(1)

GC
7.8
.H82
1994

Comparison of Neural and Control Theoretic Techniques for Nonlinear Dynamic Systems

by

He Huang

B. S., University of Science and Technology of China (1990)

Submitted in partial fulfillment of the requirements for the dual degrees of

MASTER OF SCIENCE IN OCEANOGRAPHIC ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

and the

WOODS HOLE OCEANOGRAPHIC INSTITUTION

and

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1994

Signature of Author ....[illegible]........
Department of Ocean Engineering, MIT and the MIT-WHOI Joint Program in Oceanographic Engineering

Certified by ....[illegible]........
Dr. Dana R. Yoerger
Associate Scientist, Woods Hole Oceanographic Institution
Thesis Supervisor

Certified by ....[illegible]........
Prof. Jean-Jacques E. Slotine
Associate Professor, Massachusetts Institute of Technology
Thesis Reader

Accepted by ....[illegible]........
Professor Arthur B. Baggeroer
Chairman, Joint Committee for Oceanographic Engineering, Massachusetts Institute of Technology and the Woods Hole Oceanographic Institution

1994

3401

Comparison of Neural and Control Theoretic Techniques for Nonlinear Dynamic Systems

by

He Huang

Submitted to the Massachusetts Institute of Technology/
Woods Hole Oceanographic Institution
Joint Program in Oceanographic Engineering
in May, 1994 in partial fulfillment of the
requirements for the dual degrees of

MASTER OF SCIENCE IN OCEANOGRAPHIC ENGINEERING

and

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

Abstract

This thesis compares classical nonlinear control theoretic techniques with recently developed neural network control methods based on the simulation and experimental results on a simple electromechanical system. The system has a configuration-dependent inertia, which contributes a substantial nonlinearity. The controllers being studied include PID, sliding control, adaptive sliding control, and two different controllers based on neural networks: one uses feedback error learning approach while the other uses a Gaussian network control method. The Gaussian network controller is tested only in simulation due to lack of time. These controllers are evaluated based on the amount of a priori knowledge required, tracking performance, stability guarantees, and computational requirements. Suggestions for choosing appropriate control techniques to one's specific control applications are provided based on these partial comparison results.

Thesis Supervisor: Dr. Dana R. Yoerger

Associate Scientist
Woods Hole Oceanographic Institution

Acknowledgments

I greatly appreciate the efforts of my thesis advisor and reader, Dr. Dana Yoerger and Prof. Jean-Jacques Slotine, who steered me through this difficult project bridging the worlds of underwater robotics, control theory and neural networks.

My thanks also go to everyone at the Deep Submergence Laboratory of the Woods Hole Oceanographic Institution, for all the help and support, especially from Dr. Nathan Ulrich, Dr. Louis Whitcomb, Dr. Franz Hover, Dave Mindell, Marty Marra, Todd Morrison, Tad Snow, Hanu Singh and Dan Potter.

I would like to thank Prof. Arthur Baggeroer and the MIT/WHOI Joint Program staff, who assisted me in many ways during this work.

Thank you, my friends in MIT 5-007, for all the helpful discussions.

Finally, I thank my parents, my husband Xiaoou Tang, my brother Shan, and my good friends Dan Li and Changle Fang. Without their support and encouragement, I would not have been able to complete this thesis.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Underwater vehicles are important tools for exploration of the oceans. They are being applied to a wide range of tasks including hazardous waste clean-up, dam and bridge inspections, and ocean bottom geological surveys. The Deep Submergence Lab at Woods Hole Oceanographic Institution has developed several underwater vehicles, and deployed them in dozens of ocean science expeditions.

Most of the current underwater vehicles are remotely operated through cables. Human pilots are heavily involved during the operations. Precise, repeatable computer control of the vehicles will significantly reduce the operator's workload and provide better performance. However, due to the nonlinearity and uncertainties introduced by hydrodynamic drag and effective mass properties, precise control of an underwater vehicle is very difficult to realize. Traditional well developed linear control techniques can only be applied to this highly nonlinear and uncertain scenario by compromising performance.

The same is true with the control of a manipulator on an underwater vehicle, which is different from its counterpart on the land vehicle. The hydrodynamic force and other factors add nonlinearities into the dynamics, so choosing a good nonlinear control method on the vehicle and manipulator becomes necessary and important.

1.2 Overview of Traditional Control Theory and Neural Network Control Methods

1.2.1 Traditional Control Theory

Traditional linear control, which is a well developed control technique, performs poorly on nonlinear dynamic systems like underwater vehicles because the dynamics model must be linearized within a small range of operation and consequently loses its accuracy in representing the whole physical plant. While good performance and stability can be reached within the linearized region, stability can not be guaranteed for the overall dynamic system. Hard nonlinearities and model uncertainties often make linear control unable to perform well for nonlinear systems.

Nonlinear control methodologies are thus developed for better control of nonlinear dynamic systems. There is no general method for nonlinear control designs, instead, there is a rich collection of techniques each suitable for particular class of nonlinear control problems. The most used techniques are feedback linearization, robust control, adaptive control and gain-scheduling. The first and the last control techniques are more closely related to linear control methodology and their stability and robustness are not guaranteed. So in this thesis, I choose robust control and adaptive control for the purpose of study and comparison.

A simple approach to robust control is the sliding control methodology. It provides a systematic approach to the problem of maintaining stability and consistent performance in the face of modeling imprecision. It quantifies trade-offs between modeling and performance, greatly simplifies the design process by accepting reduced information about the system. Sliding control has been successfully applied on the underwater vehicles and other nonlinear dynamic systems. It eliminates lengthy system identification efforts and reduces the required tuning of the control system. The operational system can be made robust to unanticipated changes in the vehicle's dynamic parameters. However, the upper bounds on the nonlinearities and the unknown constant or slow varying parameters in the dynamic system have to be estimated for sliding control to be successful.

Adaptive sliding control techniques have also been successfully used to deal with the uncertain and nonlinear dynamics of underwater vehicles [16]. It further provides an adaptation mechanism for the unknown parameters in the dynamic system, thus it achieves better performance if the initial parameter estimates are imprecise. It can be regarded as a control system with on-line parameter estimation. The form of the nonlinearities must be known, along with bounds on the parameter uncertainty. The unknown parameters it adapts to have to be constant or slowly-varying.

1.2.2 Neural Network Control Methods

Neural network control is a fast growing new candidate for nonlinear controls. These controllers have ability to learn the unknown dynamics of the controlled system. The parallel signal processing, computational inexpensive, and adaptive properties of the neural networks also make them appealing to the real time control of underwater vehicles. J. Yuh has developed a multi-layered neural network controller with the error estimated by a critic equation [19] [17] [18] . The only required information about the system dynamics is an estimate of the inertia terms. K. P. Venugopal *et al.* described a direct neural network control scheme with the aid of a gain layer, which is proportional to the inverse of the system Jacobian [14].

This thesis adapts a neural network control technique from Mitsuo Kawato's feedback error learning scheme which uses error backpropagation as the weight adjustment methods [3] [5] [4] [2]. No a priori information about the nonlinear system is required. The on-line neural network learns the inverse dynamics of the system based on the error signals feedback from a conventional PD controller. It simultaneously controls the motion of the system while it learns.

While the backpropagation method is theoretically proven to be convergent, there is no theory regarding its stability when implemented into real time control problems. Moreover, there is no standard criteria to choose the number of layers and nodes within the networks.

A better solution to these problems is a network of gaussian radial basis functions, called Gaussian networks [8]. It uses a network of gaussain radial basis functions to adaptively compensate for the plant nonlinearities. The a priori information about the nonlinear function of the plant is its degree of smoothness. The weight adjustment is determined using Lyapunov theory, so the algorithm is proven to be globally stable, and the tracking errors converge to a neighborhood of zero. Its another feature is that the number of nodes within the network can be decided by the desired performance and the a priori frequency content information about the nonlinear function to be approximated.

The Gaussian network is a controller which combines theoretic control and connectionists approach. It uses the network to its full advantage of learning ability while in the mean time guarantees the stability of the whole system using traditional theoretical approach. The resulting controller is thus robust and retains its high performance.

1.3 Outline of Thesis

This thesis chooses these two neural network controls to compare with the above mentioned three traditional theoretic control techniques. They are evaluated based on the amount of a priori knowledge required, tracking performance, stability guarantees, and computational requirements. Results from simulation and experiment on a simple electromechanical system are studied. But due to lack of time and computational constraints, the Gaussian network control method is not implemented into the experiments. Thus the discussion and evaluation on this controller is limited. Characteristics of these controllers are discussed based on current work and suggestions for choosing appropriate control techniques for different nonlinear dynamic systems are given based on these comparison results.

The outline of the thesis is:

Chapter 2 describes each of the five different control design theories.

Chapter 3 contains simulation results and comparisons of these controls on a sensorimotor system with a nonlinear inertia load design to introduce nonlinear dynamics.

Chapter 4 presents the experiment results and comparisons on the nonlinear sensorimotor system. The experiments of Gaussian network controller remains to be continued in future work.

Chapter 5 summarizes the results of the thesis and offers suggestions for choosing appropriate nonlinear controllers. Limitations of current work and recommendations for future work are discussed.

Chapter 2

Controller Design Techniques

There is no general method for designing nonlinear controllers, instead, there is a rich collection of nonlinear control techniques, each dealing with different classes of nonlinear problems. The techniques studied in this thesis comprised of two groups: one of traditional theoretic nonlinear control techniques, the other of recently developed neural network control approaches. The first group applies to systems with known dynamic structure, but unknown constant or slowly-varying parameters, and can deal with model uncertainties. The second group is for systems without much a priori information about their dynamic structures and parameters. This chapter describes their design principles and procedures.

2.1 Procedure for General Control Design

The objective of control design can be stated as follows: given a physical system to be controlled and the specifications of its desired behavior, construct a feedback control law to make the closed-loop system display the desired behavior. The procedure of constructing the control goes through the following steps, possibly with a few iterations:

1. specify the desired behavior, and select actuators and sensors;

2. model the physical plant by a set of differential equations;

3. design a control law for the system;
4. analyze and simulate the resulting control system;
5. implement the control system in hardware.

Generally, the tasks of control systems, i.e. the desired behavior, can be divided into two categories: stabilization and tracking. The focus of this thesis is on tracking problems. The design objective in tracking control problems is to construct a controller, called a tracker, so that the system output tracks a given time-varying trajectory.

The task of asymptotic tracking can be defined as follows:

Asymptotic Tracking Problem: Given a nonlinear dynamics system described by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{2.1}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) \tag{2.2}$$

and a desired output trajectory $\mathbf{y}_d$, find a control law for the input $\mathbf{u}$ such that, starting from any initial state in a region $\mathbf{\Omega}$, the tracking errors $\mathbf{y}(t) - \mathbf{y}_d(t)$ go to zero, while the whole state $\mathbf{x}$ remains bounded.

This chapter focuses on different control law designs. The desired behavior and dynamic model of the physical plant are assumed to be known and are the same for all different control techniques.

Let the dynamic model of the nonlinear system represented by:

$$x^{(n)}(t) = f(\mathbf{X}; t) + b(\mathbf{X}; t)u(t) + d(t) \tag{2.3}$$

where

$u(t)$ is the control input

x is the output of interest

$\mathbf{X} = [\ x \quad \dot{x} \quad \cdots \quad x^{(n-1)}\ ]^T$ is the state.

$d(t)$ is the disturbance

$f(\mathbf{X};t)$ is the nonlinear function describing the system's dynamics

$b(\mathbf{X};t)$ is the control gain

The desired behavior, i.e., the control problem is to get the state $\mathbf{X}$ to track a specific state

$$\mathbf{X}_d = [\ x_d \quad \dot{x}_d \quad \cdots \quad x_d^{(n-1)}\ ]^T \tag{2.4}$$

in the presence of model imprecision on $f(\mathbf{X};t)$ and $b(\mathbf{X};t)$, and of disturbances $d(t)$.

If we define the tracking error vector as:

$$\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{X}_d = [\ \tilde{x} \quad \dot{\tilde{x}} \quad \cdots \quad \tilde{x}^{(n-1)}\ ]^T \tag{2.5}$$

the control problem of tracking $\mathbf{X} \equiv \mathbf{X}_d$ is equivalent to reaching $\tilde{\mathbf{X}} \equiv 0$ in finite time.

2.2 PID Control

The combination of proportional control, integral control, and derivative control is called proportional-plus-integral-plus-derivative control, also called PID control. This combined control has the advantages of each of the three individual control actions. The equation of a PID controller is given by

$$u(t) = K_p\tilde{x}(t) + K_pT_d\dot{\tilde{x}}(t) + \frac{K_p}{T_i}\int \tilde{x}(t)dt \tag{2.6}$$

or the transfer function is

$$\frac{u(s)}{\tilde{x}(s)} = K_p(1 + T_ds + \frac{1}{T_is}) \tag{2.7}$$

where K_p represents the proportional sensitivity, T_d represents the derivative time, and T_i represents the integral time.

In the proportional control of a plant whose transfer function does not possess a free integrator $1/s$, there is a steady-state error, or offset, in the response to steady disturbance. Such an offset can be eliminated if the integral control action is included in the controller. On the other hand, while removing offset or steady-state error, the integral control action may lead to oscillatory response of slowly decreasing amplitude or even increasing amplitude, both of which are usually undesirable.

Derivative control action, when added to a proportional controller, provides a means of obtaining a controller with high sensitivity. It responds to the rate of change of the actuating error and can produce a significant correction before the magnitude of the actuating error becomes too large. Derivative control thus anticipates the actuating error, initiates an early corrective action, and tends to increase the stability of the system.

Although derivative control does not affect the steady-state error directly, it adds damping to the system and thus permits the use of a larger value of the gain K, which will result in an improvement in the steady-state accuracy. Since derivative control operates on the rate of change of the actuating error and not the actuating error itself, this mode is never used alone. It is always used in combination with proportional or proportional-plus-integral action.

The selection of PID parameters K_p, T_d and T_i is based on the knowledge about the dynamic systems and the desired closed-loop bandwidth.

2.3 Sliding Control

Given the perfect measurement of a linear system's dynamic state and a perfect model, the PID controller can achieve perfect performance. But it may quickly fail in the presence of model uncertainty, measurement noise, computational delays and disturbances. Analysis of the effects of these non-idealities are further complicated by nonlinear dynamics. The issue becomes one of ensuring a nonlinear dynamic system remains robust to non-idealities while minimizing tracking error.

Two major and complementary approaches to dealing with model uncertainty are *robust control* and *adaptive control.* Sliding control methodology is a method of robust control. It provides a systematic approach to the problem of maintaining stability and consistent performance in the face of modeling imprecision.

Sliding Modes are defined as a special kind of motion in the phase space of a dynamic system along a sliding surface for which the control action has discontinuities. This special motion will exist if the state trajectories in the vicinity of the control discontinuity are directed toward the sliding surface. If a sliding mode is properly introduced in a system's dynamics through active control, system behavior will be governed by the selected dynamics on the sliding surface, despite disturbances, nonlinearities, time-variant behavior and modeling uncertainties.

Let's consider the dynamic system described by equation (2.3). A time-varying sliding surface $S(t)$ in the state-space R^n is defined as

$$s(\mathbf{X};t) = 0 \tag{2.8}$$

with

$$s(\mathbf{X};t) = (\frac{d}{dt} + \lambda)^{n-1}\tilde{x}, \ \ \lambda > 0 \tag{2.9}$$

where λ is a positive constant related to the desired control bandwidth.

The tracking problem is now transformed to remaining the system state on the sliding surface $S(t)$ for all $t > 0$. This can be seen by considering $s \equiv 0$ as a linear differential equation whose unique solution is $\tilde{x} \equiv 0$, given initial condition:

$$\tilde{\mathbf{X}}|_{t=0} = 0 \tag{2.10}$$

This positive invariance of $S(t)$ can be reached by choosing the control law u of system (2.3) such that outside $S(t)$

$$\frac{1}{2}\frac{d}{dt}s^2(x;t) \leq -\eta|s| \tag{2.11}$$

where η is a positive constant. (2.11) is called the sliding condition. It constrains the trajectories of the system to point towards the surface $S(t)$.

The idea behind (2.9) and (2.11) is to pick a well behaved function of the tracking error, s, according to (2.9) and then select the feedback control law u such that s^2 remains a Lyapunov function of the closed-loop system despite the presence of model imprecision and disturbances. This guarantees the robustness and stability of the closed-loop system. Even when the initial condition (2.10) is not met, the surface $S(t)$ will still be reached in a finite time smaller than $s(\mathbf{X}(0);0)/\eta$, given the sliding condition (2.11) is verified.

The detailed controller design procedure is described in the following two sections. Section 2.3.1 shows how to select a feedback control law u to verify sliding condition (2.11) and account for the modeling imprecision. This control law leads to control chattering. Section 2.3.2 describes how to eliminate the chattering to achieve an optimal trade-off between control bandwidth and tracking precision.

2.3.1 Perfect Tracking Using Switched Control Laws

This section illustrates how to construct a control law to verify sliding condition (2.11) given bounds on uncertainties on $f(\mathbf{X};t)$ and $b(\mathbf{X};t)$.

