Adaptive Motor Control Using Predictive Neural Networks

by

Fun Wey

Submitted to the Department of Brain and Cognitive Sciences
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computational Neuroscience

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1995

Author .....
Department of Brain and Cognitive Sciences
September 7th, 1995

Certified by .....
Michael I. Jordan
Professor
Thesis Supervisor

Accepted by .....
Gerald E. Schneider
Chairman, Departmental Committee on Graduate Students

Adaptive Motor Control Using Predictive Neural Networks

by

Fun Wey

Submitted to the Department of Brain and Cognitive Sciences
on September 7th, 1995, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computational Neuroscience

Abstract

This thesis investigates the applicability of the forward modeling approach in adaptive motor control. The forward modeling approach advocates that in order to achieve effective motor control, the controller must first be able to predict the outcomes of its actions with an internal model of the system, which can be used in the search for the appropriate actions to achieve particular desired movement goals. In realistic control problems, however, the acquisition of a perfect internal dynamical model is not generally feasible. This thesis shows how an approximate forward model, obtained via a simple on-line adaptation algorithm and an on-line action-search process, is able to provide effective reaching movement control of a simulated three-dimensional, four-degree-of-freedom arm.

In the course of studying the on-line action search process, a problem which we refer to as the "control boundary problem" was identified. This problem was found to occur frequently and was found to be detrimental to the gradient-based search method. We developed a novel technique for solving the problem, referred to as the "moving-basin approach." The moving basin approach leads the system to the desired goal by adaptively creating subgoals. Results are reported that show the improvement in the action-search process using the moving basin method.

Once the control boundary problem is solved, the forward modeling approach is able to provide stable adaptive control under perturbations of various forms and magnitudes, including those that could not be handled by analytical adaptive control methods. Further investigation also revealed that changes within the parameters of a forward model can provide information about the perturbed dynamics.

Thesis Supervisor: Michael I. Jordan
Title: Professor

Acknowledgments

I am awfully short of words to fully express my gratitude towards Mike Jordan for all he has done for me throughout my study at MIT. Mike has been highly tolerant towards my incompetence and rebellious nature, and had gone through maneuvers around the department to maintain the well-being of my studentship. I will probably not be able to find another person with such a compassionate heart. In fact the most important lesson I learned from Mike is his patience and tolerance. In my future career as a leader of a research team, my experience with these important qualities will most certainly turn out to be extremely beneficial.

I would also like to thank the government of Singapore for having achieved the economic miracle for the Republic, by which my study at MIT became possible. I am from a poor family, and my parents would not have imagined that their son could one day attain a PhD degree from the world's most prestigious school of science and technology. My government has been very supportive of my study by providing me a precious scholarship that extensively covered my study at MIT. In particular, I would like to thank the Chief Defense Scientist for attending my thesis defense. This is a great honor for me.

I would like to thank the members of Jordan lab for their friendship, support and encouragement throughout these years. In particular, I would like to thank John Houde for correcting some parts of my thesis.

I would like to thank Janice Ellertsen for her graceful assistance and care throughout these years. She alleviated my workload by keeping a close watch on administrative requirement related to my study.

Finally I would like to thank all members of my thesis committee (Prof Ronald Williams, Prof Tomaso Poggio and Asst Prof Peter Dayan) for their kind comments on my thesis, and being so helpful and considerate in granting me the PhD degree.

Contents

List of Figures

List of Tables

An Overview

This thesis work is prompted by the recent growth of research into the application of neural networks for the control of nonlinear systems. Specifically, it involves research into the applicability of the forward modeling idea in adaptive motor control.

The forward modelling approach to motor learning and control is an idea similar to the indirect methods in the adaptive control literature. It advocates that in order to achieve adaptive motor control, the controller must first be able to predict the outcomes of its actions (given the state information of the environment) with a model of the system. This predictive capability should then be utilized in the search for the appropriate actions in a particular situation. The idea is conjugate to the direct methods which do not involve building a model of the dynamical system to be controlled.

The original forward modeling idea involves:

1. the off-line training of neural networks with large number of exemplars;
2. the performance of the neural networks during the applied stage with feedforward computations only.

Most other connectionist architectures for adaptive control of nonlinear systems, in either direct or indirect form, require the neural networks to assume a role similar to that postulated above. There has not been any work on investigating the performance

of these architectures with the ANN's learning of a small number of on-line exemplars. This may be due to the fact that ANN training usually requires many exemplars and epochs to approximate a target function accurately, which created the presumption that the NN's learning of a small number of on-line exemplars will not do much.

The idea has been tested in the control of simulated robotic arms. In each simulation a neural network (forward model) was trained off-line to predict the resultant joint accelerations caused by torques applied to the joints at any given state of the arm. If this prediction is highly accurate, the sequence of feedforward torques retrieved from the forward model (using the backpropagation algorithm) would lead to the accurate execution of a desired trajectory (specified in the form of a sequence of desired joint accelerations). However an exact forward model of the dynamics of a robotic arm (even a simple one) produced by off-line training alone is not possible. There is thus a need to adapt on-line the forward model if the approach is to work without incorporating additional auxiliary controllers. A simple adaptation method is to train the forward model on each exemplar collected on-line (after the execution of its action in each time-step of the control). In view of the large amount of exemplars and time taken to off-line train the forward model, I presumed that this one-exemplar-at-each-time-step learning will not work. To my surprise the forward model is able to stably adapt to various kinds of nonlinear perturbations (of high magnitudes) to the arm dynamics with this method, and in turn performs highly accurate trajectory tracking. Further investigation revealed that the structure of the forward model could provide information about what the perturbations are. It also provides impressive generalizing power such that the learning of a single exemplar could lead to improvement of prediction in a large neighborhood in the forward model's input space. This discovery leads to new research into the computational properties of neural networks in general, and in particular its applicability in adaptive control of nonlinear systems.

Chapter 1

Introduction

Whatever you can do, or dream you can do, begin it!
Boldness has genius, magic, and power in it.
Begin it now!
- Goethe

1.1 Background and Motivation

The urge to understand and better control our bodily movements has been around throughout the ages. During the warring years early in human history, a better knowledge of this field would yield better fighting forces, and this in turn could mean the dominance of a particular race or nation over its neighbors. With the advances of science and technology in modern era, we see the ongoing quest for intelligent robotics for better industrial production and military capabilities. The need to find treatments for illnesses related to human motor control, such as Parkinson and Huntington diseases, and design of actuator interfaces for the disabled, also require a better understanding of biological motor control systems.

Of the various motor faculties, arm movements have attracted the most attention among researchers. While it is known that the inherent dynamics of human arms are nonlinear and complex, we can effortlessly execute highly flexible and complex arm movements under many conditions. Research into the arms' motor control system has in recent years evolved into a multi-disciplinary field involving robotics, medical science, adaptive control theory, psychophysics and, recently, computational neuroscience, in the hope that breakthroughs could be achieved through the convergence of both *engineering* and *reverse-engineering* approaches.

The reverse-engineering approach, which consists of the search for control strategies adopted by nature by looking at the actual neural circuits in biological systems, quickly run aground because of the difficulty in linking the implementing circuits to the inherent dynamics involved. To date numerous models involving only the use of feedback control circuits (e.g. alpha-gamma coactivation - see Kandel & Schwarz 1991) to handle perturbed dynamics of the arms have been proposed; but such control schemes can only explain how local improvement in movement control can be achieved. Since it is known that biological control systems are able to generalize from a small amount of experimenting with the perturbed dynamics of the motor faculties and lead to global improvement in performance, a framework that could fully model the competence of a biological motor system would have to be made within the realm of *adaptive* motor control.

The engineering approach mainly involves the modeling of the nominal dynamics of the arms using its artificial counterpart - i.e. the robotic manipulator with revolute joints, which can be achieved using numerous formulation strategies. Of these strategies, the most notable ones are the Newton-Euler formulation and the Lagrangian formulation, by which many formal adaptive methods for robotic manipulators have been established in recent years. The fast development of adaptive control for manipulators, which is by itself a specialized field, is due to the fact that the effective control of robotic manipulators is inherently linked to industrial interests; and consequently they provide enormous support for such development. These adaptive control

methods, many of them well-established in handling certain specific domains of perturbed dynamics, provide the theoretical framework that might be adapted to model the adaptive capabilities of biological motor systems. Such attempts would have to be made within the constraint of biological plausibility at the implementational level. In particular, effort would have to be made to tackle a challenging fact - the computational modules of biological motor systems consist of huge number of processing units, each limited in computational capabilities but whose global connectedness provide powerful learning capabilities. Many such attempts were made in recent years (e.g. see Lewis 1995, Miller 1990), and it is this interface that this thesis work is aimed at.

There are core organizational principles at a conceptual level which provide constraints narrowing our scope of research directions at this interface. For example, Jordan & Rumelhart (1992) have clearly argued that forward modeling is more likely to have been adapted by motor circuits as the control strategy for certain aspects. Such control methods involve an identification procedure in which the control system forms an approximate internal model of the system that is being controlled, coupled with a search process in which the model is used to search for control signals (also see Narendra & Parthasarathy, 1990). This leads to the concept of distal supervised learning in which the controller is trained using a model of the system to be controlled.

The original forward modeling idea utilizes auxiliary feedback controllers to improve the performance of an inaccurate controller trained by an approximate forward model. The forward model and controller can then be re-trained off-line with the exemplars collected to yield better feedforward control. The core idea in this thesis stretches the approach a bit further - the feedback controller is discarded altogether so that there is only feedforward action involved in the control. The feedforward action is retrieved from the forward model using an *on-line action-search* process, coupled with the *on-line adaptation* of the forward model for improving its prediction accuracy (and in turn the precision of feedforward action). By the on-line adaptation process, the forward model is trained to interpolate through an exemplar collected

on-line at each time step of the control. This is similar to some of the formal adaptive control methods in which the improvement in performance is to be achieved solely through the updating of parameters' and does not involve feedback control at all. The success of such an approach depends on whether the process of interpolating through a small number of exemplars does lead to improvement in the forward model's global prediction accuracy of the underlying mapping. The main task of the thesis is to report on the results from simulations and give an idea of the scope of applicability of such a control method implemented using various learning architectures.

In the course of implementing the on-line action-search process, a form of local minima problem was found to occur very frequently and proved detrimental. The common approach to the local minimum problem would be to treat it as an inherent, unsolvable problem in gradient-guided processes. A simple idea called the *moving basin* mechanism was devised and solved this problem to a significant extent. Once the problem with local minima was solved, the power of the forward modeling is unleashed and the on-line adaptation process was found to yield stable adaptive control in the presence of various forms and magnitudes of perturbations.

1.2 Organization of the thesis

The claims made about the applicability of on-line adaptation in this thesis are entirely supported by empirical simulation results. In order to substantiate the credibility of such support from empirical data, the most important part of the experiments were carried out on the adaptive control of a simulated 2-joint, 4-degree-of-freedom manipulator with a configuration similar to a human arm.

Chapter 2 will give a review of the nature of dynamics involved in motor control. A review of existing analytical methods for adaptive manipulator controls will be given. Chapter 3 will introduce the idea of forward modeling in its original form, to

be followed by a description of the trajectory planner used. The preliminary result of the adaptive control of a simulated 4-dof arm are then presented.

In Chapter 4, the problem with local minima encountered during the action search process is discussed. This is followed by a description of how it was solved with the moving-basin idea.

Chapter 5 presents the main results obtained from the control of the simulated 4-dof arm in various performance measures (intra- and inter-trajectory, types of perturbations to the dynamics). The unique and desirable properties of the forward model's on-line adaptation process is also presented together with notes on why they are counter-intuitive. The most important finding of the thesis - that the success of on-line adaptation process dispels the conventional view that a neural network's training necessarily requires extensive number of exemplars and training time - will be emphasized.

Chapter 6 presents the results on the implementation of the idea using other computational architectures such as Hyper-Basis functions and local expert networks. This is followed by a brief comparison with Craig's adaptation method. Such comparisons provide a better scope of performance measure of the forward modeling approach.

Chapter 7 provides a summary of the interesting findings in the research. The feasibility/implementability of the technique investigated is also discussed.

Chapter 2

Motor Dynamics and its Adaptive Control

2.1 Introduction

This chapter will review the mathematical modeling of human arm dynamics based upon its artificial counterpart - i.e. a robotic manipulator with revolute joints. In particular the simplifications involved in the modeling will be discussed. This is to be followed by a glimpse of the complexity in the modeling of ideal multi-degree-of-freedom manipulators. We will also review some of the existing analytical adaptive control schemes specific for manipulators, and point out their limitations in handling nonlinearities. The exposition of the complexity involved in the field of manipulator controls in this chapter shall establish the justification for the use of the empirical results in supporting the claims to be made in later chapters of this thesis.

2.2 Manipulator's Dynamics

A revolute joint manipulator is usually modeled as a set of n moving *rigid* bodies connected by revolute joints in a serial chain. There is a torque actuator and friction acting at each joint. The equation of motion of such a device can be described by

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

where τ is the vector of joint torques, θ is the vector of joint positions, $\dot{\theta}$ the vector of joint velocities, and $\ddot{\theta}$ the vector of joint accelerations. The matrix $M(\theta)$ is the *manipulator mass matrix.* The vector $V(\theta, \dot{\theta})$ represents torques arising from centrifugal and Coriolis forces. The vector $F(\dot{\theta})$ represents torques due to friction acting at the joints. The vector $G(\theta)$ represents torques due to gravity, and T_d is a vector of unknown signals due to unmodeled dynamics and external disturbances. Equation 2.1 is sometimes written as

$$\tau = M(\theta)\ddot{\theta} + Q(\theta, \dot{\theta}) \tag{2.2}$$

where $Q(\dot{\theta}, \dot{\theta}) = V(\theta, \dot{\theta}) + F(\dot{\theta}) + G(\theta) + T_d$.

The major simplification involved in modeling the human arm's dynamics using ideal manipulator dynamics is in the rigidity of the links and viscosity at the joints. The mechanical links of manipulators are much more rigid than the human arms, which can be more fully modeled using flexible joint/bending modes dynamics (see Yurkovich 1990, Book 1993). As the joints in human arms are surrounded by fluids, viscous forces play a larger role in human arm dynamics than robotic arm. Viscous forces are complex and nonlinear, but in robotics they are usually considered as part of frictional forces and can be written as

$$F(\dot{\theta}) = F_m\dot{\theta} + T_{df} \tag{2.3}$$

where F_m is a diagonal matrix of viscous friction coefficients, and T_{df} is unstructured

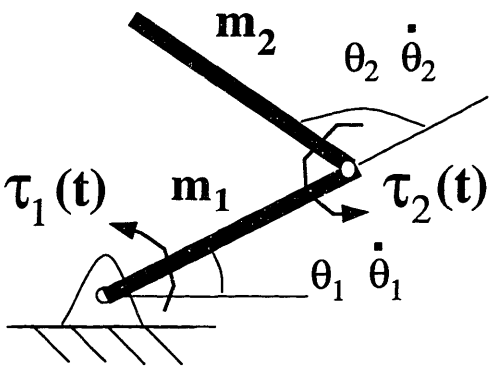


Figure 2-1: a 2-joint revolute arm.

friction effects.

2.3 A Glimpse of the Complexity involved in Motor Dynamics

Equation 2.1 may not look too intimidating at first glance. However when we look at it in detail using, say, the Newton-Euler formulation, the dynamics of the 2-joint arm in Figure 2-1 would look like this:

$$\begin{aligned}
\tau^{interactive} &= -[I_2 + m_2(l_{c2}^2 + l_1 l_{c2} cos\theta_2)]\ddot{\theta}_2 \\
\tau_1^{centrifugal} &= -m_2 l_1 l_{c2} \dot{\theta}_2^2 sin(\theta_2) \\
\tau_2^{centrifugal} &= -m_2 l_1 l_{c2} \dot{\theta}_1^2 sin(\theta_2) \\
\tau_0^{Coriolis} &= 2m_2 l_1 l_{c2} \dot{\theta}_1 \dot{\theta}_2 sin(\theta_2) \\
\ddot{\theta}_1 &= \frac{\tau_1 + \tau_1^{centrifugal} + \tau_0^{Coriolis} + \tau_0^{interactive}}{I_{2,0} + I_1} \\
\ddot{\theta}_2 &= \frac{\tau_2 + \tau_2^{centrifugal}}{I_2}
\end{aligned} \tag{2.4}$$

where I_j, l_j and m_j are respectively the moment of inertia, the length and the mass of link j, and l_{cj} is the position of the center of mass of link j along the link's length, $\tau_j^{centrifugal}$, $\tau_j^{interactive}$ and $\tau_j^{Coriolis}$ are respectively the centrifugal torque, the

interactive torque and the Coriolis torque experienced at joint j. Most adaptive control methods for direct-drive manipulators (see Asada & Slotine, 1986) were tested on 2-joint arm manipulators. We now consider the dynamics of a 3-joint arm - i.e. with the addition of one more joint:

$$
\begin{aligned}
I_{2,0} &= I_2 + m_2(l_1^2 + l_{c2}^2 + 2l_1 l_{c2} cos\theta_2) \\
I_{3,0} &= I_3 + m_3((x_3 + l_{c3} cos(\theta_1 + \theta_2 + \theta_3))^2 + (y_3 + l_{c3} sin(\theta_1 + \theta_2 + \theta_3))^2) \\
I_{3,1} &= I_3 + m_3(l_2^2 + l_{c3}^2 + 2l_2 l_{c3} cos\theta_3) \\
\tau_1^{interactive} &= -I_3(\ddot{\theta}_1 + \ddot{\theta}_2 + \ddot{\theta}_3) \\
&\quad -m_3(\ddot{\theta}_1(l_1 l_2 cos\theta_2 + l_2^2 + l_3^2 + l_1 l_{c3} cos(\theta_2 + \theta_3) + 2l_{c3} l_2 cos\theta_3) \\
&\quad +\ddot{\theta}_2(l_2^2 + l_3^2 + 2l_2 l_3 cos\theta_3) + \ddot{\theta}_3(l_3^2 + l_2 l_{c3} cos\theta_3)) \\
\tau_0^{interactive} &= -I_2\ddot{\theta}_2 - \tau_1^{interactive} - m_2\ddot{\theta}_2(l_{c2}^2 + l_1 l_{c2} cos\theta_2) \\
\tau_1^{Coriolis} &= 2m_3 l_{c3} l_2 sin\theta_3 \dot{\theta}_3 - \dot{\theta}_3 \\
\tau_0^{Coriolis} &= 2(m_3 l_{c3}(\dot{\theta}_1 + \dot{\theta}_2)\dot{\theta}_3(l_1 sin(\theta_2 + \theta_3) + l_2 sin\theta_3) + m_2 l_1 l_{c2}\dot{\theta}_1\dot{\theta}_2 sin\theta_2) \\
\tau_2^{centrifugal} &= -m_3 l_{c3}(l_1\dot{\theta}_1^2 sin(\theta_2 + \theta_3) + (\dot{\theta}_2^2 + \dot{\theta}_1^2) l_2 sin\theta_3) \\
\tau_1^{centrifugal} &= -m_2 l_1 l_{c2}\dot{\theta}_1^2 sin\theta_2 + l_{c3} m_3 \dot{\theta}_3^2 sin\theta_3 \\
\tau_0^{centrifugal} &= m_2 l_1 l_{c2}\dot{\theta}_2^2 sin\theta_2 + l_{c3} m_3 \dot{\theta}_3^2(l_1 sin(\theta_2 + \theta_3) + l_2 sin\theta_3) \\
\ddot{\theta}_1 &= \frac{\tau_1 + \tau_0^{centrifugal} + \tau_0^{Coriolis} + \tau_0^{interactive}}{I_{2,0} + I_{3,0} + I_1} \\
\ddot{\theta}_2 &= \frac{\tau_2 + \tau_1^{centrifugal} + \tau_1^{Coriolis} + \tau_1^{interactive}}{I_2 + I_{3,1}} \\
\ddot{\theta}_3 &= \frac{\tau_3 + \tau_2^{centrifugal}}{I_3}
\end{aligned}
\tag{2.5}
$$

The explosive increase in complexity from Equation 2.4 to Equation 2.5 illustrates the difficulty involved in actually applying many formal methods in the control of manipulators with large number of dofs. A normal human arm has seven major degrees-of-freedom in it, with 4 of them (three at the shoulder joint, one at the elbow) being required to overcome large centripetal forces when the arm swings at high speeds. An elaborated formulation of the dyanmics of the arm involving just

these 4 degrees-of-freedom would take many pages to list out. If more precise modeling using the flexible joint/bending modes is involved, the complexity could be far more deterring.

2.4 Review of Adaptive Control Schemes for Manipulators

The field of adaptive motor control is vast. An extensive and detailed coverage of the work in this field is beyond the scope of this thesis. This section is to provide a brief overview of the current state of the research involved.

Traditionally, most work on the problem of adaptively controlling a mechanical manipulator has simply been the application of methods that were developed for linear systems (see Craig 1988, Koivo 1983, Stoten 1990, Dubowsky & DesForges 1979). Recently, there has been a tremendous growth in the number of adaptive control schemes for manipulators proposed to handle certain types of nonlinearities in the dynamics. Virtually all of these schemes assume perfect knowledge of the *structure* of the underlying dynamics - i.e. that a perfect structural model of the manipulator dynamics can be built, and the performance errors are due entirely to parameter errors. Some (e.g. Craig's and Slotine's - see the following subsections) involve explicit identification of some of the manipulator's parameters. These schemes generally do not involve the use of feedback controllers; and rely solely on parameter adaptation coupled with the use of a control law to achieve on-line improvement in performance. Methods of these types were mathematically proven to be robust under specified and bounded disturbances and parametric uncertainties. Horowitz and Tomizuka's approach (see Horowitz & Tomizuka 1980, Sadegh & Horowitz 1987), another prominent group of work in this field, treats those parts of the dynamics depending only on manipulator position as unknown parameters, which are then adaptively identified. However their method requires that the parameters be *slowly*

varying. Some other methods (e.g. see Slotine & Li 1991,Liu & Yuan 1989) are based on sliding mode or variable structure systems, in which there is no explicit identification of systems' parameters[1]. However these chattering controls (which in theory switch at infinite frequency) cannot be implemented, and the derivative of servo error goes to zero in the mean. Such high-frequency control action may also excite unmodeled resonances.

We now present two of the more well-known schemes for manipulator's adaptive control. Both of these schemes involve the explicit on-line identification of certain unknown parameters, do not involve feedback controllers; and rely solely on the on-line update of parameters to achieve improvement in performance.

2.4.1 Craig's Adaptive Scheme

Craig(1988) proposed a scheme for adaptive manipulator control that takes full advantage of any known parameters while estimating the remaining unknown parameters. However its applicability is limited to the identification of parameters that can be linearly decoupled from the dynamical equation.

