

Learning Control for a Class of Discrete-Time, Nonlinear Systems

by

Ron Yitzhak Perel

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1999

© Ron Yitzhak Perel, MCMXCIX. All rights reserved.

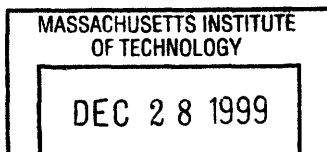
The author hereby grants to MIT permission to reproduce and to disseminate in paper and electronic forms of this thesis or any part thereof, in whole or in part.

Author
Department of Aeronautics and Astronautics
June 30, 1999

Approved by
Rami Mangoubi
Technical Supervisor, Charles Stark Draper Laboratory

Certified by
Anuradha Annaswamy
Associate Professor
Thesis Supervisor

Accepted by ...
Jaime Peraire
Chairman, Department Graduate Committee



Learning Control for a Class of Discrete-Time, Nonlinear Systems

by

Ron Yitzhak Perel

Submitted to the Department of Aeronautics and Astronautics
on June 30, 1999, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

Over the last few decades, control theory has developed to the level where reliable methods exist to achieve satisfactory performance on even the largest and most complex of dynamical systems. The application of these control methods, though, often require extensive modelling and design effort.

Recent techniques to alleviate the strain on modellers use various schemes which allow a particular system to *learn* about itself by measuring and storing a large, arbitrary collection of data in compact structures such as neural networks, and then using the data to augment a controller. Although many such techniques have demonstrated their capabilities in simulation, performance guarantees are rare. This thesis proposes an alternate *learning* technique, where a controller, based on minimal initial knowledge of system dynamics, acquires a prescribed data set on which a new controller, with guaranteed performance improvements, is based.

Thesis Supervisor: Anuradha Annaswamy

Title: Associate Professor

Technical Supervisor: Rami Mangoubi

Title: Senior Member of Technical Staff, Draper Laboratory

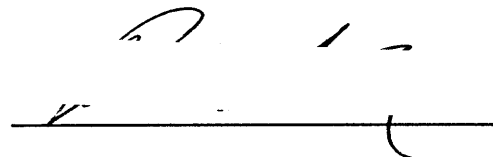
ACKNOWLEDGMENT

June 30, 1999

This thesis was prepared at The Charles Stark Draper Laboratory, Inc., under IR&D project number 18587.

Publication of this thesis does not constitute approval by Draper of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

Permission is hereby granted by the Author to the Massachusetts Institute of Technology to reproduce any or all of this thesis.

A handwritten signature, possibly "D. I.", is written above a horizontal line. The signature is in cursive and appears to be "D. I.". The horizontal line is a simple black line with a small hook at the end on the right side.

Contents

1	Introduction	13
1.1	Motivation and Problem Statement	13
1.2	Contribution	15
1.3	Outline	17
2	Discrete-Time Dynamic Systems	19
2.1	Concepts in System Dynamics	19
2.2	Recursive Input-Output Maps	21
2.3	Summary	28
3	Control Strategies	29
3.1	Control of Linear Systems	29
3.2	Feedback Linearization and Dynamic Inversion	32
3.3	Dynamic Inversion and Zero Dynamics	35
3.4	MIMO Feedback Linearization	37
3.5	Summary	38
4	Proposed Learning Strategy	39
4.1	Problem Description	39
4.2	Exact Problem Statement	40
4.3	Proof	43
4.3.1	Error Theorems	44
4.3.2	Tracking	54

4.3.3	Sampling	63
4.3.4	Approximation Theorems	81
4.3.5	Proof of Problem Statement	91
5	Implementation	95
5.1	Helicopter Dynamics	95
5.1.1	Zero Dynamics and Feedback Linearization	98
5.1.2	Simplified Model	101
5.2	Control Applications	102
5.2.1	Feedback Linearizing Controller Based on a Linear Model . . .	102
5.2.2	Adaptive Strategy	105
5.2.3	Neural Network Strategy	106
5.2.4	New Learning Strategy	108
6	Results	111
6.1	Controller Performance	111
6.2	Computational Effort	119
6.3	Design Effort	119
7	Final Comments and Further Work	121
7.1	Final Comments and Further Work	121
A	Convex Hulls	123
B	Matrix Algebra	128

List of Figures

- 5-1 Helicopter Model 96

- 6-1 Perfect Feedback Linearizing Controller 113
- 6-2 Linear Feedback Controller, Large Additive Errors 114
- 6-3 Adaptive Controller, Large Additive Errors 114
- 6-4 Neural Controller, Large Additive Errors 115
- 6-5 New Controller, Large Additive Errors 115
- 6-6 Limitation of New Controller 118

List of Tables

5.1	Longitudinal States and Controls	97
5.2	Parameter Definitions and Values	97
6.1	Relative Strengths of Various Control Strategies	111

Chapter 1

Introduction

1.1 Motivation and Problem Statement

Although great progress has been made in the last couple decades in the arena of control system design, the most predominant approach is still the traditional one of using simple, linear, feedback controllers. For most engineering applications, such controllers seem the obvious choice. The theory is well understood and has been thoroughly investigated over the years. Their application to current engineering control problems is therefore straightforward with well defined design procedures. In fact, most systems of interest are nearly linear and perform quite well with traditional controllers. The small nonlinearities present in such systems are compensated for by robustness techniques, also well defined, which offer some degree of guaranteed performance based on the magnitude of the nonlinearities.

In certain applications, though, the benefits of using well proven linear control strategies are outweighed by other factors. In fighter jets, for example, one goal is to perform extremely radical and violent maneuvers which take the jet out of the range in which its behavior is largely linear. If such maneuvers are desired, linear control strategies and associated robustness techniques may require huge gains on the control inputs, if these techniques can even be applied. Also, financial concerns are becoming an increasing factor in design. Linear controllers may be too conservative, consuming excessive fuel and energy and thereby increasing weight and cost in an effort to remain

robust. One widely investigated solution to these concerns is that of exact feedback linearization. Exact feedback linearization takes nonlinearities into account directly, and then, through a transformation of the inputs, makes the system's input-output model effectively linear ([10]). Once this step has been taken, linear techniques can be applied and achieve far better performance.

New nonlinear techniques, such as feedback linearization, and traditional linear strategies do have one significant design barrier in common. Specifically, both require extensive effort in the task of modeling system dynamics, and both can suffer greatly in terms of performance if the modeling is poor. Therefore, many new control strategies attempt to make performance robust to modeling error. Among these are parameter adaptation and neural network based controllers.

The parameter adaptation strategy assumes the model structure is known and any error exists as error in parameters, such as vehicle mass. A controller is designed in terms of unknown parameters, and then a scheme is defined to adjust the parameters' value over time, usually to achieve a performance objective such as tracking error. Extensive work has been done to apply parameter adaptation to a variety of systems, both linear and nonlinear, where the parameters may enter the system dynamics in either a linear or nonlinear fashion ([15]).

Another strategy makes fewer assumptions about system structure and assumes the system can be modeled with sufficient accuracy by a neural network ([11, 19]). In fact, it has been shown that certain classes of neural networks, collectively called Universal Approximators ([3, 7, 16]), are able to approximate continuous functions on compact domains to any desired degree of accuracy. Neural network strategies are applied when there is significant uncertainty about the system dynamics. As with parameter adaptation strategies, a scheme is devised to evolve the network over time in order to achieve some performance goal ([19]). These neural network strategies are often similar to parameter adaptation strategies where large uncertainty is treated by using a huge number of parameters. Other strategies employ neural networks in more creative ways to achieve different goals. For example, in [21] a neural network is trained in order to find a stable adaptation rule for system parameters.

The ability of neural networks to compensate for large model uncertainty raises an interesting question. Can one design a controller which is completely independent of the model to which it is applied? If the answer to this question is yes then a great achievement is possible. No longer would so much time and effort be directed toward the task of modeling. The controller design process would not begin from scratch every time a new system needs be controlled. If the effort were taken to design such a model free controller for a helicopter, then little or no effort would be needed to then port this controller to a radically different system such as a submarine. The cost saved could be tremendous. Achieving this goal as stated may be a pipe dream, but the advancing power of computers, neural networks and recent work applying them to control systems serve as the inspiration for this thesis.

My primary goal is to devise a control strategy that requires minimal knowledge of the plant model and that is applicable to as large a class of systems as possible. More specifically,

Devise a control technique which, given local knowledge of the input-output behavior of a class of discrete-time nonlinear dynamic systems, can *learn* a feedback linearizing controller that satisfies bounded tracking error for a closed set of output commands containing the region of initial local knowledge.

Furthermore, we will demonstrate the results of this thesis on a small autonomous helicopter.

1.2 Contribution

The approaches taken to solve the problem differ from the approach taken in this work. Two common approaches were mentioned above. Adaptive methods are restrictive in that they assume knowledge of the functional form of the system. Such an assumption does allow a treatment of uncertainty in the model, but still requires significant modeling effort on the part of the designer. Neural network approaches

do not make this assumption. They use neural networks as function approximators to try and directly model unknown dynamics for use by the controller. Since neural networks are known to be universal approximators, any uncertainty can be approximated by a sufficiently large neural network, though the number of parameters in the network may be large.

In each of these approaches, the essential goal of achieving some tracking or other performance objective is usually reduced to finding a stable algorithm to evolve the controller or network parameters. Doing so can be difficult for many classes of systems. Complete stability proofs are rare, and many papers often avoid the question of stable on-line training by suggesting that the necessary parameters be updated off-line, without describing how to collect the data to perform such updates. The approach suggested in this paper differs from the on-line versus off-line learning approaches in a distinct, if subtle, way. The algorithm separates the learning objective from any performance or tracking objective by providing a separate training mission. This obviously differs from on-line procedures which update parameters as a function of stability or tracking errors. It also differs from off-line methods by describing exactly how data can be stably collected. In addition, a rigorous proof is provided and gives conditions under which the training and performance objectives can be guaranteed!

The proposed algorithm may be unique in one additional and significant way. The structure used to store the *learned* controller is neither a parametric model nor a neural network. It is simply a table of acquired data points with a method of interpolating between and extrapolating from them. Using such a structure may seem inefficient, since neural networks are essentially used as compact tools to store large tables of data, eliminating redundancy in the information contained in the data points. In fact, neural networks could be used in lieu of the structure, essentially a spline, which is suggested. But such neural networks, as argued below, will always require at least the same order of size and complexity of the spline, since only a well defined set of data points is stored in the proposed structure. We are not trying to make the most of a huge, arbitrary collection of data. Required datum are specified

and sought after.

Nothing has been achieved for free. The results guaranteed by the proposed algorithm do rest on the basic assumption that a good controller already exists, at least in some small region about some operating point. *Good* and *small* are well defined and may not be very restrictive. As a matter of fact, if the Jacobian of the controller is known at any one point, these conditions can be met. The basic argument is that the boundaries of the small region can be repeatedly tested and extended to nearly any larger region of interest. Since performance measures such as tracking error can be stated explicitly in terms of controller errors, learning objectives can be well defined and all the pieces of a complete proof exist.

1.3 Outline

The Thesis is organized as follows.

Chapter 1 introduced the main goal of the thesis by describing prior work on learning based control strategies and factors motivating the associated research.

Chapter 2 provides an overview of basic concepts in dynamics and control and relates them to the classes of discrete-time nonlinear systems to which the main results of the paper apply.

Chapter 3 then describes a variety of traditional and more modern controllers which compete with the algorithm described in detail in Chapter 4.

Chapter 4 provides a general description of the capabilities of the learning strategy of this thesis, followed by an exact mathematical formulation and rigorous proof.

Chapter 5 describes the small autonomous helicopter over which the algorithm is successfully simulated. The dynamic model is derived, and the controller of Chapter 4 is constructed.

Chapter 6 presents the simulation results for the learning controller as well as additional, comparative results for traditional and adaptive algorithms.

Chapter 7 discusses these results, draws conclusions and proposes further extensions of the basic results in Chapter 4.

Chapter 2

Discrete-Time Dynamic Systems

Before stating and proving the primary results of this thesis in exact mathematical forms, certain basic ideas and concepts must be understood. The purpose of this chapter and the next is to introduce these concepts. This chapter deals specifically with system dynamics and various useful input-state and input-output system representations. Chapter 3 then builds on these ideas by introducing the linear and nonlinear control strategies and presenting various associated results.