Consider a second-order dynamic system

$$\ddot{x} = f + bu \tag{2.12}$$

The nonlinear dynamics f is not known exactly, but estimated as $\hat{f}$. The estimation error on f is assumed to be bounded by some known function F:

$$|\hat{f} - f| \leq F \tag{2.13}$$

Similarly, the control gain b is unknown but of known bounds:

$$0 < b_{min} \leq b \leq b_{max} \tag{2.14}$$

The estimate $\hat{b}$ of gain b is the geometric mean of the above bounds:

$$\hat{b} = \sqrt{b_{min} b_{max}} \tag{2.15}$$

Bounds (2.14) can then be written in the form

$$\beta^{-1} \leq \frac{\hat{b}}{b} \leq \beta \tag{2.16}$$

where

$$\beta = \sqrt{b_{max}/b_{min}} \tag{2.17}$$

In order to have the system track $x(t) \equiv x_d(t)$, we define a sliding surface $s = 0$ according to (2.9), namely:

$$s = (\frac{d}{dt} + \lambda)\tilde{x} \tag{2.18}$$

We then have:

$$\dot{s} = \ddot{x} - \ddot{x}_d + \lambda\dot{\tilde{x}} = f + bu - \ddot{x}_d + \lambda\dot{\tilde{x}} \tag{2.19}$$

The best approximation $\hat{u}$ of a continuous control law that would achieve $\dot{s} = 0$ is thus:

$$\hat{u} = \widehat{bu} = -\hat{f} + \ddot{x}_d - \lambda\dot{\tilde{x}} \tag{2.20}$$

In order to satisfy the sliding condition (2.11) despite uncertainties on the dynamics f and the control gain b, we add to $\hat{u}$ a term discontinuous across the surface $s = 0$:

$$\begin{aligned} u &= \hat{b}^{-1}[\hat{u} - k\text{sgn}(s)] && (2.21) \\ &= \hat{b}^{-1}[-\hat{f} + \ddot{x}_d - \lambda\dot{\tilde{x}} - k\text{sgn}(s)] && (2.22) \end{aligned}$$

By choosing k in (2.21) to be large enough,

$$k \geq \beta(F + \eta) + (\beta - 1)|\hat{u}| \tag{2.23}$$

we can guarantee that (2.11) is verified. Indeed, we have from (2.19) to (2.22)

$$\dot{s} = (f - b\hat{b}^{-1}\hat{f}) + (1 - b\hat{b}^{-1})(-\ddot{x}_d + \lambda\dot{\tilde{x}}) - b\hat{b}^{-1}k\text{sgn}(s) \tag{2.24}$$

In order to let

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}s^2 &= \dot{s}s \\ &= [(f - b\hat{b}^{-1}\hat{f}) + (1 - b\hat{b}^{-1})(-\ddot{x}_d + \lambda\dot{\tilde{x}})]s - b\hat{b}^{-1}k|s| \\ &\leq -\eta|s|\end{aligned}$$

k must verify

$$k \geq |\hat{b}b^{-1}f - \hat{f} + (\hat{b}b^{-1} - 1)(-\ddot{x}_d + \lambda\dot{\tilde{x}})| + \hat{b}b^{-1}\eta \tag{2.25}$$

Since $f = \hat{f} + (f - \hat{f})$, where $|f - \hat{f}| \leq F$, this leads to

$$k \geq \hat{b}b^{-1}F + |\hat{b}b^{-1} - 1| \cdot |\hat{f} - \ddot{x}_d + \lambda\dot{\tilde{x}})| + \hat{b}b^{-1}\eta \tag{2.26}$$

and using $\hat{b}b^{-1} \leq \beta$ leads to (2.23).

2.3.2 Continuous Control Laws to Approximate Switched Control

The control law derived from the above section is discontinuous across the surface $S(t)$ and leads to control chattering, which is usually undesirable in practice since it involves high control activity and may excite high-frequency dynamics neglected in the course of modeling. In this section, continuous control laws are used to eliminate the chattering.

The control discontinuity is smoothed out by introducing a thin *boundary layer* neighboring the switching surface:

$$B(t) = \{\mathbf{X}, |s(\mathbf{X};t)| \leq \Phi\}; \quad \Phi > 0 \tag{2.27}$$

where Φ is the boundary layer *thickness*, and is made to be *time varying* in order to exploit the maximum control bandwidth available. Control smoothing is achieved by choosing control law u *outside* $B(t)$ as before, which guarantees boundary layer attractiveness and hence positive invariance –all trajectories starting inside $B(t = 0)$

remain inside $B(t)$ for all $t \geq 0$–and then interpolation u inside $B(t)$, replacing the term $\text{sgn}(s)$ in the expression of u by s/Φ. As proved by Slotine(1983), this leads to *tracking to within a guaranteed precision* $\varepsilon = \Phi/\lambda^{n-1}$, and more generally guarantees that for all trajectories starting inside $B(t=0)$

$$|\tilde{x}^{(i)}(t)| \leq (2\lambda)^i \varepsilon; \;\; i = 0, \cdots, n-1 \tag{2.28}$$

The sliding condition (2.11) is now modified to maintain attractiveness of the boundary layer when Φ is allowed to vary with time.

$$|s| \geq \Phi \Rightarrow \frac{1}{2}\frac{d}{dt}s^2 \leq (\dot{\Phi} - \eta)|s| \tag{2.29}$$

The term $k(\mathbf{X};t)\text{sgn}(s)$ obtained from switched control law u is also replaced by $\bar{k}(\mathbf{X};t)sat(s/\Phi)$, where:

$$\bar{k}(\mathbf{X}_d;t) = k(\mathbf{X};t) - k(\mathbf{X}_d;t) + \frac{\lambda\Phi}{\beta_d} \tag{2.30}$$

with $\beta_d = \beta(\mathbf{X}_d;t)$.

Accordingly, control law u becomes:

$$u = \hat{b}^{-1}[\hat{u} - \bar{k}\text{sat}(s/\Phi)] \tag{2.31}$$

The desired time-history of boundary layer thickness Φ is called *balance condition* and is defined according to the value of $k(\mathbf{X}_d;t)$:

$$k(\mathbf{X}_d;t) \geq \frac{\lambda\Phi}{\beta_d} \quad \Rightarrow \quad \dot{\Phi} + \lambda\Phi = \beta_d k(\mathbf{X}_d;t) \tag{2.32}$$

$$k(\mathbf{X}_d;t) \leq \frac{\lambda\Phi}{\beta_d} \quad \Rightarrow \quad \dot{\Phi} + \frac{\lambda\Phi}{\beta_d^2} = \frac{k(\mathbf{X}_d;t)}{\beta_d} \tag{2.33}$$

with initial condition $\Phi(0)$ defined as:

$$\Phi(0) = \beta_d k(\mathbf{X}_d(0);(0))/\lambda \tag{2.34}$$

The balance conditions have practical implications in terms of design / modeling / performance trade-offs. Neglecting time-constants of order $1/\lambda$, conditions (2.32) and (2.33) can be written

$$\lambda^n \varepsilon \approx \beta_d k(\mathbf{X}_d; t) \tag{2.35}$$

that is

$$\begin{aligned} (\text{bandwidth})^n \;\; &\times \;\; (\text{tracking precision}) \\ &\approx \;\; (\text{parametric uncertainty measured along the desired trajectory}) \end{aligned}$$

It shows that the balance conditions specify the best tracking performance attainable, given the desired control bandwidth and the extent of parameter uncertainty.

2.4 Adaptive Sliding Control

In order to improve performance when large parametric uncertainties are present, an *adaptive* sliding controller is introduced, where uncertain parameters are estimated on-line based on the algebraic distance of the current state to the boundary layer. This structure leads naturally to active adaptation only when the system is outside the boundary layer, avoiding the long term drift frequently experienced in parameter estimation schemes. The developed adaptive controller structure is based on the premise that there be no adaptation to that which can be modeled but adapt only to the complex dynamic effects which cannot be simply modeled.

The basic form of the adaptive sliding controller is applicable to systems of the type

$$x^{(n)} + \sum_{i=1}^{r} a_i Y_i = bu + d \tag{2.36}$$

where Y_i are known continuous, possibly nonlinear functions of the state variables, parameters a_i and control gain b are unknown but constant, and $d = d(t)$ is a bounded disturbance.

$$|d(t)| \leq D \tag{2.37}$$

Let's consider the second order dynamic system as follows:

$$\ddot{x} + aY = bu + d \tag{2.38}$$

As in the previous section, the sliding surface $s = 0$ is defined as

$$s = (\frac{d}{dt} + \lambda)\tilde{x} \tag{2.39}$$

The control discontinuities are smoothed out inside the boundary layer $B(t)$. The boundary layer thickness Φ is constant here, since residual dynamic uncertainty would be time invariant if adaptation were perfect, as it would be owing only to $d(t)$. To derive a control law that ensures convergence to the boundary, a Lyapunov function $V(t)$ is defined as

$$V(t) = \frac{1}{2}[s_\Delta^2 + b((\hat{h} - \frac{a}{b})^2 + (\hat{b}^{-1} - b^{-1})^2)] \tag{2.40}$$

where

$s_\Delta = s - \Phi\text{sat}(s/\Phi)$ is a measure of the algebraic distance of the current state to the boundary layer

$\hat{h}$ is the estimate of the (a/b)

$\hat{b}$ is the estimate of b

The control law u is selected as

$$u = \hat{h}Y - \hat{b}^{-1}[u^* + (D + \eta)\text{sat}(s/\Phi)] \tag{2.41}$$

where

$$u^* = -\ddot{x} + \lambda\dot{\tilde{x}} \tag{2.42}$$

Noting that

$$\begin{aligned} \dot{s}_\Delta &= \dot{s} \text{ outside the boundary layer} \\ \dot{s}_\Delta &= 0 \text{ inside the boundary layer} \end{aligned}$$

we get

$$\begin{aligned}\dot{V}(t) &= (b\hat{h}-a)(Ys_\Delta+\dot{\hat{h}})-(b\hat{b}^{-1}-1)u^* s_\Delta - b\hat{b}^{-1}(D+\eta)s_\Delta \mathrm{sat}(s/\Phi)+ \\ &\quad d\cdot s_\Delta+(b\hat{b}^{-1}-1)\dot{\hat{b}}^{-1}\end{aligned}$$

where $\dot{\hat{b}}^{-1}$ is the time-derivative of $\hat{b}^{-1}$. $\hat{h}$ and $\hat{b}$ are estimated on-line according to the following adaptation laws:

$$\dot{\hat{h}} = -Ys_\Delta \tag{2.43}$$

$$\dot{\hat{b}}^{-1} = [u^*+(D+\eta)\mathrm{sat}(s/\Phi)]s_\Delta \tag{2.44}$$

which leads to

$$\dot{V}(t) = [-(D+\eta)\mathrm{sat}(s/\Phi)+d]s_\Delta \tag{2.45}$$

so that

$$\dot{V}(t) \leq -\eta|s_\Delta| \tag{2.46}$$

outside the boundary layer. The adaptation laws (2.43) and (2.44) show that the adaptation ceases as soon as the boundary layer is reached. This avoids the undesirable long term drift found in many adaptive schemes, and provides a consistent rule on when to stop adaptation. Definition (2.40) implies that $\dot{V}=0$ inside the boundary layer, which shows that (2.46) is valid everywhere and thus guarantees that trajectories eventually converge to the boundary layer.

2.5 Feedback Error Learning Control

Nonlinear control design techniques like sliding control and adaptive sliding control have been successfully used on some nonlinear systems. Yet, the system's dynamic structure has to be understood beforehand and the uncertain parameters have to be estimated. This section introduces integration of a neural networks scheme called feedback error learning into the trajectory control of nonlinear systems. No a priori information about the system dynamics is required, the network will gradually

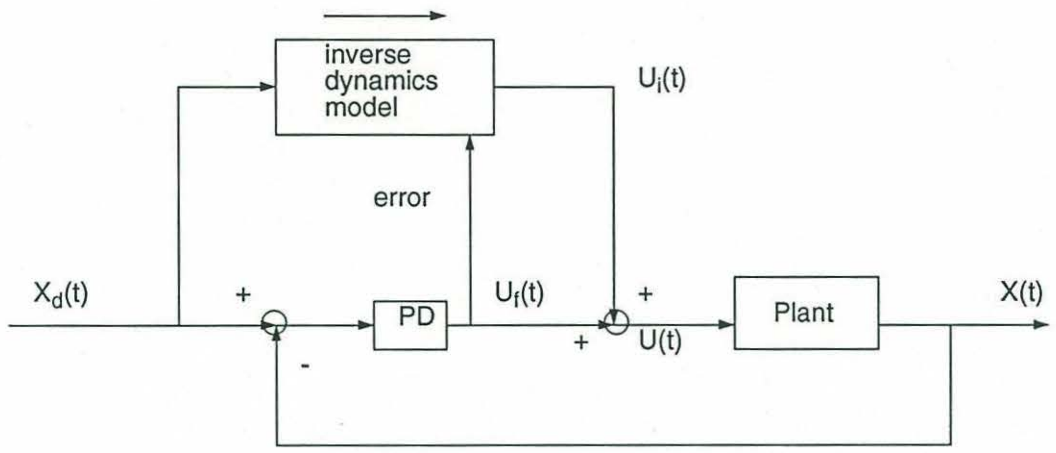


Figure 2-1: Structure of feedback error learning control

learn the inverse dynamics model from error signals feedback from a conventional PD controller. It simultaneously controls the motion of the system while it learns.

Let's consider again controlling the dynamic system represented by equation (2.3). Figure 2-1 shows the control system structure using the feedback error learning scheme. The inverse dynamics model is represented by a neural network. The total control force $U(t)$ fed to the dynamic system is the sum of the feedback control force $U_f(t)$ from the PD controller and the feedforward control force $U_i(t)$ from the neural network controller.

$$\begin{aligned} U(t) &= U_f(t) + U_i(t) && (2.47) \\ U_f(t) &= K_p(X_d(t) - X(t)) + K_d(\dot{X}_d(t) - \dot{X}(t)) && (2.48) \end{aligned}$$

where K_p and K_d are feedback gains of the proportional term and the differential term.

The inverse-dynamics model receives the desired trajectory $X_d(t)$, and uses the feedback control force $U_f(t)$ as the error signal to adjust its weights. The training algorithm is the error-back propagation method developed by Rumelhart *et al.* [10]. As learning proceeds, the feedforward network controller will provide a greater part of the control action, while the feedback signal will gradually decrease to zero. Since the

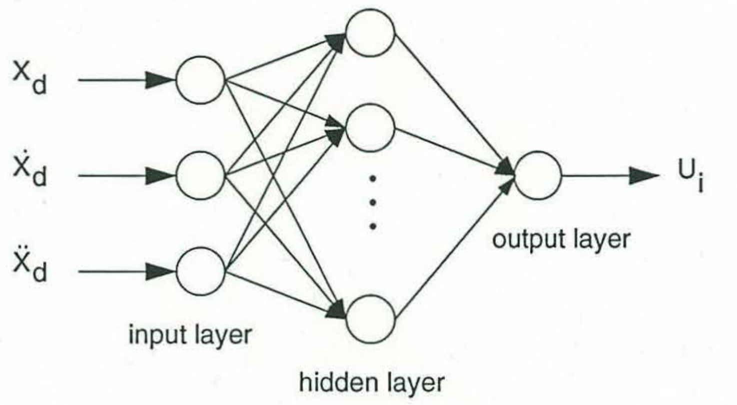


Figure 2-2: Structure of a three-layer network

network receives only the desired trajectory as the input, and produces the required control force to the system, it is a representation of the nonlinear model of the inverse dynamics instead of just being a neural network version of the PD controller.

As shown in Figure 2-2, the network of the inverse dynamics model has three layers. The input layer neurons represent desired trajectory, velocity, and acceleration. The output layer neurons represent the feedforward control forces.The nonlinear characteristics of the inverse dynamics model of the system is learned and represented by this three layered cascade of linear-weighted summation and sigmoidal nonlinearity.

Let x_j^I, y_j^I represent the weighted sum of the input and output of the j-th neuron in the input layer, their relation is:

$$y_j^I = x_j^I \tag{2.49}$$

Next, let x_k^H, y_k^H represent the weighted sum of the input and output of the k-th neuron in the hidden layer, and the weight from the j-th neuron in the input layer to the k-th neuron in the hidden layer represented by w_{jk}^{IH}, we have:

$$x_k^H = \sum w_{jk}^{IH} y_j^I \tag{2.50}$$

$$y_k^H = f(x_k^H) \tag{2.51}$$

Finally, the m-th neuron of the output layer is represented by x_m^O, and y_m^O, and the weight from the k-th neuron in the hidden layer to the m-th neuron in the output layer is w_{km}^{HO}:

$$\begin{aligned} x_m^O &= \sum w_{km}^{HO} y_k^H && (2.52) \\ y_m^O &= f(x_m^O) && (2.53) \end{aligned}$$

The control force output from the network is:

$$u_{im} = y_m^O \qquad (2.54)$$

Here, f is the widely used sigmoid function:

$$f(x) = \frac{1}{1+e^{-x}} \qquad (2.55)$$

The neuron weights are updated according to the error-back propagation method, using the feedback signal from the PD controller as the error. Let Δ represent the increment of weights and ρ be the learning rate:

$$\begin{aligned} \Delta w_{km}^{HO} &= \rho y_k^H \dot{f}(x_m^O) u_{fm} && (2.56) \\ \delta_k^H &= \sum_m w_{km}^{HO} \dot{f}(x_m^O) u_{fm} && (2.57) \\ \Delta w_{jk}^{IH} &= \rho y_j^I \dot{f}(x_k^H) \delta_k^H && (2.58) \end{aligned}$$

The control command fed to the system is the sum of the feedforward force and feedback force.

$$u_m = u_{im} + u_{fm} \qquad (2.59)$$

2.6 Gaussian Network Control

Another kind of neural network control method is studied with the previous controllers. It's called Gaussian network control, which uses a network of gaussian radial basis functions to adaptively compensate for the plant nonlinearities [8]. The a priori information about the nonlinear function of the plant is its degree of smoothness. The weight adjustment is determined using Lyapunov theory, so the algorithm is proven to be globally stable, and the tracking errors converge to a neighborhood of zero.

Sampling theory shows that bandlimited functions can be exactly represented at a countable set of points using an appropriately chosen interpolating function. When approximation is allowed, the bandlimited restriction can be relaxed and the interpolating function gets a wider selection. Specifically, the nonlinear function $f(\mathbf{x})$ can be approximated by smoothly truncated outside a compact set $\mathbf{A}$ so that the function can be Fourier transformed. Then by truncating the spectrum, the function can be bandlimited and uniformly approximated by

$$f(\mathbf{x}) = \sum_{I \in I_o} c_I g_\sigma(\mathbf{x} - \xi_I) \tag{2.60}$$

where c_I are the weighting coefficients and ξ_I form a regular lattice covering the subset $\mathbf{A}_T$, which is larger than the compact set $\mathbf{A}$ by an n-ball of radius ρ surrounding each point of $\mathbf{x} \in \mathbf{A}$. The index set is $I_o = \{I \mid \xi_I \in \mathbf{A}_T\}$.

$g_\sigma(\mathbf{x} - \xi)$ is the gaussian radial basis function given by:

$$g_\sigma(\mathbf{x} - \xi) = \exp(-\pi\sigma_\nu^2 ||\mathbf{x} - \xi||^2) = \exp[-\pi\sigma_\nu^2 (\mathbf{x} - \xi)^T (\mathbf{x} - \xi)] \tag{2.61}$$

Here ξ is the center of the radial gaussian, and σ_ν^2 is a measure of its essential width. Gaussian functions are well suited for the role of interpolating function because they are bounded, strictly positive and absolutely integrable, and they are their own Fourier transforms.