In Craig's method, the output servo error $\mathbf{E} = \theta^* - \theta$, where θ^* and θ are respectively the desired and actual joint position specified at each time step, is related to the dynamical model's error in the following form :

$$\ddot{\mathbf{E}} + \mathbf{K}_v\dot{\mathbf{E}} + \mathbf{K}_p\mathbf{E} = \hat{\mathbf{M}}^{-1}(\theta)\left[\tilde{\mathbf{M}}(\theta)\ddot{\theta} + \tilde{\mathbf{Q}}(\theta,\dot{\theta})\right] \tag{2.6}$$

where $\tilde{\mathbf{M}}(\theta)$ is the error in manipulator mass matrix, $\tilde{\mathbf{Q}}(\theta,\dot{\theta})$ is the error in the centrifugal and Coriolis forces. A necessary step in Craig's adaptive parameter identification scheme is the decoupling of the parameters' errors $\mathbf{\Phi} = P - \hat{P}$ from the

[1]These were sometimes classified as robust control instead of adaptive control.

dynamics such that the following linear relation can be achieved:

$$\ddot{\mathbf{E}} + \mathbf{K}_v\dot{\mathbf{E}} + \mathbf{K}_p\mathbf{E} = \hat{\mathbf{M}}^{-1}(\theta)\mathbf{W}(\theta, \dot{\theta}, \ddot{\theta})\mathbf{\Phi}, \tag{2.7}$$

i.e. $\mathbf{W}(\theta, \dot{\theta}, \ddot{\theta})\mathbf{\Phi} = \tilde{\mathbf{M}}(\theta)\ddot{\theta} + \tilde{\mathbf{Q}}(\theta, \dot{\theta})$. If the above is possible, the *parameter adaptation law*

$$\dot{\hat{P}} = \Gamma\mathbf{W}^T(\theta, \dot{\theta}, \ddot{\theta})\hat{\mathbf{M}}^{-1}(\theta)(\dot{E} + \Psi E) \tag{2.8}$$

can then be used in conjunction with the control law

$$\tau_{ff} = \hat{M}(\theta)\ddot{\theta}^* + \hat{Q}(\theta, \dot{\theta}) \tag{2.9}$$

to achieve stable trajectory tracking control. In the above, Γ is a diagonal matrix of non-negative constants and Ψ is a diagonal matrix of positive constants, $\ddot{\theta}^*[t+1] = \ddot{\theta}_d + K_p(\theta^*(t) - \theta[t]) + K_v(\dot{\theta}^* - \dot{\theta}[t])$, where $\theta^*(t)$, $\dot{\theta}^*(t)$ and $\ddot{\theta}_d$ are the prescribed desired joint angle, joint velocity and joint acceleration at each time-step that fully describe a reaching trajectory[2].

The advantage of Craig's scheme over most other schemes developed prior to its conception is that it has been rigorously proven stable, and it provides explicit identification of certain parameters(e.g. the viscous and Coulomb friction coefficients, v_i and k_i respectively, and the masses of the links, which were already decoupled in the original dynamics formulation.). However there are limitations as to the types of parameters that can be identified by this method. To illustrate this point, consider the problem of identifying the displacements ($\delta\theta_1$ and $\delta\theta_2$) in the joint angle readings (θ_1 and θ_2) of a 2-d.o.f. arm (see Equation 3.42, Craig 88). The goal is to obtain a formulation whereby the vector $\mathbf{\Phi}$ is in the form $[\delta\theta_1, \delta\theta_2]^T$ or $[f(\delta\theta_1), f(\delta\theta_2)]^T$, where f is a trigonometric function. Evaluating the dynamics error function of the first

[2]See Section 3.4.1.

d.o.f. yields the following :

$$
\begin{aligned}
& \left(\tilde{\mathbf{M}}(\theta)\ddot{\theta} + \tilde{\mathbf{Q}}(\theta,\dot{\theta})\right)_1 \\
= \; & m_2 l_1 l_2 (2\ddot{\theta}_1 + \ddot{\theta}_2)((\cos\theta_2 \cos\delta\theta_2 - \sin\theta_2 \sin\delta\theta_2) - \cos\theta_2) \\
& -(\dot{\theta}_2 - \dot{\theta}_1) m_2 l_1 l_2 ((\sin\theta_2 \cos\delta\theta_2 - \cos\theta_2 \sin\delta\theta_2) - \sin\theta_2) \\
& +m_2 l_2 g(((\sin\theta_1 \cos\delta\theta_1 + \cos\theta_1 \sin\delta\theta_1)(\cos\theta_2 \cos\delta\theta_2 - \sin\theta_2 \sin\delta\theta_1) \\
& +(\cos\theta_1 \cos\delta\theta_1 - \sin\theta_1 \sin\delta\theta_1)(\sin\theta_2 \cos\delta\theta_2 + \cos\theta_2 \sin\delta\theta_2)) - \sin(\theta_1 + \theta_2)) \\
& +(m_1 + m_2) l_1 g(\sin\theta_1 \cos\delta\theta_1 + \cos\theta_1 \sin\delta\theta_1 - \sin\theta_1), \qquad (2.10)
\end{aligned}
$$

The above involves $(\cos\delta\theta_1 \sin\delta\theta_2)$ and $(\sin\delta\theta_1 \cos\delta\theta_2)$ terms, which implies that the desired $\mathbf{W\Phi}$ form could not be achieved. Note that it is important for biological neuromotor systems be able to effectively handle the problems with erroneous joint positions' readings. This is because, among many species of wildlife, lesions in the afferent pathways are common.

2.4.2 Slotine's scheme

Like Craig's scheme, Slotine's approach (Slotine & Li 87) requires the unknown manipulator parameters to be linearly decoupled from the dynamical formulation, such that the relation between the vector $\hat{P}$ of unknown parameters and the robotic dynamics can be written as :

$$\tilde{\mathbf{M}}(\theta)\ddot{\theta}^* + \tilde{\mathbf{V}}_m(\theta,\dot{\theta})\dot{\theta}^* + \tilde{\mathbf{G}}(\theta) = \mathbf{Y}\hat{P} \qquad (2.11)$$

where $\tilde{\mathbf{M}}(\theta)$ is the error in the manipulator mass matrix, $\tilde{\mathbf{V}}_m(\theta,\dot{\theta})$ is the error in the matrix of involving centrifugal and Coriolis forces, $\tilde{\mathbf{G}}(\theta)$ is the error in the vector of forces due to gravity, $\theta^*(t)$ is the desired trajectory and $\mathbf{Y} = \mathbf{Y}(\theta,\dot{\theta},\dot{\theta}^*,\ddot{\theta}^*)$ is a

matrix. It has been shown that the adaptation law

$$\dot{\hat{P}} = -\Gamma^{-1}\mathbf{Y}^T(\theta, \dot{\theta}, \dot{\theta}^*, \ddot{\theta}^*)\dot{\tilde{\theta}} \tag{2.12}$$

together with the control law

$$\tau = \hat{\mathbf{M}}(\theta)\ddot{\theta}^* + \hat{\mathbf{V}}_m(\theta, \dot{\theta})\dot{\theta}^* + \hat{\mathbf{G}}(\theta) - \mathbf{K}_p\tilde{\theta} - \mathbf{K}_D\dot{\tilde{\theta}} \tag{2.13}$$

yields a globally stable adaptive controller. The $\hat{}$ in the above equation denotes the estimate of the respective terms, $\tilde{\theta} = \theta - \theta^*$ and $\mathbf{K}_p$ and $\mathbf{K}_D$ are positive definite matrices.

Slotine's scheme also has been mathematically proven to be globally stable when specified conditions are met. But, again, the scheme will not be able to handle the dynamic error due to the shift of joint angles described in Equation 2.10, or other forms of perturbations whose parameters cannot be linearly decoupled.

Chapter 3

Forward Modeling and On-line Adaptation

3.1 Introduction

This chapter introduces the core idea of forward modeling. The configuration of forward modeling for manipulator control will then be presented. This is to be followed by a discussion of related issues such as the nature of the mapping to be learned, learning efficacies, and the arbitrary distinction between learning and adaptation. The chapter will end with a description of the simulated 4-dof arm used in the experiments to be presented in the subsequent chapters.

3.2 Distal supervised learning

To control a system requires solving for a control input to the system that will yield a desired result at its output. A *feedforward controller* solves for a control signal on the

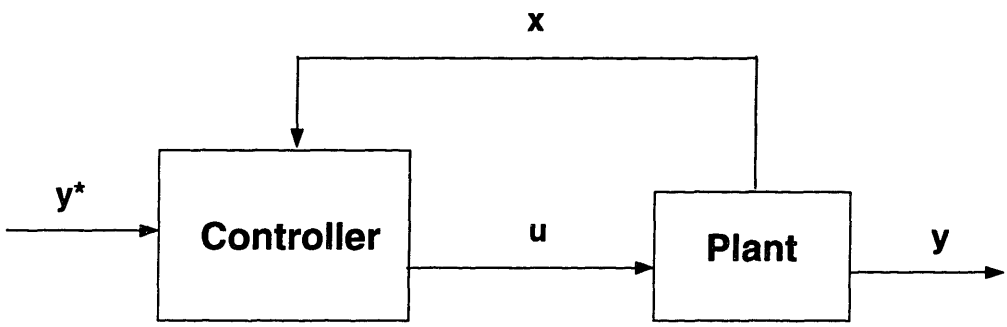


Figure 3-1: An inverse model as a feedforward controller. With the direct-inverse modeling technique, the inverse model will be trained to perform the map $\mathbf{u}[t] = \hat{f}^{-1}(\mathbf{y}[t+1], \mathbf{x}[t])$, where $\mathbf{y}[t+1]$ is the observed system's output caused by an action $\mathbf{u}[t]$ at state $\mathbf{x}[t]$.

basis of future desired values of the output. As shown in Figure 3-1, a feedforward controller is a mapping from reference signals and states to control signals:

$$\mathbf{u}[t] = g(\mathbf{y}^*[t+1], \mathbf{x}[t]). \tag{3.1}$$

where the state vector $\mathbf{x}[t]$ and the control vector $\mathbf{u}[t]$ are defined at time t, and the reference signal $\mathbf{y}^*[t+1]$ is the desired output at time $t+1$. The task is specified in a sequence of $\mathbf{y}^*[t+1]$.

Jordan & Rumelhart (1992) discuss the problem of learning a feedforward controller using supervised learning methods. It has been observed that if the required mapping is one-to-many, i.e. a set of distinct target outputs to be paired with the same input, and this set of target outputs is non-convex such that the mean of the set is outside it, then the network will not learn the mapping. A prominent example is the inverse kinematic mapping of robotic arms involving redundant degrees-of-freedom. They propose an indirect approach for training a feedforward controller in which the controller is placed in series with an internal *forward model* of the controlled system. A *forward model* predicts the behavior of the controlled system by performing the following mapping

$$\hat{\mathbf{y}}[t+1] = \hat{f}(\mathbf{u}[t], \mathbf{x}[t]). \tag{3.2}$$

where $\hat{\mathbf{y}}[t+1]$ is the predicted output at time $t+1$ after the action $\mathbf{u}[t]$ has been

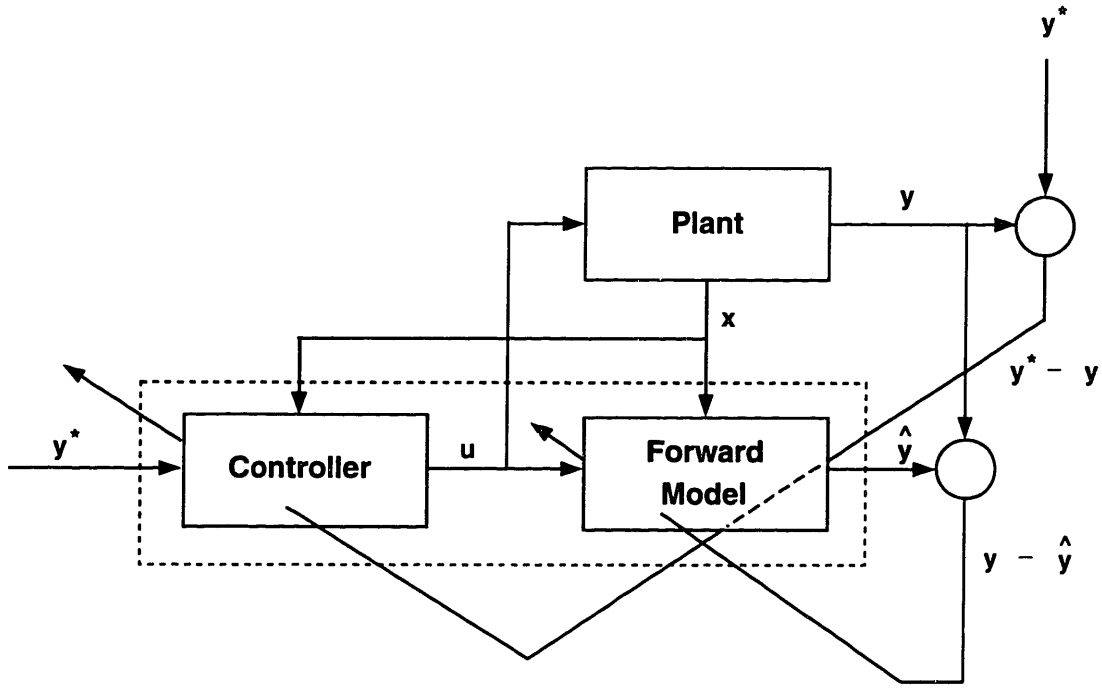


Figure 3-2: The distal supervised learning approach. The forward model is trained using the *prediction error* $\mathbf{y}[t] - \hat{\mathbf{y}}[t]$. The controller is trained by propagating the *performance error* $\mathbf{y}^*[t] - \mathbf{y}[t]$ backward through the forward model to yield an incremental adjustment to the controller. It is also possible to propagate the *predicted performance error* $\mathbf{y}^*[t] - \hat{\mathbf{y}}[t]$ backward through the forward model. This can be done iteratively to find a locally optimal control signal based on the initial control signal proposed by the controller.

elicited at time t.

The forward model is itself trained using supervised learning algorithms. With the forward model held fixed, the composite system is trained using the reference signal as both the input and the target (Figure 3-2). This procedure solves implicitly for a control signal at the interface of the controller and the forward model.[1]

It is important to note that in principle a perfect controller can be learned even if the forward model is inaccurate. This is due to two factors: first, although an inaccurate forward model may preclude steepest descent it need not prevent the algorithm from moving downhill; and second, the true output from the environment can be substituted for the inaccurate estimate in the calculation of the error term (see

[1]Although we utilize gradient-based techniques in the current thesis, it is worth noting that the distal supervised learning approach is not restricted to such techniques.

Figure 3-2). That is, the system can use the *performance error* ($\mathbf{y}^*[t] - \mathbf{y}[t]$) rather than the *predicted performance error* ($\mathbf{y}^*[t] - \hat{\mathbf{y}}[t]$), (where $\hat{\mathbf{y}}[t]$ is the output of the forward model and $\mathbf{y}[t]$ is the output of the plant). In the current thesis, however, we are concerned with finding an appropriate control signal *before* observing the output from the environment, thus we use the predicted performance error. The cost function that we utilize is the squared norm of the predicted performance error:

$$J_{pp} = \frac{1}{2}(\mathbf{y}^*[t+1] - \hat{\mathbf{y}}[t+1])^T(\mathbf{y}^*[t+1] - \hat{\mathbf{y}}[t+1]). \tag{3.3}$$

To search for a control signal, the state, the reference signal, and the weights in the forward model are held fixed while $\mathbf{u}[t]$ is varied in a direction that decreases J_{pp}.

3.3 Some Psychophysical Evidence of the existence of Forward Models

Some models of human motor controls proposed within the psychophysics community are similar to the forward modeling idea. For example in Adam's scheme (Wilberg, 1991) it was proposed that our CNS uses mental imagery as a representational system. This is because it has been observed that motor skills can be improved even in the absence of actual physical movement. We can think of the mental imagery system as a forward model which provides internal 'simulated' feedback to imagined movements.

Other works also report how the nervous systems might use internal maps for learning/calibrating motor control systems. For example, Grossberg & Kuperstein (1989) propose that each adaptive sensory-motor system, such as the eye-head and hand-arm system, computes a representation, or map, of target positions. The sensory-motor system also computes a *representation* of present position that is updated by a copy of the actuation signal to the muscles of the system called the *efferent's copy.* This representation of present position can be deemed as a forward

model that predicts the outcome of the efferent signal, and the mismatch between it and the afferent signal would be used for adapting the system. Wolpert et al. (1995) also report on results and simulations based on a novel approach that investigates the temporal propagation of errors in the sensorimotor integration process. Such findings provide direct support for the existence of an internal model. It has also been proposed (Miall et. al., 1993) that cerebellum could be a site for such a kinematic forward model.

3.4 Forward modeling for Manipulator Control

In the context of manipulator control, the state $\mathbf{x}(t)$ would be the vector $(\theta(t), \dot{\theta}(t))$, where θ is the joints' position and $\dot{\theta}$ the joints' velocity. The action $\mathbf{u}(t)$ would be the joints' torque $\boldsymbol{\tau}(t)$, and the output $\mathbf{y}$ would be the joint acceleration $\ddot{\theta}(t+1)$ to be observed at the next time step $t+1$. $J_{pp} = 0$ in Equation 3.3 would be equivalent to having

$$\tau_{ff} = \hat{M}(\theta)\ddot{\theta}^* + \hat{Q}(\theta, \dot{\theta}) \tag{3.4}$$

where τ_{ff} is the feedforward torque, $\hat{M}$ and $\hat{Q}$ are respectively the estimates of M and Q in Equation 2.2.

Equation 3.4 is sometimes referred to as the *computed torque method* of manipulator control, which works based on the assumption that desired trajectory of the free end of the manipulator is known and expressible as time functions of joint positions, velocities, and accelerations. We now briefly discuss the issue of trajectory planning in this thesis work.

3.4.1 Trajectory Planning

Trajectory planning is the time evolution of kinematics variables. Thus a trajectory has both spatial and temporal aspects : the spatial aspect is the sequence of locations of an arm or object from start to goal (a *path*), and the temporal aspect is the time dependence along a path. A trajectory planned for a hand's reaching movement can be expressed either in the desired hand's position and velocity (in Cartesian coordinate), or in desired joints' position and velocity (polar coordinate) at each time step. The latter is known as *joint interpolation* (see Hollerbach 1990), which has the advantage that complex inverse transformations can be left out of the control cycle. Since this thesis only concerns adaptive dynamical controls, a simple form of joint interpolation is used for trajectory planning for all the experiments. This trajectory planner has the form :

$$\ddot{\theta}^*[t+1] = K_p(\theta^* - \theta[t]) + K_v(\dot{\theta}^* - \dot{\theta}[t]), \tag{3.5}$$

where K_v is a constant diagonal matrix of *damping* gains, K_p is constant diagonal matrix of *position* gains, $\boldsymbol{\theta}^*$ is the target joints' position and $\dot{\theta}^*$ is the taget joints' velocity at the end of the trajectory. Note that this does not involve the explicit specification of $\theta^*(t)$ and $\dot{\theta}^*(t)$ at each time step. The path of the hand generated by such an *implicit* joint interpolation scheme is, in general, not a straight line. However it is smoothly varying and thus leads to the demand of slowly-varying $\tau(t)$. Furthermore since it is influenced by $\theta[t]$ and $\dot{\theta}[t]$, it actually contributes certain degree of feedback control to the overall reaching movement, and allows the target position to be reached even if the controller is only fairly accurate. This can be easily inferred from the fact that it has a similar form of control proposed by Arimoto and Miyazaki (see Arimoto & Miyazaki 1985), except the M and Q are not precise.

3.4.2 Nature of mapping and Learning Efficacy

Biological neural networks can handle nonlinear mappings of varying degrees of non-linearities. For example in the dynamical control of eyeballs, since the eye balls are nearly spherical and the eye balls' muscles are very strong, their control dynamics are very linear (see Robinson 1981, Galiana 1990). On the other hand, the arm's dynamics is relatively nonlinear as the geometry of its posture and joint velocities determine the gravitational, centripetal and Coriolis forces acting on the joints. A major issue is whether an artificial neural network (or some other mapping tools) can learn the forward dynamics of a human arm. This can be put in the context of the issue of whether the mapping of a nonlinear dynamical system can be learned by a neural network, which in recent years have been addressed by many papers (e.g. Zbikowski 1994, Chen & Chen 1995). These approaches of work usually involve Stone-Weierstrass Theorem and will not be discussed in this thesis.

From my empirical experience, training a forward model to high prediction accuracy can be achieved much more easily and quickly than that of an inverse model, even in cases where the inverse model's mapping is *one-to-one*. In particular, the difficulty of on-line training a CMAC inverse model has been pointed out in Kawato 1990. It is possible that this is related to the nature of the exemplar-collection process. Training exemplars collected from, say, a real robotic arm, consist of the following $:(\theta[t], \dot{\theta}[t], \tau[t]) \mapsto \ddot{\theta}[t+1]$. A property of such an exemplar set is that for a given $\theta \in \mathbf{\Theta}$, where $\mathbf{\Theta}$ is the set of all possible θ, there are exemplars with $\boldsymbol{\tau}$ well-distributed in the set of possible torques $\boldsymbol{\Gamma}$; but the set of possible $\dot{\theta}$ given a particular θ is a small subset of the global set of possible $\dot{\theta}$. Similarly, the set of possible $\ddot{\theta}$ given a particular θ is a small subset of the global set of possible $\ddot{\theta}$. It is a common practice to scale each of these values by the min/max of them, such that the activation of the input and output nodes is within the range $[-1, 1]$, to facilitate the network's training process. Within this hypercube, the $\theta[t]$ and $\boldsymbol{\tau}[t]$ dimensions are usually fully-spanned by the exemplars because of the property of the exemplar set.

On the other hand, the $\dot{\theta}[t]$ and $\ddot{\theta}[t]$ dimensions are usually poorly spanned, and the set of values occupies a narrow strip within the hypercube. The ease of a training a neural network is related to how well its input space is spanned by the training set. For a forward model (whose mapping is $\hat{\ddot{\theta}}[t+1] = \hat{f}(\theta[t], \dot{\theta}[t], \tau[t])$), the above implies that only part of the state input subspace - i.e. $\dot{\theta}[t]$ - are poorly spanned. On the other hand, an inverse model (whose mapping is $\hat{\tau}[t] = \hat{g}(\theta[t], \dot{\theta}[t], \ddot{\theta}[t+1])$) would have the $\dot{\theta}[t]$ and $\ddot{\theta}[t+1]$ dimensions of its input space poorly spanned. The result is that it is more difficult to train an inverse model than to train a forward model. For simple dynamics (such as that of a 2-joint robotic arm) the difference between the forward model's and inverse model's training is not too noticeable. In fact, virtually all reports (a recent one is Gorinesky 1993) with positive results using direct-inverse approach were from experiments or simulations on planar 2-joint arms. However for more complex dynamics, the difference becomes significant.

3.5 On-Line Adaptation

Perfect tracking control could be achieved with this method if the forward model is exact (i.e. $\hat{\mathbf{y}}[t] = \mathbf{y}[t]\ \forall \mathbf{u}[t], \mathbf{x}[t]$) and the action at each time step could be retrieved. However an exact forward model of a system of moderate complexity is very unlikely even if the system is time-invariant, as the number of exemplars and time required for the forward model's off-line training grow at an exponential rate *with respect to* the desired level of prediction accuracy. This implies that there is a need for on-line correction of the control.

At this point, an arbitrary but more formal distinction between the terms *learning* and *adaptation* is needed in order to justify the further use of the term on-line adaptation. Learning, in a motor control sense, is a process of acquiring the capability for producing skilled action (Schmidt 1982). In the sense of manipulator control, it is usually referred to (see Craig 1988) as any control scheme that improves the per-

formance of the device being controlled as actions are repeated, and does so without the necessity of a parametric model of the system. Such schemes are advantageous when the system is difficult to model parametrically, or when the system structure is unknown. The current applicability of such learning control schemes has been limited to manipulators that repeat their actions in constant cycles (e.g. see Arimoto & Kawamura 1984, Atkeson & Reinkensmeyer 1990). During the learning phase of a learning control scheme, stability should be less of a concern, as the system and environment should be well-anticipated in case of the failure of control.

Adaptive control, on the other hand, requires that the structure of the system be known *entirely*[2]; and that the imperfection in performance is due entirely to the errors in parameters. An adaptive control scheme thus requires the unknown or changing parameters to be explicitly sorted out in the formulation of the dynamics of the system. With the nature/bounds of the parameters known, we can then proceed with the proof of stability of control and devise measures of the efficiency in parameters' identification and performance, which are important in establishing the viability of an adaptive control scheme.

"Off-line training" is applied to a feedforward controller of a dynamical system when as it is not accurate enough to satisfy certain performance measures (such as the completion of reaching tasks with high probability of success). It refers to the training of a forward model with a large training set consisting of exemplars collected by (probably) *passively* observing the system. At this stage, the control of the system, if there is any, should be carried out by subsidiary tools such as feedback controllers.