2.1 Concepts in System Dynamics

This section introduces various common representations of discrete-time, time-invariant dynamic models, namely state space models and input-output mappings. The relationships between these representations are described, and properties of the systems relevant to the derivation of results in this thesis are presented. Let us begin by considering the well known state space representation of a general nonlinear discrete-time time-invariant system.

$$\begin{aligned}x(t+1) &= g[x(t), u(t)] \\ y(t) &= h[x(t), u(t)]\end{aligned}\tag{2.1}$$

where x , y and u are the state, output and input respectively. Time t is restricted to the non-zero integers and

$$x(t) \in \mathbf{X} \subset \mathbf{R}^n$$

$$y(t) \in \mathbf{Y} \subset \mathbf{R}^m$$

$$u(t) \in \mathbf{U} \subset \mathbf{R}^r$$

The functions $g : \mathbf{X} \times \mathbf{U} \rightarrow \mathbf{X}$ and $h : \mathbf{X} \times \mathbf{U} \rightarrow \mathbf{Y}$ are the one-step state transition function and output function. Each is assumed continuously differentiable. The k -step ahead state transition function $\Phi_k : \mathbf{X} \times \mathbf{U}^k \rightarrow \mathbf{X}$ can be found by repeated application of the one-step state transition function.

$$\begin{aligned} x(t) &= \Phi_0[x(t), u(t)] \\ g[x(t), u(t)] &= \Phi_1[x(t), u(t)] \\ g[x(t+1), u(t+1)] &= g[\Phi_1[x(t), u(t)], u(t+1)] \\ &= \Phi_2[x(t), u(t+1), u(t)] \\ &\vdots \\ g[x(t+k-1), u(t+k-1)] &= g[x(t+k-1), u(t+k-1)] \\ &= \Phi_k[x(t), u(t+k-1), u(t+k-2), \dots, u(t)] \end{aligned}$$

A state x_0 is an equilibrium state if there exists an input u_0 such that $x_0 = g[x_0, u_0]$. It is assumed that the system has at least one equilibrium state x_0 . We can assume, without loss of generality, that $x_0 = 0$, $y_0 = h[x_0, u_0] = 0$ and $u_0 = 0$ because a simple translation in \mathbf{X} , \mathbf{Y} and \mathbf{U} can move the equilibrium state, output and input to the origin of their respective vector spaces.

Before specializing this system to the class considered by the control algorithm presented in this paper, let's present the concepts of reachability and observability.

A state x is said to be reachable from a state x' if there exists an integer k and an input sequence $v \in \mathbf{U}^k$ such that $x = \Phi_k[x', v]$. The above system is reachable from x' if every $x \in \mathbf{X}$ is reachable from x' . The system is reachable if it is reachable from

all $x' \in \mathbf{X}$.

A state x is unobservable over k -steps if there exists a $v \in \mathbf{U}^k$ and a state $x' \neq x$ such that $h[\Phi_k[x, v]] = h[\Phi_k[x', v]]$. The above system is observable if there exists an integer k such that there are no unobservable states $x \in \mathbf{X}$.

2.2 Recursive Input-Output Maps

So far, other than the assumption that the system of Equation (2.1) has a finite number of states, the system is presented in a fairly general fashion. Now we add the crucial assumptions which bring us to the D-step ahead predictor form (an input-output mapping), and allow the derivation of the results of this thesis. Insuring that the system can be equivalently cast in a recursive input-output form is an essential element of the algorithm presented in this paper, since the input-output map is the maximum information about a system that one can measure.

The argument that follows is largely drawn from [12], with a few simplifications made for clarity. One simplifying assumption is that the system is single-input, single-output (SISO). At the end of the chapter, a discussion of the extension of the SISO results to the multiple-input, multiple-output (MIMO) case is made. The SISO assumption states that $m = r = 1$.

Let us state two theorems which prove useful in showing the existence of a recursive input-output map.

Theorem 2.2.1 *Let \mathbf{X} and \mathbf{F} be finite dimensional vector spaces and let \mathbf{W} be an open subset of \mathbf{X} containing the point x_0 . Let $f : \mathbf{W} \rightarrow \mathbf{F}$ be continuously differentiable. If the derivative of f at $x \in \mathbf{W}$, $Df(x)$, has constant rank for all $x \in \mathbf{W}$ then*

1. *There exists open $\mathbf{V} \subset \mathbf{W}$ and open $\mathbf{V}^* \subset \mathbf{F}$ such that $f(\mathbf{V}) \subset \mathbf{V}^*$. There also exist diffeomorphisms $d_1 : \mathbf{V} \rightarrow \mathbf{X}$ and $d_2 : \mathbf{V}^* \rightarrow \mathbf{F}$.*
2. *The restriction of $f|_{\mathbf{V}}$ is $f|_{\mathbf{V}} = d_2^{-1} \circ Df(x_0) \circ d_1$.*

A consequence of this theorem is that the function $f|V$ has connected level submanifolds. The level submanifolds are the set of points in \mathbf{W} that map to the same point in \mathbf{F} . Since all linear functions have connected level submanifolds, and f is diffeomorphically equivalent to a linear function in V , then f has connected level submanifolds. This fact becomes useful for finding conditions to satisfy the next theorem.

Theorem 2.2.2 *Let \mathbf{X} , \mathbf{Z} , \mathbf{F} and \mathbf{G} be finite dimensional vector spaces, and let functions $f : \mathbf{V} \rightarrow \mathbf{F}$ and $g : \mathbf{V} \rightarrow \mathbf{G}$ be continuously differentiable, where $\mathbf{V} \subset \mathbf{Z} \times \mathbf{X}$ is open. The functions $f(z, x)$ and $g(z, x)$ have derivatives $D_x f(z, x)$ and $D_x g(z, x)$ respectively, for $(z, x) \in \mathbf{V}$. Assume*

1. *rank $D_x f(z, x) = n$ for all $(z, x) \in \mathbf{V}$,*
2. *for any fixed z the submanifold of f consisting of all (z, x) mapping to the same point through f is connected, and*
3. *for all $x \in \mathbf{V}$*

$$\text{rank} \begin{bmatrix} D_x f(z, x) \\ D_x g(z, x) \end{bmatrix}$$

Let $f^ = (z, f(z, x)) : \mathbf{V} \rightarrow \mathbf{Z} \times \mathbf{F}$ have image \mathbf{W} . Then there exists a function $h : \mathbf{W} \rightarrow \mathbf{G}$ such that*

$$g = h \circ f$$

To simplify notation, let

$$u^t = \begin{bmatrix} u(t) \\ u(t-1) \\ \vdots \\ u(1) \end{bmatrix}$$

$$u_t^k = \begin{bmatrix} u(t+k-1) \\ u(t+k-2) \\ \vdots \\ u(t) \end{bmatrix}$$

$$y_t^k = \begin{bmatrix} y(t) \\ y(t+1) \\ \vdots \\ y(t+k-1) \end{bmatrix}$$

Therefore,

$$\begin{aligned} y(t) &= h[\Phi_0(x(t)), u(t)] \\ y(t+1) &= h[\Phi_1(u(t), x(t)), u(t+1)] \\ &\vdots \\ y(t+k-1) &= h[\Phi_{k-1}(u(t+k-2), \dots, u(t), x(t)), u(t+k-1)] \end{aligned}$$

and we can write

$$y_t^k = G_k(u_t^k, x(t))$$

where $G_k : \mathbf{U}^k \times \mathbf{X} \rightarrow \mathbf{Y}^k$.

Now assume that the initial state of the system is the equilibrium state at the origin $x(1) = 0$. We can now define the zero-state response function

$$x(t) = f_t(u^t) \triangleq \Phi_t[u^t, x(0)] \quad (2.2)$$

This assumption can be made without loss of generality since we assume that the system is reachable. Therefore, given a sufficiently large t , $x(t)$ can be acquired from $x(1) = 0$.

Define $z = u_t^k$ and $x = u^t$.

Lemma 2.2.1 *The rank of $D_x G_k(z, f(x))$ is less than or equal to n , the number of states of the system.*

Proof: Since the state space has dimension n , the co-domain of f_t has dimension n and

$$\text{rank } (D_x f_t(x)) \leq n$$

Using the chain rule

$$D_x G_k(z, f_t(x)) \circ D_x f_t(x)$$

Therefore

$$\text{rank } (D_x G_k(z, f_t(x))) \leq n$$

□

Define $F_{k,t}(z, x) = G_k(z, f_t(x))$.

Lemma 2.2.2

$$\text{rank } (D_x F_{k,t}(0, 0)) = n$$

Lemma 2.2.2 holds if the linearized system has state space of dimension n . Furthermore k and t are only required to be equal to n .

Now we have all the tools to show that a recursive input-output map exists.

Theorem 2.2.3 *There exists a function f such that*

$$y(t+n) = f(y_t^n, u(t+n), u_t^n)$$

in some open set about the origin.

Proof: First,

$$y_t^n = F_{n,t}(u_t^n, u^t)$$

By the Theorem 2.2.1 there exists an open $\mathbf{W}_t \in \mathbf{U}^n \times \mathbf{U}^t$ such that

- $\text{rank } D_x F_{n,t} = n$ for every $(z, x) \in \mathbf{W}_t$ and $t \geq n$
- $F_{n,t}|_{\mathbf{W}_t}$ has connected level submanifolds for any fixed z

The zero state response function of Equation (2.2) at time $t + n$ gives us

$$y(t + n) = f_{t+n}(u(t + n), u_t^n, u^{t-1})$$

Since Lemma 2.2.1 gives us

$$D_x F_{n+1,t} = n$$

and

$$D_x F_{n+1,t} = \begin{bmatrix} F_{n,t} \\ f_{t+n} \end{bmatrix}$$

then the derivative of $(F_{n,t}(u_t^n, u^t), f_{t+n}, u_t^p, u^t)$ with respect to $x = u_t^n$ has rank n on W_t . All the requirements of Theorem 2.2.2 are satisfied, so f_{t+n} is dependent on $F_{n,t}$. Therefore, there exists a function f such that

$$y(t + n) = f(y_t^n, u(t + n), u_t^n)$$

for all $(u_t^n, u^t) \in W_t$.

We have some freedom in defining the W_t . They can merely be chosen as the inputs restricting f to some open Y .

□

The above arguments can be followed, with various modifications and specializations, to arrive at a variety of forms. Some of these forms require the notion of delay in a discrete-time dynamic system. The following definition requires

Assumption 2.2.1

$$h[x(t), u(t)] = h[x(t)]$$

Definition 2.2.1 *The SISO system has relative degree d if*

$$\frac{\partial}{\partial u} (h(\Phi_{k+1}(x(t), 0, \dots, 0, u(t)))) = 0$$

for $k \mid 0 \leq k < d$, and

$$\frac{\partial}{\partial u} (h(\Phi_{d+1}(x(t), 0, \dots, 0, u(t)))) \neq 0$$

The interpretation of this definition is that $y(t+d+1)$ is the first output affected by the input $u(t)$. Now assume that g and h are analytic, implying that d is either ∞ or $d < n$. Clearly we must assume $d < n$, otherwise the control problem is hopeless.

Therefore, we can specialize Equation 2.2.3 under this conditions

$$y(t+n) = f(y_t^n, u_t^d)$$

The control algorithm feedback linearization, presented in the next section, attempts to define a $u(t+d)$ in terms of the remaining arguments of f . These arguments contain outputs $y(t+d+1)$ to $y(t+n-1)$. The difficulty becomes obvious, since our input can certainly not be determined as a function of future outputs. We must therefore write the future outputs as a function of current and past outputs. It is not difficult to do so. Simply let

$$\begin{aligned} y(t+d+1) &= f(y_{t+d+1-n}^n, u_{t+d+1-n}^d) \\ y(t+d+2) &= f(y_{t+d+2-n}^n, u_{t+d+2-n}^d) \\ &= f(y_{t+d+2-n}^{n-1}, u_{t+d+2-n}^d, f(y_{t+d+1-n}^n, u_{t+d+1-n}^d)) \\ &\triangleq f_{d+2}^*(y_{t+d+1-n}^n, u_{t+d+1-n}^{d+1}) \\ &\vdots \\ y(t+n) &= f_n^*(y_{t+d+1-n}^n, u_{t+d+1-n}^n) \end{aligned}$$

Since the system is autonomous, we can translate it in time without loss of generality, and write it in the D-step Ahead Predictor form.

$$y(t+d+1) = f_n^*(y(t), \dots, y(t-n+1), u(t), \dots, u(t-n+1))$$

Now, let us consider a further specialization. In particular, let us have the maxi-

mum finite delay.

$$y(t+n) = f(y_t^n, u(t))$$

or

$$\begin{aligned} y(t+n) &= f_n^*(y_{t-n+1}^n, u_{t-n+1}^n) \\ &= f_n^*(y(t), \dots, y(t-n+1), u(t), \dots, u(t-n+1)) \end{aligned}$$