Expansion (2.60) maps onto a network with a single hidden layer nodes. Each node represents one term in the series with the weight ξ_I connecting between the input $\mathbf{x}$ and the node. It then calculates the activation energy $r_I^2 = ||\mathbf{x} - \xi_I||^2$ and outputs a gaussian function of the activation, $\exp(-\pi r_I^2 \sigma_\nu^2)$. The output of the network is weighted summation of the output of each node, with each weight equals to c_I. Figure 2-3 shows the structure of the network described above.

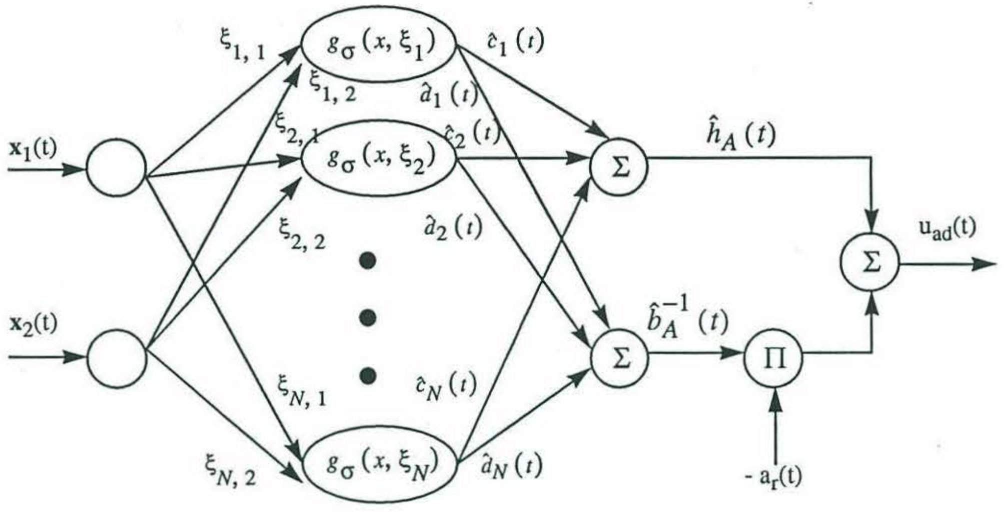


Figure 2-3: Structure of the Gaussian network

The next step is the construction of the controller. Consider the nonlinear dynamics system

$$x^{(n)}(t) + f(\mathbf{x}(t)) = b(\mathbf{x}(t))u(t) \tag{2.62}$$

define the unknown nonlinear function $h = b^{-1}f$, and let h_A and b_A^{-1} be the radial gaussian network approximations to the functions h and b^{-1} respectively with approximation error ϵ_h and ϵ_b as small as desired on the chosen set $\mathbf{A}$.

Figure 2-4 shows the structure of the control law. To guarantee the stability of the whole dynamic system, the control law is constructed as a combination of three components: a linear PD control, a sliding control and an adaptive control represented by the Gaussian network.

$$\begin{aligned} u(t) &= -k_D s(t) - \frac{1}{2}M_2(\mathbf{x}(t))||\mathbf{x}(t)||s_\Delta(t) + m(t)u_{sl}(t) \\ &\quad +(1-m(t))[\hat{h}_A(t,\mathbf{x}(t)) - \widehat{b_A^{-1}}(t,\mathbf{x}(t))a_r(t)] \end{aligned} \tag{2.63}$$

$m(t)$ is a modulation allowing the controller to smoothly transition between sliding and adaptive controls.

$$m(t) = \max(0, \mathrm{sat}(\frac{r(t)-1}{\Psi})) \tag{2.64}$$

where $r(t) = ||\mathbf{x}(t) - \mathbf{x}_0||$.

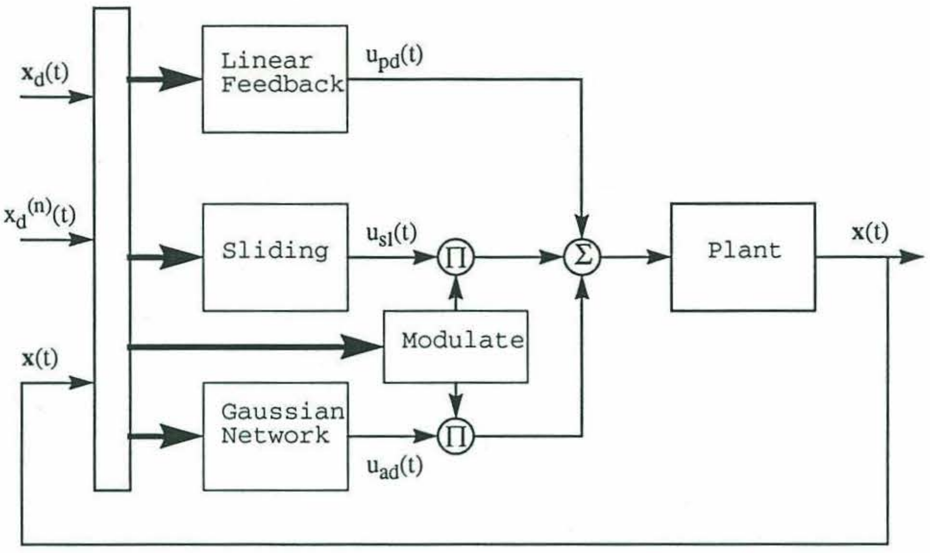


Figure 2-4: Structure of the Gaussian network controller

$$a_r(t) = \lambda_v^T \tilde{\mathbf{x}}(t) - x_d^{(n)}(t) \tag{2.65}$$

with $\lambda_v^T = [\ 0,\ \ \lambda^{n-1},\ \ (n-1)\lambda^{n-2},\ \ \cdots,\ \ (n-1)\lambda\ ]$ and $x_d^{(n)}(t)$ is the nth derivative of the desired trajectory.

The adaptive components $\hat{h}_A$ and $\widehat{b_A^{-1}}$ are realized as the outputs of a single gaussian network, with two sets of output weights: $\hat{c}_I(t)$ and $\hat{d}_I(t)$ for each node in the hidden layer.

$$\begin{aligned} \hat{h}_A(t, \mathbf{x}(t)) &= \sum_{I \in I_o} \hat{c}_I(t) g_\sigma(\mathbf{x}(t) - \xi_I) \\ \widehat{b_A^{-1}}(t, \mathbf{x}(t)) &= \sum_{I \in I_o} \hat{d}_I(t) g_\sigma(\mathbf{x}(t) - \xi_I) \end{aligned} \tag{2.66}$$

The output weights are adjusted according to the following adaptation law:

$$\dot{\hat{c}}_I(t) = -k_{a1}[(1 - m(t)) s_\Delta(t) g_\sigma(\mathbf{x}(t) - \xi_I)] \tag{2.67}$$

$$\dot{\hat{d}}_I(t) = k_{a2}\, a_r(t)[(1 - m(t)) s_\Delta(t) g_\sigma(\mathbf{x}(t) - \xi_I)] \tag{2.68}$$

where positive constants k_{a1} and k_{a2} are adaptation rates. See Figure 2-3 for the detailed structure of the adaptive control law.

As proved by Sanner and Slotine in [8], when the parameters in the controller are chosen appropriately according to the a priori knowledge about the smoothness and upper bounds of the nonlinear dynamic functions, the controller thus constructed will be stable and convergent. All the states in the adaptive system will remain bounded and the tracking errors will asymptotically converge to a neighborhood of zero.

Chapter 3

Simulation on a Sensorimotor System

In order to evaluate the performance of different controllers, a sensorimotor system is setup with a nonlinear inertia load representing the nonlinear dynamics. The detailed descriptions of the experiments are given in Chapter 4. This chapter focuses on simulations of the controller designs on a computer before implementing them on the real physical plant. This makes the process of design and analysis easy to carry out without causing physical system failures.

3.1 Nonlinear Model of the Sensorimotor Dynamic System

The nonlinear inertia load structure is shown in Figure 3-1.A weight is attached by cable to a point on the inertia disk through a small pulley. It has a simple design easy to implement, yet the nonlinearity introduced is highly complicated as will be seen from its equation of motion, which is derived by using the Lagrange's equations.

$$\mathcal{L} = \mathcal{T} - \mathcal{U} = \text{Lagrangian} \;=\; \text{Kinetic Energy} - \text{Potential Energy} \qquad (3.1)$$

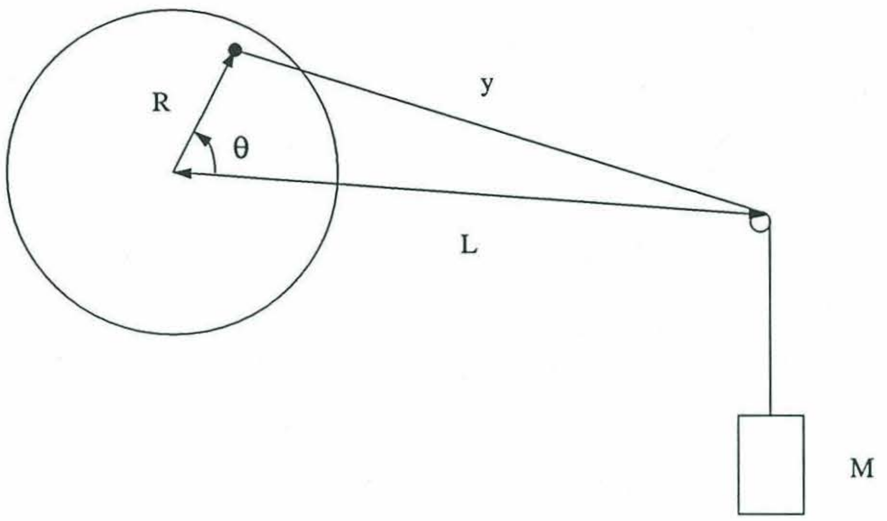


Figure 3-1: Nonlinear inertia structure of the sensorimotor system

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}}\right) - \frac{\partial \mathcal{L}}{\partial \theta} = \sum \mathcal{F} \tag{3.2}$$

where θ is a generalized coordinate which uniquely specifies the location of the object whose motion is being analyzed. In our case, θ is the angle of rotation of the load. $\mathcal{F}$ is the nonconservative generalized force corresponding to the generalized coordinate θ. In our case, it's the torque u applied to the load.

The kinetic energy and potential energy of the nonlinear system are

$$\begin{aligned} \mathcal{T} &= \frac{1}{2}J\dot{\theta}^2 + \frac{1}{2}M\dot{y}^2 \\ \mathcal{U} &= Mgy \end{aligned}$$

so we have

$$\mathcal{L} = \frac{1}{2}J\dot{\theta}^2 + \frac{1}{2}M\dot{y}^2 - Mgy \tag{3.3}$$

Since the relation between y and θ is governed by

$$y = \sqrt{R^2 + L^2 - 2RL\cos\theta} \tag{3.4}$$

$$\dot{y} = \frac{RL\sin\theta}{y}\dot{\theta} \tag{3.5}$$

we have

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \dot\theta} &= J\dot\theta + M\dot y \frac{RL\sin\theta}{y} \\
&= J\dot\theta + M(\frac{RL\sin\theta}{y})^2\dot\theta
\end{aligned}
$$

$$
\begin{aligned}
\frac{d}{dt}(\frac{\partial \mathcal{L}}{\partial \dot\theta}) &= J\ddot\theta + M(\frac{RL\sin\theta}{y})^2\ddot\theta + 2M\dot\theta\frac{RL\sin\theta}{y}[\frac{RL\cos\theta}{y}\dot\theta - \frac{RL\sin\theta}{y^2}\dot y] \\
&= [J + M(\frac{RL\sin\theta}{y})^2]\ddot\theta + \frac{2M\dot\theta^2(RL)^2\sin\theta}{y^2}[\cos\theta - \frac{RL\sin^2\theta}{y^2}]
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta} &= M\dot y[\frac{RL\cos\theta}{y}\dot\theta - \frac{RL\sin\theta}{y^2}\dot\theta\frac{RL\sin\theta}{y}] - Mg\frac{RL\sin\theta}{y} \\
&= \frac{M(RL)^2\sin\theta\cos\theta}{y^2}\dot\theta^2 - M\frac{(RL\sin\theta)^3}{y^4}\dot\theta^2 - \frac{MgRL\sin\theta}{y}
\end{aligned}
$$

Substituting the above expressions into the Lagrangian equation (3.2) yields

$$
(J + \frac{M(RL\sin\theta)^2}{y^2})\ddot\theta + \frac{M(RL)^2\sin\theta}{y^2}[\cos\theta - \frac{RL\sin^2\theta}{y^2}]\dot\theta^2 + \frac{MgRL\sin\theta}{y} = u \quad (3.6)
$$

3.2 Simulation Results

As can be seen from equation (3.6), the dynamics of the sensorimotor system is very complex and nonlinear, thus provides a good basis for evaluating different nonlinear control methodologies. The actual parameter values of the system are

$$
J = .002\ kg\cdot m^2;\ \ M = 0.4749\ km;\ \ R = .0476\ m;\ \ L = .177\ m \qquad (3.7)
$$

The inertia value of the load J is chosen to fall within the operating range of the sensorimotor. The dimensions of R, L and the mass of the weight M are designed to cause the variation of the system inertia to be half the value of J, as shown in Figure (3-2).

The different controllers are simulated on the nonlinear dynamic system (3.6) and

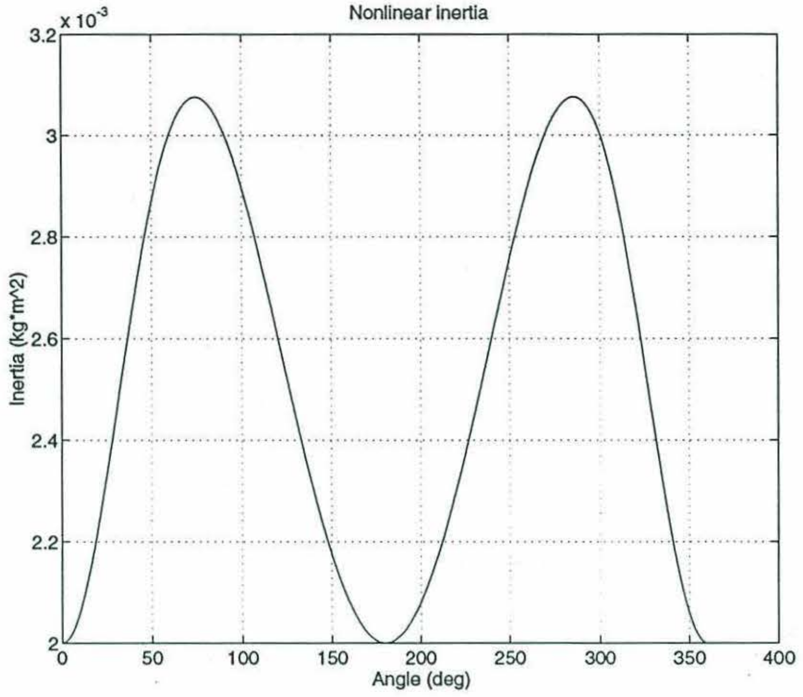


Figure 3-2: Nonlinear inertia plot

their results are compared in the following sections. To establish a fair comparison, all the controllers are designed to tracking the same trajectory shown in Figure 3-3, which is generated by a trapezoidal speed profile also shown in the same figure. The system accelerates to a constant speed and then decelerates until stops and has a duration time of 15 seconds. The closed-loop bandwidth of all controller is also the same: $\lambda = 20$ hz.

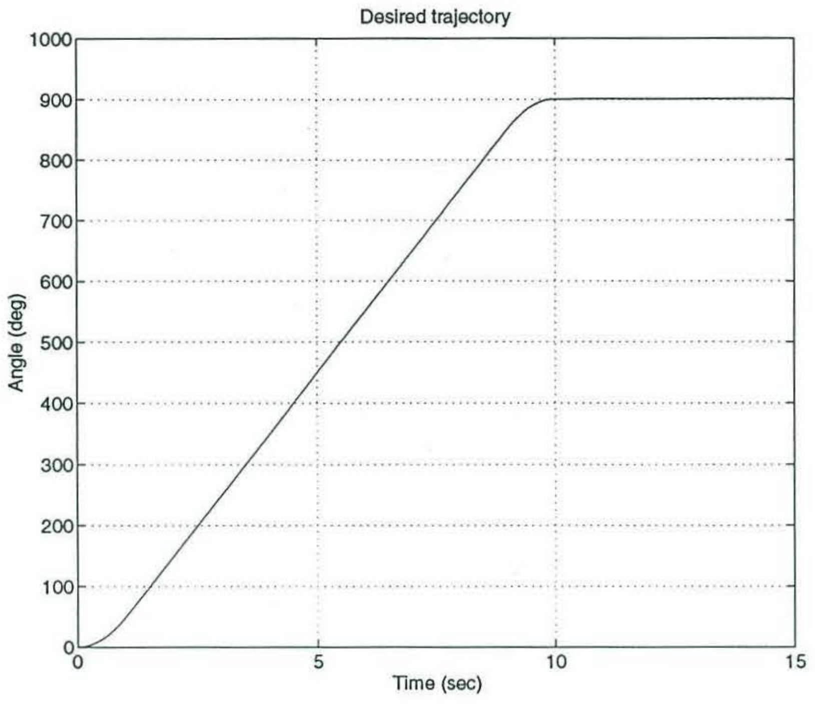
3.2.1 PID Control

The control gains of the PID controller are selected according to the bandwidth λ, time constant τ and damping constant ζ of the nonlinear system.

$$K_i = J\lambda^2/\tau \tag{3.8}$$

$$K_p = J(\lambda^2\tau + 2\zeta\lambda)/\tau \tag{3.9}$$

$$K_d = J(1 + 2\lambda\zeta\tau)/\tau \tag{3.10}$$



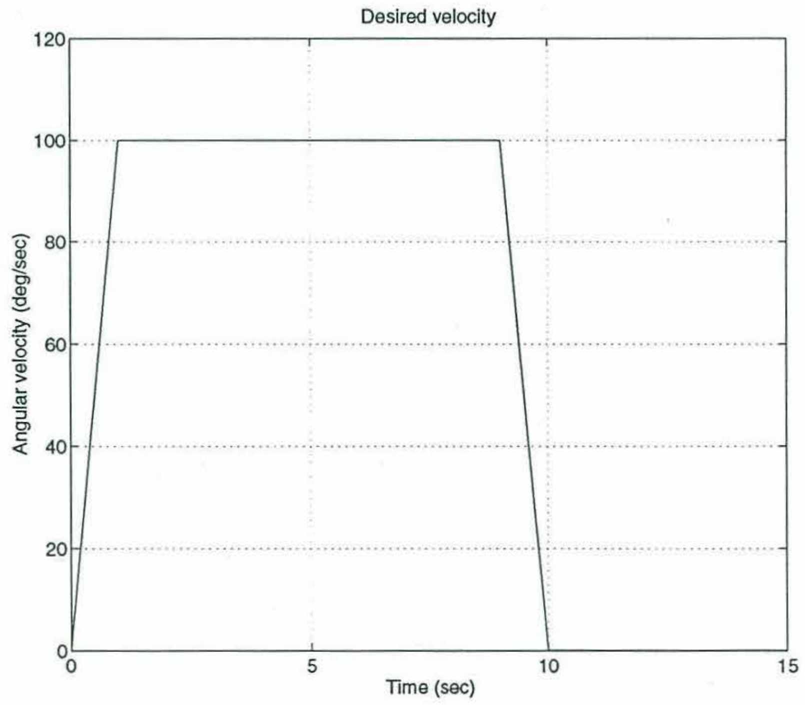


Figure 3-3: Desired trajectory and speed profile of the motion

with

$$\tau = 1.0/(\lambda \tau_f) \tag{3.11}$$

where τ_f is the time constant factor.