In the original model proposed in Jordan & Rumelhart 1992 (as well as many other neural-network-based architectures for controls - e.g. see Kawato 1990, Hollerbach 1990), fringe PID feedback controllers are to be used in conjunction with the

[2]The word "entirely" here is to include the possibility that a viable adaptive control scheme should be able to accommodate certain degree of unknown dynamics (i.e. perturbations) to the system.

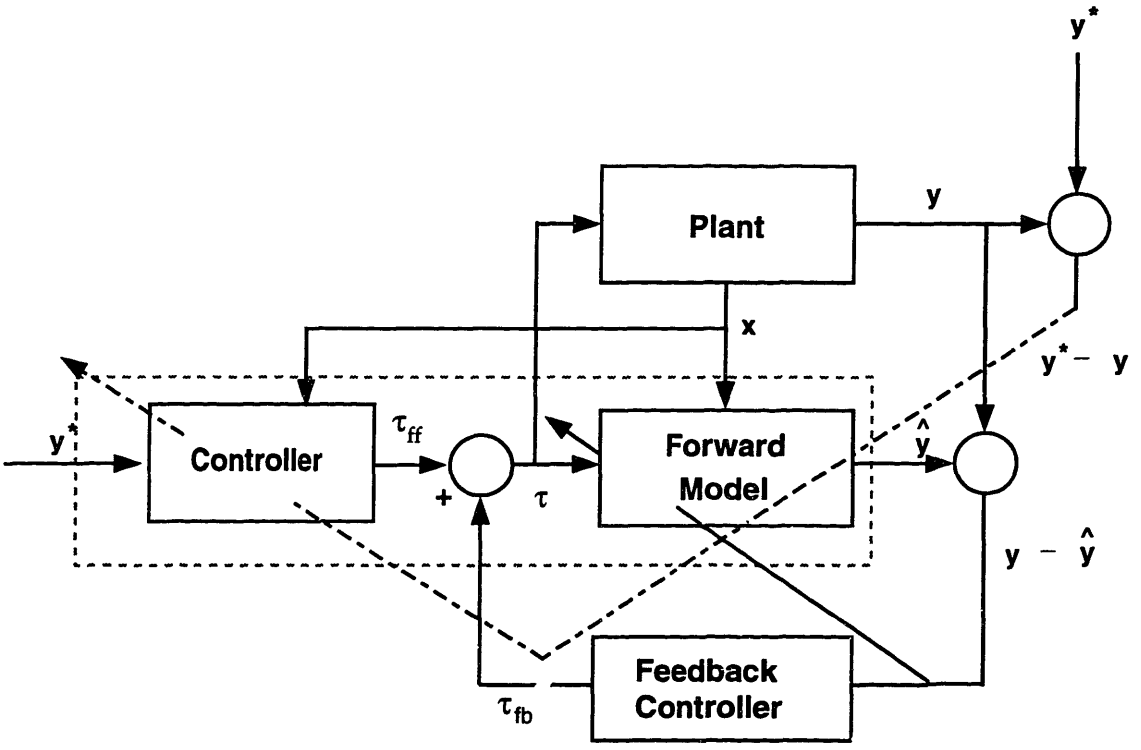


Figure 3-3: Feedback controller to correct the error in feedforward torque τ_{ff} via the feedback torque τ_{fb}.

feedforward controllers (see Figure 3-3) for correcting the inaccuracies in actions due to inaccuracies in the forward model. Feedback controllers[3] are usually of the form

$$\tau_j^{fb}(t) = K_p(\theta_j^*(t) - \theta_j(t)) + K_v\dot{\theta}_j(t) + K_i\int_{t_o}^{t}(\theta_j^*(t) - \theta_j(t))dt \qquad (3.6)$$

where τ_j^{fb} is the feedback torque at joint j. During the course of executing a reaching trajectory, τ^{fb} will be added to the feedforward torque generated by the feedforward controller. The sequence of exemplars $\{(\theta[t], \dot{\theta}[t], \tau[t]) \mapsto \ddot{\theta}[t+1]\}$ collected during the trajectory will then be used for off-line training the forward model and feedforward controller, such that future feedforward control will eventually dominate and lead to the diminishing of the feedback control. If a neural network can be off-line trained to provide the precise forward model, and the gains of the feedback controllers manually preset to certain values that ensure stability, this approach will most certainly provide

[3]Note that although some physiological observations (e.g. see Abbs & Gracco 1983) supported the hypothesis that *task-level* feedback are more likely adopted by biological systems for slow arm movements, we nontheless restrict the scope to proximal feedback for illustrative purposes.

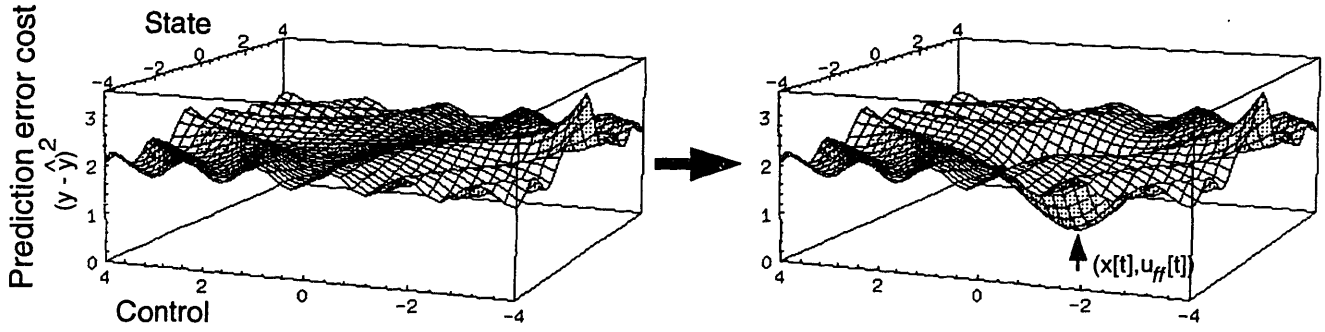


Figure 3-4: The local generalizing effect of on-line adaptation in the forward model of an single-state SISO system. The learning of an exemplar $y[t+1] = f(x[t], u_{ff}[t])$ at time step $t+1$ leads to the reduction of prediction error in a neighborhood of $(x[t], u_{ff}[t])$.

precise trajectory tracking control.

We stretch the approach further : what if we leave the feedback controllers out completely and *on-line adapt* the forward model ? i.e. the adaptation law is simply the forward model's learning of a single exemplar received after the action $\mathbf{u}_{ff}[t]$ has been executed - i.e. to *interpolate* the point $(\mathbf{u}_{ff}[t], \mathbf{x}[t], \mathbf{y}[t+1])$ with the mapping :

$$\mathbf{y}[t+1] = \hat{f}(\mathbf{u}_{ff}[t], \mathbf{x}[t], \mathbf{w}[t+1]), \tag{3.7}$$

at each time step, where $\mathbf{w}[t+1]$ is the *adapted* weight vector for next time step.

By adaptive tracking control we first require the *local* prediction accuracy of the forward model be improved. The main question of our approach is whether Equation 3.7 would:

1. improve the prediction accuracy in a neighborhood Ω of $(\mathbf{u}_{ff}[t], \mathbf{x}[t])$ of the forward model's input space at each time step (see Figure 3-4), or

2. cause severe distortion of the forward model's mapping.

If (1) could be achieved then, provided that the time step is small (such that $\mathbf{x}[t+1] \approx \mathbf{x}[t]$) and $\mathbf{y}^*[t+2] \approx \mathbf{y}^*[t+1]$, this would lead to more accurate action search in the next

time step, which in turn would result in $||\hat{\mathbf{y}}[t]-\mathbf{y}[t]|| \rightarrow 0$ ($\forall \mathbf{y}[t]$ within a neighborhood of $\mathbf{y}^*[t]$) as $t \rightarrow \infty$. Given a task specified in a smoothly varying sequence of $\mathbf{y}^*[t]$, the above implies that the feedforward action found from the forward model would produce $||\mathbf{y}[t] - \mathbf{y}^*[t]|| \rightarrow 0$ as $t \rightarrow \infty$.

Note also that the process of interpolating through a single exemplar is fast - it only takes about 5 iterations of the Scaled Conjugate Gradient Algorithm (see Moller 1990) in our simulations.

When the feedforward controller has been partially trained and has captured the underlying nominal dynamics of the system to the extent that fairly accurate feedforward control is achieved, we might then "on-line adapt" it by training it with one exemplar observed at each time step of the control. Our assessment of its success should be based on the improvement of performance over various time scales. At a short time scale, say within a trajectory, we might measure its success by whether the target state can be reached and by how far off is the actual trajectory from the intended trajectory. At a longer time scale, we might measure its success by inter-trajectory performance.

Thus, the main research work in this thesis set out to investigate the following :

1. if an accurate forward model is available, could accurate control be achieved without an accurate inverse model, i.e. solely by retrieving the control signals from the forward model ?

2. given a task specified in a sequence of $\mathbf{y}^*(t)$ that *varies smoothly* over time, would an approximate forward model be able to on-line adapt and provide the control signals for achieving $\mathbf{y}^*(t)$?

3. if the on-line adaptation is possible, to what extent of inaccuracies in the forward model be tolerated in completing a task ?

3.6 Control of a four degree-of-freedom arm in 3-D space

This section presents the results of experiments on the control of a simulated four degree-of-freedom arm in 3-D space under the influence of gravity. The arm is shown in Figure 3-5(a). The shoulder and elbow are ball joints and each has two degrees of freedom. The sampling time was set at eight milliseconds, which is coarse relative to the amount of time required for a full-span traverse at maximum acceleration (approximately 100 milliseconds).

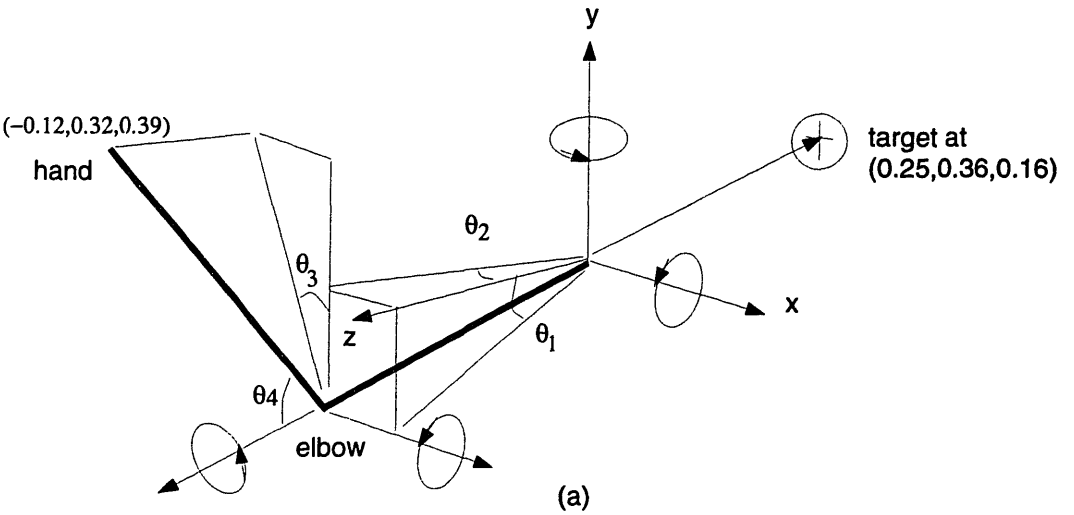
3.6.1 Forward model

The forward model was a feedforward network of three hidden layers of sigmoidal units, the first with 20 nodes and the second and third with 15 nodes. The input consisted of 21 linear units (4 of which are action inputs, 16 of which are state inputs and 1 as bias) while the output consisted of 4 linear units. The activation of the action units are constrained to be within τ_{min} and τ_{max}. The network had 1005 connections and learned to perform the following mapping:

$$\hat{\ddot{\boldsymbol{\theta}}}[t+1] = \hat{f}(\boldsymbol{\tau}[t], \boldsymbol{\theta}[t], \dot{\boldsymbol{\theta}}[t], \mathbf{w}[t]), \tag{3.8}$$

where $\hat{\ddot{\boldsymbol{\theta}}}$ is the vector of estimated joint accelerations, $\boldsymbol{\tau}$ is the torque vector, and the state variables $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ are the vectors of joint positions and velocities. The latter three variables were scaled and presented as inputs to the network. The input also consisted of sines and cosines of the components of $\boldsymbol{\theta}$.

The forward model was trained off-line with 30,000 exemplars until the relative



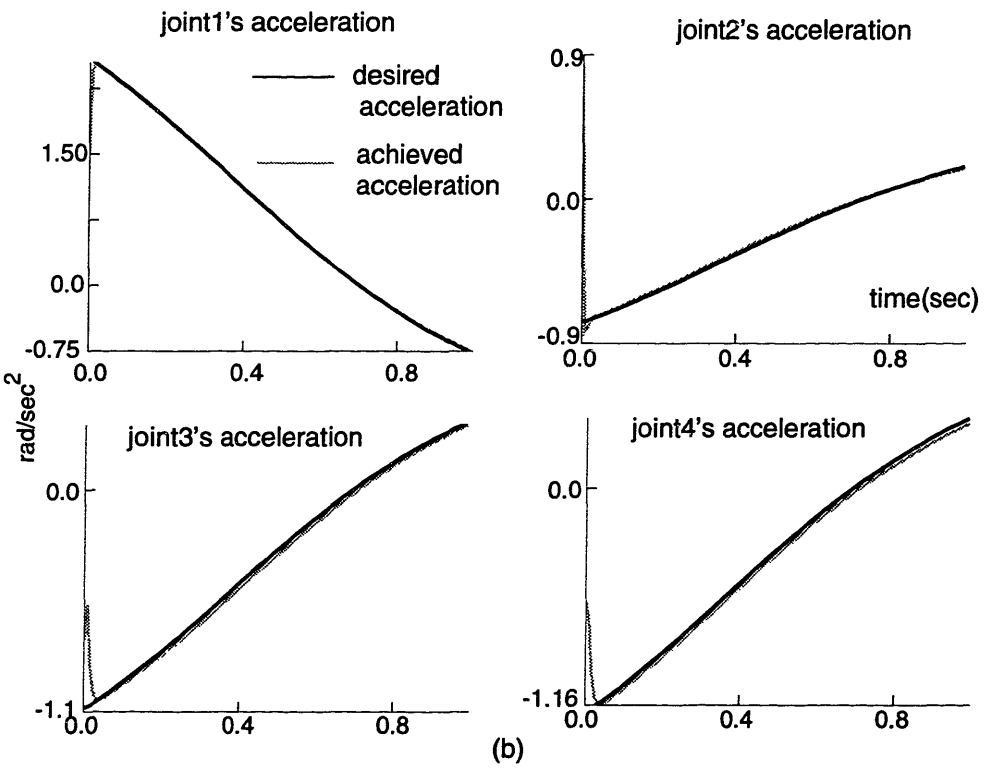


Figure 3-5: (a) The arm and the reaching task. The length of each link is 0.3 m. The center of mass of each link is at the center of the link. The maximum torques at the shoulder are 91.3 N m, whereas those at elbow are 15.6 N m. The experiments on robustness involved a reaching movement from the point $(-0.12, 0.32, 0.39)$ to the point $(0.25, 0.36, 0.16)$. The joint angles in the initial configuration were $(\theta_1, \theta_2, \theta_3, \theta_4) = (0.14, 0.24, 0.04, 1.04)$. (b) Performance of the controller with nominal dynamics in the reaching task. Note that the scales of the graphs of each joint are not the same. The performance error at the beginning of the trajectory is 2.66 rad/s^2. The target is reached in 1.05 seconds.

error[4] is ≈ 0.027. The scaled conjugate gradient(SCG) algorithm of Møller (1990) was used for off-line training, on-line adaptation and action search. This algorithm is much faster than the conventional back-propagation algorithm in the action search process. This is due to the fact that by the time the forward model's output's error was back-propagated(through all the hidden layers) to its input nodes, the magnitude of the error signal usually becomes too small, and it is thus difficult to determine a suitable fixed step size beforehand that would achieve a compromise between speeding up the process and stability. SCG uses the second order information of the process to automatically scale up the step size of control adjustment, and in practice a 75-80% reduction in the number of iterations required for a typical action search is common.

3.6.2 Controller

For the experiments reported here, *no* separate controller was used. Rather, the moving basin search process was initialized with a zero torque vector at each time step and the process was iterated until a control signal was found.

The reference signal for the controller was the desired acceleration computed by Equation 3.5 with $K_p = 3.5I$ and $K_v = 1.31I$, where I is the identity matrix.

The results for the reaching task under nominal dynamics are shown in Figure 3-5(b). As can be seen, the system moves the arm stably along the desired trajectory. Note that the forward model is not accurate enough to provide purely feedforward control - i.e. the arm would not be able to reach the target without its on-line adaptation.

[4]The relative error measure of network's accuracy is :

$$\epsilon_r = \frac{\sum_{i=1}^{N}(\mathbf{y}_i^* - \hat{f}(\mathbf{x}_i))^2}{\sum_{i=1}^{N}(\mathbf{y}_i^* - \overline{\hat{f}(\mathbf{x}_i)})^2} \tag{3.9}$$

where $(\mathbf{x}_i$ and $\mathbf{y}_i), i = 1,..,N$ are the training data and $\hat{f}$ is the approximating function. $\overline{\hat{f}}$ is the mean of $\hat{f}$.

Chapter 4

The Moving Basin Mechanism

If Mohammad had not gone to the mountain,
the mountain would have come to Mohammad.

4.1 Introduction

In the previous chapter we have presented simulation results that showed how the forward modeling idea has worked in accommodating a certain degree of prediction inaccuracies. However, as with most applied gradient-guided search processes, there are problems with local minima which could completely wreck the applicability of the approach. Indeed, in simulations of the adaptive control of four-joint robot manipulators, it has been observed that gradient-based neural network algorithms are not always able to find control signals that achieve specified endpoint accelerations throughout the state space. The problem is particularly acute early in learning, when the controller is often unable to propose an initial control signal that lies sufficiently close to a solution. As will be shown below, in robotic control the problem is not due to the presence of local minima, but arises instead because the search process reaches

the boundary of the search space. Whether the problem is caused by local minima or by control boundaries, failure to find an appropriate control signal can lead to instability.

This chapter describes an approach for achieving more effective search in gradient-based adaptive control. The approach is related to continuation methods in optimization theory (Allgower & Georg, 1990). We propose that the system begins by setting a target that is known to be achievable at the current point in the state space. The system then sets a succession of intermediate targets that can be achieved by a linked succession of "easy" searches. Within the framework of gradient-based optimization, this approach has an interpretation in terms of a "moving basin" of the cost function. The moving basin mechanism is shown to be able to solve adaptive control problems that simple gradient descent cannot.

4.2 The control boundary problem

In preliminary simulation experiments involving the control of a four-dof robot arm, we have found that the action-search process often causes one (or more) degree-of-freedom to be required to produce a torque that is greater or smaller than possible. One possible cause for this problem is that the desired output is actually not achievable. We have found, however, that the problem exists even in cases where the desired output is achievable, and would be predicted to be so by an accurate forward model if the initial predicted output were sufficiently near to the desired output. Moreover, the problem is not due to the presence of local minima in the form of stationary points, because it can be shown from a Lagrangian formulation of robot arm dynamics that joint accelerations are *monotonic* functions of the joint torques (Appendix A). Hence stationary points cannot appear in the action-search process if the forward model has been well-trained.

The control boundary problem can be understood by considering the dynamics of the action-search process in the forward model's output space. Define the *predicted achievable target set* $\hat{\mathcal{Y}}$ of a forward model $\hat{f}$ at time t as:

$$\hat{\mathcal{Y}} = \{\hat{\mathbf{y}} = \hat{f}(\mathbf{u}, \mathbf{x}[t], \mathbf{w}[t]) \mid u_j^{min} \leq u_j \leq u_j^{max}, 1 \leq j \leq n\}. \tag{4.1}$$

where $\mathbf{u} = (u_1, ..., u_n)^T$ is the control vector, u_j^{min} and u_j^{max} are the minimum and maximum values of u_j respectively, $\mathbf{w}[t]$ is the current weight vector of the forward model, and $\mathbf{x}[t]$ is the state at time t. For a system whose dynamics is monotonic, at least one of the u_j's is equal to its minimum or maximum value at any point on the boundary of $\hat{\mathcal{Y}}$.

For a fixed value of t, the action-search process yields a sequence of control vectors $\{\mathbf{u}^{(i)}[t]\}$:

$$\mathbf{u}^{(i)}[t] = \mathbf{u}^{(i-1)}[t] + \alpha \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T \Big|_{i-1} (\mathbf{y}^*[t+1] - \hat{\mathbf{y}}^{(i)}[t+1]), \tag{4.2}$$

where $\mathbf{u}^{(0)}[t]$ is the initial control vector provided by the feedforward controller, α the step size and $\{\hat{\mathbf{y}}^{(i)}[t+1]\}$ is the corresponding sequence of outputs obtained from the forward model:

$$\hat{\mathbf{y}}^{(i+1)}[t+1] = \hat{f}(\mathbf{u}^{(i)}[t], \mathbf{x}[t], \mathbf{w}[t]). \tag{4.3}$$

Because the search process utilizes the transpose Jacobian $(\partial \hat{\mathbf{y}}/\partial \mathbf{u})^T$ rather than the inverse Jacobian $(\partial \hat{\mathbf{y}}/\partial \mathbf{u})^{-1}$, the path $\{\hat{\mathbf{y}}^{(i)}[t+1]\}$ followed in the output space is generally nonlinear even for *affine* systems (see Appendix A). Let the angle between the lines $\mathbf{v} = \mathbf{y}^*[t+1] - \hat{\mathbf{y}}^{(i)}[t+1]$ and $\mathbf{z} = \hat{\mathbf{y}}^{(i+1)}[t+1] - \hat{\mathbf{y}}^{(i)}[t+1]$ be γ :

$$cos(\gamma) = \frac{< \mathbf{z}, \mathbf{v} >}{\|\mathbf{z}\| \, \|\mathbf{v}\|} \tag{4.4}$$

For an affine system, Equations 4.2 and 4.3 yield :

$$\mathbf{z} = \alpha \left. \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}} \right|_{i-1} \left. \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T \right|_{i-1} \mathbf{v}$$

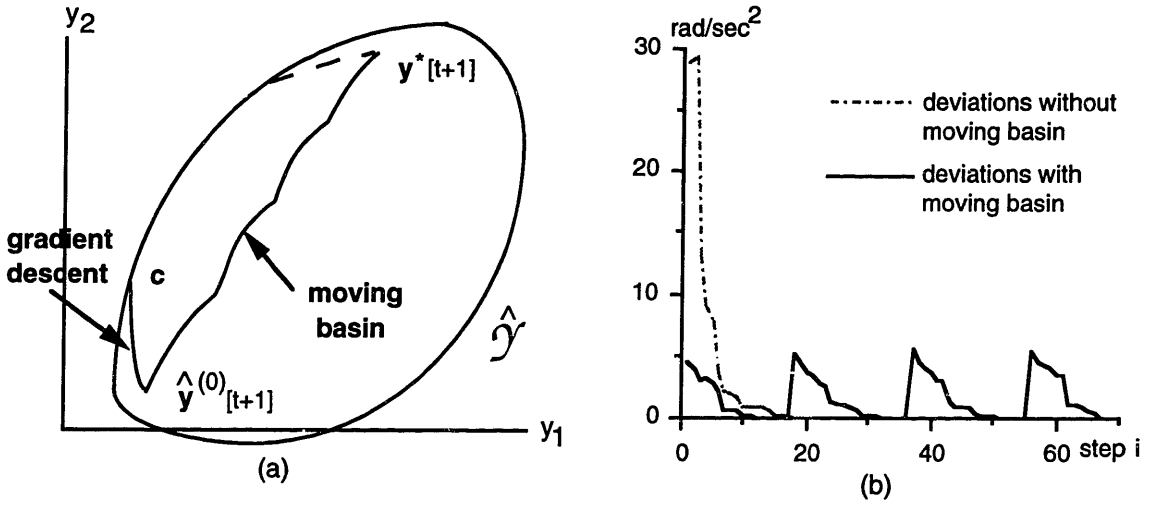


Figure 4-1: (a). The path $\{\hat{\mathbf{y}}^{(i)}[t+1]\}$ taken by steepest descent is generally nonlinear and often reaches the boundary of the predicted achievable target set $\hat{\mathcal{Y}}$. The moving basin mechanism tends to straighten the path, allowing the target to be reached. (b) A plot of the deviations of $\hat{\mathbf{y}}^{(i)}[t+1]$ from the line joining $\hat{\mathbf{y}}^{(0)}[t+1]$ and $\mathbf{y}^*[t+1]$ in an actual action-search process, with and without the use of moving basin mechanism. With a 4-step target-shift the deviation is reduced by approximately 83 percent.

$$\Rightarrow \quad |cos(\gamma)| = \frac{\left| \mathbf{v}^T \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}\right|_{i-1} \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T\right|_{i-1} \mathbf{v} \right|}{\|\mathbf{v}\| \left\| \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}\right|_{i-1} \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T\right|_{i-1} \mathbf{v} \right\|} \leq 1$$

$$\Rightarrow \quad \left| \mathbf{v}^T \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}\right|_{i-1} \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T\right|_{i-1} \mathbf{v} \right| \leq \|\mathbf{v}\| \left\| \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}\right|_{i-1} \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T\right|_{i-1} \mathbf{v} \right\|$$

The equality for the above (i.e. $\gamma = 0°$) holds only when $\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T$ is a *scaling* matrix - i.e. $\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T = \beta \mathbf{I}$, where β is a scalar and $\mathbf{I}$ is the identity matrix. However in general $\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T \neq \beta \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^{-1}$, which implies $\gamma \neq 0°$. We often find in practice that these nonlinear paths reach the boundary of the predicted achievable target set $\hat{\mathcal{Y}}$ if $\left\| \mathbf{y}^*[t+1] - \hat{\mathbf{y}}^{(0)}[t+1] \right\|$ is large, as illustrated in Figure 4-1(a).