This case is important because the $u(t-1)$ through $u(t-n+1)$ can be written exclusively in terms of $y(t+n-1)$ through $y(t-n+1)$. The existence of such a transformation is crucial to the proofs presented in Chapter 4. Suppose that any $y(t) \in \mathbf{Y}$ can be reached at any t . Therefore the u are restricted to \mathbf{U} , the set of $u(t)$ which take any $[y(t), \dots, y(t-n+1)] \in \mathbf{Y}^n$ to any $y(t+n) \in \mathbf{Y}$. Since we assumed that $\frac{\partial f}{\partial u} \neq 0$, the implicit function theorem guarantees that there exists a function $U : \mathbf{Y}^{n+1} \rightarrow \mathbf{U}$ satisfying

$$u(t) = U(y(t+n), \dots, y(t))$$

Therefore we have a transformation T

$$\begin{aligned} \begin{bmatrix} y(t) \\ \dots \\ y(t-n+1) \\ u(t-1) \\ \dots \\ u(t-n+1) \end{bmatrix} &= \begin{bmatrix} y(t) \\ \dots \\ y(t-n+1) \\ U(y(t+n-1), \dots, y(t-1)) \\ \dots \\ U(y(t+1), \dots, y(t-n+1)) \end{bmatrix} \\ &= T(y(t+n-1), \dots, y(t-n+1)) \end{aligned}$$

Furthermore T is a diffeomorphism since the transformation T^{-1} exists satisfying.

$$\begin{aligned} \begin{bmatrix} y(t+n-1) \\ \dots \\ y(t+1) \\ y(t) \\ \dots \\ y(t-n+1) \end{bmatrix} &= \begin{bmatrix} f(y(t+n-2), \dots, y(t-1), u(t-1)) \\ \dots \\ f(y(t), \dots, y(t-n+1), u(t-n+1)) \\ y(t) \\ \dots \\ y(t-n+1) \end{bmatrix} \\ &= T(y(t+n-1), \dots, y(t-n+1)) \end{aligned}$$

2.3 Summary

We have taken a system written in a well known and understood state space form and show that it can be recast in the input-output form known as the D-step ahead predictor form. Working with this new form has several advantages. Most importantly, it is important to understand that the input-output model represents all the knowledge we can perceive given exclusively measurements of the input and output. In addition, we will see that using the D-step ahead predictor form in particular allows us to write an exact input-output feedback linearizing controller exclusively as a function of past inputs and outputs. This result is the primary contribution of the following chapter.

Chapter 3

Control Strategies

Now the concepts of the previous chapter can be built upon to devise methods of controlling a particular system. This chapter begins by introducing some simple concepts regarding linear control systems, and then moves on to describe in more detail the input-output feedback linearizing strategy as it applies to the state-space system and the equivalent D-step ahead predictor form.

3.1 Control of Linear Systems

Much of the current control theory makes the assumption that the system under consideration is linear, or nearly linear, with respect to the state variables and inputs.

$$g[x(t), u(t), t] = A(t)x(t) + B(t)u(t)$$

$$h[x(t), u(t), t] = C(t)x(t) + D(t)u(t)$$

Very few systems are actually linear, but many may be well approximated by some linear system. Due to the depth of research focusing on linear systems, many tools exist to design $u(t)$ such that the system's states and outputs behave in some desired fashion. The most basic design goals are stability and tracking. Before continuing, let us make the assumption that the system is time-invariant, as we have done with the system of Chapter 2. This assumption is a reasonable one, since otherwise the

identification problem, the primary goal of the results of this thesis, would not be possible. Therefore,

$$A(t) = A, \quad B(t) = B, \quad C(t) = C, \quad , D(t) = D$$

Since the transient behavior of the above linear system is invariant with respect to a translation of axes in the state and input, we can assume that the stability problem is that of stabilizing the origin. The most common of approaches to solving the stability problem is to allow $u(t)$ to be a linear combination of the states of the system.

$$u(t) = k^T x(t)$$

If such a strategy were not sufficient, designers have the option of augmenting $x(t)$ with additional states whose dynamics also depend linearly on states and inputs. If we let the augmented state vector be $z(t)$ then the new system can be written.

$$z(t+1) = A_z z(t) + B_z(t)u(t)$$

$$z(t+1) = C_z z(t) + D_z(t)u(t)$$

and $u(t) = k^T z(t)$. Eliminating $u(t)$ from the above expression

$$z(t+1) = (A_z + B_z k^T)z(t)$$

$$z(t+1) = (C_z + D_z k^T)z(t)$$

It is well known that the stability of the above system can be written in terms of the eigenvalues of the state coefficient matrix

$$\text{eig}(A_z + B_z k^T) < 1$$

This condition guarantees

$$y(t+1) < y(t)$$

and

$$y(t+1) \rightarrow 0, \text{ as } t \rightarrow \infty$$

And the system is stable.

If the goal is acquire the origin in finite time, we can reintroduce the notion of reachability. In term of the linear system, the reachability problem takes on a particularly simple form. Let's progress by first constructing the k step response function of the above system

$$\begin{aligned} x(t+k) &= A^k x(t) + \begin{bmatrix} A^{k-1}B & | & A^{k-2} & | & \dots & | & B \end{bmatrix} \begin{bmatrix} u(t) \\ u(1) \\ \vdots \\ u(t+k-1) \end{bmatrix} \\ &= A^k x(t) + R_k u_k \end{aligned}$$

Therefore, if there exists a k such that R_k has rank n , then all $x(t+k)$ would be in the range of R_k . Since without loss of generality, $x(t)$ can be set to zero, any $x(t+k)$ can be reached from any $x(t)$. It is obvious that reachability of the origin implies reachability of any state.

Observability also plays an important role, and as with reachability, the condition takes a particularly simple form for the case of linear systems. We require observability since the above feedback control strategies required access to all the states. Observability is the condition under which the states can be constructed from the outputs. Begin by collecting the k step output response functions into a vector form. Suppose $u(t)$ and $y(t)$ are known. Then,

$$\begin{bmatrix} y(t) \\ y(t+1) \\ \vdots \\ y(t+k-1) \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{k-1} \end{bmatrix} x(t) + \begin{bmatrix} D & 0 & 0 & \dots & 0 \\ CB & D & 0 & \dots & 0 \\ \vdots & & & & \vdots \\ CA^{k-2}B & CA^{k-3}B & CA^{k-4}B & \dots & D \end{bmatrix}$$

Since the last term above is known, we can write the above equation in the following form.

$$y = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{k-1} \end{bmatrix} x(t) = O_k x(t)$$

If rank O_k is n , the number of states, for some k , then each $x(t)$ maps to a different y and $x(t)$ can be recovered from y . It is also clear that observability of any state implies observability of all the states.

With respect to the small autonomous helicopter of this section, the linear control design depending essentially on the concepts so far presented in this section, is based on a linear approximation of the vehicle about the hover condition (zero velocity, angular rates and attitude). Reasonable linear approximations can be constructed as first order Taylor's series expansions of the system. Since such approximations apply with arbitrarily small error in arbitrarily small open sets about the point of linearization, the resulting controller's performance can only be guaranteed locally. Robustness techniques do exist, but they are often very conservative in nature, and required extensive modeling of errors. There do exist control strategies which take the nonlinear structure of the system directly into account and provide global stability and tracking guarantees. One of these is that of feedback linearization, discussed in the following section.

3.2 Feedback Linearization and Dynamic Inversion

Since non-linear systems are so difficult to analyze, and linear systems have been thoroughly studied and are relatively simple to analyze and control, an obvious control strategy to employ on non-linear systems is the strategy of feedback linearization ([2], [9], [11], [13], [21]). In essence, a transformation of the control inputs is found which makes the transformed system linear. Then, traditional control strategies can be

employed. A specialization of this strategy is dynamic inversion, where the dynamics are completely inverted and the output at time $t + \Delta t$ is simply the transformed input at time t . Desired response characteristics are then achieved through appropriate definition of the input. These algorithms are obviously very powerful. Their strengths, though, are tempered by some significant weaknesses, all of which shall be discussed in this section.

The conditions under which feedback linearization and Dynamic inversion can be employed are now described. The derivations of these controllers shall be presented for both a state space model, and for a D-step ahead predictor model. This step is taken for the sake of completeness.

Let us begin by input output linearizing a non-linear discrete time state space SISO model.

$$\begin{aligned}x(t+1) &= g[x(t), u(t)] \\y(t) &= h[x(t)]\end{aligned}$$

Define a transformation of variables

$$\begin{aligned}z(t) &= \Phi(x(t)) \\ \Phi(x(t)) &= \begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_n(x) \end{bmatrix}\end{aligned}$$

where $\phi_i(x) = \Phi_i(x)$ the i step ahead state transition function for $i \in [1, d]$. If the delay d is finite it is always possible to find $\phi_i(x)$ for $i \in [d+1, n]$ such that $D_x \Phi(0)$ has rank n in some open set \mathbf{M} about $x = 0$. Furthermore, the $\phi_i(x)$ for $i \in [d+1, n]$ can be chosen such that $\phi_i(f(x, u)) = (\phi_i \circ f)(x)$ and $\Phi(x)$ is a diffeomorphism on \mathbf{M} . In the new coordinates, the system can be written

$$\begin{aligned}z_1(t+1) &= z_2(t) \\ z_2(t+1) &= z_3(t) \\ &\vdots\end{aligned}$$

$$\begin{aligned}
z_d(t+1) &= [h \circ f^d](\Phi^{-1}(z(k)), u(k)) \\
z_{d+1}(t+1) &= \phi_{d+1}(\Phi^{-1}(z)) \\
&\vdots \\
z_n(t+1) &= \phi_n(\Phi^{-1}(z)) \\
y(t) &= z_1(t)
\end{aligned}$$

Since $\frac{\partial \phi_d(x, u)}{\partial u} \neq 0$ by assumption, the implicit function theorem guarantees the existence of a function g such that

$$\phi_d(x, g(x, v)) = v(t)$$

in some open convex set. Therefore, we can define our input $u(t)$ as $g(x, v)$ and the input output map of the above system becomes

$$y(t+d) = v(t)$$

The function g is the input output linearizing transformation of the control input u .

The argument can be similarly applied to the recursive input output maps derived in Chapter 2. Consider the d-step ahead predictor form

$$y(t+d) = f_n^*(y(t), \dots, y(t-n+1), u(t), \dots, u(t-n+1))$$

Since $\frac{\partial f}{\partial u(t)} \neq 0$ in some open convex neighborhood of the origin, as shown in Chapter 2, then the implicit function theorem guarantees that there exists a function U such that

$$u(t) = U(y(t+d), y(t), \dots, y(t-n+1), u(t-1), \dots, u(t-n+1))$$

and the input output model has been linearized.

3.3 Dynamic Inversion and Zero Dynamics

Although the Dynamic inversion algorithm seems promising, since it can take a large class of systems and define their input output behavior as desired, there are some issues that must be dealt with. In particular, the feedback linearizing transformation of the control may render some states of the system unobservable. Thus we need some way to guarantee the behavior of these states is stable in some sense.

Let us return to the feedback linearized state space system. Define

$$\begin{aligned}\zeta &= [z_1, \dots, z_d]^T \\ \eta &= [z_{d+1}, \dots, z_n]\end{aligned}$$

Let the system start at the origin $z = 0$ and let the control $v(t) = 0$ for all t . The zero dynamics of the system are thus defined as the dynamics of the η under the above initial condition and control.

$$\eta(t+1) = q(\eta(t)) \triangleq \begin{bmatrix} \phi_{d+1}(\Phi^{-1}(0, \eta(t))) \\ \vdots \\ \phi_n(\Phi^{-1}(0, \eta(t))) \end{bmatrix}$$

We would like the above system to be asymptotically stable. If it is, the system is referred to as being minimum phase.

If we are only given an input output model such as the d-step ahead predictor form, we must first find a state space description in order to define the zeros dynamics of the input output feedback linearized system. Such a description is easy to find. In particular, let

$$x(t) \triangleq [y(t+d-1), y(t+d-2), \dots, y(t-n+1), u(t-1), \dots, u(t-n+1)]$$

$$A \triangleq \begin{bmatrix} \begin{bmatrix} 0 & \dots & 0 \\ & I & \vdots \\ & & 0 \end{bmatrix} & 0_{(n+d-1) \times (n-1)} \\ 0_{(n-1) \times (n+d-1)} & \begin{bmatrix} 0 & \dots & 0 \\ & I & \vdots \\ & & 0 \end{bmatrix} \end{bmatrix}$$

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

A state space realization of the d step ahead predictor form is thus

$$x(t+1) = Ax(t) + B \begin{bmatrix} f_n^*(y(t), \dots, y(t-n+1), u(t), \dots, u(t-n+1)) \\ U(y(t+d), y(t), \dots, y(t-n+1), u(t-1), \dots, u(t-n+1)) \end{bmatrix}$$

and the unobservable states are the $u(t)$. If we partition the state vector into observable ζ and unobservable η states, as above, the zero dynamics thus become

$$\eta(t+1) = A \begin{bmatrix} 0 \\ \eta(t) \end{bmatrix} + B \begin{bmatrix} 0 \\ U(0, \eta(t)) \end{bmatrix}$$

and we call the system minimum phase if the above dynamics are asymptotically stable.

