182-1197, August 1976.

[9] M. Athans et al., "The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control (MMAC) Method—Part I: Equilibrium Flight," *IEEE Transactions on Automatic Control*, AC-22: 768-780, 1977.

[10] B. Maartensson, *Adaptive Stabilization*, Ph.D. thesis, Lund Institute of Technology, Lund, Sweden, 1986.

[11] R.H. Middleton, G.C. Goodwin, D.J. Hill, and D.Q. Mayne, "Design Issues in Adaptive Control," *IEEE Transactions on Automatic Control*, AC-33(1): 50-58, January 1988.

[12] A.S. Morse, D.Q. Mayne, and G.C. Goodwin, "Applications of Hysteresis Switching in Parameter Adaptive Control," *IEEE Transactions on Automatic Control*, AC-37(9): 1343-1354, September 1992.

[13] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*, Prentice-Hall Inc., Englewood Cliffs, NJ 07632, 1989.

[14] K.S. Narendra and S. Mukhopadhyay, "Adaptive Control of Nonlinear Multivariable Systems Using Neural Networks," *Neural Networks*, 7(5): 737-752, 1994.

[15] K.S. Narendra and S. Mukhopadhyay, "Intelligent Control Using Neural Networks," in M.M. Gupta and N.K. Sinha, eds., *Intelligent Control*, IEEE Press, 1995. To be published.

[16] K.S. Narendra and S.-M. Li, "Neural Networks in Control Systems," in P. Smolensky et al., eds., *Mathematical Perspectives on Neural Networks*, Lawrence Erlbaum, 1995. To be published.

[17] K. Hornik, M. Stinchcombe, and H. White, "Universal Approximation of an Unknown Mapping and its Derivatives Using Multilayer Feedforward Networks," *Neural Networks*, 3:551—560, 1990.

[18] A.R. Barron, "Universal Approximation Bounds for Superpositions of a Sigmoidal Function," *IEEE Transactions on Information Theory*, 39(3): 930-945, May 1993.

[19] S. Mukhopadhyay and K.S. Narendra, "Disturbance Rejection in Nonlinear Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, 4(1): 63-72, January 1993.

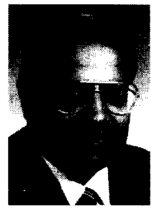
[20] K.S. Narendra and S. Mukhopadhyay, "Approximate Models for the Identification and Control of Nonlinear Dynamical Systems," technical report, Center for Systems Science, Yale University, New Haven, CT. Under preparation.

[21] R. Ortega and M.W. Spong, "Adaptive Motion Control of Rigid Robots," in *Proc. IEEE 27th CDC*, Austin, TX, December 1988.

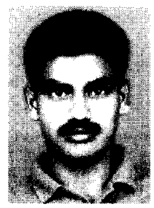
[22] K. Ciliz and K.S. Narendra, "Multiple Model Based Adaptive Control of Robotic Manipulators," in *Proc. IEEE 33rd CDC*, Lake Buena Vista, FL, December 1994.

[23] K. Ciliz and K.S. Narendra, "Intelligent Control of Robotic Manipulators using Multiple Models and Switching," Technical Report 9313, Center For Systems Science, Yale University, New Haven, 1993.

[24] R.H. Middleton and G.C. Goodwin, "Adaptive Computed Torque Control for Rigid Link Manipulators," *Systems and Control Letters*, 10:9-16, 1988.



Kumpati S. Narendra received the Ph.D. degree from Harvard University in 1959. He is currently professor of electrical engineering and director of the Center for Systems Science at Yale University. Narendra is the author of three books and more than 150 technical papers in the areas of stability theory, learning automata, adaptive systems, and neural networks. He has also edited four volumes in these areas. Narendra serves on the scientific advisory boards of several national and international institutions and is the recipient of numerous awards from IEEE, AACC, and INNS. He will deliver the Bode Lecture at the 1995 CDC.



Jeyendran Balakrishnan received his BE degree in electronics and communication engineering from the College of Engineering, Madras, India, in 1987, and the ME degree in electrical communication engineering from the Indian Institute of Science, Bangalore, India, in 1989. In the first half of 1989, he worked on digital beamforming at the Central Research Laboratory, Bangalore, India. He is currently completing the requirements for the Ph.D. degree in electrical engineering at Yale University. His research interests are in the general area of control of complex systems, and include adaptive control and signal processing, switching systems, and neural networks.

M. Kemal Ciliz received the B.S. degree in 1985 from Middle East Technical University, Turkey, and the M.S. and Ph.D. degrees in 1987 and 1990, respectively, from Syracuse University, NY, all in electrical engineering. Between 1988 and 1989 he worked for Borg Warner Corp., Ithaca, NY, as a control, systems engineer. Since 1991 he has been with Bogazici University, Turkey, where he is currently an associate professor. Ciliz is a member of Tau Beta Pi and Eta Kappa Nu. He was a visiting research fellow at Yale University during 1993 and 1994. His research interests include neural networks, non-linear systems, and dynamics of economic systems.

Sampled Data

When the Truth Just Won't Do

You're called upon for an opinion of a friend who is extremely lazy. You don't want to lie, but you also don't want to risk losing even a lazy friend. Try this line: "In my opinion," you say as sincerely as you can manage, "you will be very fortunate to get this person to work for you."

If you're frustrated about this occupational hazard for teacher—having to write letters of recommendation for people with dubious qualifications—use some of these techniques for collecting an arsenal of statements that can be read two ways. It's called the Lexicon of Inconspicuously Ambiguous Recommendations—LIAR.

LIAR may be used to offer a negative opinion of the personal qualities, work habits, or motivation of the candidate, while allowing the candidate to believe that it is high praise. Some examples include:

- To describe a person who is totally inept: "I most enthusiastically recommend this candidate with no qualifications whatsoever."
- To describe an ex-employee who had problems getting along with fellow workers: "I am pleased to say that this candidate is a former colleague of mine."
- To describe a candidate who is so unproductive that the job would be better left unfilled: "I would urge you to waste no time in making this candidate an offer of employment."
- To describe a person with lackluster credentials: "All in all, I cannot say enough good things about this candidate or recommend him too highly."

LIAR is not only useful in preserving friendships, but also can help avoid serious legal trouble in a time when laws have eroded the confidentiality of letters of recommendation. In most states, job applicants have the right to read the letters of recommendation and can even file suit against the writer if the contents are negative. When writers use LIAR, however, whether perceived correctly or not by the candidate, the phrases are virtually litigation-proof.

—Taken from a United Press International story about the "LIAR System" designed by R. Thornton