One approach to dealing with the problem would be to project the gradient onto $\hat{\mathcal{Y}}$ and to continue the search along the boundary $\partial \hat{\mathcal{Y}}$ (cf. Rosen, 1967). However, because it cannot be guaranteed that an approximate forward model yields a convex

predicted achievable target set, this approach is not feasible.[1] The search could not continue outside the boundary because this would require the forward model performing extrapolation, and, due to the nonlinear mapping of the forward model, the search may not return to $\hat{\mathcal{Y}}$. Furthermore if the action nodes are logistic[2], $\hat{\mathcal{Y}}$ will become an open set (because at $\partial\hat{\mathcal{Y}}$ one or more of the action nodes' output will be at its limit, and this is possible only if the magnitude of node's activation $\rightarrow \infty$), and it is possible that the action search process causes $\hat{\mathbf{y}}^{(i)}[t+1]$ approach $\partial\hat{\mathcal{Y}}$ asymptotically. In this case action search will fail even if $\hat{\mathcal{Y}}$ is convex.

The control boundary problem is problematic for the distal supervised learning approach in both on-line adaptation and off-line training of the controller. In on-line adaptation, because the system fails to move along the path leading to the goal, the forward model fails to receive useful training data. An even more serious problem is that the controller tends to have its weights set to values of high magnitude in order to produce actions that are near the boundary. The control boundary problem therefore tends to persist in subsequent time steps and the system can become unstable.

The control boundary problem is an example of a larger class of problems that are encountered when using gradient search techniques for nonlinear adaptive control. Gradient techniques are based on estimates of the partial derivatives of the plant output with respect to the control variables (cf. Equation 4.2). Control variables tend to be small in number and therefore act as a bottleneck for the gradient search process. This leads to performance surfaces that are complex as viewed through the bottleneck. The approach often taken to deal with complex performance surfaces–that of increasing the dimensionality of the search space–is not an option in the adaptive control problem because the dimensionality of the control space is fixed by the nature of the plant.

---

[1]Indeed we find empirically that the set $\hat{\mathcal{Y}}$ is generally nonconvex for a partially-trained forward model. The convexity of $\mathcal{Y}$ could not be easily captured because the forward model is presented only with exemplars within the boundary of $\mathcal{Y}$, whereas it needs training with exemplars on both sides of $\partial\mathcal{Y}$ in order to effectively produce accurate mapping around $\partial\mathcal{Y}$.

[2]See Appendix B for a discussion on the advantages of using logistic action nodes.

4.3 The moving basin mechanism

We have developed a very simple approach for retrieving the wanted information from the forward model in an efficient way that avoids the control boundary problem. This idea is somewhat analogous to Werbos' notion of "moving targets" (Werbos, 1983). Werbos used the term "moving targets" to describe the qualitative idea that the network should "set itself intermediate objectives, and vary these objectives as information is accumulated on their attainability and their usefulness for achieving overall objectives." Our algorithm works as follows. For a fixed value of t and a fixed value of the state $\mathbf{x}[t]$, the system chooses a *grounding action* $\mathbf{u}_g[t]$. This can be chosen arbitrarily or can be taken to be the output of the feedforward controller. The forward model is used to estimate the *grounding output* $\hat{\mathbf{y}}_g[t+1]$ that would be obtained if that action were used as a control signal:

$$\hat{\mathbf{y}}_g[t+1] = \hat{f}(\mathbf{u}_g[t], \mathbf{x}[t], \mathbf{w}[t]). \tag{4.5}$$

As shown in Figure 4-2, the pairing of the grounding action and the grounding output essentially induces an error surface whose minimum is located at the known location $\mathbf{u}_g[t]$. That is, if the system wished to achieve $\hat{\mathbf{y}}^{(1)}[t]$ as a target, then the grounding action $\mathbf{u}_g[t]$ would be the appropriate action. The system now effects a series of steps in which a sequence of virtual targets are interpolated along the path between the grounding output and the desired output $\mathbf{y}^*[t+1]$. At each step, the action-search process is utilized to find a locally-optimal control action. This action is then used to initialize the search at the next step. As shown in Figure 4-2, this process has an interpretation in terms of a moving basin of the cost function that is tracked by a linked sequence of action searches.

By setting intermediate targets along the line joining the commanded goal $\mathbf{y}^*[t+1]$ and the grounding goal $\hat{\mathbf{y}}_g[t+1]$, the system reduces the effect of the nonlinearity of the search path. By following a straighter path, the system reduces the probability

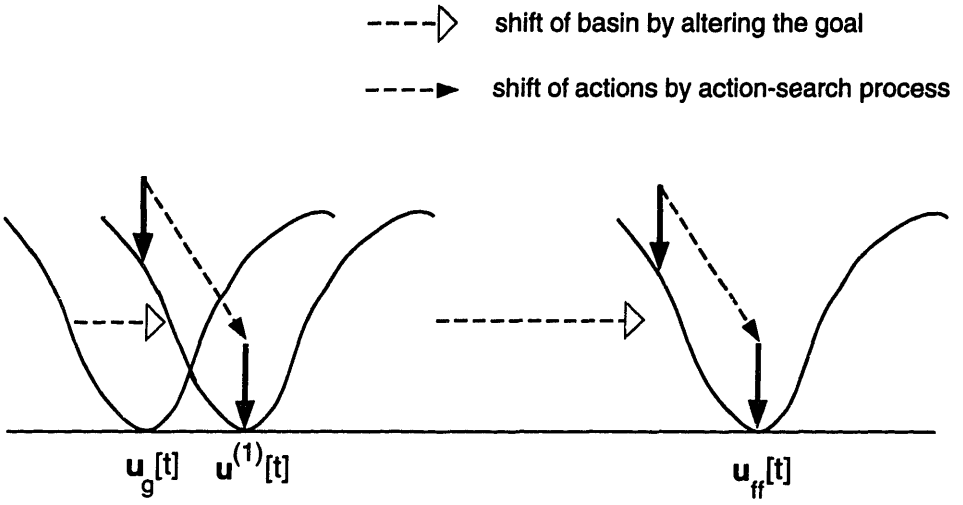


Figure 4-2: The basin on the left corresponds to the cost J when $\hat{\mathbf{y}}_g[t+1]$ is the target, and the basin on the right corresponds to the cost J when $\mathbf{y}^*[t+1]$ is the target. The basin of J is shifted by slowly shifting the virtual target from $\hat{\mathbf{y}}_g[t+1]$ to $\mathbf{y}^*[t+1]$, and the action-search finds the locally-optimal action $\mathbf{u}^{(i)}[t]$ for each virtual target until the feedforward action $\mathbf{u}_{ff}[t]$ is found. Note that the state and the forward model's parameters are invariant during the search process.

of meeting the boundary of the predicted achievable target set (see Figure 4-1).

4.3.1 Choosing the starting point

There are advantages to choosing the *null* or *neutral* action as the grounding action. When the predicted achievable target set is nonconvex, a neutral grounding action produces a grounding goal which is near the geometric center of the set, and the straightened search path has a better chance of reaching all of the points in the set[3]. Also, in problems involving a redundant plant, choosing a grounding action at the origin of the control space increases the tendency of the action-search process to find solutions in which the norm of the control signal is small (i.e., "least-effort" solutions - also see Appendix B). The symmetry of the actual achievable target set $\mathcal{Y}$ is easily captured by the forward model during its off-line training because the geometric center

[3]The dynamics of a serial-link arm has this property of symmetry in the controls (see Appendix A).

of $\mathcal{Y}$ is well-interpolated by the exemplars in the training set. The utilization of this property by the action search process thus compensates the difficulty encountered in capturing the convexity of $\mathcal{Y}$ by the forward model.

4.3.2 Stationary points

The action search would succeed for dynamical systems of more general form:

$$\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t), t) + \mathbf{b}(\mathbf{x}(t), \mathbf{u}(t), t);$$

where $\mathbf{b}(\mathbf{x}(t), \mathbf{u}(t), t)$ is at least C^1 in $\mathbf{u}(t) = (\bar{u_1}, .., \bar{u_n})^T(t) \in \mathcal{U}$(The feasible control set), and

$$\left.\frac{\partial \hat{\mathbf{y}}}{\partial u_i}^T\right|_{\bar{\mathbf{u}}} \neq 0, i = 1, ..., n; \forall \mathbf{u}$$

-i.e. the dynamics is *monotonic* in the controls.

The moving basin is not capable of handling local minima in the form of stationary points, which exists if the plant's dynamics is *non-monotonic* in the controls - i.e. :

$$\exists \mathbf{u} \text{ such that } \left.\frac{\partial \hat{\mathbf{y}}}{\partial u_i}^T\right|_{\bar{\mathbf{u}}} = 0, \text{ for some } i = 1, ..., n;$$

, These non-monotonicities correspond to *folds* in the achievable target set. In that case the common approach of including a momentum term in the action search process may be effective in overcoming the problem of stationary points :

$$\mathbf{u}^{(i)}[t] = \mathbf{u}^{(i-1)}[t] + \beta \Delta \mathbf{u}^{(i-1)}[t];$$

$$\Delta \mathbf{u}^{(i-1)}[t] = \alpha \left.\frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{u}}^T\right|_{i-1} (\mathbf{y}^*[t+1] - \hat{\mathbf{y}}^{(i)}[t+1]) + \Delta \mathbf{u}^{(i-2)}[t];$$

where α, β are constants, and $\Delta \mathbf{u}^{(0)}[t] = 0$.

4.4 Simulation Results

In order to observe how much improvement can be gained with the use of moving basin idea, it was applied on the control of the simulated 4-dof arm described in Section 3.6. The moving basin algorithm utilized a linear interpolatory process with a variable step size. A recursive procedure was used to refine the step size up to a certain maximum resolution before the algorithm gave up and issued the feedforward action that was predicted to achieve the virtual target closest to the commanded goal, plus a small stochastic component. In the following subsections we first show the results comparing the performance with and without the use of the mechanism within a specific trajectory in details. This is to be followed by results comparing inter-trajectory performance, giving a more global view of the effectiveness of the idea.

4.4.1 Performance within a trajectory

The moving basin idea was applied onto a reaching task depicted in Section 3.6, with the torque vector at the start of the action-search process at each time step of the control biased by a certain amount. Figure 4-3 shows the difference in performance with and without the moving basin. In the case where the moving basin is not utilized, the initial torque was biased to 95% of the maximum torque (i.e. $\hat{\mathbf{y}}^{(0)}[t+1]$ is close to the boundary).

4.4.2 Global Performance

As has been stated earlier, the occurrence of control boundary problem also depends on the initial $\hat{\mathbf{y}}^{(0)}[t+1]$. A comparison of the performance with and without using the moving basin is made in 500 reaching tasks using the simulated 4-dof arm, whose

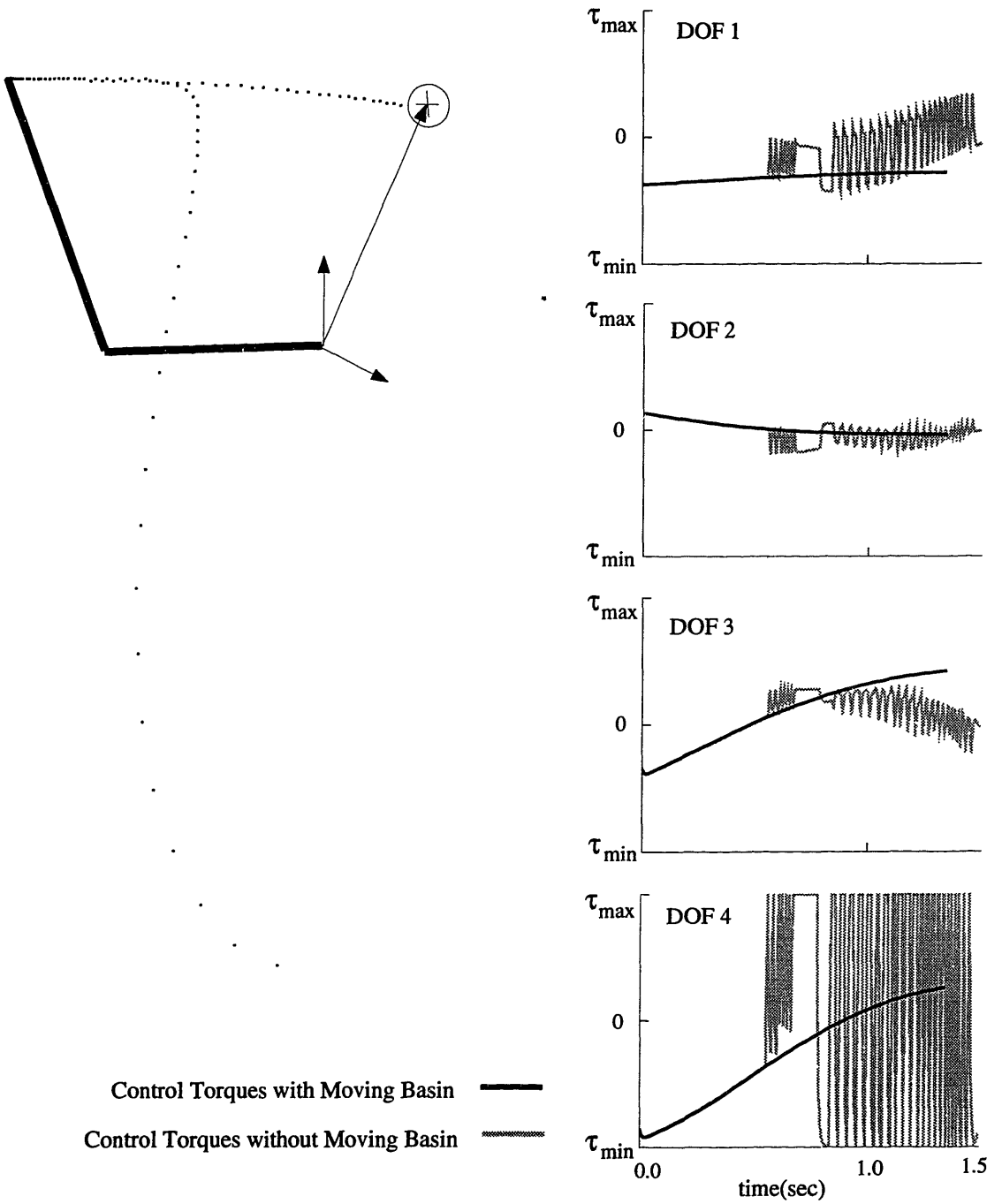


Figure 4-3: (a) Performance with and without the moving basin. The hand is able to reach the target with a fairly straight path with using the moving basin. Without the moving basin the control is able to track the desired trajectory up to $\approx 0.5sec$, after which the control boundary problem occurs and the hand strays and exhausts one of its degrees of freedom.

α	0.0	0.3	0.5	0.7	0.8	0.9	0.98
Without moving basin	0	10	36	49	39	109	195
With moving basin	0	0	0	0	1	31	79

each initial joint position is randomly set to within 90% of the span of each joint angle, and the desired acceleration computed as follows :

$$\ddot{\boldsymbol{\theta}}^*[t+1] = 4(\boldsymbol{\theta}^* - \boldsymbol{\theta}[t]) + \frac{8}{3}(\dot{\boldsymbol{\theta}}^* - \dot{\boldsymbol{\theta}}[t]), \tag{4.6}$$

The initial grounding action $\mathbf{u}^{(0)}[t]$ is :

$$\mathbf{u}^{(0)}[t] = \alpha \times \mathbf{u}^{max} \tag{4.7}$$

α is a bias factor within the range [0, 1) and $\mathbf{u}^{max}$ the maximum torque vector. The torque of the forth degree of freedom has been increased threefold. Larger α means that the $\hat{y}^{(0)}[t+1]$ is closer to the boundary, and hence has a higher chance of being trapped. The table below shows the number of failed trajectories with different α:

The reaching task is considered a failure if one or more of the joints reaches its limit. Note that since the actual task is to achieve the desired acceleration at each time step, the more appropriate measurement of the approaches' success should be the amount of reduction in the total deviations from desired accelerations for all the trajectories. However this could not reflect the true performance either, as it also depends on the adaptive capability of the forward model. The result nontheless indicates that the distance of $\hat{\mathbf{y}}^{(0)}[t+1]$ from the boundary has a significant effect on the attainability of the solutions. It is also interesting to note that there is no failed trajectory with $\alpha = 0$ even without the moving basin, indicating that $\hat{\mathcal{Y}}$ is fairly symmetric[4] and curved paths of $\hat{\mathbf{y}}^{(i)}[t+1]$ are tolerated by the approach. However the use of moving basin does reduce the number of failed trajectories for higher

[4]The scaling of the control torques affects the dynamics in an affine manner. Hence the symmetry of $\hat{\mathcal{Y}}$ is not affected.

α, indicating that in situations where $\hat{\mathcal{Y}}$ is not symmetric, the moving basin would increase the reachability of any point within $\hat{\mathcal{Y}}$ with $\mathbf{u}^{(0)}[t] = \mathbf{0}$.

4.5 Discussion

This chapter describes a computational technique that allows gradient-based methods to be used more effectively in nonlinear adaptive control problems. By utilizing the moving-basin search procedure and the symmetry of the dynamics, the control system in our simulations is able to avoid the control boundaries and thereby stably track the reference trajectory.

Although the simulations studied a simple version of the moving-basin algorithm in which the search procedure was initialized with the zero torque vector at each time step, it is straightforward to cache the results of the search at one time step to initialize the search at the following time step. Because reference signals tend to change smoothly in time, the cached value will often be sufficiently close to a solution such that the local gradient search will succeed. This implies that the moving-basin search will generally be required only at the onset of a movement and at other points where the desired trajectory changes unpredictably.

Chapter 5

Adaptive Control of a 4-dof arm

It is a profound and necessary truth that the deep things in science are not found because they are useful; they are found because it was possible to find them.

- Oppenheimer

5.1 Introduction

The single-trajectory performance of the control of the simulated 4-dof arm described in Chapter 3 provided preliminary results showing the robustness of the use of forward model with on-line adaptation. This chapter reports on the more comprehensive and detailed empirical results obtained via the adaptive control of the simulated 4-dof arm. I show that with the inclusion of noise in the inputs during the on-line adaptation process, certain types of perturbations to the modeled generic dynamics can be reflected in the weight changes in the forward model. These findings, together with the attempt to explain them, will be presented in this chapter.

5.2 Robustness to Perturbed Dynamics

A series of experiments is performed to investigate the robustness of the control system with respect to inaccuracies in the forward model. The experiments involved training a forward model to perform the mapping of nominal dynamics of the 4-dof arm described in Section 3.6 and then assessing how it performs when the plant dynamics are altered in various ways. Subsections 5.2.1, 5.2.2 and 5.2.3 report on the three sets of experiments performed on a single trajectory for the reaching task described in section 3.6 to provide an intra-trajectory performance measure of the robustness of the approach. The performance error is measured using $\|\ddot{\theta}^*[t] - \ddot{\theta}[t]\|_2$ (in rad/sec^2) along the trajectory. Note that the maximum $\|\ddot{\theta}^*[t] - \ddot{\theta}[t]\|_2$ under nominal condition is ≈ 3 rad/sec^2. Section 5.3 reports the inter-trajectory performance obtained through simulations.

5.2.1 Performance with changes of the links' masses

In this set of experiments the mass of the first link of the arm was scaled by a certain constant factor without (off-line) re-training of the forward model. Figure 5-1 shows the performance with changes in the mass of link 1. The trajectory is completed stably with up to 510 percent increase of the mass of the first link or 5 percent increase of the mass of the second link, or with 50 percent decrement of either mass. Beyond these values the system becomes unstable. The smaller tolerance with respect to increases in the mass of the second link is due to the fact that near-boundary torque is required at the second joint to achieve the desired initial acceleration. Note that it takes less than 0.2 sec (< 25 exemplars) to achieve near-perfect tracking.

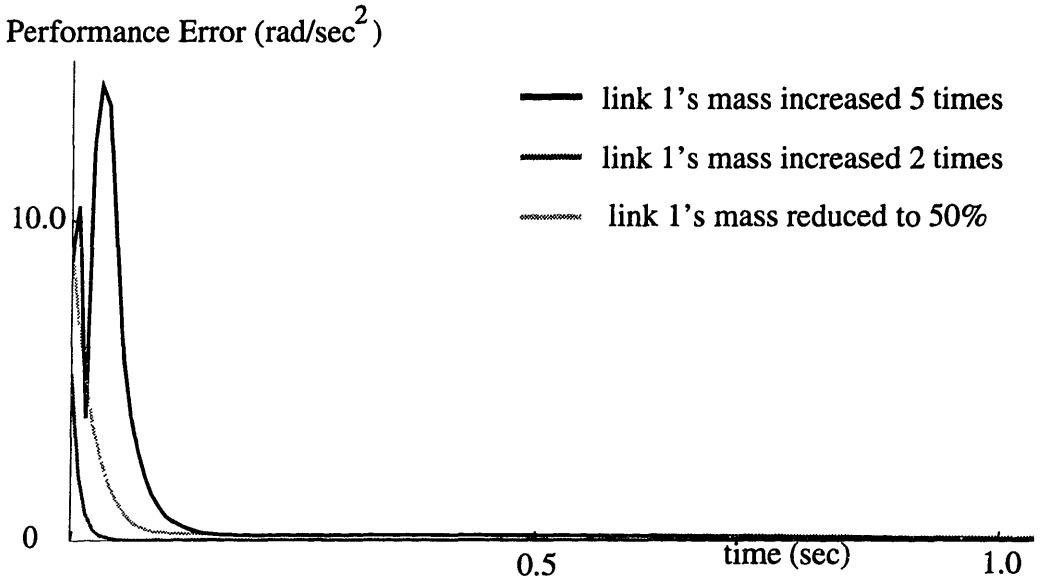


Figure 5-1: Performance with mass of link 1 changed by a constant factor. The target is reached within 1.3 seconds in all cases.