Solutions to deal with unstable zero dynamics usually involve the addition of *outer* control loops which shape the type of control inputs $y_{desired}$ as a function of the unobservable states. These states are obviously measured in the outer control

loop. An example of this strategy is implemented on the helicopter in Chapter 6. It is clear that if pitch rate is treated as an output, its dynamics must be chosen carefully if the pitch is not to go unbounded. An outer feedback loop defines the desired pitch rate to achieve a given desired horizontal velocity.

3.4 MIMO Feedback Linearization

So far, the discussion of feedback linearization has treated only the SISO case. We need to extend the above notions to the MIMO systems. See reference [13].

$$\begin{aligned}x(t+1) &= f(x(t), u(t)) \\ y_i(t) &= h_i(x(t))\end{aligned}\tag{3.1}$$

where $i \in [1, \dots, m]$, $x \in \mathbf{R}^n$, $u \in \mathbf{R}^m$, $y_i \in \mathbf{R}$ and f and h are analytic. Delay for the MIMO case is defined similar to the SISO case.

Definition 3.4.1 *The output y_i of the MIMO system has relative degree d_i if*

$$\frac{\partial}{\partial u_j}(h_i(\Phi_{k+1}(x(t), 0, \dots, 0, u(t)))) = 0$$

for $k \in [0, \dots, d-1]$ and $j \in [1, \dots, m]$, and there exists $\bar{j} \in \{1, \dots, m\}$ such that

$$\frac{\partial}{\partial u_{\bar{j}}}(h_i(\Phi_{d_i+1}(x(t), 0, \dots, 0, u(t)))) \neq 0$$

for all $x \in \mathbf{R}^n$ and $u \in \mathbf{R}^m$.

Definition 3.4.2 *The input output decoupling matrix is*

$$A(x, u) = \left\{ \frac{\partial}{\partial u_j}(h_i(\Phi_{d_i+1}(x(t), 0, \dots, 0, u(t)))) \right\}_{i,j}$$

If the origin is reachable, and $A(x, u)$ has rank m then there exists a transformation of the input such that the system is input-output decoupled ([13]). The new

transformed system takes the form

$$\begin{aligned}\eta_i(t+1) &= v_i(t) \\ \zeta_j(t+1) &= F(\eta(t), \zeta(t), v(t)) \\ y_i &= \eta_i\end{aligned}$$

for $i \in \{1, \dots, m\}$, $j \in \{m+1, \dots, n\}$, and

$$\eta = \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_m \end{bmatrix}, \quad \zeta = \begin{bmatrix} \zeta_{m+1} \\ \vdots \\ \zeta_n \end{bmatrix}$$

Thus the system has been feedback linearized.

3.5 Summary

Now we have both the system and control background required to move on to an exact mathematical formulation and proof of the algorithm proposed by this thesis. This algorithm, which can be discussed in more specific terms, takes the system of Chapter 2 and shows how and under what conditions we can identify the feedback linearizing controller of this chapter and use it to control the system.

Chapter 4

Proposed Learning Strategy

4.1 Problem Description

This section introduces the learning control algorithm of this thesis. The presentation is in now way complete or precise. An exact mathematical formulation is presented in the next section. What follows is intended to provide an intuitive feel of the proposed algorithm.

The primary objective is to find a controller which guarantees bounded tracking error for a given system in a given controllable region of the state space. The algorithm described in the next two sections can accomplish this goal given a minimal amount of prior information about the desired controller. In particular, if a sufficiently good approximation to the *ideal* (zero tracking error) controller is known in some small region of the state space, the proposed controller can extend the initial small region to the desired region, while at the same time decreasing error to any arbitrarily small value. Furthermore, the algorithm can be applied to a large class of non-linear, feedback linearizable systems which can be written in the *D-step ahead predictor* form described in detail in Section 2.2

The basic idea for the algorithm is as follows. If we have good knowledge of the controller in some small region of the desired state space, then we can *fly* our system within that space. By flying near the boundaries, sample data can be collected and then extrapolated outside the boundaries of the small initial region. Using the new

extrapolated data, the boundaries can be pushed further and further until the data spans the entire desired region of the state space.

4.2 Exact Problem Statement

An exact mathematical formulation of the above strategy follows. This formulation consists of a description of the system under consideration followed by a precise claim of the control strategy's capabilities. The problem statement does not specify the controller, but claims the existence of the controller. The control strategy is then constructed by means of the proof in the next section.

The class of systems considered is all systems which can be written in the *D-step ahead predictor* form with delay equal to the order of the system.

Let y_t and u_t be the output and input of the system, respectively. For now, only consider the case where y_t and u_t are real scalars. The proof can be extended to the case of vector valued input and outputs under certain minimal restrictions. These restrictions include that the input and output have equal numbers of scalar components. For simplicity, only consider the scalar case. The system is

$$\begin{aligned} y_{t+n} &= f(u_t, w_t) \\ w_t &\triangleq [y_{t+n-1}, \dots, y_{t-n+1}] \end{aligned} \tag{4.1}$$

where $n \geq 1$. The function $f : \mathbf{Y}^{2n-1} \times \mathbf{U} \rightarrow \mathbf{R}$ is a continuous function with continuous first and second derivatives. The sets $\mathbf{Y} \subseteq \mathbf{R}$ and $\mathbf{U} \subseteq \mathbf{R}$ are open, and y_t and u_t are the input and output of the system at time t respectively. Now make the following assumption.

Assumption 4.2.1

$$\frac{\partial f}{\partial u_t}(u_t, 0) \neq 0, \quad \forall u_t \in \mathbf{U}$$

As a consequence of the above assumption and the *Inverse Function Theorem* [17, p. 345], there exist closed intervals $\mathbf{D}_y \subseteq \mathbf{Y}$ and $\mathbf{D}_u \subseteq \mathbf{U}$, and a function $U : \Omega_y^{2n} \rightarrow \mathbf{R}$

such that

$$f(U(v_t, w_t), w_t) = v_t \quad (4.2)$$

for all

$$\begin{aligned} v_t &\in \mathbf{D}_y \\ w_t &\in \mathbf{D}_y^{2n-1}, \end{aligned}$$

where v_t is an exogenous input. Furthermore, this function is continuous with continuous derivatives, and

$$\frac{\partial U}{\partial v_t}(v_t, w_t) \neq 0 \quad (4.3)$$

This result states, in words, that there exists an exact, input-output feedback linearizing transformation U of the control u_t . To see this, let $u_t = U(v_t, w_t)$, where v_t is the new control variable. Therefore

$$y_{t+n} = f(U(v_t, w_t), w_t) = v_t.$$

The equation $y_{t+n} = v_t$ is clearly linear. The implications for control are clear. If the functions U and f , and all past outputs and inputs were known precisely, any desired output $y_{des_{t+n}}$ could be achieved exactly by letting the input be

$$U(y_{des_{t+n}}, w_t)$$

where w_t is known since past outputs $\{y_{t-n+1}, \dots, y_t\}$, can be measured and the future outputs $\{y_{t+1}, \dots, y_{t+n-1}\}$, can be calculated since

$$\begin{aligned} y_{t+1} &= f(u_{t+1-n}, y_t, \dots, y_{t+1-(2n-1)}) \\ &\vdots \\ y_{t+n-1} &= f(u_{t-1}, y_{t+n-2}, \dots, y_{t-2n-2}). \end{aligned}$$

In reality, though, this condition would never exist. Any controller would have to make do with approximations of f and U on restricted domains. We now make the assumption that such approximations exist.

Assumption 4.2.2 *There exist known functions \hat{U}_0 and \hat{f}_0 and closed intervals*

$$\begin{aligned}\hat{\mathbf{D}}_{y,0} &\subset \mathbf{D}_y \text{ and} \\ \hat{\mathbf{D}}_{u,0} &\subset \mathbf{D}_u\end{aligned}$$

such that

$$\begin{aligned}|\hat{U}_0(y_{t+n}, w_t) - U(y_{t+n}, w_t)| &\leq \delta_{u,0} \\ |\hat{f}_0(u_t, w_t) - f(u_t, w_t)| &\leq \delta_{f,0}\end{aligned}$$

for all

$$\begin{aligned}y_{t+n} &\in \hat{\mathbf{D}}_{y,0} \\ w_t &\in \hat{\mathbf{D}}_{y,0}^{2n-1} \\ u_t &\in \hat{\mathbf{D}}_{u,0}.\end{aligned}$$

These approximations may be poor, and the intervals $\hat{\mathbf{D}}_{y,0}$ and $\hat{\mathbf{D}}_{u,0}$ may be small. The goal of this thesis is to present a scheme to generate successively better approximations

$$\begin{aligned}\hat{f}_0, \hat{f}_1, \dots, \hat{f}_N, \text{ and} \\ \hat{U}_0, \hat{U}_1, \dots, \hat{U}_N\end{aligned}$$

on successively larger intervals

$$\begin{aligned}\hat{\mathbf{D}}_{y,0} \subset \hat{\mathbf{D}}_{y,1} \subset \dots \subset \hat{\mathbf{D}}_{y,N} \text{ and} \\ \hat{\mathbf{D}}_{u,0} \subset \hat{\mathbf{D}}_{u,1} \subset \dots \subset \hat{\mathbf{D}}_{u,N}.\end{aligned}$$

This goal is stated in Theorem 4.2.1 as the primary result of this thesis.

Theorem 4.2.1 *There exist numbers, δ_f , δ_u and C , depending only on properties of the functions f and U , such that if*

$$\begin{aligned}\delta_{u,0} &\leq \delta_u \\ \delta_{f,0} &\leq \delta_f \\ \max(y \in D_0) - \min(y \in D_0) &\geq C\delta_u,\end{aligned}$$

then there exists a sequence of inputs u_t and corresponding outputs y_t from which we can construct functions $\hat{U}_N(y_{t+n}, w_t) : \hat{\mathbf{D}}_{y,N}^{2n} \rightarrow \mathbf{R}$ and $\hat{f}_N(u_t, w_t) : \hat{\mathbf{D}}_{y,N}^{2n-1} \times$

$\hat{\mathbf{D}}_{u,N}(w_t) \rightarrow$ satisfying

$$\begin{aligned} |U(y_{t+n}, w_t) - \hat{U}_N(y_{t+n}, w_t)| &\leq \delta_{u,N} \\ |f(u_t, w_t) - \hat{f}_N(u_t, w_t)| &\leq \delta_{f,N} \end{aligned}$$

for any

$$\delta_{u,N} \in (0, \delta_{u,0}],$$

$$\delta_{f,N} \in (0, \delta_{f,0}]$$

and for all

$$y_{t+n} \in \hat{\mathbf{D}}_{y,N}$$

$$w_t \in \hat{\mathbf{D}}_{y,N}^{2n-1}$$

$$u_t \in \hat{\mathbf{D}}_{u,N}(w_t)$$

where $\hat{\mathbf{D}}_{y,N} \in \mathbf{D}_y$ is closed and contains $\hat{\mathbf{D}}_{y,0}$ and

$$\hat{\mathbf{D}}_{u,N}(w_t) = \{u_t = U(y_{t+n}, w_t) | y_{t+n} \in \hat{\mathbf{D}}_{y,N}\}.$$

□

In itself, the statement that we can achieve successively better approximations of the functions U and f seem to imply very little. Appearances aside, the consequences are significant. As we will see in Section 4.3.2, a controller can be constructed from these approximations, and both the tracking error and the set of commandable outputs y depend directly on the approximation errors of the \hat{U} and \hat{f} . Essentially, the smaller the approximation errors, the smaller the tracking errors, and the larger the $\hat{\mathbf{D}}_{y,N}$ the larger the set of commandable outputs.

4.3 Proof

The proof of Theorem 4.2.1 presented in Section 4.2 can be separated into a few main elements. These elements are divided among the remaining sections of this chapter. Although they may be complete and self contained, their individual influences on the big picture, and their use in proving Theorem 4.2.1 may not be immediately apparent.

The following outline is included specifically to provide clarification on these issues. Keep the outline in mind as you progress through the remainder of this chapter.

Section 4.3.1 shows how an approximation to a function can be constructed from samples of the function. The theorems contained in the section give bounds on the error between the function and its approximation. These theorems are of a general nature and do not make reference to the system of Equation 4.1.

Section 4.3.2 describes a control algorithm which can be constructed from the approximations \hat{f}_k and \hat{U}_k , and shows that the better the approximations, the better the tracking error of the control algorithm.