In the simulation, the constant values are chosen as $\tau_f = 0.1$ and $\zeta = 0.707$. The duration of the motion is 15 seconds with a time step of .0005 seconds.

Thus the control law for the PID controller is

$$u = K_i \int \tilde{\theta} dt + K_p \tilde{\theta} + K_d \dot{\tilde{\theta}} \tag{3.12}$$

The simulation results of the PID controller are shown from Figure 3-4 to Figure 3-5.

3.2.2 Sliding Control

To use the sliding method described in Chapter 2, the dynamic function (3.6) can be written as

$$\ddot{\theta} = f + bu \tag{3.13}$$

where

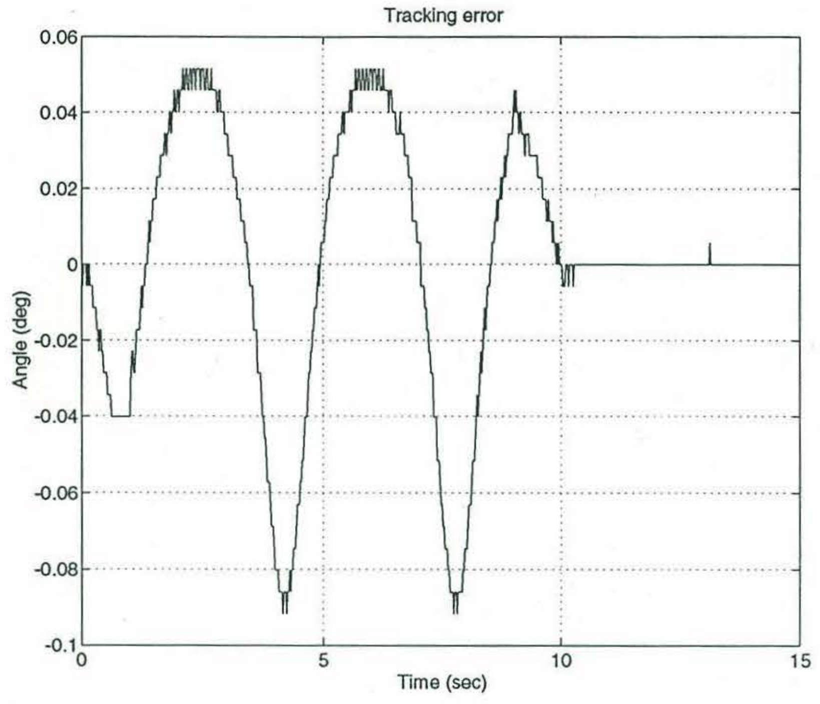
$$b = (J + \frac{M(RL\sin\theta)^2}{y^2})^{-1} \tag{3.14}$$

$$f = -b(\frac{M(RL)^2\sin\theta}{y^2}[\cos\theta - \frac{RL\sin^2\theta}{y^2}]\dot{\theta}^2 + \frac{MgRL\sin\theta}{y}) \tag{3.15}$$

Assuming the exact values of J, M, R and L are not known, thus the exact values of the control gain b and the nonlinear function f are unknown, but have a priori bounds as:

$$325 < b < 500 \tag{3.16}$$

$$-72 < f < 72 \tag{3.17}$$



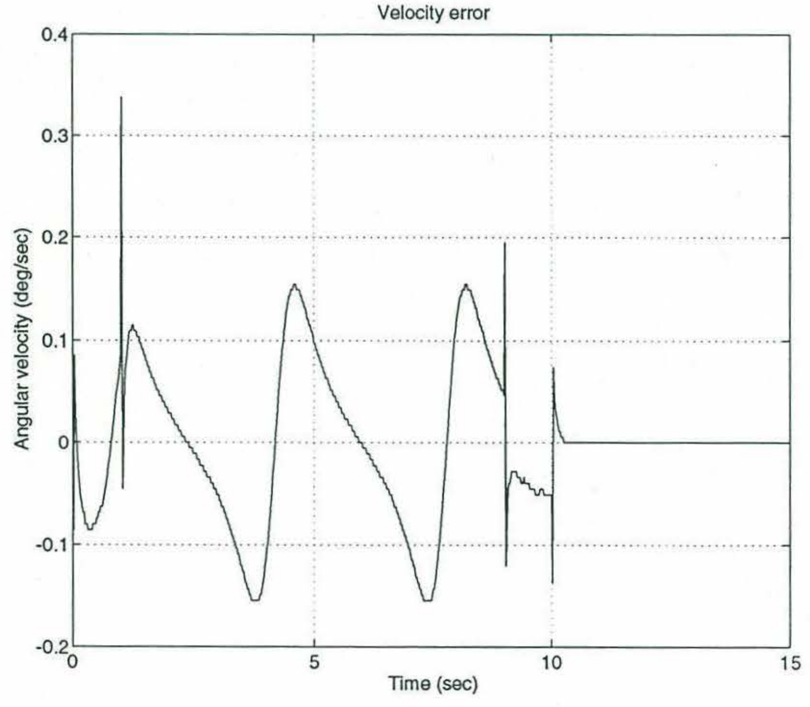


Figure 3-4: Tracking error and velocity error for PID control

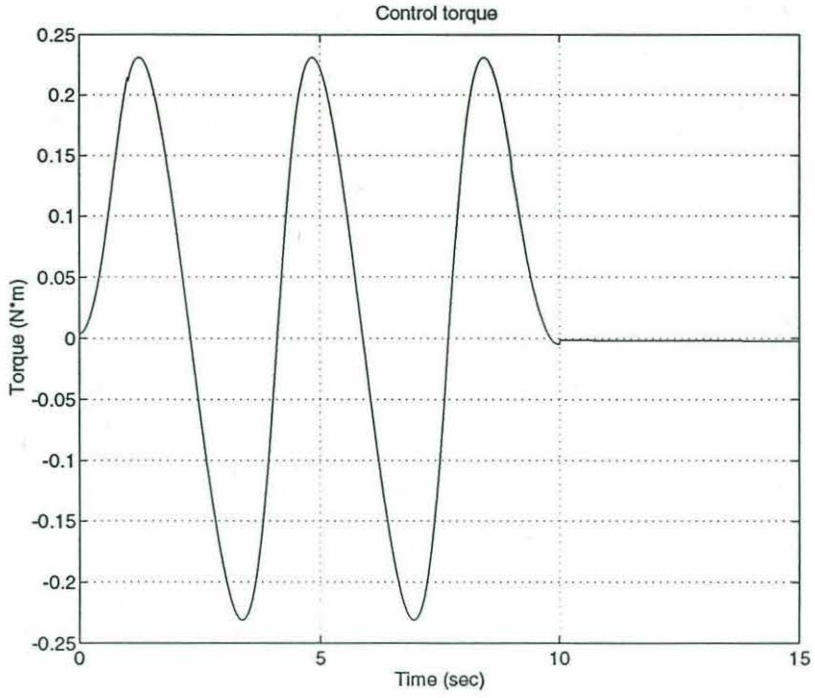


Figure 3-5: Control torque for PID control

The estimated values of b and f are

$$\begin{aligned} \hat{b} &= \sqrt{b_{max} b_{min}} = 400 && (3.18) \\ \hat{f} &= -72 \sin\theta && (3.19) \end{aligned}$$

and, accordingly,

$$\begin{aligned} \beta &= \sqrt{b_{max}/b_{min}} = 1.25 && (3.20) \\ |f - \hat{f}| &\leq F = 25 && (3.21) \end{aligned}$$

Defining s as $s = \dot{\tilde{\theta}} + \lambda\tilde{\theta}$, computing $\dot{s}$ explicitly, and proceeding as described in Chapter 2, a control law satisfying the sliding condition can be derived as

$$u = \hat{b}^{-1}[72 \sin\theta + \ddot{\theta}_d - \lambda\dot{\tilde{\theta}} - \bar{k}\mathrm{sat}(s/\Phi)] \tag{3.22}$$

The constant values are chosen as $\eta = 50$ and $\lambda = 120$. The total time duration is 15 seconds with a time step of .001 seconds.

The simulation results of the sliding controller are shown from Figure 3-6 to Figure 3-8.

3.2.3 Adaptive Sliding Control

The case for the adaptive sliding controller is similar to the sliding control simulation, except that the value of b is updated in each step of the control according to the following adaptation law:

$$\dot{\hat{b}}^{-1} = b_n^2[u^* + \eta \text{sat}(s/\Phi)]s_\Delta \tag{3.23}$$

with

$$u^* = -\ddot{\theta} + \lambda\dot{\tilde{\theta}} \tag{3.24}$$

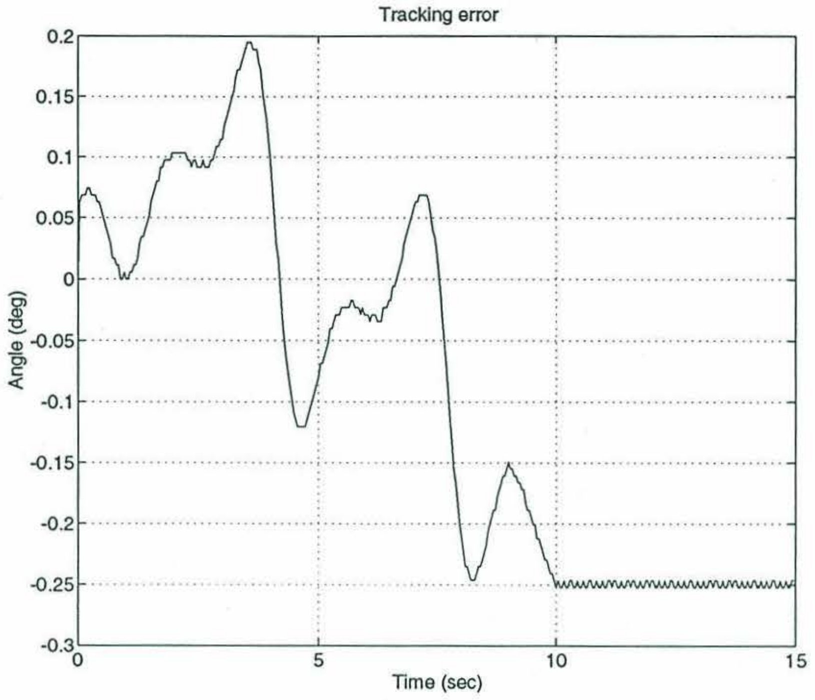
and a constant $b_n^2 = .00005$ to control the adaptation rate.

The simulation results of the adaptive sliding controller are shown from Figure 3-9 to Figure 3-12.

3.2.4 Feedback Error Learning Control

The network consists of 3 input neurons corresponding to desired displacement θ_d, velocity $\dot{\theta}_d$ and acceleration $\ddot{\theta}_d$, 15 neurons in the middle layer and 1 output neuron which is the feedforward control torque. Each neuron has a bias term in its weights. The starting initial weights are randomly selected between 0 and 1.

The inverse-dynamics model is acquired by repetitively experiencing the single movement pattern, while receiving the desired trajectory and monitoring the feedback control value as the error signal. The desired movement trajectory is the same as before, shown in Figure 3-3. The movement has a duration time of 15 seconds and the sampling time is 0.006 second. So the weights are adjusted 2500 times for each iteration. The learning rate and momentum term are adjustable at the beginning of



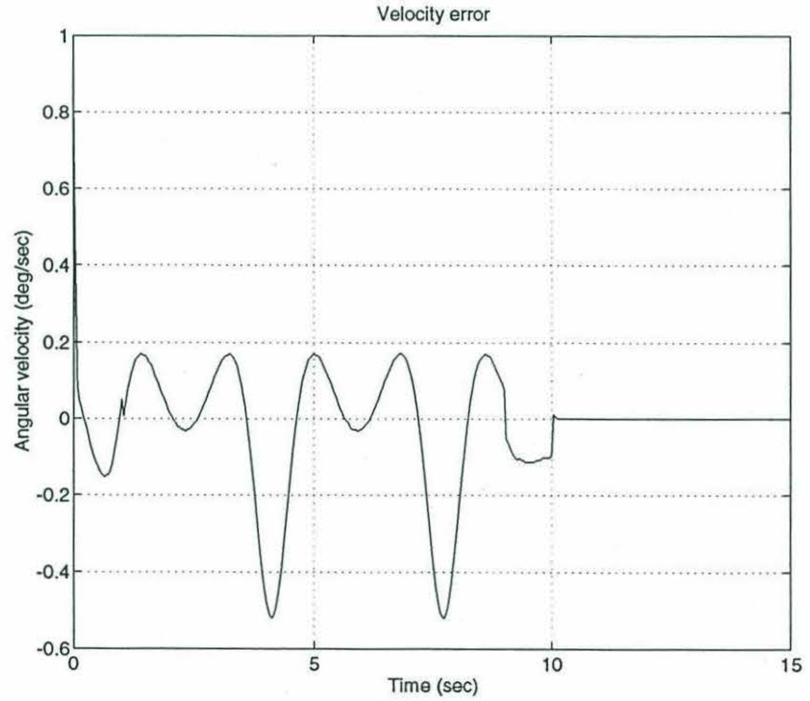


Figure 3-6: Tracking error and velocity error for sliding control

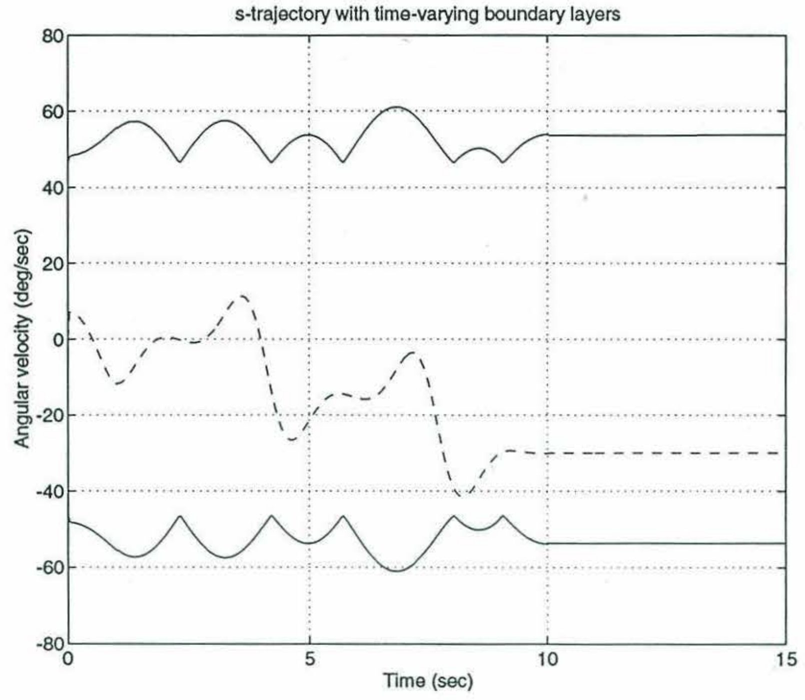


Figure 3-7: Boundary layers (solid lines) and s trajectory (dashed line) for sliding control

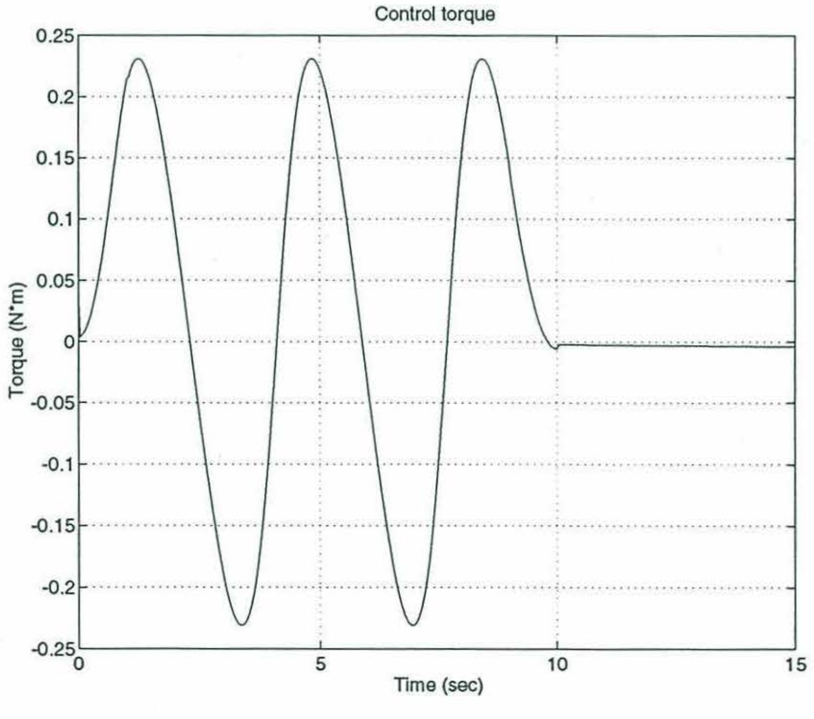
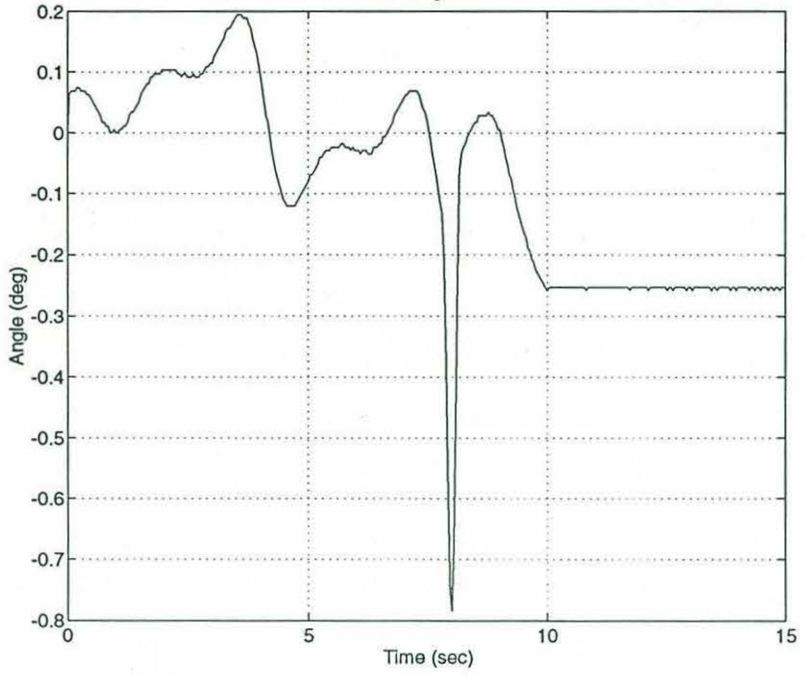