5.2.2 Performance with errors in the joint angle sensors

In this set of experiments the values of θ were either scaled by a constant factor or displaced by a constant amount.[1] As shown in Figures 5-2 and 5-3, the trajectory is completed stably when θ is scaled by as much as 1.65 or displaced by as much as 1 radian. This is significant robustness, given that the span of the joint angles are ± 1.57 radians.

5.2.3 Performance with errors in joint velocity sensors

In this set of experiments the readings of $\dot{\theta}$ were either scaled by a constant factor or displaced by a constant amount. The target is reached stably with a scaling factor of up to 20 (Figure 5-4), or displacements of as much as 14.5 rad/sec (Figure 5-5).

[1]Note that in the experiments with perturbations to the sensor readings, the information provided to the trajectory planner (i.e. Equation 3.5) remains accurate, such that the arm would reach the target if the desired accelerations at each time step were achieved.

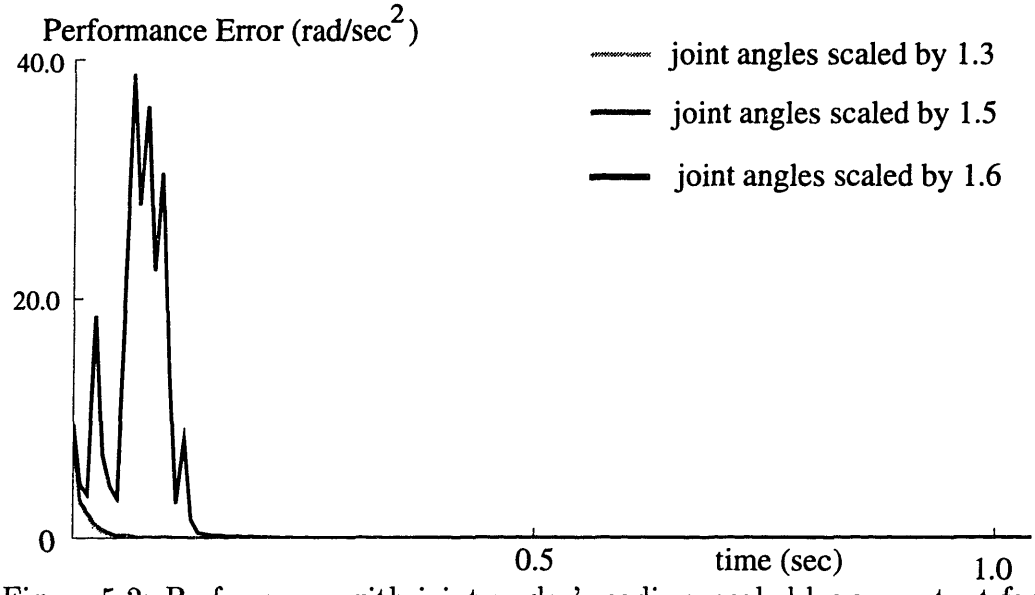


Figure 5-2: Performance with joint angles' readings scaled by a constant factor.

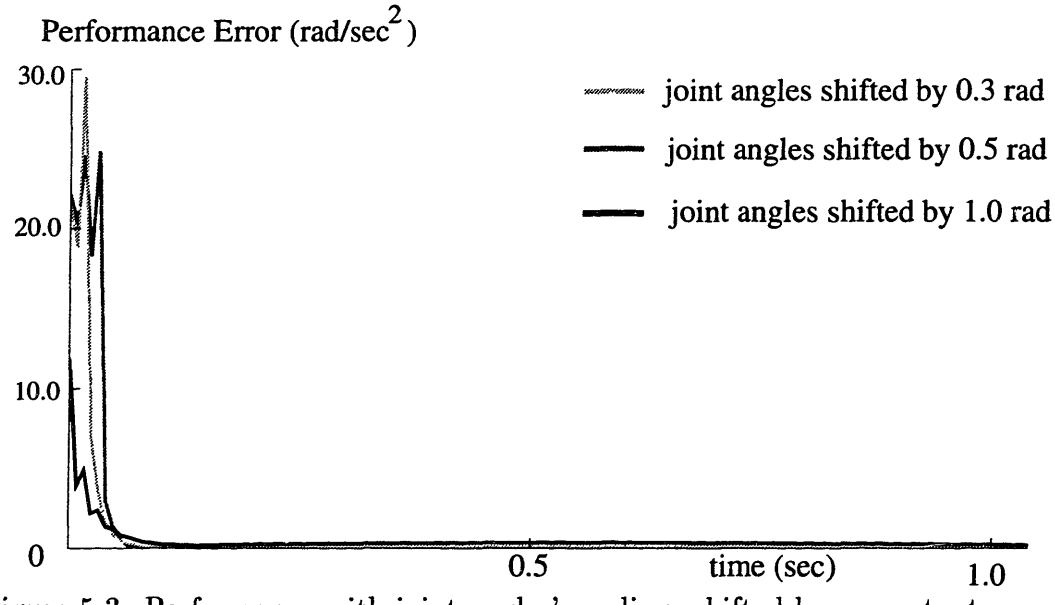


Figure 5-3: Performance with joint angles' readings shifted by a constant amount.

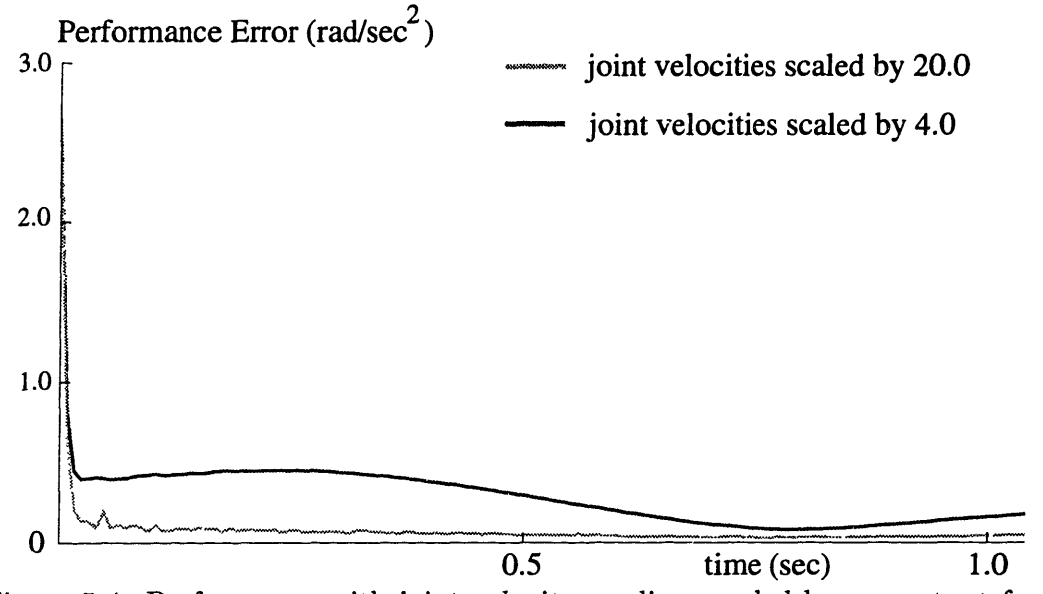


Figure 5-4: Performance with joint velocity readings scaled by a constant factor.

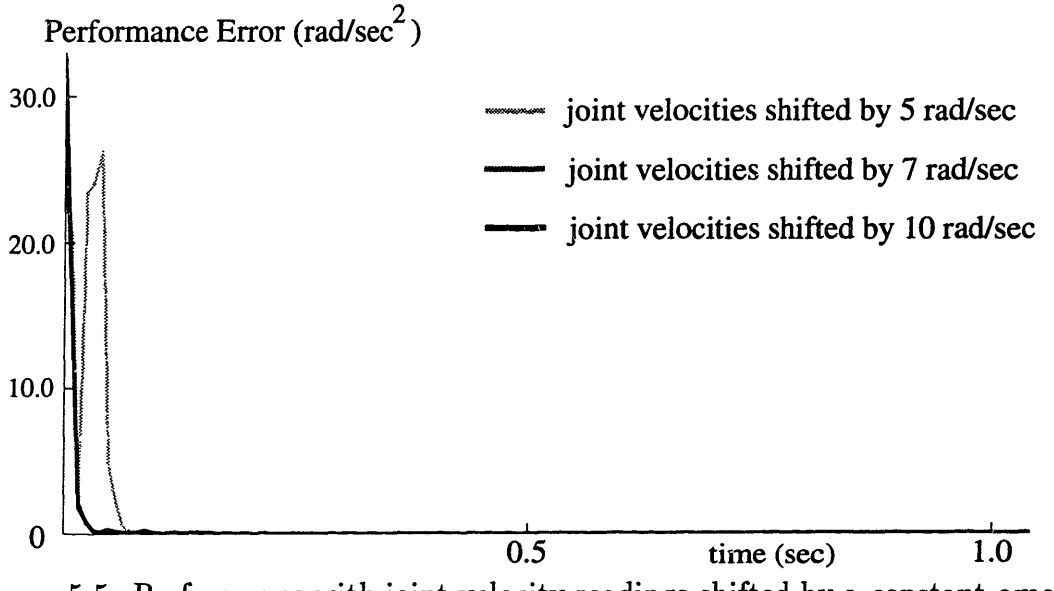


Figure 5-5: Performance with joint velocity readings shifted by a constant amount.

5.3 Inter-Trajectory Performance

As stated in Chapter 3, the utilization of a non-adaptive control method with fringe PID feedback controllers could also yield local improvement in performance within a trajectory such as those reported in Section 5.2.1 through 5.2.3. Viable adaptive control schemes should not only yield such local improvement, but also global improvement in performance via on-line adaptation through a relatively small number of on-line exemplars. A measure of this aspect could be established via the assessment of performance improvement across trajectories.

Since the current main interest is in assessing the robustness of the control in the domain where the forward model has been well-trained and the task well-defined and achievable, the global performance evaluation (which requires a good trajectory planner, an integral part of a good overall control scheme) has not been extensively studied. Nonetheless it was found that with appropriate choices (by manual search) of the coefficients in the simple trajectory planner used (Equation 3.5), most reaching tasks within reachable space can be completed within seconds under nominal dynamics. Figure 5-6a shows the inter-trajectory performance of the approach with the fourth degree-of-freedom's strength increased by threefold. The 400 trajectories are randomly-selected reaching tasks with the starting posture restricted to within 90% of the joints' spans.[2] Equation 3.5 with the coefficients set to $4I$ and $\frac{4}{3}I$ respectively is used as the trajectory planner. The moving average of the maximum performance error length is calculated with a window size of 25 by :

$$e_p(i) = \frac{\sum_{j=i-12}^{i+13} \|\ddot{\boldsymbol{\theta}}[t] - \ddot{\boldsymbol{\theta}}[t]^*\|_\infty^j}{25}, 13 \leq i \tag{5.1}$$

[2]The control is considered a failure if one or more of the joints reaches its boundary. In this sense it is not a very fair assessment of the adaptive capability if the initial posture is restricted to within 100% of the joints' spans, as there would be cases where the arm's first exploratory move cause one or more of the joints to reach its boundary.

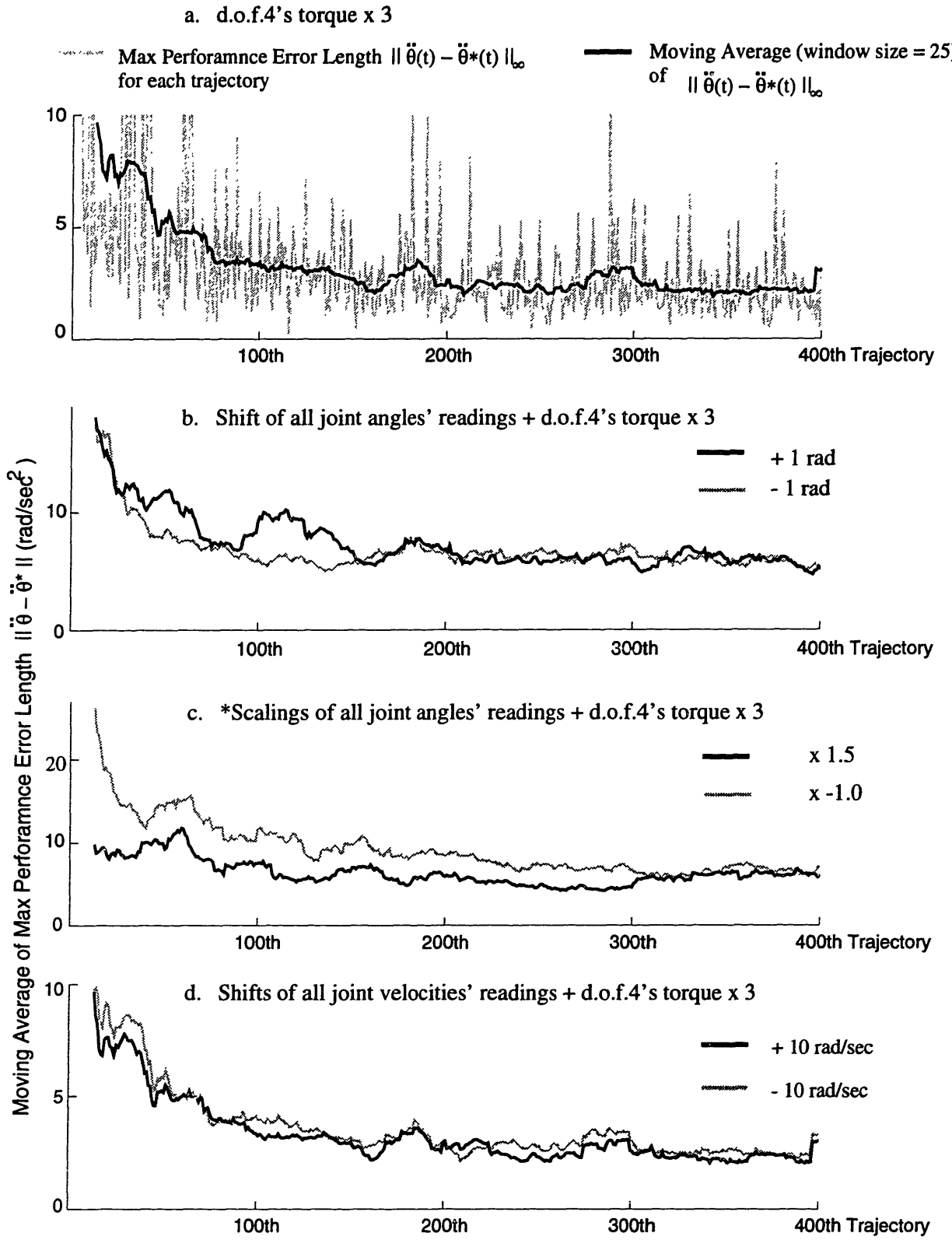


Figure 5-6: Performance for distinct trajectories under various perturbations. (a) shows the average (in dark line) of the maximum performance errors encountered in each trajectory (in grey line) when the d.o.f.4's torque is increased threefold. (b) to (d) show the averages of the max performance errors under respective listed perturbations to the dynamics.

where

$$\|\ddot{\boldsymbol{\theta}}[t] - \ddot{\boldsymbol{\theta}}[t]^*\|_\infty^j = \max_t \|\ddot{\boldsymbol{\theta}}[t] - \ddot{\boldsymbol{\theta}}[t]^*\|^j$$

is the maximum performance error length of the jth trajectory. Figure 5-6b, c and d show the inter-trajectory performances with shifts and scalings of the joint angles' readings and shifts of the joint velocity readings (in addition to the threefold increase of d.o.f. 4s' torque) respectively. Note that the target is reached in all the trajectories executed except in the first and forth trajectories of the case where the joint angles' readings are scaled by -1. The results show that the performance errors tend to be reduced as more trajectories are executed - i.e. the on-line adaptation of the forward model does lead to improvement in performance across trajectories.

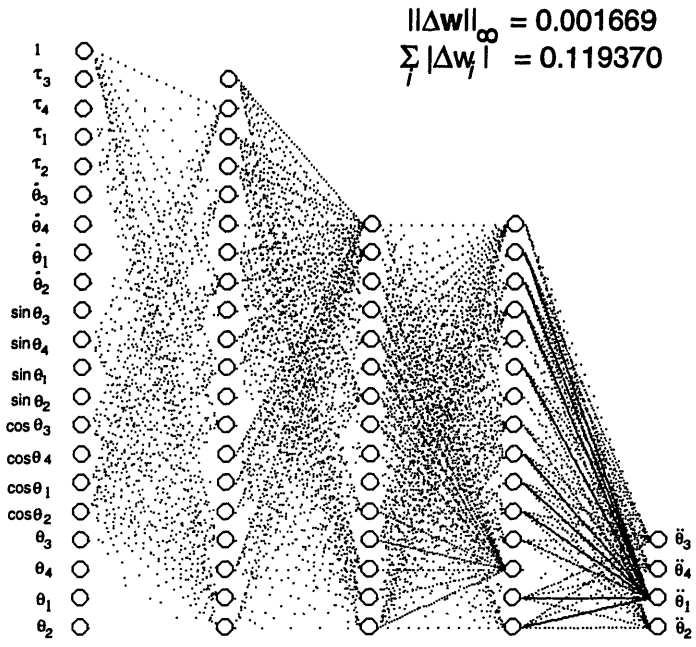
5.4 Properties of the adapted network

We further investigate the features of changes within the forward model after certain amounts of on-line adaptation. Such effort is made in the hope that it might lead to more formal characterization of the robustness and stability of the approach.

5.4.1 Distributed changes

As expected, the on-line adaptation process led to distributed changes throughout the whole network (see Figure 5-7), which is the result of the massive amount of crosstalk within the forward model. The maximum magnitude of a single weight's change is small compared to the maximum magnitude of a connection in the original network and to the sum of the magnitudes of the weight changes. However, the property of backpropagation training is reflected in the network's changes - that larger magnitude changes in the weights occurred at the layers closer to the output.

(a) Nominal dynamics



(b) with scaling of link1's mass by 5.0

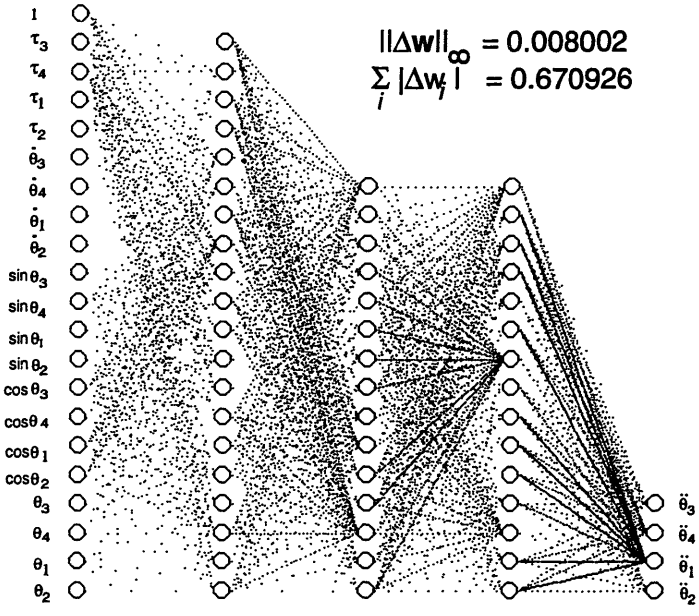
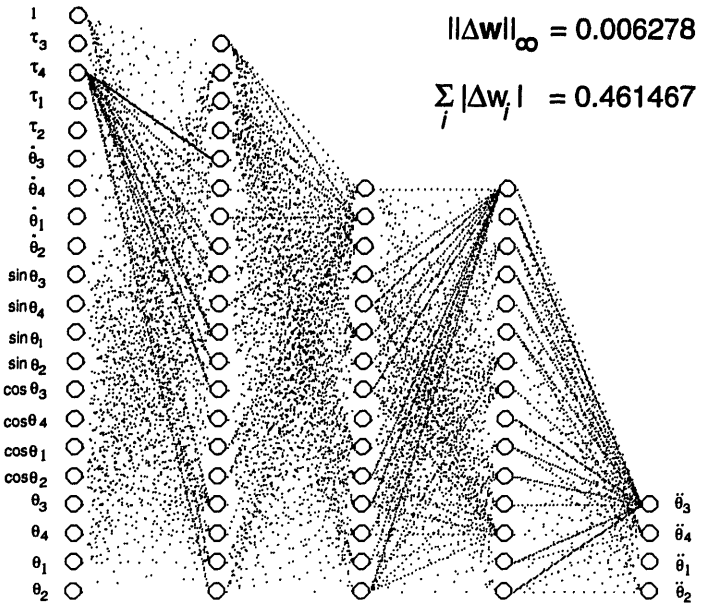


Figure 5-7: The changes in the forward model after the arm has reached the target in the reaching task. The density of a connecting line is proportional to the magnitude of change of the connection. The darkest line corresponds to the maximum magnitude of weight change ($\|\Delta \mathbf{w}\|_\infty$). (a) shows the changes under nominal dynamics. (b) shows the changes with scaling of link1's mass by 5.0 (ref Figure 5-1). In both cases the $\|\Delta \mathbf{w}\|_\infty$ is small compared to the original forward model's maximum weight of 2.377.

(a) DOF4's torque scaled by 3.0



(b) Combined DOF4's torque scaled by 3.0 and white noise at input

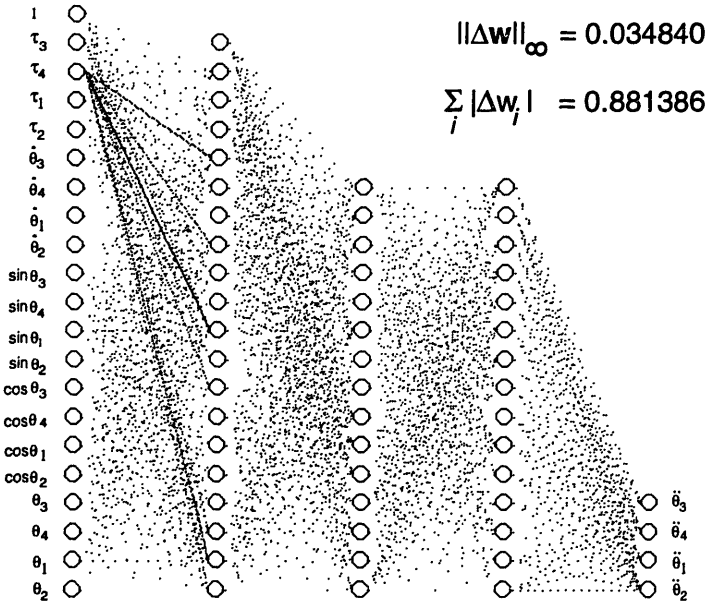


Figure 5-8: The changes in the forward model after the completion of the second reaching task. The density of a connecting line is proportional to the magnitude of change of the connection. The darkest line corresponds to the maximum magnitude of weight change ($\|\Delta \mathbf{w}\|_\infty$). (a) shows the much-distributed changes with scaling of DOF4's torque by 3.0 only. The actual perturbation in the plant cannot be observed. (b) shows the changes with scaling of DOF4's torque by 3.0 and white noise at state input. The much larger changes at the connections proximal to DOF4's input indicates that the main problem is at DOF4's input. Note also that although $\sum_i |\Delta w_i|$ increases by only 91% in (b), $\|\Delta \mathbf{w}\|_\infty$ increases by 455%.