Section 4.3.3 describes how samples of f and U are acquired using the control algorithm of Section 4.3.2. This sampling is central to the proof, since if these samples can be used to generate better approximations of f and U , then it may be possible to repeatedly sample and generate improved approximations. As will be shown, this is indeed the case. Section 4.3.4 shows that the acquired samples can be applied to the results of Section 4.3.1 in order to generate improved approximations of f and U , and the final section brings together the results of the previous sections to complete the proof of Theorem 4.2.1.

4.3.1 Error Theorems

The following theorems show how to take samples of some function f and construct an approximation \hat{f} which satisfies certain error properties. The first two theorems, Theorem 4.3.1 and Theorem 4.3.2, treat functions whose domain is a set of scalars. Theorem 4.3.3 and its corollaries extends these theorems to functions with vector domains by reducing the vector problem to the scalar problem treated in Theorems 4.3.1 and 4.3.2.

Theorem 4.3.1 *Let $f : \mathbf{D} \rightarrow \mathbf{R}$ where $\mathbf{D} \triangleq [a, b]$. We are given samples of f at two different points x_1, x_2 in \mathbf{D} :*

$$y_1 = f(x_1), \quad y_2 = f(x_2).$$

Without loss of generality, we can assume $x_1 < x_2$. Define the function $f : \mathbf{D} \rightarrow \mathbf{R}$ as

$$\hat{f}(x) = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1). \quad (4.4)$$

Suppose f is continuous with continuous first derivative, and $\left| \frac{d^2 f}{dx^2} \right| \leq L$. Then

$$\left| f(x) - \hat{f}(x) \right| \leq \frac{3L}{2}(b - a)^2, \quad \forall x \in \mathbf{D}. \quad (4.5)$$

Proof: Applying Taylor's Theorem (with remainder) [17, p.185] to $f(x)$ and about x_1 for some arbitrary $x \in \mathbf{D}$ yields

$$f(x) = f(x_1) + \frac{df}{dx}(x_1)(x - x_1) + \frac{1}{2} \frac{d^2 f}{dx^2}(\zeta_1)(x - x_1)^2, \quad (4.6)$$

for some ζ_1 on the interval between x_1 and x . Since $\frac{d\hat{f}}{dx}(x) = \frac{y_2 - y_1}{x_2 - x_1}$, we can rewrite Equation (4.4) as

$$\hat{f}(x) = f(x_1) + \frac{d\hat{f}}{dx}(x)(x - x_1). \quad (4.7)$$

Subtracting Equation (4.7) from Equation (4.6) yields

$$\begin{aligned} \left| f(x) - \hat{f}(x) \right| &= \left| f(x_1) + \frac{df}{dx}(x_1)(x - x_1) + \frac{1}{2} \frac{d^2 f}{dx^2}(\zeta_1)(x - x_1)^2 \right. \\ &\quad \left. - f(x_1) - \frac{d\hat{f}}{dx}(x)(x - x_1) \right| \\ &\leq \left| \frac{df}{dx}(x_1) - \frac{d\hat{f}}{dx}(x) \right| |x - x_1| + \left| \frac{1}{2} \frac{d^2 f}{dx^2}(\zeta_1) \right| |x - x_1|^2. \end{aligned} \quad (4.8)$$

Since $\mathbf{D} = [a, b]$ it follows that

$$|x' - x''| \leq b - a, \quad \text{for any } x', x'' \in \mathbf{D}. \quad (4.9)$$

Therefore Equation (4.8) becomes

$$\left| f(x) - \hat{f}(x) \right| \leq \left| \frac{df}{dx}(x_1) - \frac{d\hat{f}}{dx}(x) \right| |x - x_1| + \frac{1}{2} L(b - a)^2. \quad (4.10)$$

To bound the first term on the right hand side of Equation (4.10) we proceed as follows. As a consequence of the Mean-Value Theorem [18, p. 89], there exists an

$x_3 \in [x_1, x_2]$ such that

$$\frac{df}{dx}(x_3) = \frac{d\hat{f}}{dx}(x) \quad (4.11)$$

By applying Taylor's Theorem to $f(x)$ about x_3 and x_1 respectively, and substituting Equation (4.11) into the result, we obtain

$$\begin{aligned} f(x_1) &= f(x_3) + \frac{d\hat{f}}{dx}(x)(x_1 - x_3) + \frac{1}{2} \frac{d^2\hat{f}}{dx^2}(\zeta_2)(x_1 - x_3)^2 \\ f(x_3) &= f(x_1) + \frac{d\hat{f}}{dx}(x_1)(x_3 - x_1) + \frac{1}{2} \frac{d^2\hat{f}}{dx^2}(\zeta_3)(x_3 - x_1)^2 \end{aligned} \quad (4.12)$$

for some $\zeta_2, \zeta_3 \in [x_1, x_3]$. From Equations (4.12) we can derive the following:

$$\begin{aligned} \left| \frac{df}{dx}(x_1) - \frac{d\hat{f}}{dx}(x) \right| &= \left| \frac{1}{2} \left(\frac{d^2\hat{f}}{dx^2}(\zeta_2) + \frac{d^2\hat{f}}{dx^2}(\zeta_3) \right) (x_3 - x_1) \right| \\ &\leq \frac{1}{2} \left(\left| \frac{d^2\hat{f}}{dx^2}(\zeta_2) \right| + \left| \frac{d^2\hat{f}}{dx^2}(\zeta_3) \right| \right) |x_3 - x_1| \\ &\leq L(b - a) \end{aligned} \quad (4.13)$$

Plugging Equation (4.13) into Equation (4.10) and applying Equation (4.9) yields the desired inequality (4.5).

□

Theorem 4.3.2 *Let $f : \mathbf{D} \rightarrow \mathbf{R}$ where $D \triangleq [a, b]$. We are given approximations of f at two points x_1, x_2 satisfying*

$$|y_1 - f(x_1)| \leq \epsilon_1, |y_2 - f(x_2)| \leq \epsilon_2, x_2 - x_1 \geq c > 0. \quad (4.14)$$

Define the function $\hat{f} : \mathbf{D} \rightarrow \mathbf{R}$ as

$$\hat{f}(x) = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1). \quad (4.15)$$

Suppose f is continuous with continuous first derivatives, and $\left| \frac{d^2f}{dx^2} \right| \leq L$. Then

$$\left| f(x) - \hat{f}(x) \right| \leq 2 \frac{b-a}{c} \max(\epsilon_1, \epsilon_2) + \frac{3L}{2}(b-a)^2, \quad \forall x \in \mathbf{D}. \quad (4.16)$$

Proof: Define an intermediate function $g : \mathbf{D} \rightarrow \mathbf{R}$ as

$$g(x) = f(x_1) + \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1). \quad (4.17)$$

First find the error between \hat{f} and g by subtracting Equation (4.17) from Equation (4.15).

$$\begin{aligned} |\hat{f}(x) - g(x)| &= \left| y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) - f(x_1) - \frac{f(x_2) - f(x_1)}{x_2 - x_1}(x - x_1) \right| \\ &= \left| (y_1 - f(x_1)) \frac{x_2 - x}{x_2 - x_1} + (y_2 - f(x_2)) \frac{x - x_1}{x_2 - x_1} \right| \\ &\leq |y_1 - f(x_1)| \left| \frac{x_2 - x}{x_2 - x_1} \right| + |y_2 - f(x_2)| \left| \frac{x - x_1}{x_2 - x_1} \right| \end{aligned} \quad (4.18)$$

Applying bounds (4.14) and Equation (4.9) to Equation (4.18) yields

$$\begin{aligned} |\hat{f}(x) - g(x)| &\leq \frac{\epsilon_1}{c}(b - a) + \frac{\epsilon_2}{c}(b - a) \\ &\leq 2 \frac{b - a}{c} \max(\epsilon_1, \epsilon_2) \end{aligned} \quad (4.19)$$

From Theorem 4.3.1,

$$|g(x) - f(x)| \leq \frac{3L}{2}(b - a)^2 \quad \forall x \in \mathbf{D}. \quad (4.20)$$

Combine Equations (4.20) and (4.19) to obtain inequality (4.16).

□

Theorem 4.3.3 *Let $g : \mathbf{D}_N \rightarrow \mathbf{R}$, where $\mathbf{D}_N \subset \mathbf{R}^N$. Suppose we are given approximations, $\{p_0, \dots, p_N\}$ of g at $N + 1$ points $\{z_0, \dots, z_N\}$ such that*

$$|p_i - g(z_i)| \leq \epsilon_1, \quad i = 0, 1, \dots, N. \quad (4.21)$$

Define the matrix Z as

$$\begin{bmatrix} z_0 & \dots & z_N \\ 1 & \dots & 1 \end{bmatrix}$$

and assume that Z is invertible. Define the set \mathbf{Z}_m for $m = 0, 1, \dots, N$ as

$$\mathbf{Z}_m = \{z_0, \dots, z_m\}. \quad (4.22)$$

Define the function $\hat{g} : \mathbf{R}^N \rightarrow \mathbf{R}$ as

$$\hat{g}(z) = \left(\begin{bmatrix} p_0 & \dots & p_N \end{bmatrix} Z^{-1} \right) \begin{bmatrix} z \\ 1 \end{bmatrix},$$

Suppose g is continuous with continuous first derivatives. Suppose, furthermore, that the second derivative of g along any straight line in \mathbf{D}_N has a bounded absolute value such that

$$\left| \frac{d^2 g}{dx^2}(xz' + (1-x)z'') \right| \leq L_1 |z' - z''|^2, \quad \forall z', z'' \in \mathbf{D}_N \text{ and } x \in \mathbf{R}. \quad (4.23)$$

Define the set $\text{Co}(\mathbf{Z}_m)$ as the convex hull¹ of \mathbf{Z}_m . Suppose

$$|g(\mathbf{z}) - \hat{g}(\mathbf{z})| \leq \epsilon_2, \quad \forall \mathbf{z} \in \text{Co}(\mathbf{Z}_{m-1}). \quad (4.24)$$

Then

$$|g(z) - \hat{g}(z)| \leq 2 \max(\epsilon_1, \epsilon_2) + \frac{3L_1}{2} \max \|z - z_m\|_2^2, \quad \forall z \in \text{Co}(\mathbf{Z}_m), \quad \forall \mathbf{z} \in \text{Co}(\mathbf{Z}_{m-1}). \quad (4.25)$$

Proof: Let $z \in \text{Co}(\mathbf{Z}_m)$. From Corollary A.0.1 in Appendix A, z can be expressed as

$$z = xz' + (1-x)z_m$$

for some $\mathbf{z}' \in \text{Co}(\mathbf{Z}_{m-1})$ and $x \in [0, 1]$. The functions $f(x)$ and $\hat{f}(x)$ can be defined as

$$\begin{aligned} f(x) &\triangleq g(xz' + (1-x)z_m) \\ \hat{f}(x) &\triangleq \hat{g}(xz' + (1-x)z_m). \end{aligned}$$

¹Definitions and properties of convex hulls are presented in Appendix A

Therefore Equation (4.25) can be established if it can be shown that

$$|f(x) - \hat{f}(x)| \leq 2 \max(\epsilon_1, \epsilon_2) + \frac{3L_1}{2} \max \|\mathbf{z}' - z_m\|_2^2, \quad \forall x \in [0, 1].$$

Since $\hat{g}(z_i) = [p_0 \cdots p_N] Z^{-1} \begin{bmatrix} z_i \\ 1 \end{bmatrix}$ for each $i = 0, \dots, N$ we have that

$$[\hat{g}(z_0) \cdots \hat{g}(z_N)] = [p_0 \cdots p_N] Z^{-1} Z = [p_0 \cdots p_N]. \quad (4.26)$$

Therefore $\hat{f}(0) = \hat{g}(z_m) = p_m$. Setting $x_1 = 0$, $x_2 = 1$, $y_1 = \hat{f}(0)$ and $y_2 = \hat{f}(1)$, and noting that $f(x_1) = g(z_m)$, $y_1 = \hat{f}(0) = p_m$, $f(x_2) = g(\mathbf{z}')$ and $y_2 = \hat{g}(\mathbf{z}')$, we have that

$$\begin{aligned} |y_1 - f(x_1)| &\leq \epsilon_1 \text{ from Equation (4.21),} \\ |y_2 - f(x_2)| &\leq \epsilon_2 \text{ from Equation (4.24).} \end{aligned}$$