Figure 3-8: Control torque for sliding control



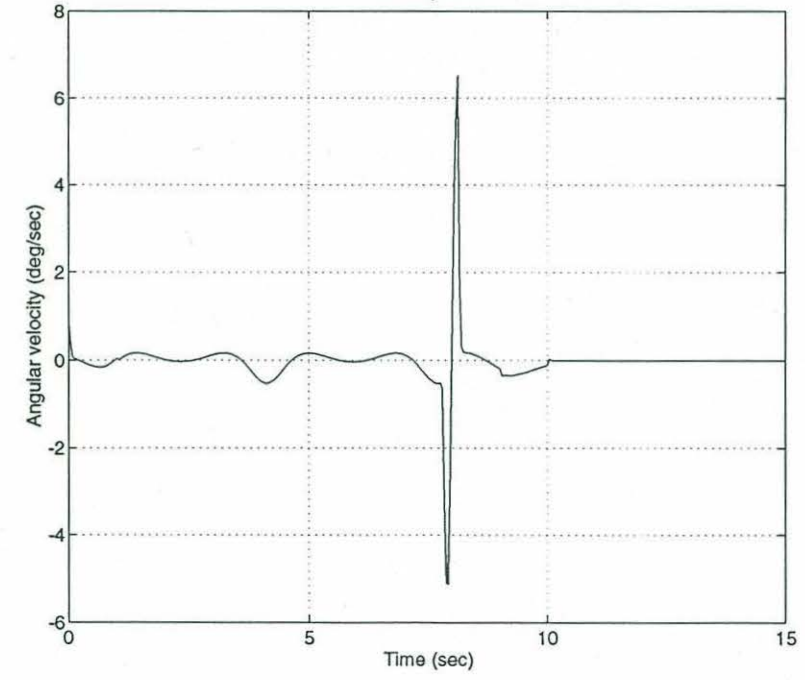


Figure 3-9: Tracking error and velocity error for adaptive sliding control

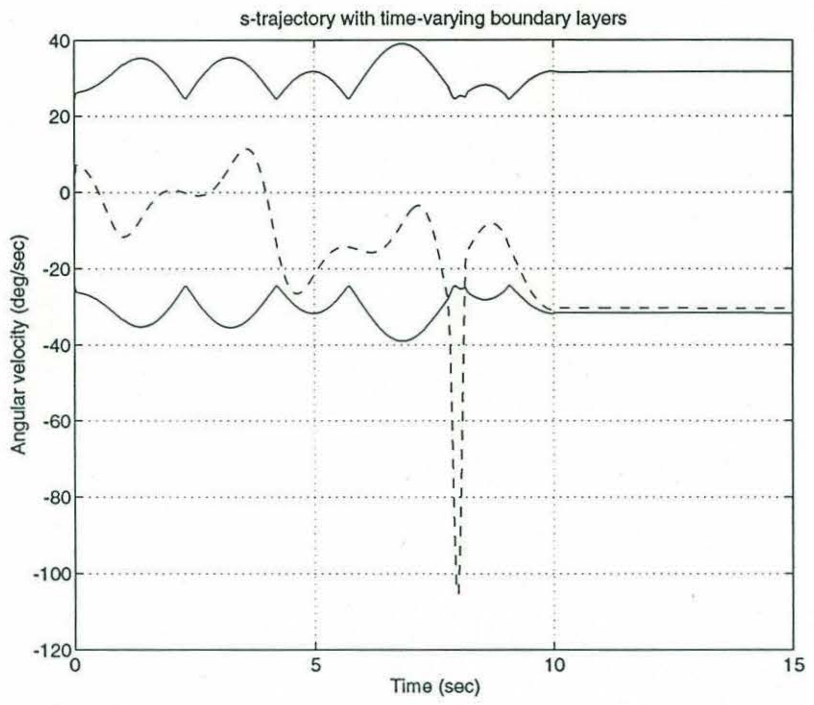


Figure 3-10: Boundary layers (solid lines) and s trajectory (dashed line) for adaptive sliding control

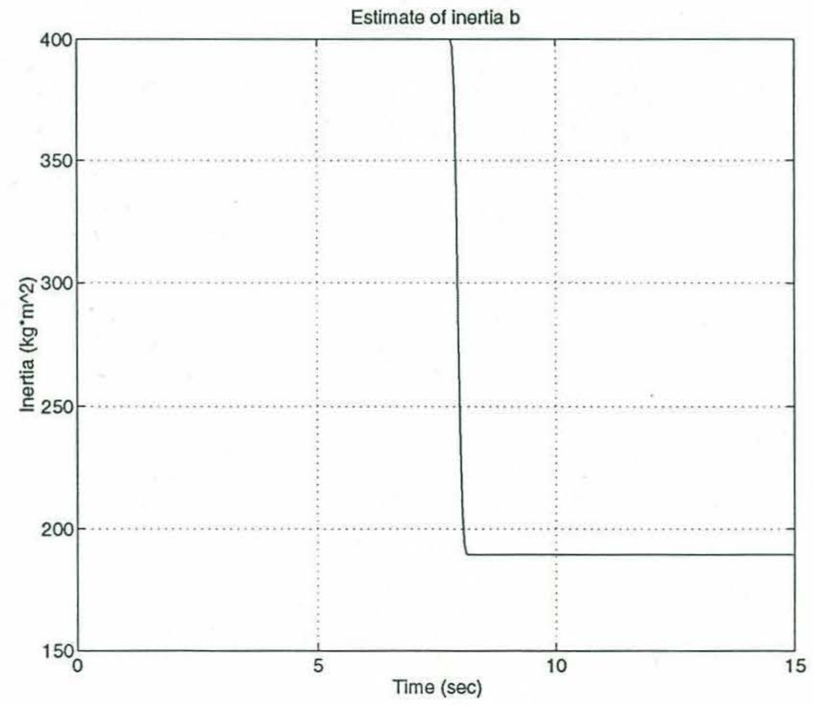


Figure 3-11: Inertia estimation for adaptive sliding control

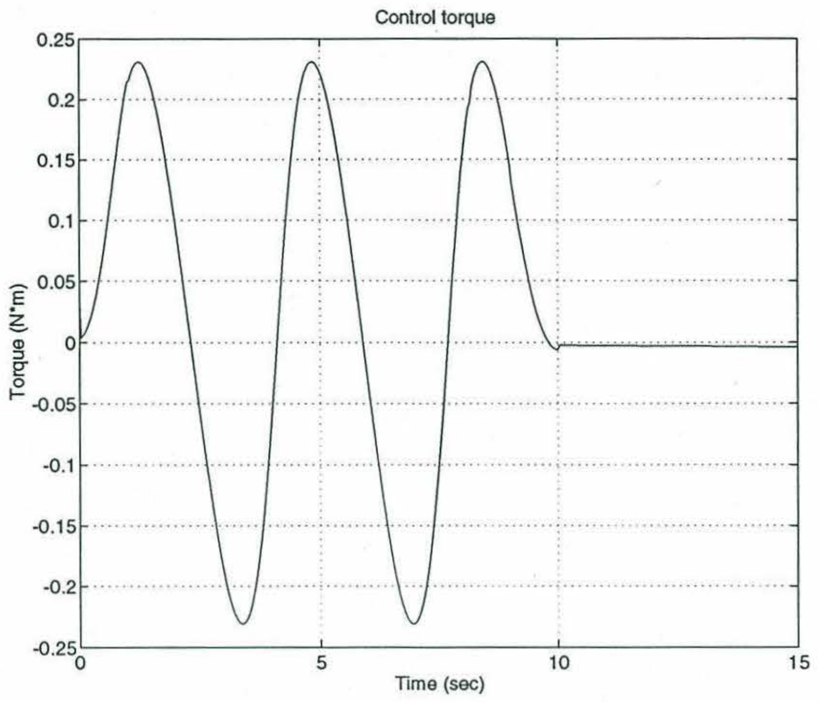


Figure 3-12: Control torque for adaptive sliding control

each iteration according to the performance of the last iteration.

In the simulation, the learning rate and momentum term are both chosen to be 0.01 for the first 4 iterations. Then they are decreased to .008 for the rest 6 iterations. The network with 15 units in the hidden layer converges. For a smaller number of hidden units, the convergence error is bigger, while for larger number of hidden units, the model behaves unstable and is difficult to converge.

Figure 3-13 to Figure 3-19 are the results from the feedback error learning control. Figure 3-13 and Figure 3-14 show the trajectory and velocity following errors after learning. As shown in Figure 3-15 and Figure 3-16, the performance improves a lot compared to the PID controller. The control force information is shown in Figure 3-17 to Figure 3-19. The feedback control force dominates when the learning just begins. The neural network quickly learns the inverse dynamics of the vehicle during the iteration, it takes the place of the feedback as a main controller and produces a majority of the control action.

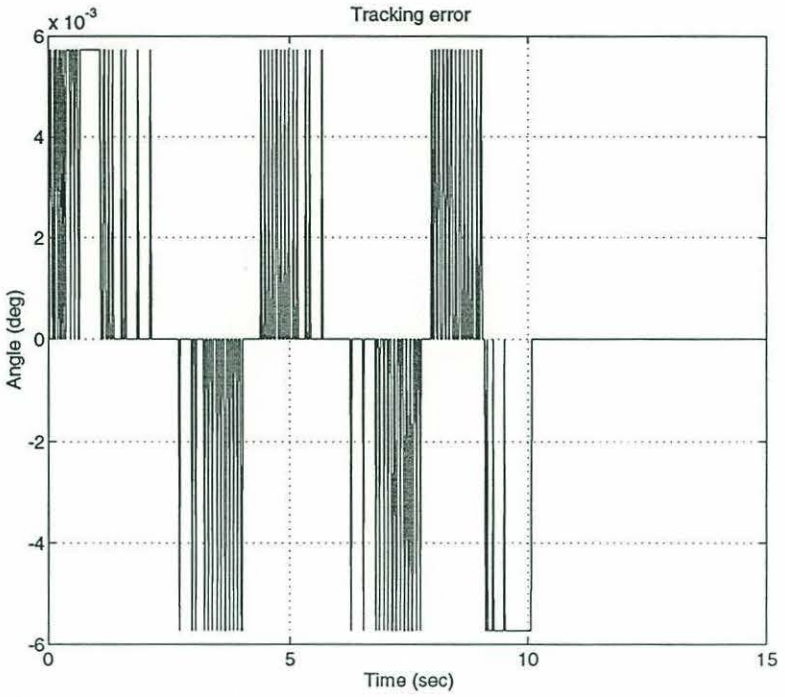


Figure 3-13: Tracking error for feedback error learning control

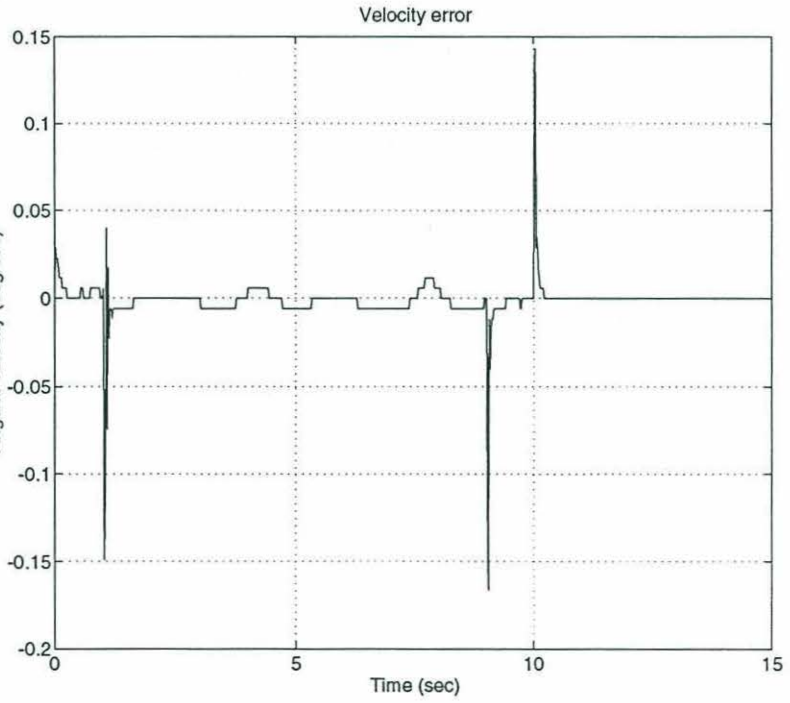


Figure 3-14: Velocity error for feedback error learning control

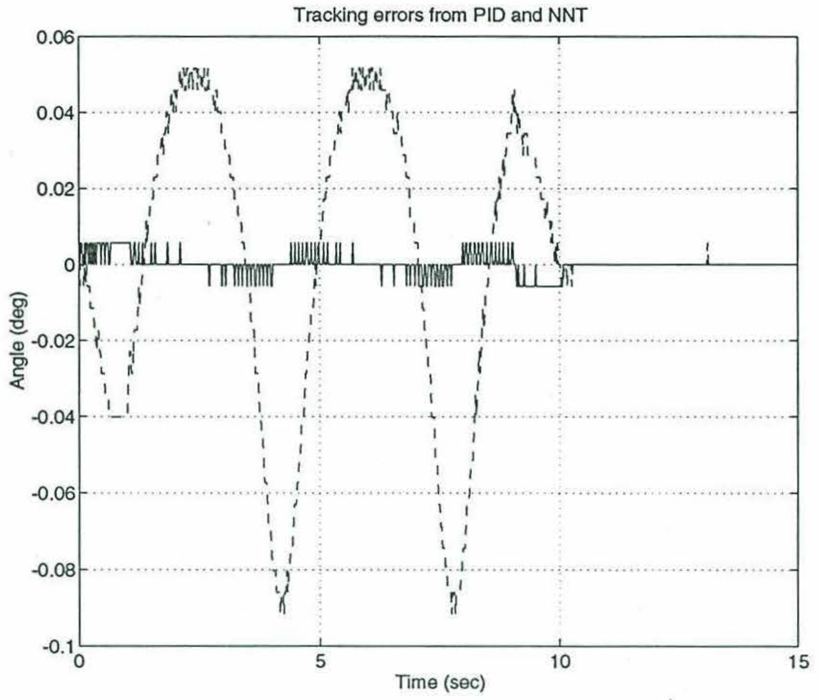


Figure 3-15: Comparison of tracking errors for PID control (dashed line) and feedback error learning control (solid line)

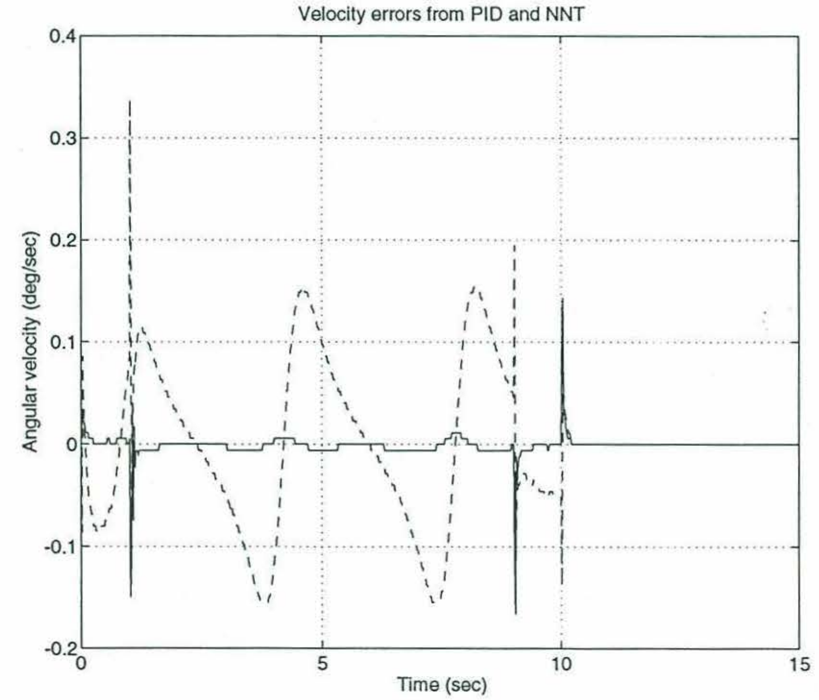


Figure 3-16: Comparison of velocity error for PID control (dashed line) and feedback error learning control (solid line)