5.4.2 "Shaking out" of perturbation information with white noise

It was also found that the on-line adaptation process with white noise present at the input sensors results in the information about certain kinds of perturbations in the actual plant being "shaken out" (by the forward model's experimentation with actions), as reflected in the localization of connection changes after the completion of a single reaching task. Figure 5-8 shows the changes in the forward model after the completion of a reaching task which starts at the hand position $(0.25, -0.01, -0.8)$ to the point $(-0.45, -0.31, 0.15)$. If there is no white noise present, the changes within the forward model after the completion of the trajectory is rather distributed (see Figure 5-8a). However if white noise were present at the state inputs, the on-line adaptation process results in localized changes concentrated at the weights directly connected to d.o.f. 4's torque input, hence providing a well-contrasted image reflecting the persistent error in d.o.f. 4's input (see Figure 5-8b). This contrast can be quantified by the ratio of average magnitude of change of weights connected to node j to the average magnitude of change of weights of the rest of the network :

$$H(node_j) = \frac{\frac{1}{n_j}\sum_{i=1}^{n_j}|\Delta w_{ji}|}{\frac{1}{N-n_j}\sum_{i,k\neq j}^{N}|\Delta w_{ik}|} \tag{5.2}$$

where n_j is the number of connections from node j, w_{ji} are connections from node j and N is the total number of connections of the network. For example $\max_j H(node_j)$ of the network in Figure 5-8b is $H(\tau_4) = 11.75$.

There are some peculiar points about the significance of this result. As described in Section 5.4.1, the largest magnitudes of change of weights normally occurs at the layers proximal to the output layer. Furthermore how could the prediction errors (caused by persistent perturbations), reflected at the outputs, find their way through the 3 hidden layers and accumulate at the appropriate sites at the input layer?

It is well known that the addition of noise to the input data of a neural net-

work during training can, in some circumstances, lead to significant improvements in generalization performance. Previous work (see Bishop, 1994) has shown that such training with noise is equivalent to an *explicit* form of regularization in which an extra term is added to the error function. A property of this extra regularization term is that its effect diminishes with respect to the order of the derivatives.

It is quite obvious that certain types of dynamical perturbations could be deemed as a form of *more-structured* noise of the underlying mapping, and thus there might be a link between the regularization effect of noisy input and the "shaking out of structural information" phenomenon. The following subsections describe an attempt to explain such a phenomenon by linking the rigorous analysis of the noisy input's regularization effect to the empirical observations.

Tikhonov Regularization and Training with Noise

In Bishop (1994), it has been shown that the inclusion of noise in the inputs while training a neural network basing on minimizing the sum-of-squares error function

$$E = \frac{1}{2}\int\int \|\hat{\mathbf{y}}(\mathbf{x}) - \mathbf{t}\|^2 p(\mathbf{x},\mathbf{t}) d\mathbf{x}d\mathbf{t} \tag{5.3}$$

$$= \frac{1}{2}\sum_k \int\int \{\hat{y}_k(\mathbf{x}) - t_k\}^2 p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k, \tag{5.4}$$

is equivalent to adding a *Tikhonov regularizer* term of the form

$$E^R = \frac{1}{2}\eta^2 \int\int \sum_k \sum_i \left\{ \left(\frac{\partial \hat{y}_k}{\partial x_i}\right)^2 + \frac{1}{2}\{\hat{y}_k(\mathbf{x}) - t_k\}\frac{\partial^2 \hat{y}_k}{\partial x_i^2} \right\} p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k \tag{5.5}$$

to the error term - i.e. the resultant error function to be minimized is

$$\tilde{E} = E + E^R \tag{5.6}$$

In the above, t_k is the kth component of the target output, $p(\mathbf{x},\mathbf{t})$ is the probability density of the exemplars in the joint input-target space, $p(t_k|\mathbf{x})$ denotes the condi-

tional density for t_k given the value of $\mathbf{x}$ and $p(\mathbf{x})$ denotes the unconditional density of $\mathbf{x}$, and η is controlled by the amplitude of the noise.

We now investigate the above relation between training with noise and regularization in greater detail. Let the noise on the input vector be described by the random vector ξ. The error function when training with noise can then be written in the form

$$\tilde{E} = \frac{1}{2}\int\int\int \sum_k \{\hat{y}_k(\mathbf{x}+\xi) - t_k\}^2 p(t_k|\mathbf{x})p(\mathbf{x})\tilde{p}(\xi)d\mathbf{x}dt_k d\xi \tag{5.7}$$

where $\tilde{p}(\xi)$ denotes the density function of the noise. The noise distribution is generally chosen to have zero mean, and to be uncorrelated between different inputs. Thus we have

$$\int \xi_i \tilde{p}(\xi)d\xi = 0; \qquad \int \xi_i\xi_j \tilde{p}(\xi)d\xi = \eta^2\delta_{ij} \tag{5.8}$$

We now assume that the noise amplitude is small, and expand the network function as a Taylor series in powers of ξ to give

$$\hat{y}_k(\mathbf{x}+\xi) = \hat{y}_k(\mathbf{x}) + \sum_i \xi_i \left.\frac{\partial \hat{y}_k}{\partial x_i}\right|_{\xi=0} + \frac{1}{2}\sum_i\sum_j \xi_i\xi_j \left.\frac{\partial^2 \hat{y}_k}{\partial x_i \partial x_j}\right|_{\xi=0} + \mathcal{O}(\xi^3) \tag{5.9}$$

Substituting the above expansion in the error function (ignoring the higher order terms) yields

$$\begin{aligned}
\tilde{E} &= \frac{1}{2}\sum_k \int\int\int \left\{(\hat{y}_k(\mathbf{x}) - t_k) + \left(\sum_i \xi_i \left.\frac{\partial \hat{y}_k}{\partial x_i}\right|_{\xi=0} + \frac{1}{2}\sum_i\sum_j \xi_i\xi_j \left.\frac{\partial^2 \hat{y}_k}{\partial x_i \partial x_j}\right|_{\xi=0}\right)\right\}^2 \\
&\qquad p(t_k|\mathbf{x})p(\mathbf{x})\tilde{p}(\xi)d\mathbf{x}dt_k d\xi \\
&= \frac{1}{2}\sum_k \int\int\int \left\{2(\hat{y}_k(\mathbf{x}) - t_k)\left(\sum_i \xi_i \left.\frac{\partial \hat{y}_k}{\partial x_i}\right|_{\xi=0} + \frac{1}{2}\sum_i\sum_j \xi_i\xi_j \left.\frac{\partial^2 \hat{y}_k}{\partial x_i \partial x_j}\right|_{\xi=0}\right)\right. \\
&\qquad +(\hat{y}_k(\mathbf{x}) - t_k)^2 \\
&\qquad \left. + \left(\sum_i \xi_i \left.\frac{\partial \hat{y}_k}{\partial x_i}\right|_{\xi=0} + \frac{1}{2}\sum_i\sum_j \xi_i\xi_j \left.\frac{\partial^2 \hat{y}_k}{\partial x_i \partial x_j}\right|_{\xi=0}\right)^2\right\} \\
&\qquad p(t_k|\mathbf{x})p(\mathbf{x})\tilde{p}(\xi)d\mathbf{x}dt_k d\xi
\end{aligned} \tag{5.10}$$

We now analyse the terms within the bracket on the righthand side. The second term

within the bracket simply becomes E in Equation 5.4. Making use of Equation 5.8, the first term within the bracket becomes

$$\frac{1}{2}\eta^2 \int\int \sum_k \sum_i \{\hat{y}_k(\mathbf{x}) - t_k\} \frac{\partial^2 \hat{y}_k}{\partial x_i^2} p(t_k|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dt_k \tag{5.11}$$

The third term within the bracket can be expressed as

$$\begin{aligned} \left(\sum_i \xi_i \left. \frac{\partial \hat{y}_k}{\partial x_i} \right|_{\xi=0} \right)^2 \quad &+ \quad \sum_i \xi_i \left. \frac{\partial \hat{y}_k}{\partial x_i} \right|_{\xi=0} \sum_m \sum_j \xi_m \xi_j \left. \frac{\partial^2 \hat{y}_k}{\partial x_m \partial x_j} \right|_{\xi=0} \\ &+ \quad \frac{1}{4} \left(\sum_m \sum_j \xi_m \xi_j \left. \frac{\partial^2 \hat{y}_k}{\partial x_m \partial x_j} \right|_{\xi=0} \right)^2 \end{aligned} \tag{5.12}$$

We now further analyse the terms of Equation 5.12. The first term can be further expressed as

$$\xi_1^2 \left(\frac{\partial \hat{y}_k}{\partial x_1} \right)^2 \Bigg|_{\xi=0} + \xi_1 \xi_2 \left. \frac{\partial \hat{y}_k}{\partial x_1} \frac{\partial \hat{y}_k}{\partial x_2} \right|_{\xi=0} + \ldots \tag{5.13}$$

which, making use of Equation 5.8, becomes

$$\frac{1}{2}\eta^2 \int\int \sum_k \sum_i \left(\frac{\partial \hat{y}_k}{\partial x_i} \right)^2 p(t_k|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dt_k \tag{5.14}$$

which has the form of a first-order regularization term.

The second term in Equation 5.12 can be expressed as :

$$\begin{aligned} &\left(\xi_1 \frac{\partial \hat{y}_k}{\partial x_1} + \xi_2 \frac{\partial \hat{y}_k}{\partial x_2} + \ldots \right) \left(\xi_1^2 \frac{\partial^2 \hat{y}_k}{\partial x_1^2} + \xi_1 \xi_2 \frac{\partial^2 \hat{y}_k}{\partial x_1 \partial x_2} + \ldots \right) \Bigg|_{\xi=0} = \\ &\left. \xi_1^3 \frac{\partial \hat{y}_k}{\partial x_1} \frac{\partial^2 \hat{y}_k}{\partial x_1^2} + \xi_1^2 \xi_2 \frac{\partial \hat{y}_k}{\partial x_1} \frac{\partial^2 \hat{y}_k}{\partial x_1 \partial x_2} + \xi_1 \xi_2 \xi_3 \frac{\partial \hat{y}_k}{\partial x_1} \frac{\partial^2 \hat{y}_k}{\partial x_2 \partial x_3} + \ldots \right|_{\xi=0} \end{aligned} \tag{5.15}$$

Assuming the noise's distribution is symmetric about its mean:

$$\int \xi_i \xi_j \xi_k \tilde{p}(\xi) d\xi = 0 \qquad \forall i, j, k; \tag{5.16}$$

The terms in Equation 5.15 thus vanish as we integrate.

The third term in Equation 5.12 can be expressed as :

$$\frac{1}{4}\left(\xi_1^2\frac{\partial^2\hat{y}_k}{\partial x_1^2}+\xi_1\xi_2\frac{\partial^2\hat{y}_k}{\partial x_1\partial x_2}+\ldots\right)\left(\xi_1^2\frac{\partial^2\hat{y}_k}{\partial x_1^2}+\xi_1\xi_2\frac{\partial^2\hat{y}_k}{\partial x_1\partial x_2}+\ldots\right)\Bigg|_{\xi=0} = \frac{1}{4}\left(\xi_1^4\left(\frac{\partial^2\hat{y}_k}{\partial x_1^2}\right)^2+\xi_1^3\xi_2\frac{\partial^2\hat{y}_k}{\partial x_1^2}\frac{\partial^2\hat{y}_k}{\partial x_1\partial x_2}+\xi_1^2\xi_2^2\left(\frac{\partial^2\hat{y}_k}{\partial x_1\partial x_2}\right)^2+\ldots\right)\Bigg|_{\xi=0} \tag{5.17}$$

Note that, with the same assumption used in Equation 5.8, all the coefficients of the derivative terms in the above that involve odd powers of any ξ_j will become zero when integrated with respect to ξ - i.e.

$$\begin{aligned}
\int \xi_i^3\xi_j\tilde{p}(\xi)d\xi &= 0; & i\neq j\\
\int \xi_i^2\xi_j\xi_k\tilde{p}(\xi)d\xi &= 0; & i\neq j\neq k\\
\int \xi_i\xi_j\xi_k\xi_l\tilde{p}(\xi)d\xi &= 0; & i\neq j\neq k\neq l
\end{aligned}$$

whereas

$$\begin{aligned}
\int \xi_i^2\xi_j^2\tilde{p}(\xi)d\xi &\geq 0; & i\neq j\\
\int \xi_i^4\tilde{p}(\xi)d\xi &\geq 0;
\end{aligned}$$

which implies that Equation 5.17 involves $\left(\frac{\partial^2\hat{y}_k}{\partial x_i^2}\right)^2$ and $\left(\frac{\partial^2\hat{y}_k}{\partial x_i\partial x_j}\right)^2 (i\neq j)$ terms only - i.e. it is positive definite and only has second-order regularizer terms.

From the above, we can see that E^R in Equation 5.5 is actually

$$E^R=\int\int\sum_k\sum_i\left\{\frac{1}{2}\eta^2\left\{\left(\frac{\partial\hat{y}_k}{\partial x_i}\right)^2+\{\hat{y}_k(\mathbf{x})-t_k\}\frac{\partial^2\hat{y}_k}{\partial x_i^2}\right\}+\mu\left(\frac{\partial^2\hat{y}_k}{\partial x_i^2}\right)^2+\nu\sum_j\left(\frac{\partial^2\hat{y}_k}{\partial x_i\partial x_j}\right)^2\right\}p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k \tag{5.18}$$

where μ and ν are some non-negative scalars that are monotonic functions of the amplitude of the noise. For noise of small magnitude, the second-order regularizer terms would become insignificant. Bishop pointed out that even though the term $\{\hat{y}_k(\mathbf{x}) - t_k\}\frac{\partial^2 \hat{y}_k}{\partial x_i^2}$ is not pdf, if the magnitudes of noise and $\{\hat{y}_k(\mathbf{x}) - t_k\}$ are small, this term will be insignificant.

Linear Perturbation with Noise

We now consider the case where the activations of the nth input has been scaled by a constant factor of small magnitude, together with superposition of noise on all inputs except the nth[3]. In such a case we have

$$\begin{aligned}
\int \xi_i \tilde{p}(\xi) d\xi &= \begin{cases} 0 & i \neq n \\ cx_n & i = n \end{cases} \\
\int \xi_i \xi_j \tilde{p}(\xi) d\xi &= \begin{cases} \eta^2 \delta_{ij} & i, j \neq n \\ c^2 x_n^2 & i = j = n \end{cases} \\
\int \xi_n^3 \tilde{p}(\xi) d\xi &= c^3 x_n^3 \\
\int \xi_n^2 \xi_j^2 \tilde{p}(\xi) d\xi &= \begin{cases} \eta^2 c^2 x_n^2 & j \neq n \\ c^4 x_n^4 & j = n \end{cases}
\end{aligned} \tag{5.19}$$

where c is a constant.

The first term within the bracket of Equation 5.10 becomes

$$\begin{aligned}
&\frac{1}{2}\eta^2 \int\int \sum_k \sum_{i \neq n} \{\hat{y}_k - t_k\} \frac{\partial^2 \hat{y}_k}{\partial x_i^2} p(t_k|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dt_k \\
&\quad + \frac{1}{2} \int\int \sum_k \{\hat{y}_k(\mathbf{x}) - t_k\} \left\{ 2cx_n \frac{\partial \hat{y}_k}{\partial x_n} + c^2 x_n^2 \frac{\partial^2 \hat{y}_k}{\partial x_n^2} \right\} p(t_k|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dt_k
\end{aligned} \tag{5.20}$$

[3]Note that the assumption that the noise at action inputs can be ignored is a biologically plausible one. This is because the action vector is generated proximally within the neural network, not input via distal afferent circuits, and thus the magnitude of its noise would probably be small compared to those at state inputs.

The first term in Equation 5.12 becomes

$$\frac{1}{2}\eta^2 \int\int \sum_k \sum_{i\neq n} \left(\frac{\partial \hat{y}_k}{\partial x_i}\right)^2 p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k$$
$$+\frac{1}{2}\int\int \sum_k c^2 x_n^2 \left(\frac{\partial \hat{y}_k}{\partial x_n}\right)^2 p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k \tag{5.21}$$

The second term in Equation 5.12 becomes

$$\frac{1}{2}c^3 \int\int \sum_k x_n^3 \frac{\partial \hat{y}_k}{\partial x_n}\frac{\partial^2 \hat{y}_k}{\partial x_n^2} p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k \tag{5.22}$$

Those terms related to x_n within the third term in Equation 5.12 are

$$\frac{1}{8}\int\int \sum_k \left\{ c^4 x_n^4 \left(\frac{\partial^2 \hat{y}_k}{\partial x_n^2}\right)^2 + c^2\eta^2 x_n^2 \sum_{j\neq n} \left(\frac{\partial^2 \hat{y}_k}{\partial x_n \partial x_j}\right)^2 \right\} p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k \tag{5.23}$$

The regularization term thus becomes:

$$E^R = \int\int \sum_k \sum_{i\neq n} \left\{ \frac{1}{2}\eta^2 \left\{ \left(\frac{\partial \hat{y}_k}{\partial x_i}\right)^2 + \{\hat{y}_k - t_k\}\frac{\partial^2 \hat{y}_k}{\partial x_i^2} \right\} \right.$$
$$\left. +\mu \left(\frac{\partial^2 \hat{y}_k}{\partial x_i^2}\right)^2 + \nu \sum_{j\neq n} \left(\frac{\partial^2 \hat{y}_k}{\partial x_i \partial x_j}\right)^2 \right\} p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k$$
$$+\int\int \sum_k \left\{ \{\hat{y}_k - t_k\} c x_n \frac{\partial \hat{y}_k}{\partial x_n} + \frac{1}{2}c^2 x_n^2 \left(\frac{\partial \hat{y}_k}{\partial x_n}\right)^2 \right.$$
$$+\frac{1}{2}\left(c^2 x_n^2 \{\hat{y}_k - t_k\} + c^3 x_n^3 \frac{\partial \hat{y}_k}{\partial x_n}\right)\frac{\partial^2 \hat{y}_k}{\partial x_n^2} + \frac{1}{8}c^4 x_n^4 \left(\frac{\partial^2 \hat{y}_k}{\partial x_n^2}\right)^2$$
$$\left. +\frac{1}{8}c^2\eta^2 x_n^2 \sum_{j\neq n} \left(\frac{\partial^2 \hat{y}_k}{\partial x_n \partial x_j}\right)^2 \right\} p(t_k|\mathbf{x})p(\mathbf{x})d\mathbf{x}dt_k \tag{5.24}$$

For finite $\{\hat{y}_k - t_k\}$ and $\frac{\partial \hat{y}_k}{\partial x_n}$, Equation 5.24 is bounded below, even though it is not pdf. The terms $c^2 x_n^2 \left(\frac{\partial \hat{y}_k}{\partial x_n}\right)^2$ and $c^4 x_n^4 \left(\frac{\partial^2 \hat{y}_k}{\partial x_n^2}\right)^2$ signify that the significance of the regularization related to the nth input varies according to the activation x_n and the magnitude of c.

Furthermore, if [4]

1. $\frac{\partial \hat{y}_k}{\partial x_n}$ is approximately constant with respect to x_n - this would be the case if the input being scaled is an action input to an *affine* dynamical systems;

2. the mapping learned prior to the input's scaling is accurate, and the current prediction error, $\tilde{y}_k$, is enirely due to the input's scaling;

3. the target output is relatively noiseless and the nominal mapping is not one-to-many - i.e. $t_k = y_k^*$;

then $\{\hat{y}_k - t_k\} = \tilde{y}_k = \frac{\partial \hat{y}_k}{\partial x_n} c x_n$ (or at least they have the same sign) and $\frac{\partial^2 \hat{y}_k}{\partial x_n^2} \approx 0$. The second term in Equation 5.24 becomes

$$\int\int \sum_k \left\{ \frac{3}{2} c^2 x_n^2 \left(\frac{\partial \hat{y}_k}{\partial x_n} \right)^2 + \frac{1}{8} c^2 \eta^2 x_n^2 \sum_{j \neq n} \left(\frac{\partial^2 \hat{y}_k}{\partial x_n \partial x_j} \right)^2 \right\} p(t_k|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dt_k; \qquad (5.25)$$

which is pdf. We cannot assume cx_n is of the same order as the noise, but we can deduce from Equation 5.25 that the effect of cx_n on the first-order regularization term is three times as significant as the noise.

At this point we might conjecture that the contrast could be *genuine reflection* of the changes in the dynamics - i.e. if the input to the τ_4 node is scaled by a factor c larger than 1, we should expect that the magnitude of weights directly connected to the corresponding input node, $|\mathbf{w}_{\tau_4}|$, would become larger. By the same conjecture, if the scaling factor is smaller than 1, we should expect that the $|\mathbf{w}_{\tau_4}|$ would become smaller. An investigation into how the changes in $|\mathbf{w}_{\tau_4}|$ correspond to the scaling factor might cast some light on this; as well as to provide the clue as to how the enhancement in the first-order regularization would result in the sharpening of contrast. This was carried out by observing the changes of $|\mathbf{w}_{\tau_4}|$ in 1046 distinct trajectories, each with a constant scaling factor c randomly assigned a value between 0.5 and 2.5

[4]These conditions are consistent with those described in section 5.4.2

at the start of trajectory. Of these 1046 trajectories, 251 were with scaling factor $0.5 \leq c \leq 1$. It was found that the average change in $\mathbf{w}_{\tau_4}$, $E(|\mathbf{w}_{\tau_4}(T)| - |\mathbf{w}_{\tau_4}(0)|)$, of these 251 trajectories were +0.002257 - i.e. $|\mathbf{w}_{\tau_4}|$ increases on average even when $c < 1$. In fact $|\mathbf{w}_{\tau_4}|$ decreases in only 17 of these 251 trajectories. Of the other 795 trajectories where $1 < c \leq 2.5$, $E(|\mathbf{w}_{\tau_4}(T)| - |\mathbf{w}_{\tau_4}(0)|) = 0.09$.

From the above observation, we can safely conclude that the increase in contrast does not genuinely reflect the changes in the dynamics as conjectured. However the fact that $|\mathbf{w}_{\tau_4}|$ increases on average, even in cases where $c < 1$, does provide support for the following hypothesis : in the course of minimizing E^R during training, there is a tendency for the output's activation to be forced away from the more linear range (where $\frac{\partial \hat{y}_k}{\partial x_i}$ is largest) of the logistic function. Such tendency could be manifested by having larger magnitude of activation of the hidden nodes further away from the output layer, such that there is greater chance of having larger magnitude of output (and hence smaller $\frac{\partial \hat{y}_k}{\partial x_i}$) in subsequent layers consisting of logistic nodes. The localized change of weights at the input layer (which is farthest away from the output layer) might be a result of this tendency. The analysis, however, does not fully explain how the network's training could lead to the manifestation of such a tendency.

"Shaking out" other forms of perturbations

We now look at the changes in the forward model after executing the same reaching task while two of the inputs, τ_4 and $\dot{\theta}_1$, are simultaneously scaled by two constant factors, 3.0 and 35.0 respectively. Figure 5-9 shows the simultaneous reflection of the 2 sources of persistent errors in the forward model.[5] Note that the influence of joint velocities on the arm's dynamics is in the form of quadratic terms, and for small $\dot{\theta}_1$ values, $\frac{\partial \hat{y}_k}{\partial \dot{\theta}_1} \approx$ constant. However, $\int \xi_n^q \tilde{p}(\xi) d\xi \neq (cx_n)^q$ for $q > 1$, where node n is the $\dot{\theta}_1$ input, because there is noise present at $\dot{\theta}_1$ as well.

[5]With a small scaling factor (< 10) of $\dot{\theta}_1$, the changes in weights proximal to $\dot{\theta}_1$'s input are too faint to be observed from the adapted forward model.