Since, in addition,

$$\left| \frac{d^2 f}{dx^2} \right| \leq L_1 \max \|\mathbf{z} - z_m\|_2^2, \quad \forall x \in [0, 1], \text{ from Equation (4.23),}$$

f satisfies the assumptions of Theorem 4.3.2. Therefore, Equation (4.16) becomes

$$|f(x) - \hat{f}(x)| \leq 2 \max(\epsilon_1, \epsilon_2) + \frac{3L_1}{2} \max \|\mathbf{z} - z_m\|_2^2, \quad \forall x \in [0, 1].$$

which proves Equation (4.25). □

Corollary 4.3.1 *If the assumptions of Theorem 4.3.3 hold for $\epsilon_1 = 0$ and $m = 1$, then*

$$|g(z) - \hat{g}(z)| \leq (2^m - 1) \frac{3L_1}{2} d^2, \quad \forall z \in Co(\mathbf{Z}_m), \quad (4.27)$$

and for all m in $\{1, 2, \dots, N\}$, where

$$d = \max \|z' - z''\|_2, \quad \forall z', z'' \in Co(\mathbf{Z}_m). \quad (4.28)$$

Proof: The proof is by induction. First, assuming Equation (4.27) holds for $m = k$, for some $k \in \{1, 2, \dots, N-1\}$, and choosing $\epsilon_2 = (2^k - 1)\frac{3L_1}{2}d^2$ it follows that

$$|g(z) - \hat{g}(z)| \leq \epsilon_2, \quad \forall z \in \text{Co}(\mathbf{Z}_k),$$

and from Equation (4.28), that

$$\max \|z - z_k\|_2 = d \quad \forall z \in \text{Co}(\mathbf{Z}_k),$$

where \mathbf{Z}_k is defined in Equation 4.22. Therefore, from Theorem 4.3.3 we have that

$$|g(z) - \hat{g}(z)| \leq 2 \max(\epsilon_1, (2^k - 1)\frac{3L_1}{2}d^2) + \frac{3L_1}{2}d^2, \quad \forall z \in \text{Co}(\mathbf{Z}_{k+1}). \quad (4.29)$$

Since $\epsilon_1 = 0$ by assumption, Equation (4.29) can be reduced to

$$|g(z) - \hat{g}(z)| \leq (2^{k+1} - 1)\frac{3L_1}{2}d^2, \quad \forall z \in \text{Co}(\mathbf{Z}_{k+1}). \quad (4.30)$$

Equation (4.30) shows that an inequality of the form of Equation (4.27) holds for the case $m = k + 1$. To complete the induction proof, it suffices to show that Equation (4.27) holds for $m = 1$.

From Equation 4.26 and the assumption that $\epsilon_1 = 0$, it follows that

$$|g(z_i) - \hat{g}(z_i)| = 0, \quad \text{for } i = 0, 1. \quad (4.31)$$

From the definition of $\text{Co}(\mathbf{Z}_0)$, Equation 4.31 also implies that

$$|g(z) - \hat{g}(z)| = 0, \quad \forall z \in \text{Co}(\mathbf{Z}_0),$$

and hence $\epsilon_2 = 0$, where ϵ_2 is defined as in Theorem 4.3.3. It follows from Theorem 4.3.3 that

$$|g(z) - \hat{g}(z)| \leq \frac{3L_1}{2}d^2 = (2^1 - 1)\frac{3L_1}{2}d^2, \quad \forall z \in \text{Co}(\mathbf{Z}_1). \quad (4.32)$$

Equation (4.32) is exactly Equation (4.27) for $m = 1$ and the proof is complete. □

Definition 4.3.1 *The ball of radius r and center $z_c \in \mathbf{R}^N$ is*

$$B(z_c, r) = \{z \mid \|z - z_c\|_2 \leq r, z \in \mathbf{R}^N\}$$

Corollary 4.3.2 *If the assumptions of Corollary 4.3.1 hold and*

$$B(z_c, d_l) \subset \mathbf{Co}(\mathbf{Z}_N) \subseteq \mathbf{G} \subseteq \mathbf{D}_N \subset \mathbf{R}^N, \quad (4.33)$$

for some compact and convex set \mathbf{G} , and some d_l and z_c then

$$|g(z) - \hat{g}(z)| \leq (2^N - 1) \frac{3L_1 d_u}{4d_l} d^2 + \frac{3L_1}{2} d_u^2, \quad \forall z \in \mathbf{G} \quad (4.34)$$

where d is given in Equation 4.28 and

$$d_u = \max \|z' - z''\|_2, \quad \forall z', z'' \in \mathbf{G}. \quad (4.35)$$

Proof: Choose some arbitrary point $z \in \mathbf{G}$. The straight line passing through points z and z_c , and extending to the boundary of \mathbf{G} , intersects the boundary of $B(z_c, d_l)$ at exactly two points z_{b1} and z_{b2} , with $\|z_{b1} - z_{b2}\|_2 = 2d_l$. We can write this line as a function of a scalar x .

$$z(x) = z_{b1} + (z_{b2} - z_{b1})x \quad (4.36)$$

Since \mathbf{G} is convex, and $z(0) = z_{b1}$ and $z(1) = z_{b2}$, $z(x) \in \mathbf{G}$ implies $x \in [a, b]$ for some $a \leq 0$ and $b \geq 1$. Define

$$\begin{aligned} f(x) &\triangleq g(z_{b1} + (z_{b2} - z_{b1})x) \\ \hat{f}(x) &\triangleq \hat{g}(z_{b1} + (z_{b2} - z_{b1})x), \quad \forall x \in [a, b]. \end{aligned}$$

Since z was chosen arbitrarily, Equation (4.34) can be obtained by showing

$$\left|f(x) - \hat{f}(x)\right| \leq (2^N - 1) \frac{3L_1 d_u}{4d_l} d^2 + \frac{3L_1}{2} d_u^2, \quad \forall x \in [a, b].$$

Setting $x_1 = 0$, $x_2 = 1$, $y_1 = f(0)$, $y_2 = f(1)$ and noting that from Assumption 4.33 and Definition 4.36 that

$$\begin{aligned} z(x_1) &= z(0) = z_{b1} \in \text{Co}(\mathbf{Z}_N) \text{ and} \\ z(x_2) &= z(1) = z_{b2} \in \text{Co}(\mathbf{Z}_N), \end{aligned}$$

it follows from Corollary 4.3.1 that

$$\begin{aligned} \left|y_1 - \hat{f}(x_1)\right| &\leq (2^N - 1) \frac{3L_1}{2} d^2 \text{ and} \\ \left|y_2 - \hat{f}(x_2)\right| &\leq (2^N - 1) \frac{3L_1}{2} d^2. \end{aligned}$$

Therefore the ϵ_1 and ϵ_2 from Theorem 4.3.2 are

$$(2^N - 1) \frac{3L_1}{2} d^2.$$

Since, in addition, Assumption 4.23 yields

$$\left|\frac{d^2 f}{dx^2}\right| \leq L_1 \|z_{b1} - z_{b2}\|_2^2,$$

it follows from Theorem 4.3.2 that

$$\left|f(x) - \hat{f}(x)\right| \leq (b-a)(2^N - 1) \frac{3L_1}{2} d^2 + \frac{3L_1}{2} \|z_{b1} - z_{b2}\|_2^2 (b-a)^2, \quad \forall x \in [a, b]. \quad (4.37)$$

Since $2d_l = \|z_{b1} - z_{b2}\|_2$ and $\|z(a) - z(b)\|_2 \leq d_u$ we have that

$$(b-a) \leq \frac{d_u}{2d_l}.$$

and therefore Equation (4.37) becomes

$$|f(x) - \hat{f}(x)| \leq (2^N - 1) \frac{3L_1 d_u}{4d_l} d^2 + \frac{3L_1}{2} d_u^2, \quad \forall x \in [a, b].$$

Equation (4.34) follows.

□

4.3.2 Tracking

Theorem 4.2.1 tells us that, under certain conditions, we can find approximations \hat{f}_N and \hat{U}_N of f and U which are 'better' than the original approximations \hat{f}_0 and \hat{U}_0 . The proof of this theorem, to be presented in the final section of this chapter, will be made using an induction argument. Therefore, the assumptions and theorems of this and other sections are concerned with properties of intermediate approximations \hat{f}_i and \hat{U}_i for $i \in \{0, \dots, N\}$.

This section deals primarily with the tracking error associated with the System (4.1) using \hat{U}_i and \hat{f}_i for control. We begin by making an assumption of the approximation error of \hat{U}_i and \hat{f}_i .

Assumption 4.3.1 *There exist known functions \hat{U}_i and \hat{f}_i , and closed interval $\hat{\mathbf{D}}_{y,i}$ satisfying*

$$\hat{\mathbf{D}}_{y,i} \subseteq \hat{\mathbf{D}}_{y,N} \quad (4.38)$$

such that

$$\left| \hat{U}_i(y_{dest+n}, w_t) - U(y_{dest+n}, w_t) \right| \leq \delta_{u,i} \quad (4.39)$$

$$\left| \hat{f}_i(u_t, w_t) - f(u_t, w_t) \right| \leq \delta_{f,i} \quad (4.40)$$

for all $y_{dest+n} \in \hat{\mathbf{D}}_{y,i}$, $u_t \in \hat{\mathbf{D}}_{u,i}(w_t)$, and $w_t \in \hat{\mathbf{D}}_{y,i}^{2n-1}$, where $\hat{\mathbf{D}}_{y,N}$ is as in Theorem 4.2.1 and

$$\hat{\mathbf{D}}_{u,i}(w_t) = \{u_t = U(y_{t+n}, w_t) | y_{t+n} \in \hat{\mathbf{D}}_{y,i}\} \quad (4.41)$$

The notion of a sample of a function is now defined.

Definition 4.3.2 *A sample of a function $g : \mathbf{A} \rightarrow \mathbf{B}$ is an input-output pair (a, b) of g , where $a \in \mathbf{A}$ is a sample input and $b = g(a) \in \mathbf{B}$ is a sample output.*

We now proceed to derive an expression for the tracking error of System (4.1) controlled by \hat{U}_i . Such an expression is pivotal to the completion of the proof since it describes our ability to acquire prescribed samples of U and f . Theorem 4.3.5 which

states the desired expression, relies upon certain properties of the System (4.1). These properties are presented in the required form as a theorem.

Theorem 4.3.4 *Given System (4.1) and Assumption 4.2.1, the following inequalities hold for some N_f , N_u , N_{fl} and N_{fu} , and all $u_t \in \hat{\mathbf{D}}_{u,N}(w_t)$, $w_t \in \hat{\mathbf{D}}_{y,N}^{2n-1}$ and $y_i \in \hat{\mathbf{D}}_{y,N}$, $i \in \{t - n + 1, \dots, t + n\}$.*

$$\begin{aligned}
\left| \frac{\partial f}{\partial y_i}(u_t, w_t) \right| &\leq N_f \\
\left| \frac{\partial U}{\partial y_i}(y_{t+n}, w_t) \right| &\leq N_u \\
\left| \frac{\partial f}{\partial u_t}(u_t, w_t) \right| &\leq N_{fu} \\
\left| \frac{\partial f}{\partial u_t}(u_t, w_t) \right| &\geq N_{fl}
\end{aligned} \tag{4.42}$$

Also, assume $N_u \geq 1$. If 4.42 is satisfied with $N_u < 1$ then we can always choose $N_u = 1$. The set $\hat{\mathbf{D}}_{u,N}$ is as in Theorem 4.2.1 and

$$\hat{\mathbf{D}}_{u,N}(w_t) = \{u_t | f(u_t, w_t) \in \hat{\mathbf{D}}_{y,N}\}$$

Proof: From [1, p. 122], the continuity of f and U and their first partial derivatives on closed sets is sufficient to imply that N_u , N_f and N_{fu} exist. The constant N_{fl} exists since Assumption 4.2.1 implies that $\frac{\partial f}{\partial u_t}$ exists and is bounded away from zero on a compact set.

□

The following Lemma is useful in the proofs of the remaining theorems of this section.