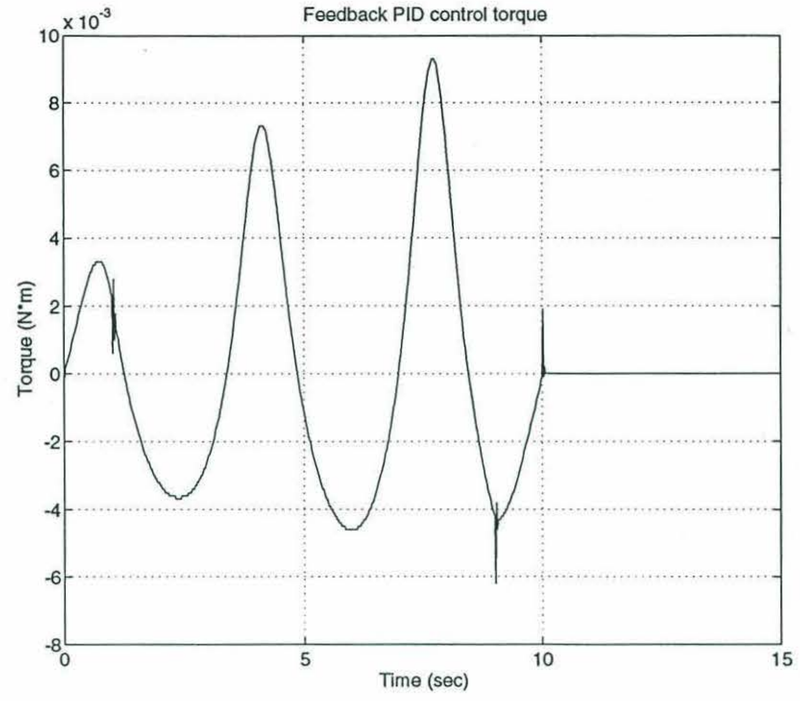


Figure 3-17: Feedback PID control torque for feedback error learning control

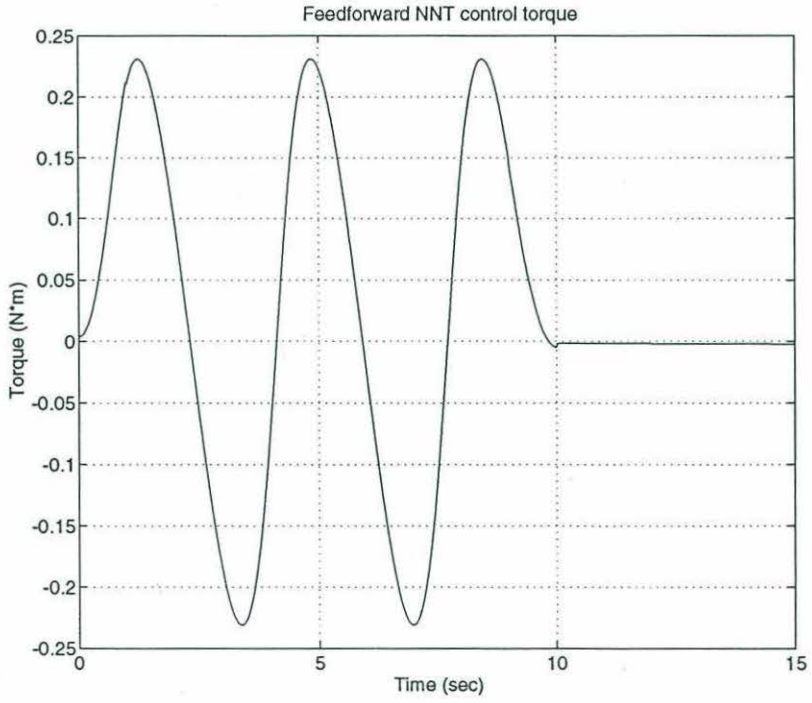


Figure 3-18: Feedforward network control torque for feedback error learning control

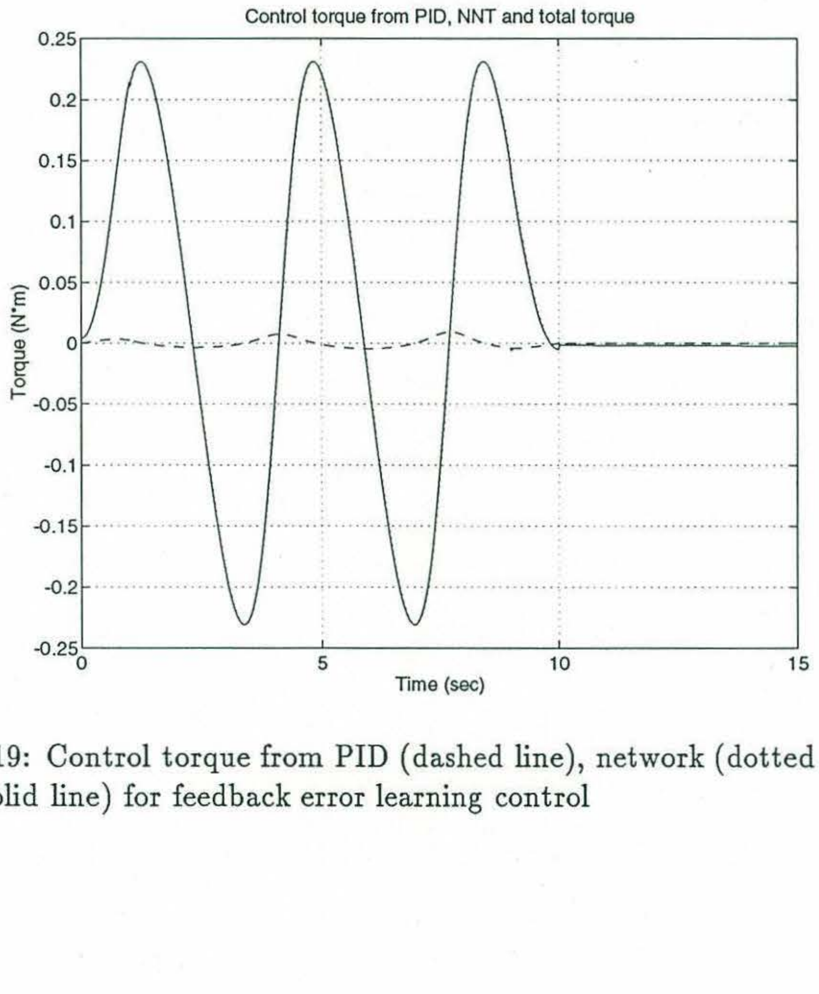


Figure 3-19: Control torque from PID (dashed line), network (dotted line) and total torque (solid line) for feedback error learning control

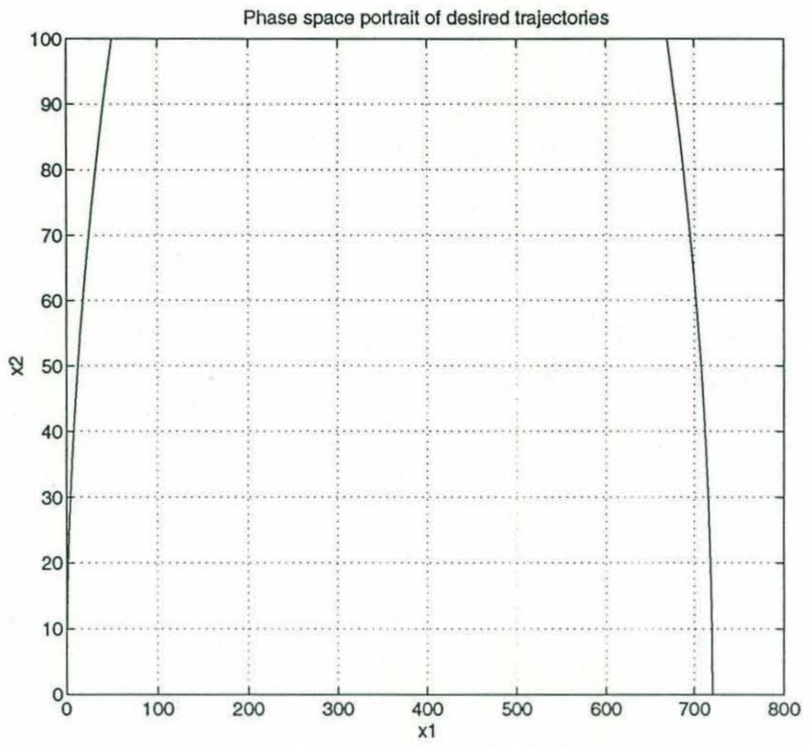


Figure 3-20: Phase space portrait of the desired trajectory

3.2.5 Gaussian Network Control

Figure 3-20 shows the phase space plot of the desired position and velocity, which is the same as used in the other controllers for the fair comparison purpose. The set $\mathbf{A}_d$ is thus chosen to be:

$$||\mathbf{x}||_{\mathbf{w}} = \max(\frac{|x|}{360}, \frac{|\dot{x}|}{50}) \tag{3.25}$$

with its center at $\mathbf{x}_0 = [360,\ 50]^T$. This represents a rectangular subset of the state space, $\mathbf{A}_d = [0,\ 720] \times [0,\ 100]$. The transition region between adaptive and sliding control modes is $\Psi = 0.1$ as the value in equation (2.64). So we have

$$\mathbf{A} = \{\mathbf{x} \mid ||\mathbf{x} - \mathbf{x_0}||_{\mathbf{w}} \leq 1.1\} \tag{3.26}$$

The maximum desired acceleration is $|\ddot{x}_d|_{max} \leq 100$ as shown in the desired trajectory plot in Figure 3-3 , which sets $|a_r| \leq 42$ for all $\mathbf{x} \in \mathbf{A}$ and the error bound to be $\epsilon_r = \epsilon_h + 42\epsilon_b$.

To achieve asymptotic tracking accuracy of 0.5 degrees, taking $k_D = \lambda = 2$ requires that $\epsilon_r \leq 0.3$. Assuming the frequency contents of h and b^{-1} on $\mathbf{A}$ are $\beta_x = 2.5$ and $\beta_{\dot{x}} = 2.5$ with uniform error no worse than 0.001 and an upper bound of 120 for the transform of h and 0.03 for the transform of b^{-1}, a close following to the parameter selection procedure in [8] shows that the choices $\theta_x = \theta_{\dot{x}} = 2$ and $l_x = l_{\dot{x}} = 5$ should be sufficient to ensure the required bound on ϵ_r. This leads to a network of gaussian nodes with variances $\sigma_x^2 = \sigma_{\dot{x}}^2 = 20$ and mesh sizes $\Delta_x = \Delta_{\dot{x}} = 0.1$. So the network includes all the nodes contained in the set $\mathbf{A}_T = [-5\Delta_x, 131\Delta_x] \times [-5\Delta_{\dot{x}}, 23\Delta_{\dot{x}}]$, for a total of 3973 nodes.

The control laws are given by equation (2.63) and (2.66) in Chapter 2, using a value of $\Phi = 0.02$ as the sliding controller boundary layer, with adaptation rates $k_{a1} = k_{a2} = 200$. Assuming uniform upper bounds of $M_0(\mathbf{x}) = 2.5$, $M_1(\mathbf{x}) = .003$ and $M_2(\mathbf{x}) = 0.005$, the sliding controller gains are taken as $k_{sl} = 2.5 + 0.003|a_r|$.

Using Fourier transformation, it's easy to find that the above spectra assumptions are justified for both h and b^{-1}. The simulation results are shown from Figure 3-21 to Figure 3-25.

3.3 Comparison of Simulation Results

Comparison are made about a priori information requirements, closed-loop bandwidth, speed of convergence, memory size, control effort, computation load, and tracking performance of these five controllers.

The PID controller needs little information about the nonlinearity, instead it requires a rough estimate of the inertia of the system. For the sensorimotor system, the constant inertia value $J = .002\ kg \cdot m^2$ is used. It then selects the control gains based on the closed-loop bandwidth and this inertia value. The tracking performance is shown to be good and the bandwidth can reach to 20 hz. The memory size is small and the computation is fast.

The sliding controller is provided with the estimated structure of the nonlinearities and the error bounds. The tracking error is slightly larger than PID controller due

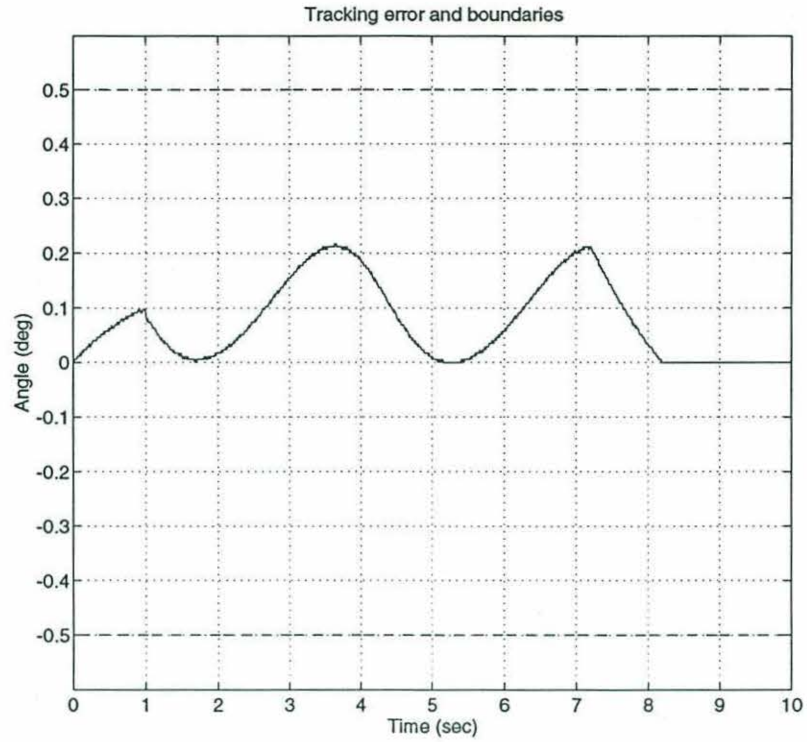


Figure 3-21: Tracking error (solid line) and boundaries (dashed lines) for Gaussian network control

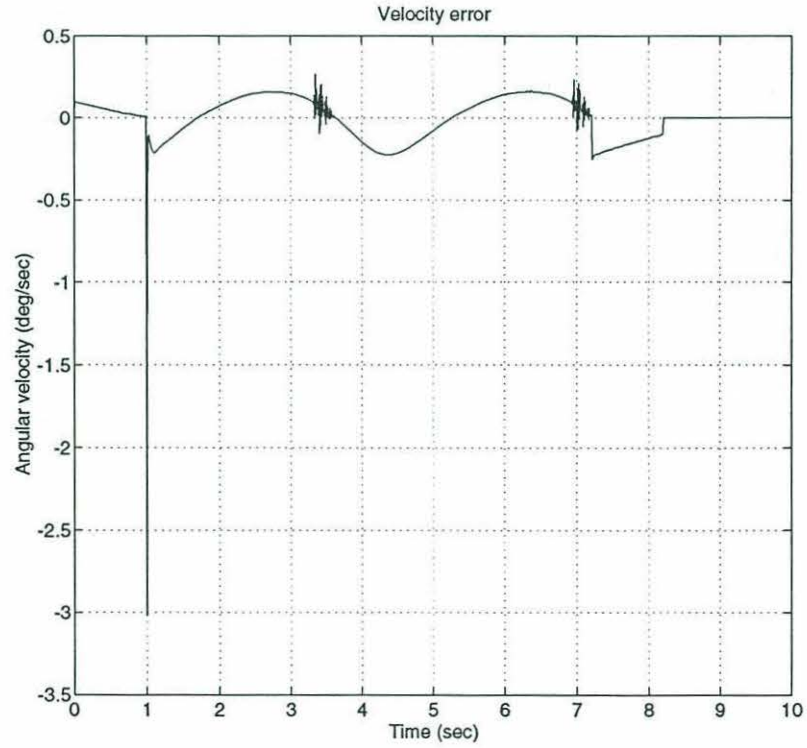


Figure 3-22: Velocity error for Gaussian network control

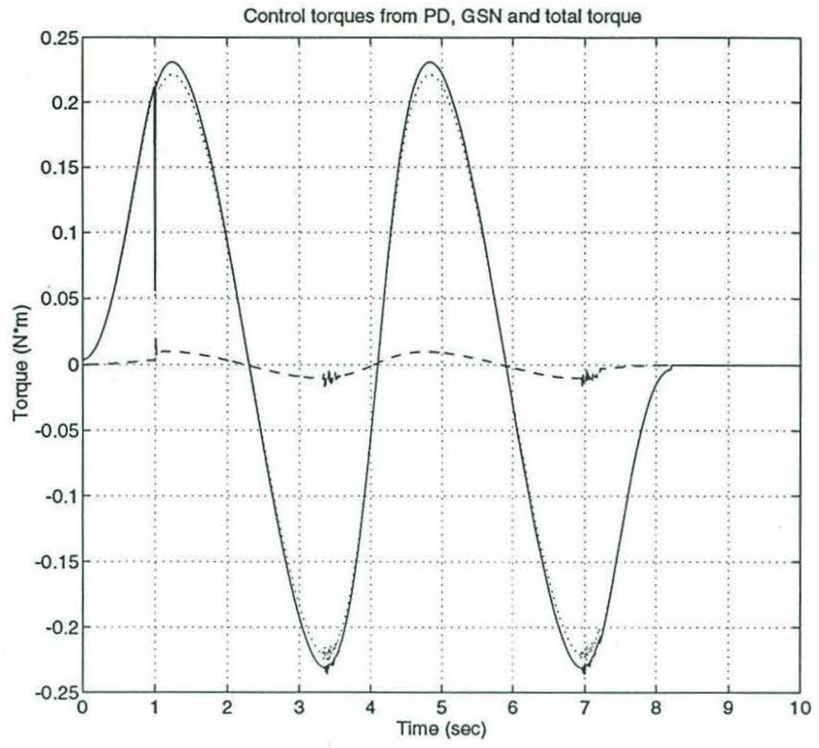


Figure 3-23: Control torque from PD (dashed line), Gaussian network (dotted line) and total torque (solid line) for Gaussian network control