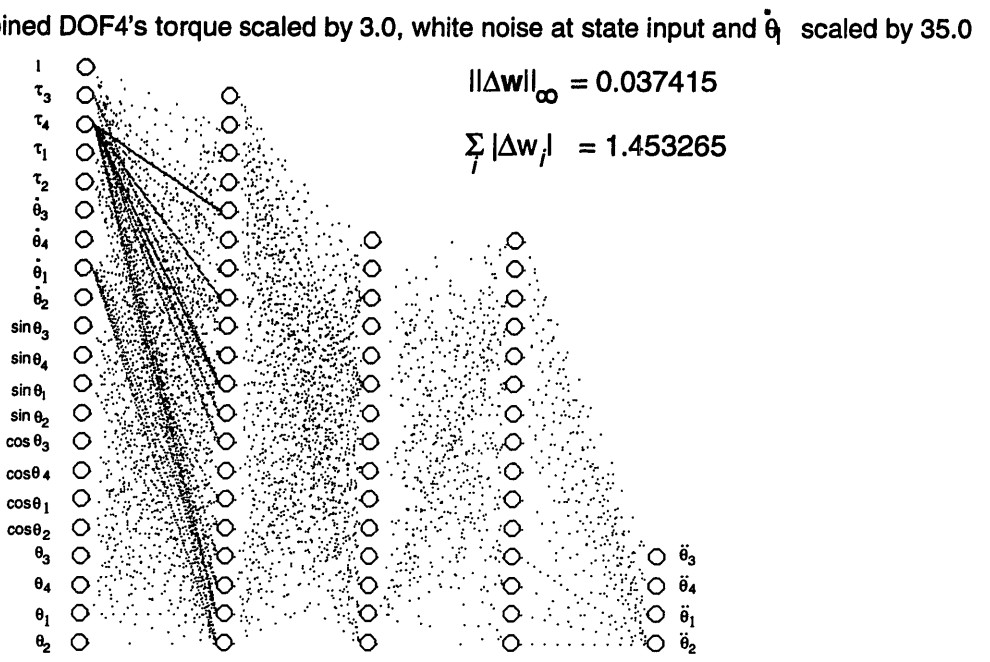


Figure 5-9: The changes in the forward model after the completion of the second reaching task with scaling of DOF4's torque by 3.0, white noise at state input and scaling of $\dot{\theta}_1$ by 35.0. In this case $H(\tau_4) = 8.82$ and $H(\dot{\theta}_1) = 4.21$, which are the highest and second highest of all Hs respectively.

It is also worthwhile to note that the reflection can be achieved even when noise is *not* present, but only after numerous distinct trajectories has been executed, by which time sufficient "actual spanning" of the forward model's input space is achieved. We can see that in such a case

$$E^R = \frac{3}{2} \int \int \sum_k c^2 x_n^2 \left(\frac{\partial \hat{y}_k}{\partial x_n} \right)^2 p(t_k|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dt_k; \tag{5.26}$$

and the delay in achieving the contrast might be due to the lack of other regularization terms in Equation 5.24.

Some other forms of perturbation are less directly interpretable than the cases mentioned above. For e.g. if the perturbation is a *displacement* in the joint velocities' readings, the appropriate compensation in the forward model should be the distributed changes in the bias of the nodes in the first hidden layer. On the other hand the changes in the bias of the nodes in the first hidden layer could not be easily interpreted as to which input has been displaced. The findings nonetheless indicate

that there is structural information within the forward model to be exploited. For example a mapping between various forms and magnitudes of perturbations and the forward model's changes can be built via simulation, which may then be used in fault identification or anticipatory compensation of known perturbations of a plant.

In recent years there has been a growth in research work linking the role of persistent excitation to the use of neural networks for robotic control (see Lewis & Liu 1995, Craig 1988). Intuitively the noise at the training inputs would provide a form persistent excitation, as it "urges" the controller to explore its parameter space more via interacting with the environment than when it is not present. This is certainly an area worth looking into in future.

5.4.3 Effect on global prediction accuracy

The on-line adaptation process also leads to effective generalization over a wide neighborhood in the input space, as shown in Figure 5-10. The statistics reported were taken after the adaptation process at 0.4 sec after the initiation of a trajectory depicted in Figure 3-5 (for (a), (b) and(c)) or one described in subsection 5.4.2 (by which time the forward model is providing accurate control). The sampling points were taken within the hypercube with each edge spanning between [-1,1] of the forward model's input space, which is effectively 12-dimensional. The diagonal of this hypercube is thus $\sqrt{48} \approx 7$ in length. Note that the range of joint velocities for each joint in most joint angle configurations is much smaller than the range of possible joint velocities (whose limits are used for the scalings of the sensory information prior to its being input to the forward model). The set of possible scaled inputs is thus a small subset of the hypercube. The restriction of the statistical sampling to within the hypercube of the input space is therefore insufficient in providing a precise measurement of the prediction accuracy. Comparisons at a distance too far from the point $(\mathbf{x}', \tau')$, where the forward model would be performing extrapolation, is therefore avoided. Note also that the prediction accuracies at larger distance are similar

in both cases, indicating that local adaptation does not result in "trading" global accuracy for local accuracy in these cases.

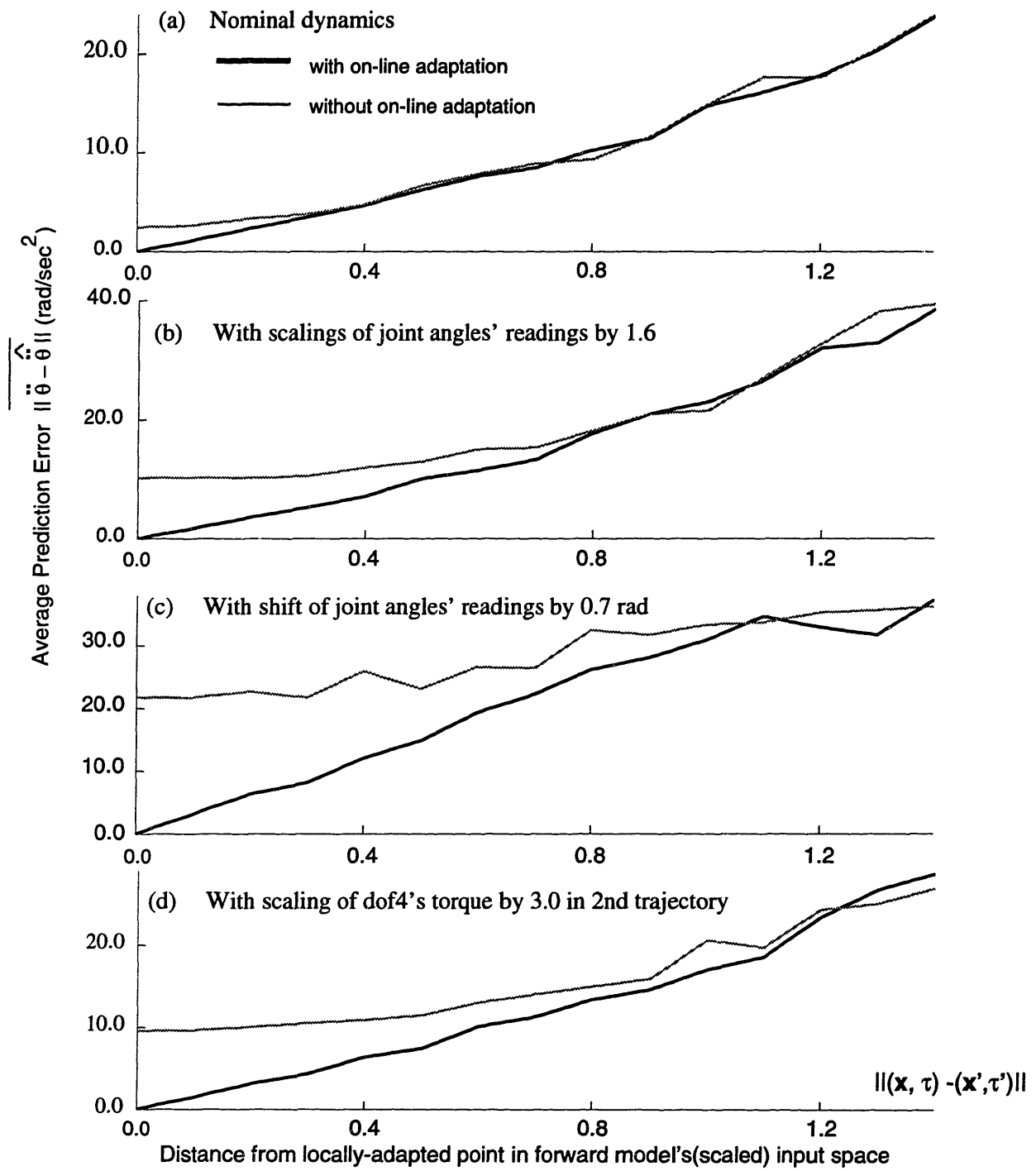


Figure 5-10: Typical influence of the local adaptation in global prediction accuracy. (a) shows the improvement in global prediction at a distance of up to ≈ 0.73 from $(\mathbf{x}', \tau')$ when performing under nominal dynamics. (b) the improvement at a distance of up to ≈ 0.9 when performing with the scaling of joint angles by 1.6. (c) improvement at up to ≈ 1.05 when performing with the shifting of joint angles by 0.7 rad. (d) the improvement at up to ≈ 1.2 with threefold increase of DOF4's torque.

Chapter 6

Performance Using Various Learning Architectures

6.1 Introduction

This chapter presents the empirical results showing how neural networks, hyper basis functions and local experts perform as the forward model in the adaptive control of a simulated 2-joint arm. Such extensive effort is made in the hope of establishing the applicability of the forward modeling approach in view of the lack of formal analysis about its stability. The dynamics of the simulated 2-joint arm was described in section 2.3. The nominal range of torques at joint 1 and 2 are [-20.0, 20.0]Nm and [-10.0, 10.0]Nm respectively, and the masses of both links are 1 kg. In each case the forward model was trained using the same exemplar set (containing 30,000 exemplars), until a local minimum of the prediction error cost is reached. The forward model is then applied in 1000 distinct reaching tasks specified by the starting and target joint positions in conjunction with the use of the trajectory planner given by Equation 3.5, where the coefficients are set to $K_p = 20I$ and $K_v = 13.33I$. The

control is considered a failure in tracking a trajectory when one or more of the joints reaches its limit, or when the time taken is more than 160 seconds, before the target position is reached. Three measures of performance are taken for comparison :

1. number of trajectories successfully completed,
2. average number of time steps per successful trajectory;
3. average performance error per time step of the successful trajectories.

For each family of learning architecture, numerous networks of distinct configurations were trained, and the one that performed best under nominal condition was then chosen for comparison under perturbed conditions. The perturbed conditions for the experiments are

1. the mass of link1 increased by three times ($m_1 \times 3$),
2. joint angle readings shifted by 1 radian ($\theta + 1rad$),
3. joint velocity readings shifted by 1 rad/sec ($\dot{\theta} + 1rad/sec$).

To provide an idea of the contribution of on-line adaptation, the performance under nominal condition *with* (w/ o-l) and *without* on-line adaptation (w/o o-l) of the forward model are also presented.

6.2 Performance of neural networks

Numerous feedforward neural networks of different architectures were trained. Table 6.1 shows the performance of two of the networks : one with a single hidden layer (consisting of 12 logistic units), and one with two hidden layers (first hidden layer has 20 units; second layer has 10 units). It is observed that the network with single

# hidden layers	1	2
# free parameters	156	440
Max weight magnitude	48.34	2.94
Average weight magnitude	1.88	0.53
Off-line Relative error	0.015	0.007
# successful trajectories	998	619
Average $\frac{\#time\ steps}{successful\ trajectory}$	258.40	416.31
Average $\frac{Performance\ Error}{time\ step}(rad/sec^2)$	2.40	8.87

Table 6.1: Performance of two neural networks under nominal condition

Conditions	Nominal		$m_1 \times 3$	$\theta+$	$\dot{\theta}+$
	w/o o-l	w/ o-l		$1rad$	$1rad.sec^{-1}$
# successful trajectories	177	998	998	983	984
Average $\frac{\#time\ steps}{successful\ trajectory}$	128.66	258.40	249.00	277.19	248.57
Average $\frac{Performance\ Error}{time\ step}$	7.31	2.40	2.68	5.32	2.38

Table 6.2: Performance of NN under nominal and perturbed conditions.

hidden layer, though it has larger off-line relative error, performs better than the one with 2 hidden layers.

The 1-hidden layer NN was then applied in the control of the arm under various perturbed conditions described in section 6.1, whose performance is as shown in Table 6.2.

6.3 Performance of Local Expert networks

In the experiments using local expert networks (LEN; see Jacobs & Jordan 91, Haykin 94), gating networks with softmax output nodes and no hidden layer were used in conjunction with linear expert networks. Numerous LENs, whose architectures differ only in the number (between 2 to 15) of local experts, were initially chosen and trained. The one with 4 local experts was deemed to give the best performance[1]

[1]In fact if the number of local experts is greater than 4, the training would result in only 4 of them remaining active.

Conditions	Nominal		$m_1 \times 3$	$\theta+$	$\dot{\theta}+$
	w/o o-l	w/ o-l		$1rad$	$1rad.sec^{-1}$
# successful trajectories	246	955	940	957	957
Average $\frac{\#time\ steps}{successful\ trajectory}$	330.42	307.33	344.24	261.02	372.45
Average $\frac{Performance\ Error}{time\ step}$	8.31	0.42	0.65	1.13	0.49

Table 6.3: Performance of LEN (with 4 local experts) under nominal and perturbed conditions

# of Nodes	2	3
# free parameters	224	336
Off-line Relative error	0.040	0.038
# successful trajectories	1000	1000
Average $\frac{\#time\ steps}{successful\ trajectory}$	229.27	200.71
Average $\frac{Performance\ Error}{time\ step}(rad/sec^2)$	0.27	0.26

Table 6.4: Performance of two HBF networks under nominal condition

under nominal condition. Table 6.3 shows its performance under various conditions.

6.4 Performance of Hyper-Basis Functions

Two HBF networks (see Poggio & Girosi 90, Poggio & Girosi 92), one with 2 nodes and the other with 3 nodes, were initially chosen and trained for this set of experiments. Table 6.4 shows the performance of the two networks under nominal condition.

Although the 3-node network provides better performance, the two-node network was chosen to obtain further performance measures because its number of parameters is closer to those of the representatives of other mapping tools. The two-node network was then applied onto the adaptive control under the same perturbed conditions described in section 6.2. It is observed (see Table 6.5) that the HBF network performs better than the neural networks and local experts in all measures under all the conditions investigated, even though its off-line relative error is larger than others. Table 6.6 provides a comparison of performance of the HBF network under various perturbations with and without on-line adaptation. The table shows that

Conditions	Nominal		$m_1 \times 3$	$\theta+$	$\dot{\theta}+$
	w/o o-l	w/ o-l		$1rad$	$1rad.sec^{-1}$
# successful trajectories	343	1000	1000	1000	1000
Average $\frac{\#time\ steps}{successful\ trajectory}$	266.63	229.27	230.02	206.29	201.56
Average $\frac{Performance\ Error}{time\ step}$	7.08	0.27	0.30	3.00	3.01

Table 6.5: Performance of HBF network under nominal and perturbed conditions

Conditions	$m_1 \times 3$		$\theta + 1rad$		$\dot{\theta} + 1rad.sec^{-1}$	
	w/o o-l	w/ o-l	w/o o-l	w/ o-l	w/o o-l	w/ o-l
# successful trajectories	376	1000	141	1000	303	1000
Average $\frac{\#time\ steps}{successful\ trajectory}$	257.60	230.02	170.65	206.29	221.33	201.56
Average $\frac{Performance\ Error}{time\ step}$	7.45	0.30	13.16	3.00	6.52	3.01

Table 6.6: Performance using HBF under perturbed conditions with (w/ o-l) and without (w/o o-l) online adaptation.

there is larger performance error/time step without on-line adaptation. Recall that the trajectory planner described by Equation 3.5 provides a certain degree of feedback compensation. The observation means that there is larger degree of feedback compensation involved in completing the reaching tasks succeeded without on-line adaptation.

6.5 Comparing to Craig's method

This chapter will not be complete without some comparison of performance with a prominent analytical method. Craig's method is chosen for this purpose. The following aspects were investigated :

1. *Stability of parameter update law under nominal dynamics* - Craig's method requires the the input be sufficiently *persistently exciting* in order for the parameters to converge. An explicit bound on the unknown parameters is also required to prevent them from becoming very large (i.e. diverging - see Figure 6-1), which may occur even under benign perturbed conditions. Craig's

thesis (Craig 1988) reported the observation that in the case where the manipulator is hanging straight down (so gravity causes no sag) and the desired trajectory is constant (i.e. the manipulator is not moving), the input is not sufficiently exciting. In such case the noise present in the system causes the tendency of parameters to *diverge*, even though the performance error remain quite small. This is not surprising because the unknown parameters are treated as control inputs to a linear dynamical system, and the parameter adaptation law (Equation 2.8) is a linear feedback term for its control.

The on-line adaptation of a HBF forward model under similar circumstance shows sign of parameter initially *drifting*, but not diverging - i.e. there is no sign of the rate of drift of parameter values increasing as the adaptation proceeds. From simulation with input noise amplitude at 5% of input amplitude, it has been observed that the HBF forward model parameters initially drifted (see Figure 6-2), but the drift eventually stops. Forward modeling hence appears to be more stable in this aspect.

2. *Explicit identification of parameters.* Craig's method can lead to fast and accurate identification of important manipulator parameters such as the masses of links, viscous and Coulomb frictional coefficients; while the distributed representation within a neural network, LEN or HBF does not give such clear interpretation. However, the observation of the "shaking out" phenomenon described in section 5.4.2 does point to the possibility that certain types of system's parameters might be extracted from an NN forward model.

3. *Applicability to perturbations whose parameters cannot be linearly decoupled.* As explained in section 2.4.1, Craig's method requires the unknown parameters to be explicitly decoupled from the dynamics equation, and thus cannot handle perturbations such as shifts in joint sensor readings. The results presented in Chapter 5 and 6 show that the forward modeling is, to a large extent, able to handle such perturbations. This is a big plus for the forward modeling approach, and the use of learning tools, such as HBF and neural networks, for mapping

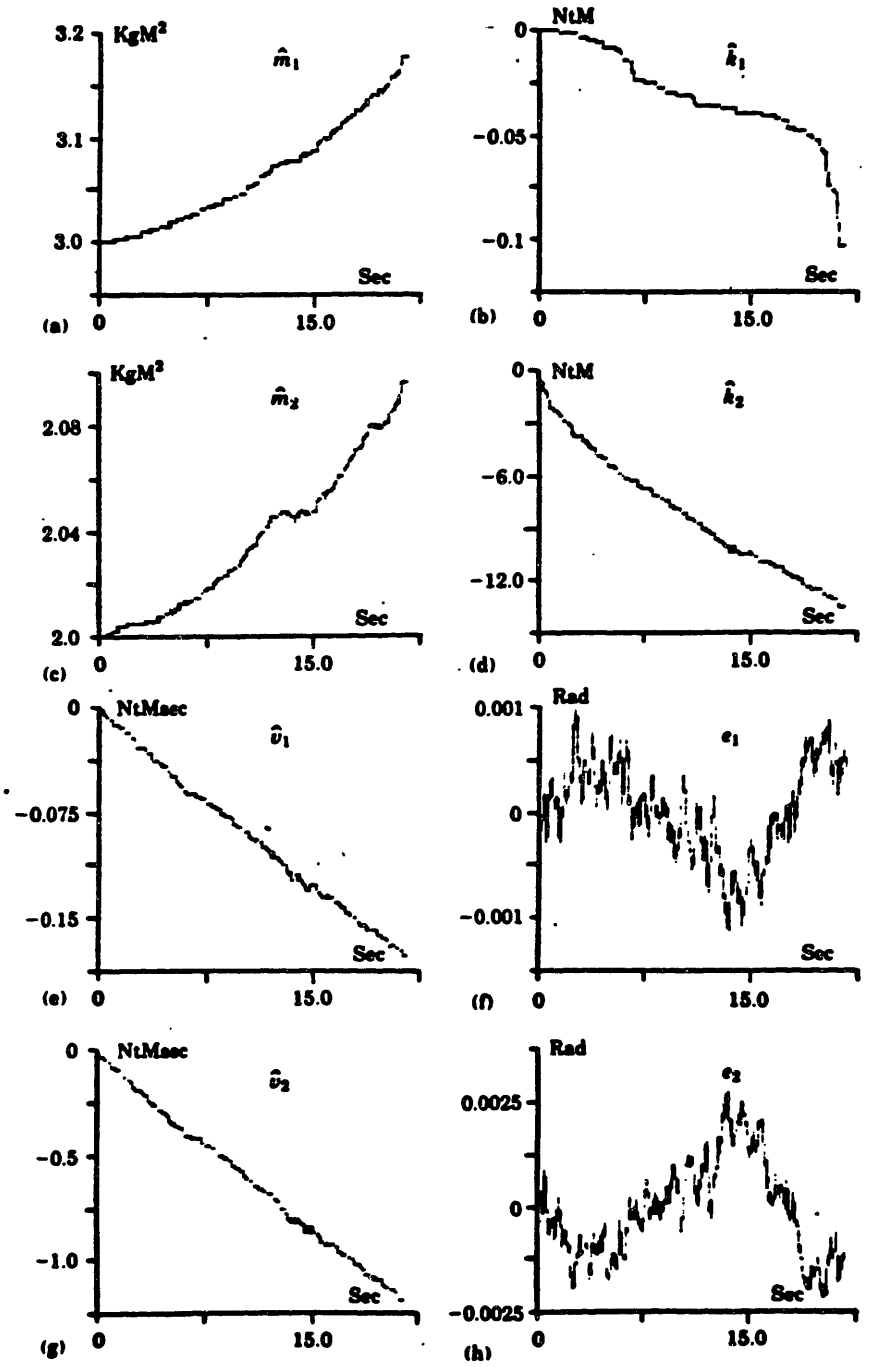


Figure 6-1: In Craig's control scheme, parameters diverge when excitation is lacking. *Adapted from Craig 1988.*

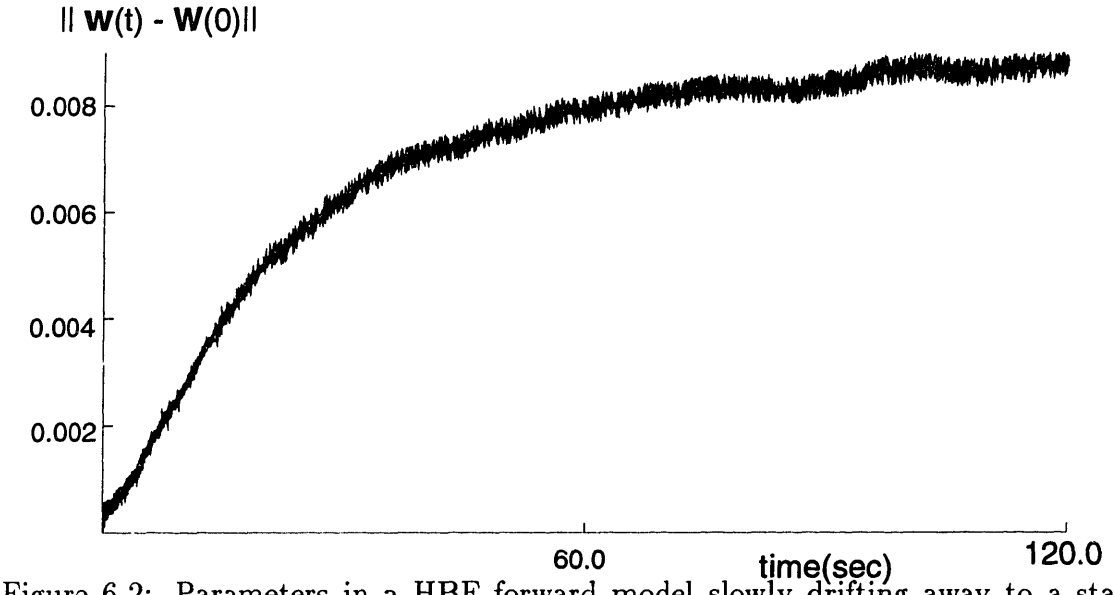


Figure 6-2: Parameters in a HBF forward model slowly drifting away to a stable point due to noise present in the inputs, even when the arm is not moving. Note that the drifted amount is small compared to the squared length of the parameters: $\|\mathbf{w}\| = 21.15$, while $\|\Delta\mathbf{w}\| < 0.009$.

of nonlinear time-varying functions in general.