Lemma 4.3.1 *If the function $g(x, z) : \mathbf{R}^n \rightarrow \mathbf{R}$ is continuous with continuous first partial derivatives, where*

$$x = (x_1, \dots, x_m) \in \mathbf{R}^m$$

and $z \in \mathbf{R}^{n-m}$, then

$$\begin{aligned}
g(x, z) &= g(x', z) \\
&\quad + \frac{\partial g}{\partial x_1}(\eta_1, x_2, \dots, x_m, z)(x_1 - x'_1) + \dots \\
&\quad + \frac{\partial g}{\partial x_k}(x'_1, \dots, x'_{k-1}, \eta_k, x_{k+1}, \dots, x_m, z)(x_k - x'_k) + \dots \\
&\quad + \frac{\partial g}{\partial x_m}(x'_1, \dots, x'_{m-1}, \eta_m, z)(x_m - x'_m)
\end{aligned} \tag{4.43}$$

where $\eta_k \in [x_k, x'_k]$ for $k = 1, \dots, m$ ² and

$$x' = (x'_1, \dots, x'_m) \in \mathbf{R}^m$$

Proof: An application of Taylor's Theorem to

$$g(x'_1, \dots, x'_{k-1}, x_k, \dots, x_m, z) \tag{4.44}$$

about the point

$$(x'_1, \dots, x'_k, x_{k+1}, \dots, x_m, z)$$

yields

$$\begin{aligned}
g(x'_1, \dots, x'_{k-1}, x_k, \dots, x_m, z) &= g(x'_1, \dots, x'_k, x_{k+1}, \dots, x_m, z) \\
&\quad + \frac{\partial g}{\partial x_k} g(x'_1, \dots, x'_{k-1}, \eta_k, x_{k+1}, \dots, x_m, z)(x_k - x'_k)
\end{aligned} \tag{4.45}$$

where $\eta_k \in [x_k, x'_k]$. Define

$$\begin{aligned}
y_k &= (x'_1, \dots, x'_k, x_{k+1}, \dots, x_m, z) \\
\zeta_k &= (x'_1, \dots, x'_{k-1}, \eta_k, x_{k+1}, \dots, x_m, z)
\end{aligned} \tag{4.46}$$

Therefore Equation (4.45) becomes

$$g(y_{k-1}) = g(y_k) + \frac{\partial g}{\partial x_k}(\zeta_k)(x_k - x'_k) \tag{4.47}$$

²In this proof it is assumed that $x'_k > x_k$. A similar proof can be given if $x_k > x'_k$ with the modification $\eta_k \in [x'_k, x_k]$.

for each $k = 0, \dots, m$. Equation (4.47) is a recursive relationship which yields

$$\begin{aligned}
g(y_0) &= g(y_1) + \frac{\partial g}{\partial x_1}(\zeta_1)(x_1 - x'_1) \\
&= g(y_2) + \frac{\partial g}{\partial x_2}(\zeta_2)(x_2 - x'_2) + \frac{\partial g}{\partial x_1}(x_1 - x'_1) \\
&\vdots \\
&= g(y_m) + \frac{\partial g}{\partial x_m}(\zeta_m)(x_m - x'_m) + \dots + \frac{\partial g}{\partial x_1}(x_1 - x'_1)
\end{aligned}$$

Substituting Equations (4.46) proves Equation (4.43). □

Definition 4.3.3 Given an interval S , define the set $S'(\delta)$ as

$$S'(\delta) \triangleq \{y \mid [y - \delta, y + \delta] \subseteq S\}.$$

Theorem 4.3.5 Assume that the System (4.1) satisfies Assumptions 4.2.1 and 4.3.1.

Define

$$\delta_{y,j} \triangleq |y_{t+j} - \hat{y}_{t+j}|, \quad j \in \{1, \dots, n-1\} \quad (4.48)$$

where \hat{y}_{t+j} is an estimate of y_{t+j} ³.

$$\hat{y}_{t+j} = \hat{f}_i(u_{t+j-n}, \hat{y}_{t+j-1}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t+j-2n+1}), \quad j \in \{1, \dots, n-1\}$$

Set

$$\delta_{tracking} \triangleq N_{fu}\delta_{u,i} + \frac{N_{fu}N_u}{N_f}((N_f + 1)^{n-1} - 1)\delta_{f,i} \quad (4.49)$$

and let

$$u_t = \hat{U}_i(y_{des_{t+n}}, \hat{y}_{t+n-1}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t-n+1}), \quad \forall y_{des_{t+n}} \in \hat{\mathbf{D}}'_{y,i}(\delta_{tracking}) \quad (4.50)$$

where N_f , N_u and N_{fu} are the constants satisfying the inequalities (4.42). Then

$$|y_{t+n} - y_{des_{t+n}}| \leq \delta_{tracking}. \quad (4.51)$$

³It is necessary for any controller to use estimates of y_{t+j} for any $j > 0$ since the controller must be causal.

and

$$y_{t+n} \in \hat{\mathbf{D}}_{y,i} \quad \forall y_{dest+n} \in \hat{\mathbf{D}}'_{y,i}(\delta_{tracking}). \quad (4.52)$$

Proof: Define u_{dest} as the input which takes us to y_{dest+n} .

$$u_{dest} = U(y_{dest+n}, y_{t+n-1}, \dots, y_{t-n+1}) \quad (4.53)$$

We note that Equations (4.1) and (4.53) and an application of Taylor's Theorem to f about u_{dest} for some u_t yields.

$$|y_{t+n} - y_{dest+n}| = \left| \frac{\partial f}{\partial u_t}(\mu, y_{t+n-1}, \dots, y_{t-n+1}) \right| |u_t - u_{dest}| \quad (4.54)$$

where μ is on the interval between u_t and u_{dest} . Using Equations (4.50) and (4.39)

$$\begin{aligned} |u_t - u_{dest}| &= |\hat{U}_i(y_{dest+n}, \hat{y}_{t+n-1}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t-n+1}) \\ &\quad - U(y_{dest+n}, y_{t+n-1}, \dots, y_{t-n+1})| \\ &\leq |U(y_{dest+n}, \hat{y}_{t+n-1}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t-n+1}) \\ &\quad - U(y_{dest+n}, y_{t+n-1}, \dots, y_{t-n+1})| + \delta_{u,i} \end{aligned} \quad (4.55)$$

since

$$|\hat{U}_i(x) - U(z)| \leq |\hat{U}_i(x) - U(x)| + |U(z) - U(x)|$$

for any real x and z . By using Lemma 4.3.1 with

$$x = (\hat{y}_{t+1}, \dots, \hat{y}_{t+n-1}), x' = (y_{t+1}, \dots, y_{t+n-1}) \text{ and } z = (y_{dest+n}, y_t, \dots, y_{t-n+1}),$$

Equation (4.55) can be written as

$$\begin{aligned} |u_t - u_{dest}| &\leq \left| \frac{\partial U}{\partial y_{t+n-1}}(y_{dest+n}, \eta_{n-1}, \hat{y}_{t+n-2}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t-n+1})(\hat{y}_{t+n-1} - y_{t+n-1}) \right. \\ &\quad + \dots \\ &\quad \left. + \frac{\partial U}{\partial y_{t+n-1}}(y_{dest+n}, \dots, y_{t+2}, \eta_1, y_t, \dots, y_{t-n+1})(\hat{y}_{t+1} - y_{t+1}) \right| + \delta_{u,i} \end{aligned} \quad (4.56)$$

where $\eta_k \in [y_{t+k}, \hat{y}_{t+k}]$, $k = 1, \dots, n-1$ ⁴. Noting Equation (4.48) and applying the

⁴In this proof it is assumed that $\hat{y}_{t+k} > y_{t+k}$. A similar proof can be given if $y_{t+k} > \hat{y}_{t+k}$ with

bound in (4.42) to Equation (4.56) yields

$$|u_t - u_{des_t}| \leq N_u(\delta_{y,1} + \cdots + \delta_{y,n-1}) + \delta_{u,i} \quad (4.57)$$

Now we find upper bounds for the terms $\delta_{y,i}$ for $j \in \{1, \dots, n-1\}$ as follows. Using Equations (4.48) and (4.40)

$$\begin{aligned} \delta_{y,j} &= |f(u_{t+j-n}, y_{t+j-1}, \dots, y_{t+j-2n+1}) \\ &\quad - \hat{f}_i(u_{t+j-n}, \hat{y}_{t+j-1}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t+j-2n+1})| \\ &\leq |f(u_{t+j-n}, y_{t+j-1}, \dots, y_{t+j-2n+1}) \\ &\quad - f(u_{t+j-n}, \hat{y}_{t+j-1}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t+j-2n+1})| + \delta_{f,i} \end{aligned} \quad (4.58)$$

since

$$|f(x) - \hat{f}_i(z)| \leq |f(x) - f(z)| + |f(z) - \hat{f}_i(z)|$$

for any real x and z . As before, by using Lemma 4.3.1 with

$$x = (\hat{y}_{t+1}, \dots, \hat{y}_{t+j-1}), x' = (y_{t+1}, \dots, y_{t+j-1}) \text{ and } z = (u_{t+j-n}, y_t, \dots, y_{t+j-2n+1}),$$

there exists $\zeta_k \in [y_{t+k}, y_{t+k}]$ for $k = 1, \dots, j-1$, such that Equation (4.58) becomes

$$\begin{aligned} \delta_{y,j} &\leq \left| \frac{\partial f}{\partial y_{t+j-1}}(u_{t+j-n}, \zeta_{j-1}, \hat{y}_{t+j-2}, \dots, \hat{y}_{t+1}, y_t, \dots, y_{t+j-2n+1})(\hat{y}_{t+j-1} - y_{t+j-1}) \right. \\ &\quad + \cdots \\ &\quad \left. + \frac{\partial f}{\partial y_{t+1}}(u_{t+j-n}, y_{t+j-1}, \dots, y_{t+2}, \zeta_1, y_t, \dots, y_{t+j-2n+1})(\hat{y}_{t+1} - y_{t+1}) \right| \\ &\quad + \delta_{f,i} \end{aligned} \quad (4.59)$$

Noting Equation (4.48), and applying the bounds of (4.42) to Equation (4.59) yields

$$\delta_{y,j} \leq N_f(\delta_{y,1} + \cdots + \delta_{y,j-1}) + \delta_{f,i} \quad (4.60)$$

the modification $\eta_k \in [\hat{y}_{t+k}, y_{t+k}]$.

Since Equation (4.60) holds for all $j = 1, \dots, n - 1$, it follows that

$$\delta_{y,j} \leq (N_f + 1)\delta_{y,j-1} \quad (4.61)$$

Noting that $\delta_{y,1} \leq \delta_{f,i}$, Equation (4.61) reduces to

$$\delta_{y,j} \leq (N_f + 1)^{j-1}\delta_{f,i}, \quad j \in \{0, \dots, n - 1\} \quad (4.62)$$

The proof is now completed by combining Equations (4.42), (4.54) and (4.57) as follows.

$$\left| y_{t+n} - y_{des_{t+n}} \right| \leq N_{f,u} (\delta_{u,i} + N_u(\delta_{y,1} + \dots + \delta_{y,n-1}))$$

Using Equation (4.62), it follows that

$$\left| y_{t+n} - y_{des_{t+n}} \right| \leq N_{f,u}\delta_{u,i} + N_{f,u}N_u \left(\sum_{k=1}^{n-1} (N_f + 1)^{k-1} \right) \delta_{f,i}$$

Using the identity

$$\sum_{k=1}^{n-1} (N_f + 1)^{k-1} = \frac{1}{(N_f + 1)^{n-1} - 1} N_f$$

it holds that

$$\left| y_{t+n} - y_{des_{t+n}} \right| \leq N_{f,u}\delta_{u,i} + \frac{N_{f,u}N_u}{N_f} ((N_f + 1)^{n-1} - 1)\delta_{f,i}$$

which proves Equation (4.51). Since we assumed $y_{des_{t+n}} \in \hat{\mathbf{D}}'_{y,i}(\delta_{tracking})$ and since Equation (4.51) holds, we have from Definition 4.3.3 that Equation (4.52) holds.

□

Before moving on to the next section, two more results are introduced. Although they are not directly related to tracking, the methods are similar to Theorem 4.3.5 so the presentation is made in this section.

Theorem 4.3.6 *Let*

$$\begin{aligned} y'_{t+n} &= f(u'_t, y'_{t+n-1}, \dots, y'_{t-n+1}) \in \hat{\mathbf{D}}_{y,N} \\ y''_{t+n} &= f(u''_t, y''_{t+n-1}, \dots, y''_{t-n+1}) \in \hat{\mathbf{D}}_{y,N} \end{aligned}$$

and assume

$$|y'_{t+j} - y''_{t+j}| \leq d, \quad j \in \{-n+1, \dots, n-1\}. \quad (4.63)$$

If

$$D_l \leq |y'_{t+n} - y''_{t+n}| \leq D_u \quad (4.64)$$

then

$$\frac{D_l}{N_{fu}} - \frac{N_f}{N_{fu}}(2n-1)d \leq |u'_t - u''_t| \leq \frac{D_u}{N_{fl}} + \frac{N_f}{N_{fl}}(2n-1)d \quad (4.65)$$

where N_f , N_{fl} and N_{fu} are defined by Equations (4.42).