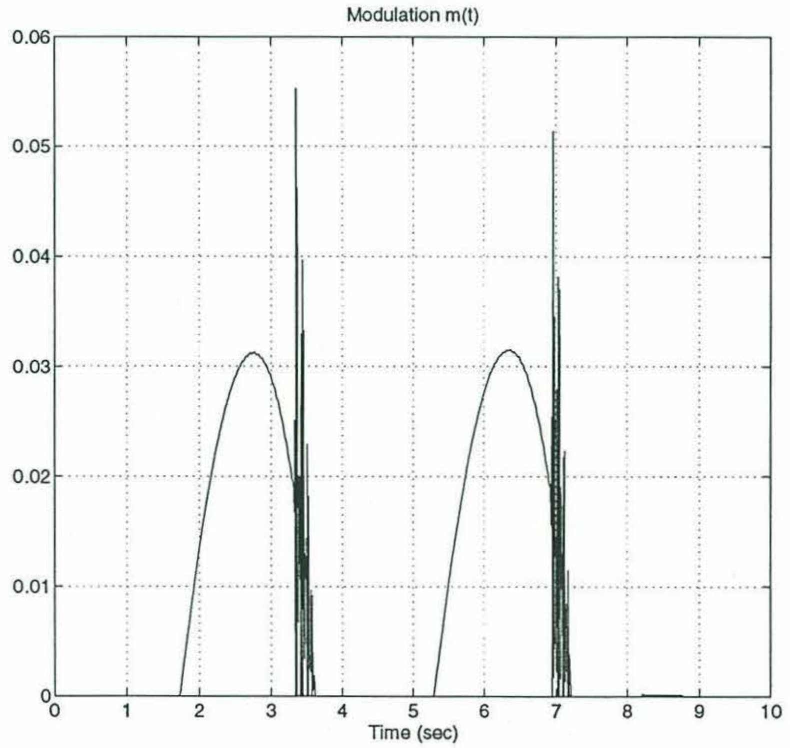


Figure 3-24: Modulation for Gaussian network control

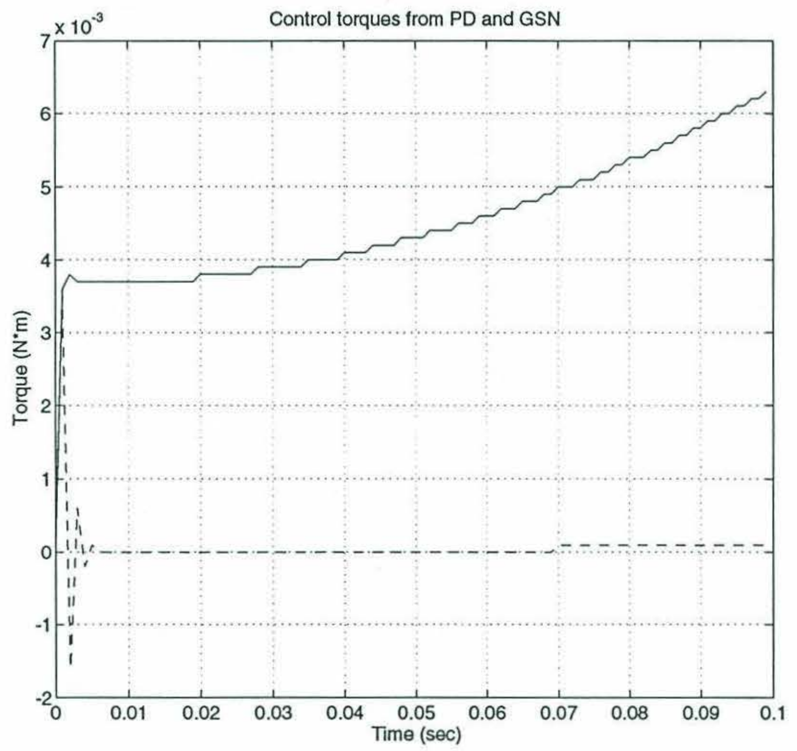


Figure 3-25: Control torque from PD (dashed line) and Gaussian network (solid line) during the early learning period of the Gaussian network control

to the large estimation error bounds. It also reaches a high closed-loop bandwidth of 20 hz and needs a slightly more computation time than PID. Unlike PID controller, the performance of sliding controller can be increased given better nonlinearity estimation. There is a quantified relationship between the performance and uncertainty trade off given by equation (2.35).

Adaptive sliding control has the capability to quickly learn and adapt to parameter uncertainties. Although it requires slightly more memory for adaptation, the tracking performance improves a lot after the parameters adapt to their real values. Its closed-loop bandwidth is also the same as the sliding controller. Both sliding control and adaptive sliding control are theoretically proven to be stable. So the overall system stability is assured.

While previous control techniques may fail to achieve the desired performance in the presence of unmodeled dynamics, the advantage of using a neural network for the nonlinear control is that it doesn't require any information about the nonlinear dynamics which is difficult to obtain. Its performance is much better than the above three controllers after learning. Yet the computations time is long, with control step of 0.006 seconds. Moreover, there is no theory to provide information about the structure of the network, i.e., the number of the layers and number of nodes within each layer. The numbers chosen are reached only from trail and error. There is either no theory developed regarding to the stability of the system.

Gaussian network control has precisely what the error backpropagation network needs, a systematic approach to choose the structure of the network while assuring the overall system stability. Its performance is good compared to the first three controllers and the system adapts much more quickly than the previous network. As shown in Figure 3-25, the adaptive control law takes most of the control action shortly after the first 0.05 seconds. The knowledge required for the controller is estimation of the frequency contents of the nonlinear function to be approximated. The structure of the Gaussian network is chosen based on this knowledge and the required performance. More work should be continued to study on reducing the size of the network and achieving fast computation while maintaining the desired performance.

Chapter 4

Experimental Comparison and Results

The best way to compare different controllers is to implement them on a real physical system and evaluate their performances. The system chosen for this thesis research is a sensorimotor system manufactured by the Seiberco Inc. Sensorimotors have an extremely simple and compact design. Unlike conventional DC brushless motors, the sensorimotor provides rotor position and velocity feedback without the use of hall effect devices, encoders, or resolvers. This feedback can easily be used to develop different controllers without introducing other sensors and devices.

According to Seiberco, the inherent position feedback information is derived from the permeance variation within the permanent magnet rotor and wound stator air gap. This position information is identical to that of a brushless resolver that outputs two sinusoidal signals phase shifted 90 degrees from each other.

The motors are frameless, i.e., the stator and rotor are provided without bearings or housings. They represent a compromise between direct drive, and small, high speed motors. The laminated stator has 24 windings, of which 4 are used for sensing and the others for power. The rotor is composed of 18 samarium-cobalt magnets epoxeid to a cold-rolled steel core. The magnets are skewed to reduce torque ripple. The skewing does reduce torque output by roughly 10% when compared to a motor without a skewed rotor, but nonetheless, the high energy rare earth magnets give the motor a very high torque-to-weight ratio.

4.1 Experimental Setup

The experimental setup used for the control system development includes the following elements:

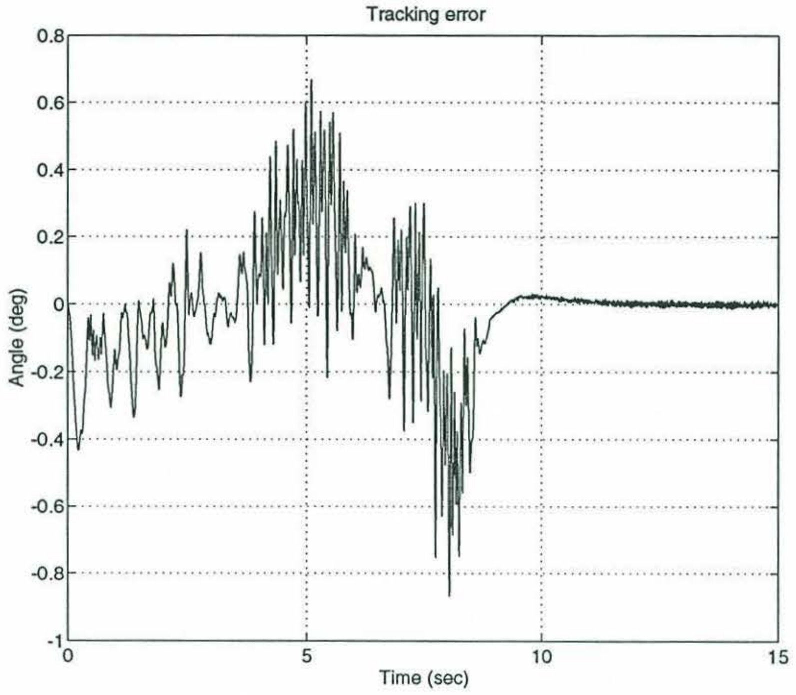
1. A transputer-based controller and data logger. The setup has analog inputs and outputs (12 bit), and general purpose digital IO ports. The system can run a full controller and state estimator at 2000 hz and log up to 100000 floating point data values.
2. The basic IO, sensor processing, state estimation, control, and data logging software. The data logger outputs directly to MATLAB for analysis and plotting.
3. A flexible mechanical setup that allows us to change motors, inertia loads, and external sensors quickly.

4.2 Experimental Results and Comparison

Based on the simulation results from the previous chapter, three control techniques are selected to be implemented on the sensorimotor systems: PID control, adaptive sliding control and error backpropagation network control. Sliding control is not select since adaptive sliding control will represent most of its features. Gaussian network control is not implemented due to the large computation requirement.

4.2.1 Performance of PID Control

The performance of PID control is shown from Figure 4-1 to Figure 4-2. The maximum bandwidth realized is 15 hz. The control step is 0.0005 seconds.



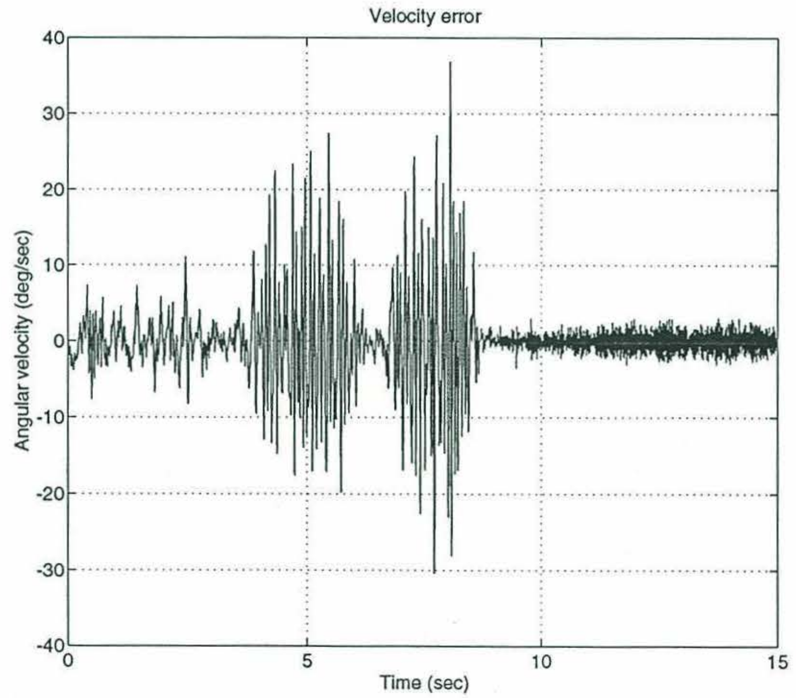


Figure 4-1: Experiment tracking error and velocity error for PID control

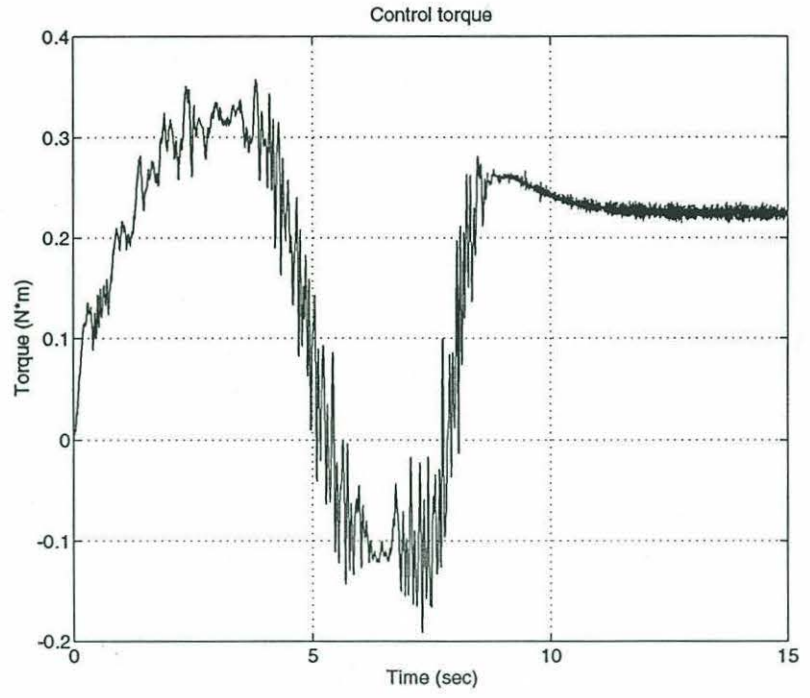


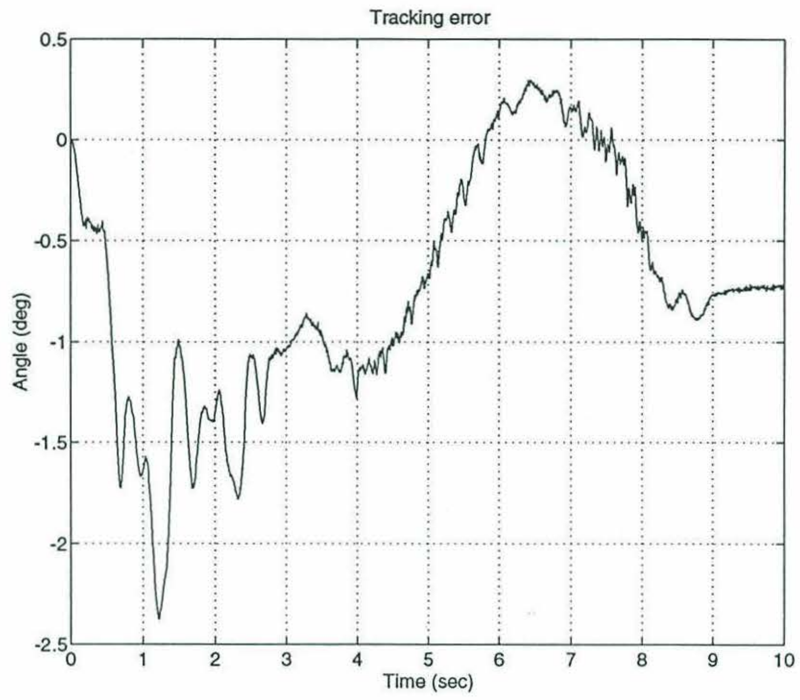
Figure 4-2: Experiment control torque for PID control

4.2.2 Performance of Adaptive Sliding Control

The performance of adaptive sliding control is shown from Figure 4-3 to Figure 4-6. The maximum bandwidth realized is 13 hz. The control step is 0.001 seconds, which represents larger memorize and more computation time than the PID controller. The tracking performance is comparable to the PID control, and it improves after the parameter adaptation.

4.2.3 Performance of Feedback Error Learning Control

The performance of the feedback error learning control is shown from Figure 4-7 to Figure 4-9. The maximum bandwidth realized is 4 hz, much smaller than the previous two controller. The control step is 0.006 seconds, represents even more larger memory size and computation time than the last two control techniques. Due to the noise in the real time sensorimotor control system, although the network adapts to most the nonlinear part of the unknown dynamics, as shown in Figure 4-9, its performance is



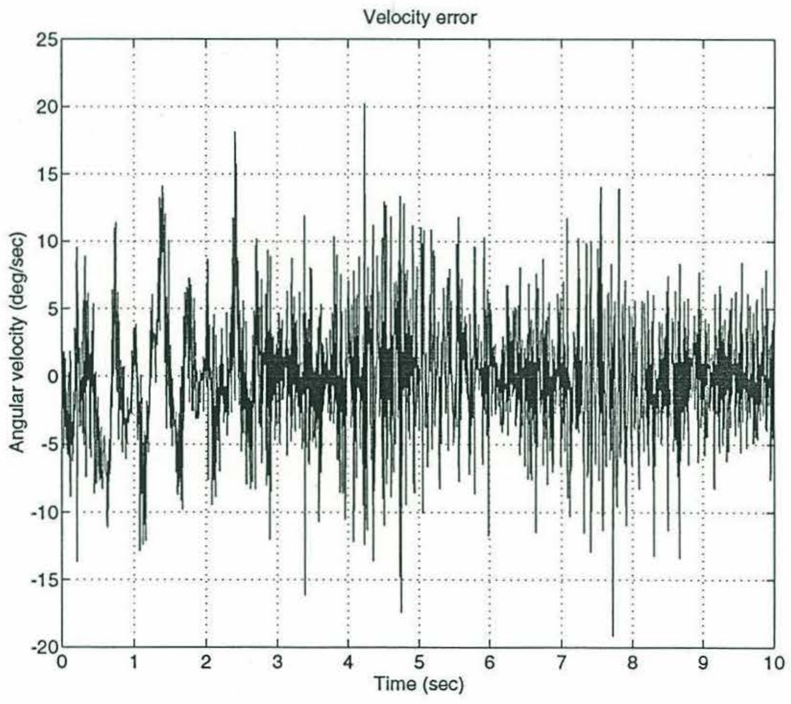


Figure 4-3: Experiment tracking error and velocity error for adaptive sliding control

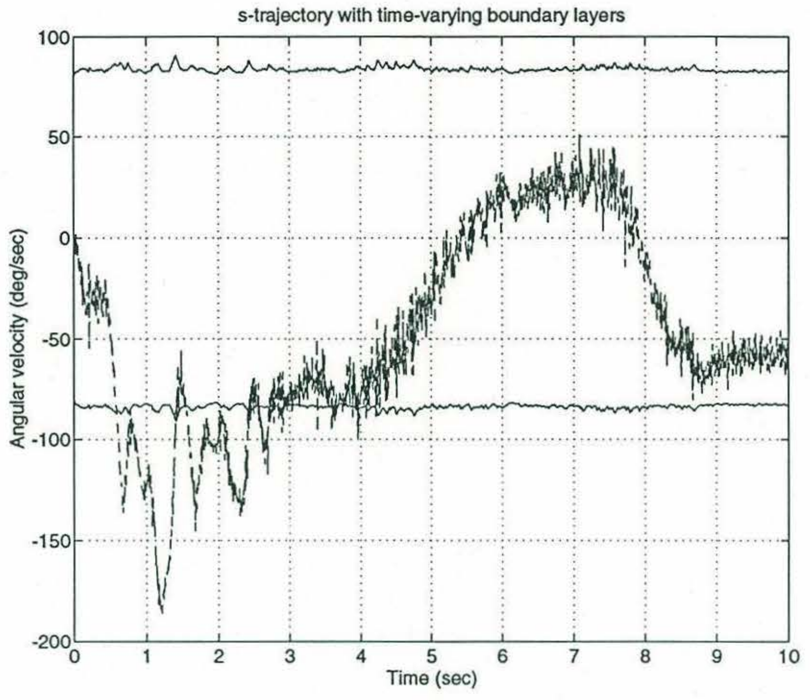


Figure 4-4: Experiment boundary layers (solid lines) and s trajectory (dashed line) for adaptive sliding control