4. *Rigorous proof of stability.* Craig's method has been proven to be stable under certain specified conditions, while the forward model's on-line adaptation lacks such rigorous proof of stability. This area of work is inherently tied to theoretical foundation of neural networks and machine learning in general. Much effort has been made in the attempt to establish the proof of the stability, but it has not been fruitful.

Chapter 7

Conclusion

To know that we know what we know, and that we do not know what we do not know, that is true knowledge.

- Thoreau

Although the approach has only been applied on simulated manipulator control, the simulations nonetheless demonstrated its potential in performing the highly non-linear and complex adaptation process that could be applied to the control of other types of dynamical systems. In particular, the results presented in Sections 5.2.2 and 5.2.3 revealed that the approach is able to handle perturbations whose parameters cannot be linearly decoupled from the dynamical formulation, as required in both Craig's and Slotine's methods. A main and general result is that there is no need for an explicit feedback controller for achieving improvement in performance. In chapter 5 we also see how information as to certain types of perturbations can be reflected in the weight changes within a forward model.

We have yet to establish analytical conditions under which the on-line adaptation provide stable adaptive motor control. However what the empirical results suggest is that the on-line action search and adaptation leads to the effective exploitation of the

forward model. This procedure leads to a more extensive form of exploration, which creates feedback from the environment that more accurately pinpoints the error in the prediction made in previous time steps. Provided that the task is specified in the form of slowly-varying desired outputs, this pinpointing may shift the basin (see Figure 3-4) in the forward model's input space appropriate to the completion of the task. The overall process is *effortful learning* that leads to the fast convergence rate of performance errors observed in both intra- and inter-trajectory performance, which can be viewed as a form of goal-directed behavior (see Wilberg 1991).

The fact that the spontaneous interpolation of a small number of exemplars by a partially-trained network could be a convenient solution to such a complex adaptive control problem also opens up questions about the power of neural networks that has not been exploited. A more detailed study of this aspect of neural network training could be complementary to the currently dominant trend of looking at a network being trained with large number of exemplars.

The claims made in this thesis rely mainly on empirical results obtained from simulations, and these claims need to be substantiated in tests on other dynamical systems. Continued empirical success in the domain of nonlinear control would attract support from the scientific community for further development of the approach.

Appendix A

Monotonicity, convexity and symmetry of direct-drive manipulator's dynamics

The Lagrangian formulation [Section 5.2, Asada 86] of an n-degree-of-freedom manipulator's dynamics is given as follows in computed-torque form :

$$\sum_{j=1}^{n} H_{ij}\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} h_{ijk}\dot{q}_j\dot{q}_k + G_i = Q_i \qquad i = 1, ..., n \tag{A.1}$$

where $q_i, \dot{q}_i, \ddot{q}_i$ are the displacement, velocity and acceleration of joint i respectively, and Q_i is the *generalized force* at joint i. H_{ij} is the $[i, j]$ component of the *manipulator*

inertia tensor :

$$\mathbf{H} = \sum_{i=1}^{n} (m_i \mathbf{J}_L^{(i)^T} \mathbf{J}_L^{(i)} + \mathbf{J}_A^{(i)^T} \mathbf{I}_i \mathbf{J}_A^{(i)}).$$

$$\text{and } \mathbf{J}_L^{(i)} = [\mathbf{J}_{L1}^{(i)} ... \mathbf{J}_{Li}^{(i)}, \mathbf{0} ... \mathbf{0}], \quad \mathbf{J}_{Lj}^{(i)} = \begin{cases} \mathbf{b}_{j-1} & \text{for a prismatic joint} \\ \mathbf{b}_{j-1} \times \mathbf{r}_{j-1,ci} & \text{for a revolute joint} \end{cases}$$

$$\mathbf{J}_A^{(i)} = [\mathbf{J}_{A1}^{(i)} ... \mathbf{J}_{Ai}^{(i)}, \mathbf{0} ... \mathbf{0}], \quad \mathbf{J}_{Aj}^{(i)} = \begin{cases} \mathbf{0} & \text{for a prismatic joint} \\ \mathbf{b}_{j-1} & \text{for a revolute joint} \end{cases}$$

$$h_{ijk} = \frac{\delta H_{ij}}{\delta q_k} - \frac{1}{2} \frac{\delta H_{jk}}{\delta q_i}$$

where $\mathbf{r}_{j-1,ci}$ is the position vector of the centroid of link i referred to the link $j-1$, and $\mathbf{b}_{j-1}$ is the 3×1 unit vector along joint axis $j-1$. $\mathbf{I}_i$ is the *inertia tensor* of the link i, and G_i is the *gravity torque* term :

$$G_i = \sum_{j=1}^{n} m_j \mathbf{g}^T \mathbf{J}_{Li}^{(j)}. \tag{A.2}$$

In fact the dynamics of the serial-link arm has the following *affine* form :

$$\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t), t) + \mathbf{b}(\mathbf{x}(t), t)\mathbf{u}(t) \tag{A.3}$$

in which the dynamics is linear in the control, and hence has the following properties:

1. Monotonicity

   Since H_{ij}, h_{ijk} and G_i are not dependent on $\ddot{q}_j$ in Equation A.1, and are fixed for a particular state $(\mathbf{q}, \dot{\mathbf{q}})$, each $\ddot{q}_j$ in Equation A.1 is either a non-decreasing (if $H_{ij} \geq 0$) or non-increasing (if $H_{ij} \leq 0$) linear function of Q_i.

2. Convexity

   Equation A.1 may be rewritten as :

$$\mathbf{H}\ddot{\mathbf{q}} + \mathbf{C} = \mathbf{Q}; \tag{A.4}$$

where

$$\begin{aligned}\ddot{\mathbf{q}}^T &= (\ddot{q}_1, ..., \ddot{q}_n), \quad \mathbf{Q}^T = (Q_1, ..., Q_n), \\ \mathbf{C}^T &= \sum_{j=1}^{n}\sum_{k=1}^{n} \dot{q}_j \dot{q}_k (h_{1jk}, ..., h_{njk}) + (G_1, ..., G_n).\end{aligned}$$

Let $\mathbf{Q}_1^T = (Q_1^{(1)}, ..., Q_n^{(1)})$ and $\mathbf{Q}_2^T = (Q_1^{(2)}, ..., Q_n^{(2)})$ be any two torque vectors that achieve joint accelerations $\ddot{\mathbf{q}}_1$ and $\ddot{\mathbf{q}}_2$ respectively at the same state :

$$\mathbf{H}\ddot{\mathbf{q}}_1 + \mathbf{C} = \mathbf{Q}_1.$$
$$\mathbf{H}\ddot{\mathbf{q}}_2 + \mathbf{C} = \mathbf{Q}_2.$$

Any joint acceleration $\ddot{\mathbf{q}}_3$ that lies between $\ddot{\mathbf{q}}_1$ and $\ddot{\mathbf{q}}_2$ takes the form :

$$\ddot{\mathbf{q}}_3 = \ddot{\mathbf{q}}_1 + \delta(\ddot{\mathbf{q}}_2 - \ddot{\mathbf{q}}_1), \qquad \text{where } 0 < \delta < 1 \tag{A.5}$$

Let $\ddot{\mathbf{q}}_3$ be achieved by $\mathbf{Q}_3$ at the same state :

$$\begin{aligned}\mathbf{Q}_3 &= \mathbf{H}\ddot{\mathbf{q}}_3 + \mathbf{C} \\ &= (1-\delta)\mathbf{H}\ddot{\mathbf{q}}_1 + \delta\mathbf{H}\ddot{\mathbf{q}}_2 + \mathbf{C} \\ &= \mathbf{Q}_1 + \delta(\mathbf{Q}_2 - \mathbf{Q}_1)\end{aligned} \tag{A.6}$$

which is of the same form as Equation A.5 - i.e. $\mathbf{Q}_3$ lies between $\mathbf{Q}_1$ and $\mathbf{Q}_2$ at the same positional ratio δ. Hence the achievable joint acceleration set is convex and symmetric for a convex and symmetric torque set.

Appendix B

Utilizing logistic action nodes

This appendix elaborates on how the utilization of logistic input (action) nodes in the forward model could influence the action search process involving redundant degrees-of-freedom.

Suppose a forward model is to perform the forward kinematic function $f : \Re^3 \mapsto \Re^2$ that predicts the position of the hand $\mathbf{x} = (x, y)^T$ given a joint position $\theta = (\theta_1, \theta_2, \theta_3)^T$ of a three-joint arm :

$$\mathbf{x} = f(\theta) \tag{B.1}$$

The action-search process described in Section 4.2 is then applied to find a joint position θ^* for reaching a desired hand position $\mathbf{x}^*$ from the forward model. As this problem involves redundant degrees-of-freedom, there are infinite number of θ^*s that would provide the same $\mathbf{x}^*$. If linear input nodes are used, the dynamics of the search process will be dominated solely by the modeled f, and the resulting θ^* may be a reaching posture with excessively acute or obtuse joint positions. A way to improve the chance of retrieving a more relaxed posture is to start the action search process

with initial input of neutral or relaxed joint positions of each joint. This "tendency" of the action search process producing a more relaxed reaching posture can be further enhanced if the following conditions are met :

1. the input nodes of the forward model are logistic, i.e. their output function is:

$$l(v_j) = \frac{2}{1 + e^{-\beta v_j}} - 1; \tag{B.2}$$

where $v_j \in (-\infty, \infty)$ is the activation of node j and β a constant. The first derivative of the logistic function is $l'(x) = \frac{\partial l}{\partial x} = \frac{\beta(l(x)+1)(1-l(x))}{2} > 0 \ \forall x$, and has the properties

$$l'(x) \to \max_x(l'(x)) = \frac{\beta}{2} \text{ as } |x| \to 0, \text{ and } \lim_{|x|\to\infty} l'(x) \to 0;$$

Each v_j is updated at the ith step of the iterative action search process (*cf* Equation 4.2) by :

$$v_j^{(i+1)} = v_j^{(i)} + \alpha\delta_j^{(i)};$$

and

$$\delta_j^{(i)} = l'(v_j^{(i)}) \sum_k \delta_k^{(i)} w_{kj}; \tag{B.3}$$

where δ_j^i is the back-propagated error signal (see pp 327, Rumelhart 86) of input node j, δ_k^i is the error signal of node k at the layer proximal to the input layer, α a step size and w_{kj} is the weight of connection joining node j to node k.

2. each joint position θ_j is represented as v_j of the respective jth input node via the affine scaling :

$$v_j = c(\theta_j - \bar{\theta}_j); \tag{B.4}$$

where $\bar{\theta}_j$ is the neutral joint position (i.e. the most relaxed position) and c is a scaling constant that is to be determined in conjunction with β of Equation B.2

such that $l(c(\theta_j^{max} - \bar{\theta}_j) \approx 1$, where θ_j^{max} is the maximum of θ_j;

3. the search starts with $v_j = 0, \forall j$;

4. the action search process leads to unidirectional changes of the activations at each input node j - i.e. for $i = 1, ..., m-1, \delta_j^{(i)}\delta_j^{(i+1)} > 0$; and $v_j^{(i)}$ is monotonic in i in Equation 4.2;

For each input node j, β in Equation B.2 can be set independently to reflect the degree to which each joint should compromise. Consider the following *deviation* cost function for joint j :

$$J_j(v_j^*) = (v_j^* - v_j^{(0)})^2 \tag{B.5}$$

Let the action search ends after $m(< \infty)$ iterations[1] - i.e. $v_j^{(m)} = v_j^*$. Then

$$v_j^* - v_j^{(0)} = \alpha \sum_{i=0}^{m-1} \delta_j^{(i)} = \alpha \sum_{i=0}^{m-1} l'(v_j^{(i)}) \sum_k \delta_k^{(i)} w_{kj} \tag{B.6}$$

Note that

$$\frac{\partial^2 l(x)}{\partial x \partial \beta} = \frac{\partial l'(x)}{\partial \beta} = l'(x)(\frac{1}{\beta} - x l(x));$$

Let $\beta = \frac{1}{\epsilon l(\epsilon)}$ for some $\epsilon > 0$. Since $l'(x) > 0$ and $-xl(x) \leq 0 \ \forall x$, the above implies

$$\frac{\partial l'(x)}{\partial \beta} \begin{cases} < 0, & |x| > \epsilon \\ \geq 0, & |x| \leq \epsilon \end{cases}$$

Let p be the number of iterations taken for $\left|\alpha \sum_i \delta_j^{(i)}\right|$ to first exceed ϵ - i.e.

$$v_j^* - v_j^{(0)} = \alpha \sum_{i=0}^{p-1} \delta_j^{(i)} + \alpha \sum_{i=p}^{m-1} \delta_j^{(i)}, \tag{B.7}$$

where

$$\frac{\partial l'(v_j^{(i)})}{\partial \beta} \begin{cases} > 0, & i = 1, ..., p-1 \\ \leq 0, & i = p, ..., m-1 \end{cases}$$

[1]This is possible with $\beta \to \infty$ only if there is redundant degrees-of-freedom.

For $i \geq p$, the above implies $l'(v_j^{(i)}) \rightarrow 0$ as $\beta_j \rightarrow \infty$, and since $\left|\delta_k^{(i)} w_{kj}\right| < \infty$ and m is finite, $\sum_{i=p}^{m-1} \delta_j^{(i)} \rightarrow 0$ as $\beta_j \rightarrow \infty$, and $\left|v_j^* - v_j^{(0)}\right| \rightarrow \epsilon$.

For $i < p$(where $\frac{\partial l'(v_j^{(i)})}{\partial \beta} > 0$), if α can be set to arbitrarily small values for any chosen $\beta < \infty$ such that

$$\alpha \left| l'(v_j^{(i)}) \sum_k \delta_k^{(i)} w_{kj} \right| < \xi, \forall i = 0, ..., p-1;$$

for some $\xi \leq \epsilon$, then $\sum_{i=0}^{p-1} \left|\delta_j^{(i)}\right| < \epsilon \rightarrow 0$ as $\beta_j (= \frac{1}{\epsilon l(\epsilon)}) \rightarrow \infty$. The above implies that the deviation of v_j^* from $v_j^{(0)}$ could be arbitrarily small as we increase β_j.

Condition (4) takes into consideration the practical situation where $\exists i$ such that $v_j^{(i)} \delta_j^{(i)} < 0$. Since for large $|v_j^{(i)}|, l'(v_j) \rightarrow 0$ for large β, which implies that $|v_j^{(i+1)}|$ reduces slowly when $v_j^{(i)} \delta_j^{(i)} < 0$. - i.e. the property of $l'(v_j) \rightarrow 0$ as $|v_j| \rightarrow \infty$ is desirable for achieving smaller $|v_j|$ only when $v_j^{(i)} \delta_j^{(i)} > 0 \forall i = 1, ..., m-1$.

Bibliography

[1] Abbs, J.H., Gracco, V.L. (1983). Sensorimotor actions in the control of multi-movement speech gestures. *Trends in Neuroscience* 6, 391-395.

[2] Allgower, E. L., Georg, K. (1990). *Numerical Continuation Methods.* Berlin: Springer-Verlag.

[3] Anderson, Brian D.O. (1977). Exponential Stability of Linear Equations Arising in Adaptive Identification. *IEEE Transactions on Automatic Control*, Feb 1977; AC-22: 83-88.

[4] Arimoto, S., Kawamura, S., Miyazaki, F. (1984). Bettering Operation of Robots by Learning. *Journal on Robotic Systems.* 1 & 2: 123-140.

[5] Arimoto, S., Miyazaki, F. (1985). Stability and Robustness of PID Feedback Control for Robotic Manipulators of Sensory Capability. Third International Symposium of Robotics Research, Gouvieux, France, July 1985.

[6] Asada, H, Slotine, J.J., (1986). *Robot Analysis and control.* John Wiley.

[7] Atkeson, C.G., Reinkensmeyer, D.J. (1990). Using Associative Content-Addressable Memories to Control Robots. *Neural Networks for Control*; 255-286. Cambridge: MIT Press.

[8] Beer, R. D., Ritzmann R.Ee., McKenna T., 1993. *Biological Neural Networks in Invertebrate Neuroethology and Robotics.* San Diego: Academic Press.

[9] Bishop, C.M. (1994). Training with Noise is Equivalent to Tikhonov Regularization. Neural Computing Res Group Report 4290. Birmingham, England.

[10] Book, W. (1993). Recursive Lagrangian Dynamics of Flexible Manipulator Arms. *Robot Control: Dynamics, Motion Planning and Analysis.* New York: IEEE Press.

[11] Chen, T. & Chen, R. (1995). Universal Approximation to Nonlinear Operators by Neural Networks with Arbitrary Activation Functions and Its Application to Dynamical Systems. *IEEE Transactions on Neural Networks.* In press.

[12] Craig, J.J. (1988). *Adaptive Control of Mechanical Manipulators.* Reading, MA: Addison Wesley.

[13] Dubowsky S., DesForges, D.T., (1979). The Application of Model-Referenced Adaptive Control to Robotic Manipulators. ASME Journal of Dynamical Systems Measurements and Control; 101.

[14] Fun, W., Jordan, M. I. (1992). *The moving basin: Effective action-search in forward models.* MIT Computational Cognitive Science Technical Report #9205, Cambridge, MA.

[15] Galiana, H.L. (1990). Oculomotor Control. *Visual Cognition and Action.* Cambridge: MIT Press.

[16] Gorinesky, D.M. (1993). Modeling of direct motor program learning in fast human arm motions. *Biological Cybernetics.* 69, 219-228.

[17] Grossberg, S., Kuperstein, M., (1989). *Neural Dynamics of Adaptive Sensory-Motor Control.* London: Pergamon Press.

[18] Haykin, S. (1994). *Neural Networks : A comprehensive Foundation.* MacMillan.

[19] Hollerbach, J.M., (1990). Fundamentals of Motor Behavior. *Visual Cognition and Action*, V2. Cambridge: MIT Press.

[20] Horowitz, R., Tomizuka, M., (1980). Discrete time Model Reference Adaptive Control of Mechnical Manipulators. ASME Paper no. 80-WA/DSC-6.

[21] Isidori, A. (1985). *Nonlinear Control Systems.* Berlin: Springer-Verlag.

[22] Jacobs, R.A., M.I. Jordan, S.J. Nowlan, and G.E. Hinton, (1991). Adaptive mixtures of local experts. *Neural Computation 3*, 79-87.

[23] Jordan, M. I., Rumelhart, D. E. (1992). Forward models: Supervised learning with a distal teacher. *Cognitive Science, 16*, 307-354.

[24] Kawato, M. (1990). Computational Schemes and Neural Network Models for Formation and Control of Multijoint Arm Trajectory. Chapter 9, *Neural Networks for Control.* Cambridge: MIT Press.

[25] Katayama, M., Kawato, M. (1991). Learning Trajectory and Force Control of an Artificial Muscle Arm by Parallel-hierarchical Neural network Model. Cognitive Processes Department, ATR Lab, Kyoto, Japan.

[26] Kandel, E.R., Schwartz, J.H., Jessel, T.M. (1991). *Principles of Neural Science.* Elsevier.

[27] Kreisselmeier, G. (1977). Adaptive Observers with Exponential Rate of Convergence. *IEEE Trans on Auto Control*, Feb 1977.

[28] Koivo, A.J., & Guo, T., (1983). Adaptive Linear Controller for Robotic Manipulators. *IEEE Trans on Automatic Control.* AC-28(2).

[29] Lewis, F.K., Liu, K., Yesildirek, A., (1995). Neural Net Robot Controller with Guaranteed Tracking Performance. *IEEE Trans on Neural Networks.* Vol.6, No.3.

[30] Liu, J.S., Yuan, K. Y. (1989). On Tracking control for affine nonlinear systems by sliding mode. *Systems & Control Letters 13.* North Holland.

[31] Miall, R., Weir,D., Wolpert,D., Stein, J. (1993). Is the cerebellum a Smith Predictor? *Journal of Motor Behavior*, 25(3):203-216.

[32] Miller, W.T., Sutton, R.S., Werbos, P.J., (1990). *Neural Networks for Control.* M.I.T. Press.

[33] Møller, M. F. (1990). A scaled conjugate gradient algorithm for fast supervised learning. (Tech. Rep. PB-339), Computer Science Department, University of Aarhus, Denmark.

[34] Narendra, K. S., Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks, 1,* 4-27.

[35] Narendra, K. S. (1991). Adaptive Control Using Neural Networks. *Neural Networks for Control.* M.I.T. Press.

[36] Poggio, T., Girosi, F. (1990). Extensions of a Theory of Networks for Approximation and Learning: dimensionality reduction and clustering. A.I. Memo No. 1167. M.I.T.

[37] Poggio, T., Girosi, F. (1992). Networks for Approximation and Learning. *Foundations of Neural Networks.* IEEE Press.

[38] Robinson, D.A. (1981). The use of control system analysis in the neurophysiology of eye movements. *Annual Review of Neuroscience* 4, 463-503.

[39] Rosen, J. B. (1967). Optimal control and convex programming. In J. Abadie, (Ed.), *Nonlinear Programming.* New York: John Wiley.

[40] Sadegh, N., Horowitz, R. (1987). Stability Analysis of an Adaptive Controller for Robotic Manipulators. *Proceedings of IEEE ROBOT'87,* Raleigh, NC, pp. 1223-1229.

[41] Schmidt, R.A. (1982). *Motor Control and Learning.* Human Kinetics.

[42] Slotine J-J.E., Li, W. (1987). On the Adaptive Control of Robot Manipulators". *International Journal of Robotic Research.* vol. 6, no. 3.

[43] Slotine J-J.E., Li, W. (1991). *Applied Nonlinear Control.* NJ: Prentice Hall.

[44] Stoten, D.P., (1990). *Model Reference Adaptive Control of Manipulators.* John Wiley.

[45] Werbos, P. (1983). *Energy Models and Studies.* B. Lev. Ed., North Holland.

[46] Wilberg, R.B. (1991). *The Learning, Memory, and Perception of Perceptual-Motor Skills.* North-Holland.

[47] Wolpert, D.M., Ghahramani, Z., Jordan, M. (in press). Forward dynamic models in human motor control:Psychophysical evidence. *Advances in Neural Information Processing Systems 7.* Cambridge: MIT Press.

[48] Yurkovich, S., Tzes, A.P. (1990). Experiments in Identification and Control of Flexible-Link Manipulators. *IEEE Control Engineering.* v10 no.2, pp.41-47, Feb. 1990.

[49] Zbikowski, R.W. (1994). *Recurrent Neural Networks: Some Control Aspects.* PhD dissertation, Glasgow University, Glasgow, Scotland.

THESIS PROCESSING SLIP

FIXED FIELD: ill _____ name _____

index _____ biblio _____

▶ COPIES (Archives) Aero Dewey Eng Hum

Lindgren Music Rotch (Science) (Scher-Plough)

TITLE VARIES: ▶☑ See Fiche only degree Book

NAME VARIES: ▶☐ _____

IMPRINT: (COPYRIGHT) _____

▶ COLLATION: 101 p.

▶ ADD. DEGREE: _____ ▶ DEPT.: _____

SUPERVISORS: _____

NOTES:

cat'r: date:

page: 559

▶ DEPT: BCS

▶ YEAR: 1995 ▶ DEGREE: Ph.D.

▶ NAME: FUN, ~~We~~ Wey