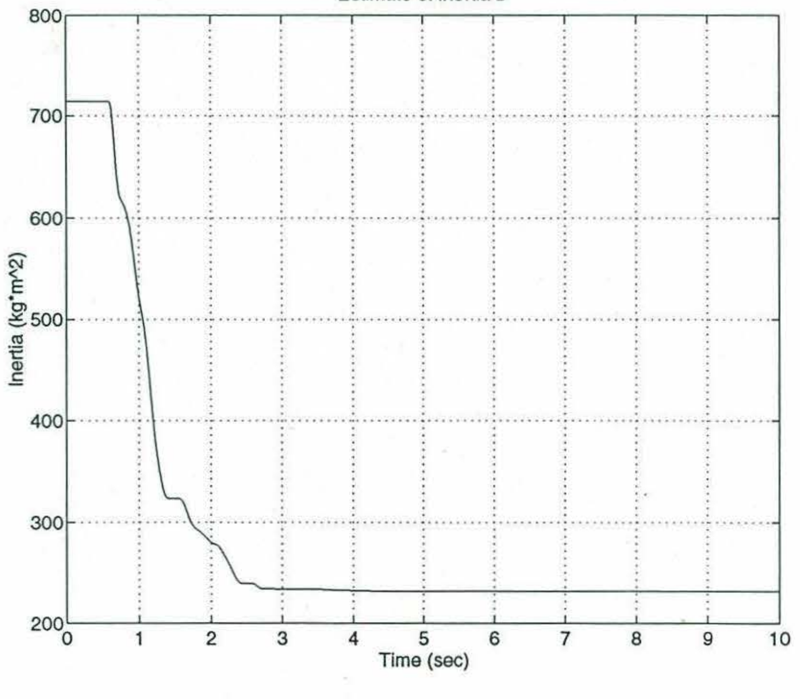


Figure 4-5: Experiment inertia estimation for adaptive sliding control

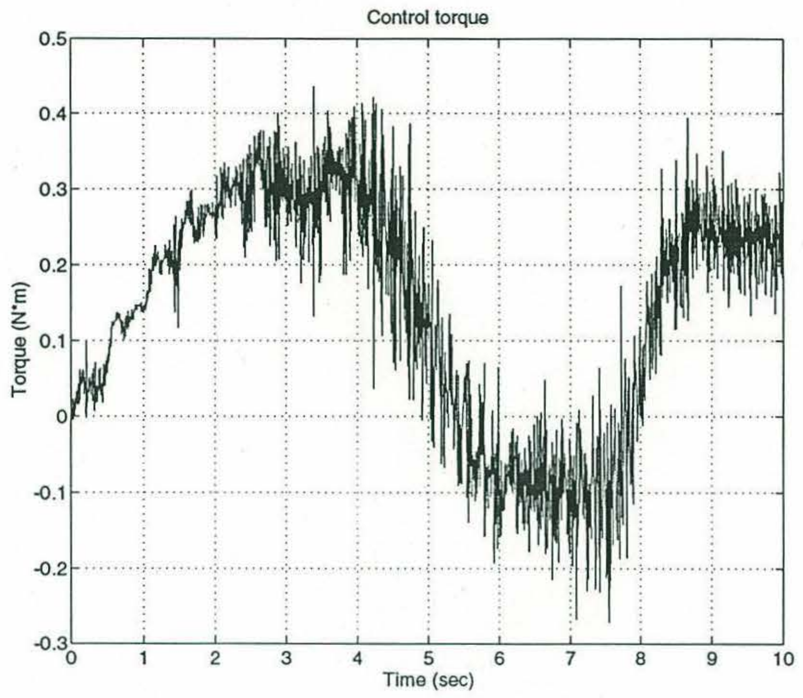


Figure 4-6: Experiment control torque for adaptive sliding control

not as good as results from the simulation and doesn't improve much with increasing learning iterations.

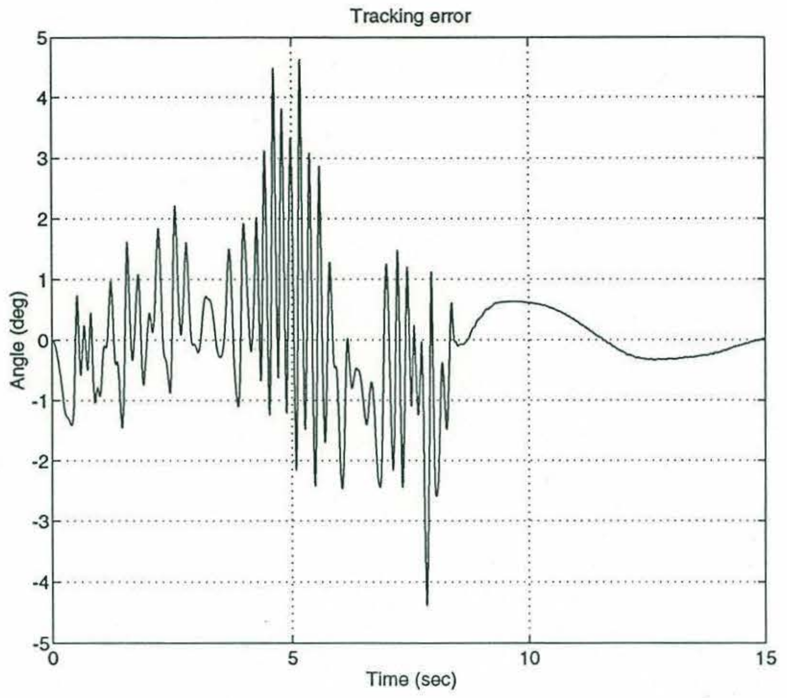


Figure 4-7: Experiment tracking error for feedback error learning control

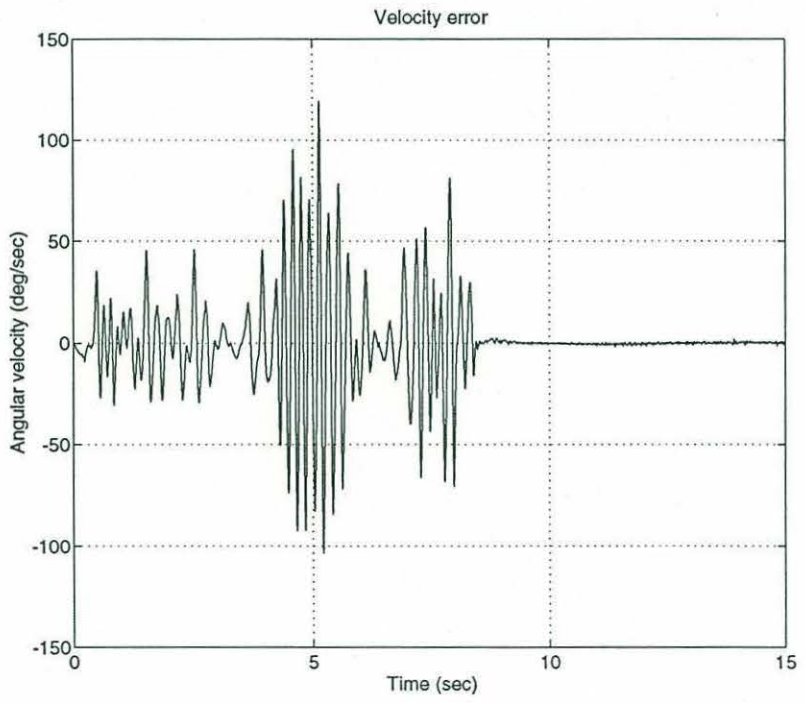


Figure 4-8: Experiment velocity error for feedback error learning control

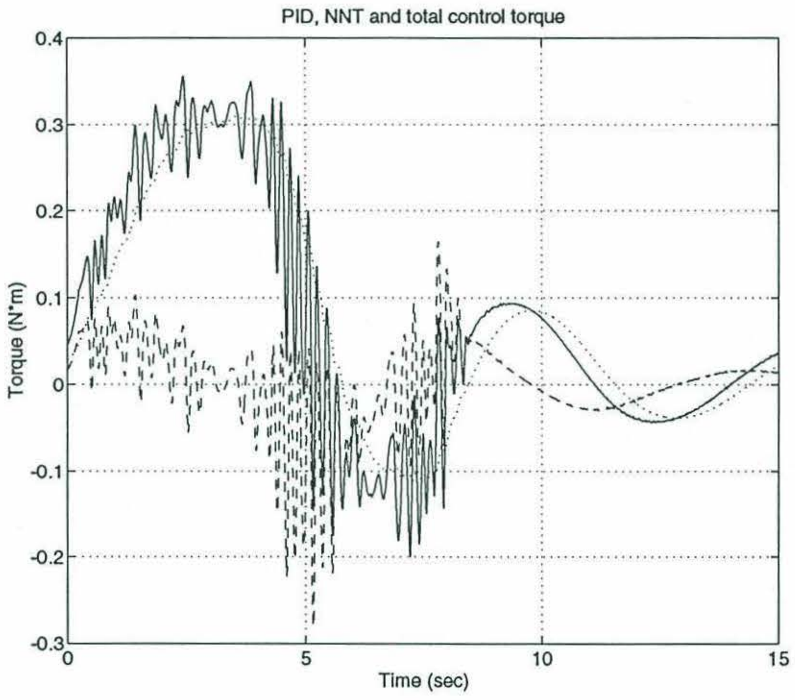


Figure 4-9: Experiment control torque from PID (dashed line), network (dotted line) and total torque (solid line) for feedback error learning control

Chapter 5

Summary and Recommendations for Future Work

5.1 Summary

The past decade has seen a dramatic increase in interest in neural networks systems. They are actively explored in psychology, signal processing, pattern classification, pattern recognition and optimization problems. There are also lots of interests growing in the control community to apply neural networks to nonlinear dynamic systems. Since control theory is a well-developed field with a large literature and solid mathematical foundation, instead of approaching neural networks as a blanket solution to control problems, we should make connections to existing control theory, illustrate their relationships to conventional and adaptive control techniques, and directly compare the new neural network approach to more traditional control system designs.

This thesis tries to study three conventional nonlinear control methodologies and two neural network approaches, compare their performances on a nonlinear dynamic system, discuss their strength and weakness, and suggest choosing appropriate control techniques for different nonlinear control applications. Due to the lack of experimental results and the very limited analysis performed in this thesis on the Gaussian network control method, the discussion of this controller is limited and serves only as a reference.

Depending on how much we know about the nonlinear dynamic system and what computation resources are available, we can make choices of the control methods that are best for our applications.

The PID control is a linear control method with gains chosen based on the information of the plant and the desired closed-loop bandwidth. The results from simulation and experiment show that it needs just a rough estimate of the inertia term, with little knowledge about the nonlinearity. Depending on the significance of the nonlinearity and the bandwidth constraints, it can be implemented on real time control with little amount of computation and memory requirements. The tracking performance is good, but can't be improved very much even if additional information about the nonlinearity is available. It's a good controller for simple control tasks on simple dynamic systems, and stability can be guaranteed for linear systems. Yet for complicated nonlinear dynamic system, performance is limited and no stability is guaranteed for closed-loop bandwidth.

Sliding control is a systematic nonlinear approach to nonlinear control problems. It requires information about the structure of the nonlinearity, but need only an estimation of the parameters involved and their estimation bounds. The tracking performance is improved and the stability of the overall system is maintained. There is a quantified trade off between the performance and the model precision. This approach can also be implemented into real time control, computation time is small. Overall, it's best for nonlinear systems with a good knowledge of the nonlinearity structure, but with just estimations of the parameters.

Adaptive sliding control is an extension to sliding control. It dynamically adapts to the unknown parameters of the nonlinear systems. If initial estimates are poor, the performance improves greatly after parameter adaptation. If the structure of the system is known, but the coefficients are hard to obtain, like the case with underwater vehicle's hydrodynamic forces, then adaptive sliding control can be a good choice.

A backpropagation network is a candidate for control problems with no prior knowledge of the structure of the system. It can be realized in on-line control with the feedback control as a guidance. The number of nodes in the network is limited.

Computation loads are much higher than the previous three controllers so the closed loop bandwidth may be low. Moreover, there is no theoretical proof about its stability and robustness. There is no established standard as how many layers and nodes are needed.

Gaussian network control serves as a link between the theoretic control and the neural network control. It requires a little more information about the nonlinearity, namely its frequency contents and some upper bounds. It adapts more quickly than the backpropagation network. Since the weight adjustment is determined using traditional Lyapunov theory, the overall system stability is assured and the tracking errors converge to a neighborhood of zero. Unlike the previous network approach, the number of nodes within the gaussian network can be decided by the desired performance and the a priori frequency content information about the nonlinear function to be approximated. It indeed needs more nodes inside the network in order to approximate the nonlinear function better, and consequently increases the computation time. Yet with the development of parallel microprocessors, this problem will be solved soon and becomes the advantage of this network controller.

The above summary is based on the simulation and experimental results shown earlier in this thesis. Figure 4-1 shows the experiment tracking performance of PID control. The tracking error gets bigger when larger nonlinearity occurs, showing that the PID doesn't compensate for the nonlinearity since it's not provided with that prior knowledge and the performance is not improving with time since there is no learning within the controller. Figure 4-3 shows that the adaptive sliding controller which is provided with a general form of the nonlinearity adapts quickly to the unknown parameters during the first 4 seconds of the experiment, and the performance improves after the adaptation. As for feedback error learning control, the experimental results shown in Figure 4-7 is not better than the previous methods, but the controller has no a priori information about the system structure. The Figure 4-9 shows the feedforward network component has the capability to learning the nonlinearity and provides most part of the control action. But choosing the structure of the network can only be realized by trial and error, and care should be taken when the system's stability is not maintained. This shortcoming may cause serious problems in real applications. The Gaussian network performance can only be seen in simulation, but it still gives us general perception of this controller. As shown in Figure 3-21, the tracking error is well within the design error bounds. The Gaussian networks quickly learns the nonlinearity during the first 0.1 seconds, as shown in Figure 3-25, and provides almost all the required control actions in Figure 3-23. The advantages of this controller are: it assures system stability and there is established standard to choose nodes and parameters for the network structure.

The computation requirements for different controllers can also be seen from the results in the thesis. The PID controller runs fast with a time step of 0.0005 seconds. The sliding controller and adaptive sliding controller have the same time step of 0.001 seconds, showing more computations time than PID. The adaptive controller further requires more memory to accommodate adaptations of the unknown parameters. The feedback error learning controller has even higher computation load, with a control step of 0.006 seconds during the experiments. The memory size is also larger to store the weights for the network. As for Gaussian network control, further work will be continued to study its computation requirement and associated performance trade-offs and realize real time control of this technique.

The current results of the comparison is that when we have the ability to understand the nonlinear dynamic system's structure, the sliding controller and adaptive sliding controller are well established for such systems with good performance and assured overall system stability. When less information is available about the system nonlinearities, neural network approaches can be applied to learn and adapt to these unknown structures.

5.2 Recommendations for Future Work

This thesis offers a preliminary comparison of some of the mostly used traditional nonlinear controllers and two examples from connectionist approaches. Results from simulation and experiment suggest when and which controllers we should use depending on our prior knowledge of the nonlinearity and the computational resources available.

The current work is greatly limited by the absence of experimental results of the Gaussian network control. There is lots of work need to be done for the completion of this comparative study. More experiments of the Gaussian network will be taken, with the main focus on reducing the number of nodes within the network and improving the computation speed, trading off parts of the dynamics that are known with those that are unknown. Parallel structure based microprocessors can be implemented to rapidly decrease the computation time and to realize the real time control of large networks.

Bibliography

[1] David M. Delonga. *A Control System Design Technique for Nonlinear Discrete Time Systems.* PhD dissertation, Massachusetts Institute of Technology, Department of Mechanical Engineering, November 1988.

[2] M. Kawato. *Neural Networks for Control*, chapter 9. The MIT Press, 1990.

[3] M. Kawato, K. Furukawa, and R. Suzuki. A hierarchical neural-network model for control and learning of voluntary movement. *Biological Cybernetics*, 57:169–185, 1987.

[4] M. Kawato, Y. Uno, M. Isobe, and R. Suzuki. Hierarchical neural network model for voluntary movement with application to robotics. *IEEE Control Systems Magazine*, pages 8–16, 1988.

[5] H. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki. Feedback-error-learning neural network for trajectory control of a robotic manipulator. *Neural Networks*, 1:251–265, 1988.

[6] Katsuhiko Ogata. *Modern Control Engineering.* Prentice-Hall Electrical Engineering Series. Prentice-Hall, Englewood Cliffs, NJ, 1970.

[7] D. E. Rumelhart, J. L. McCelland, and the PDP Research Group. *Parallel Distributed Processing*, volume 1. The MIT Press, Cambridge, MA, 1986.

[8] Robert M. Sanner and Jean-Jacques E. Slotine. Gaussian networks for direct adaptive control. *IEEE Transaction of Neural Networks*, November 1992.

[9] Jean-Jacques E. Slotine and J. A. Coetsee. Adaptive sliding controller synthesis for non-linear systems. *International Journal of Control*, 43(6):1631–1651, 1986.

[10] Jean-Jacques E. Slotine and Weiping Li. *Applied Nonlinear Control.* Prentice-Hall, Englewood Cliffs, NJ, 1991.

[11] Jon M. Smith. *Mathematical Modeling and Digital Simulation for Engineers and Scientists.* Wiley-Interscience, New York, NY, second edition, 1987.

[12] John G. Cooke V. Incorporating thruster dynamics in the control of an underwater vehicle. Master's thesis, Massachusetts Institute of Technology, Department of Ocean Engineering, September 1989.

[13] Gregory M. Vaughn. Hybrid state estimators for the control of remotely operated underwater vehicles. Master's thesis, Massachusetts Institute of Technology, September 1988.

[14] K. P. Venugopal, R. Sudhakar, and A. S. Pandya. On-line learning control of autonomous underwater vehicles using feedforward neural networks. *IEEE Journal of Oceanic Engineering*, 17(4):308–319, October 1992.

[15] David A. White and Donald A. Sofge, editors. *Handbook of Intelligent Control.* Van Nostrand Reinhold, New York, NY, 1992.

[16] Dana R. Yoerger and Jean-Jacques E. Slotine. Adaptive sliding control of an experimental underwater vehicle. In *Proc. of 1991 IEEE International Conference on Robotics and Automation*, pages 2746–2751, 1991.

[17] J. Yuh. A neural net controller for underwater robotic vehicles. *IEEE Journal of Oceanic Engineering*, 15(3):161–166, July 1990.

[18] J. Yuh and R. Lakshmi. Design of an intelligent control system for remotely operated vehicles. In *Proc. of IEEE conference on Neural Networks for ocean Engineering Symposium on the Theory of Computing*, pages 151–160, 15–17 August 1991.

[19] J. Yuh and R. Lakshmi. An intelligent control system for remotely operated vehicles. *IEEE Journal of Oceanic Engineering*, 18(1):55–62, January 1993.