Proof: An application of Lemma 4.3.1 to f with

$$x = (u'_t, y'_{t-n+1}, \dots, y'_{t-n-1}), x' = (u''_t, y''_{t-n+1}, \dots, y''_{t-n-1}) \text{ and } z = ()$$

yields

$$\begin{aligned} |y'_{t+n} - y''_{t+n}| &= \left| \frac{\partial f}{\partial u_t}(\eta, y'_{t+n-1}, \dots, y'_{t-n+1})(u'_t - u''_t) \right. \\ &\quad + \frac{\partial f}{\partial y_{t+n-1}}(u''_t, \zeta_{n-1}, y'_{t+n-2}, \dots, y'_{t-n+1})(y'_{t+n-1} - y''_{t+n-1}) \\ &\quad + \dots \\ &\quad \left. + \frac{\partial f}{\partial y_{t-n+1}}(u''_t, y''_{t+n-1}, \dots, y''_{t-n+2}, \zeta_{-n+1})(y'_{t-n+1} - y''_{t-n+1}) \right| \end{aligned} \quad (4.66)$$

where η is on the interval between u'_t and u''_t and ζ_j is on the interval between y'_{t+j} and y''_{t+j} for each $j \in \{-n+1, \dots, n-1\}$. Applying Equations (4.42), (4.63) and (4.64) to Equation (4.66) yields

$$D_l \leq N_{fu} |u'_t - u''_t| + N_f(2n-1)d \quad (4.67)$$

$$D_u \geq N_{fl} |u'_t - u''_t| - N_f(2n-1)d \quad (4.68)$$

Solving for $|u'_t - u''_t|$ in each of Equation (4.67) and (4.68) produces Equation (4.65).

□

Theorem 4.3.7 *Let*

$$\begin{aligned} y'_{t+n} &= f(u'_t, y'_{t+n-1}, \dots, y'_{t-n+1}) \in \hat{\mathbf{D}}_{y,N} \\ y''_{t+n} &= f(u''_t, y''_{t+n-1}, \dots, y''_{t-n+1}) \in \hat{\mathbf{D}}_{y,N} \end{aligned}$$

If

$$|y'_{t+j} - y''_{t+j}| \leq d_j \quad (4.69)$$

where $j = -n + 1, n$ and , then

$$|u'_t - u''_t| \leq N_u \sum_{j=-n+1}^n d_j \quad (4.70)$$

where N_u is defined by Equations (4.42).

Proof: An application of Lemma 4.3.1 to U with

$$x = (y'_{t-n+1}, \dots, y'_{t+n}), x' = (y''_{t-n+1}, \dots, y''_{t+n}) \text{ and } z = ()$$

yields

$$\begin{aligned} |u'_t - u''_t| &= \left| \frac{\partial U}{\partial y_{t+n}}(\zeta_n, y'_{t+n-1}, \dots, y'_{t-n+1})(y'_{t+n} - y''_{t+n}) \right. \\ &\quad + \dots \\ &\quad \left. + \frac{\partial U}{\partial y_{t-n+1}}(y''_{t+n}, \dots, y''_{t-n+2}, \zeta_{-n+1})(y'_{t-n+1} - y''_{t-n+1}) \right| \end{aligned} \quad (4.71)$$

where ζ_j is on the interval between y'_{t+j} and y''_{t+j} for each $j \in \{-n + 1, \dots, n\}$.

Applying Equations (4.42) and (4.69) to Equation (4.71) yields Equation (4.70)

□

4.3.3 Sampling

The purpose of this section is to describe exactly how samples of U and f are collected, and demonstrate that the acquired samples satisfy certain properties which will prove useful in the application of the theorems of Section 4.3.1 to the samples.

First a bit of notation is introduced. Let t_k be such that y_{t_k} is an output y at some time $t+\Delta$, where Δ denotes a translation in time. Since Δ 's can be arbitrary and need not be sequential, $y_t, y_{t_2}, \dots, y_{t_k}$ need not denote consecutive values of y . Since at time t_k , the functions f and U of the system in Equation (4.1) satisfy Assumption 4.2.1, we have

$$\begin{aligned} y_{t_k+n} &= f(u_{t_k}, y_{t_k+n-1}, \dots, y_{t_k-n+1}) \\ u_{t_k} &= U(y_{t_k+n}, \dots, y_{t_k-n+1}) \end{aligned}$$

Now we collect the inputs to f and u into vectors and denote them as z_{t_k} and x_{t_k} respectively. That is,

$$\begin{aligned} x_{t_k} &= \begin{bmatrix} y_{t_k+n} \\ \vdots \\ y_{t_k-n+1} \end{bmatrix} \\ z_{t_k} &= \begin{bmatrix} u_{t_k} \\ y_{t_k+n-1} \\ \vdots \\ y_{t_k-n+1} \end{bmatrix} \end{aligned}$$

Since we assume that all outputs y and inputs u can be measured, the act of sampling is simply the act of measuring and storing values of x_{t_k} , z_{t_k} , u_{t_k} and y_{t_k+n} , where, using Definition 4.3.2, the pair

$$(x_{t_k}, u_{t_k}) \tag{4.72}$$

is a sample of U and the pair

$$(z_{t_k}, y_{t_k+n}) \tag{4.73}$$

is a sample of f .

Sampling Procedure: Let a set of desired input samples to U be given.

$$\mathbf{X}_{des} = \{x_{des_{t_0}}, \dots, x_{des_{t_{2n}}}\}$$

where each desired input sample $x_{des_{t_j}}$ can be written

$$x_{des_{t_k}} = \begin{bmatrix} y_{des_{t_k+n}} \\ \vdots \\ y_{des_{t_k-n+1}} \end{bmatrix}$$

At each time t_k+m-n for $k = 0, \dots, 2n$ and $m = n, \dots, -n+1$ let u_{t_k+m-n} be the control input of Equation (4.50) with $y_{des_{t_k+n}} = y_{des_{t_k+m}}$. We can then measure values of x_{t_k} , z_{t_k} , u_{t_k} and y_{t_k+n} for each $k = 1, \dots, 2n$. Assume that this measurement can be taken with perfect accuracy. It is also important to note that under the conditions outlined in Theorem 4.3.5, that the tracking error satisfies

$$|y_{t_k+m} - y_{des_{t_k+m}}| \leq \delta_{tracking}$$

for $k = 0, \dots, 2n$, and $m = -n+1, \dots, n$. The term $\delta_{tracking}$ is described explicitly in Equation (4.49).

Now, for any given $x_{des_{t_{(2n+1)j}}}$, we specify $x_{des_{t_{(2n+1)j+k}}}$ for $k = 1, \dots, 2n$ as

$$x_{des_{t_{(2n+1)j+k}}} = x_{des_{t_{(2n+1)j}}} + \delta e_k^{2n} \quad (4.74)$$

where

$$\delta = \delta_{tracking} \max(10n^2, 40n^3 N_u N_{fu} + N_f(2n-1) + 2) \quad (4.75)$$

for N_f , N_u and N_{fu} are defined in Equation (4.42) and

$$e_k^{2n} = [0 \cdots 010 \cdots 0] \quad (4.76)$$

is a vector of length $2n$ with all zero entries except in the k_{th} position. Define the set of $2n + 1$ desired samples of Equation (4.74) as

$$\mathbf{X}_{des_j} = \{x_{dest_{(2n+1)j}}, \dots, x_{dest_{(2n+1)j+2n}}\} \quad (4.77)$$

Note that the factor $2n + 1$ is added so that the intersection of the sets \mathbf{X}_{des_j} and $\mathbf{X}_{des_{j+1}}$ is empty. Therefore we can define numerous sets indexed by integers j which have no implied dependence on each other.

Now a critical theorem is presented. This theorem states that the acquired samples satisfy properties which will be useful in proving the statement of Theorem 4.2.1.

Theorem 4.3.8 *Let \mathbf{X}_{des_0} ⁵ be defined as in Equation (4.77). Acquire the corresponding samples*

$$(x_{t_k}, u_{t_k}), \quad (z_{t_k}, y_{t_k+n})$$

according to the scheme outlined above in Sampling Procedure . Assume that each $y_{des_{t_k+m}}$ for $k = 0, \dots, 2n$, $m = n, \dots, -n + 1$ is in the set $\hat{\mathbf{D}}'_{y,i}(\delta_{tracking})$ as defined by Equation (4.38) and Definition 4.3.3. Then the acquired samples satisfy the following properties.

A: *The matrix*

$$X' = \begin{bmatrix} x_{t_0} & \cdots & x_{t_{2n}} \\ 1 & \cdots & 1 \end{bmatrix} \quad (4.78)$$

is invertible.

B: *The set $Co(x_{t_0}, \dots, x_{t_{2n}})$ is such that*

$$B(x_c, r_x) \subseteq Co(x_{t_0}, \dots, x_{t_{2n}}) \quad (4.79)$$

for

$$r_x \geq \frac{\sqrt{n(n+1)}}{2n+1} \delta_{tracking} \quad (4.80)$$

⁵We let the subscript j be zero without loss of generality. The subscript j will prove useful in the next section to distinguish between numerous \mathbf{X}_{des_j} .

C: The matrix

$$Z' = \begin{bmatrix} z_{k_0} & \cdots & z_{t_{2n}} \\ 1 & \cdots & 1 \end{bmatrix} \quad (4.81)$$

is invertible.

D: The set $Co(z_{t_0}, \dots, z_{t_{2n}})$ is such that

$$B(z_c, r_z) \subseteq Co(z_{t_0}, \dots, z_{t_{2n}}) \quad (4.82)$$

for

$$r_z \geq \frac{\sqrt{n(n+1)}}{2n+1} \delta_{tracking} \quad (4.83)$$

E: Let

$$G_x = \left\{ x = x_{dest_0} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix} \text{ where } g_i \in [-\delta - 2\delta_{tracking}, \delta + 2\delta_{tracking}] \quad i = 1, \dots, 2n \right\} \quad (4.84)$$

Then

$$G_x \subseteq Co(x_{t_0}, \dots, x_{t_{2n}}) \quad (4.85)$$

and satisfies

$$\|x' - x''\|_2 \leq \sqrt{2n}(2\delta + 4\delta_{tracking}) \quad (4.86)$$

for $x', x'' \in G_x$.

F: Let

$$G_z = \left\{ z = \begin{bmatrix} u_{t_0} \\ y_{dest_k+n-1} \\ \vdots \\ y_{dest_k-n+1} \end{bmatrix} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix} \right\} \quad (4.87)$$

where

$$\begin{aligned} g_1 &\in [-4nN_u(\delta + 3\delta_{tracking}), 4nN_u(\delta + 3\delta_{tracking})] \\ g_i &\in [-(\delta + 2\delta_{tracking}), \delta + 2\delta_{tracking}] \quad i = 2, \dots, 2n \end{aligned}$$

Then

$$G_z \subset Co(z_{t_0}, \dots, z_{t_{2n}}) \quad (4.88)$$

and

$$\|z' - z''\|_2 \leq 8nN_u\sqrt{2n}(\delta + 3\delta_{tracking}) \quad (4.89)$$

for $z', z'' \in G_z$. Furthermore,

$$x_t = \begin{bmatrix} y_{t+n} \\ \vdots \\ y_{t-n+1} \end{bmatrix} \in G_x$$

implies

$$z = \begin{bmatrix} U(x) \\ y_{t+n-1} \\ \vdots \\ y_{t-n+1} \end{bmatrix} \in G_z \quad (4.90)$$

Proof: ⁶ We begin by first establishing a few useful inequalities.

- Since $y_{dest_{k+m}} \in \hat{D}'_{y,i}(\delta_{tracking})$ by assumption, we can apply Equation (4.51) of Theorem (4.3.5) to yield

$$|y_{t_k+m} - y_{dest_{k+m}}| \leq \delta_{tracking} \quad (4.91)$$

⁶Throughout this proof we make reference to N_u , N_f and N_{fu} of Equation (4.42).

for $k = 0, \dots, 2n$, and $m = -n + 1, \dots, n$. Therefore

$$\begin{aligned}
|y_{t_k+m} - y_{t_p+m}| &= |y_{t_k+m} - y_{t_p+m} \\
&\quad + y_{des_{t_p+m}} - y_{des_{t_k+m}}| \\
&\leq |y_{des_{t_k+m}} - y_{des_{t_p+m}}| \\
&\quad + |y_{t_k+m} - y_{des_{t_k+m}}| \\
&\quad + |y_{des_{t_p+m}} - y_{t_p+m}| \\
&\leq |y_{des_{t_k+m}} - y_{des_{t_p+m}}| + 2\delta_{tracking}
\end{aligned} \tag{4.92}$$

for $k = 1, \dots, 2n$, $m = -n + 1, \dots, n$, $p = 1, \dots, 2n$, and $p \neq k$. Similarly,

$$|y_{t_k+m} - y_{t_p+m}| \geq |y_{des_{t_k+m}} - y_{des_{t_p+m}}| - 2\delta_{tracking} \tag{4.93}$$

Together, Equation (4.92) and (4.93) yield

$$y_{t_k+m} - y_{t_p+m} = y_{des_{t_k+m}} - y_{des_{t_p+m}} + \gamma \tag{4.94}$$

where $|\gamma| \leq 2\delta_{tracking}$ for $k = 1, \dots, 2n$, $m = -n + 1, \dots, n$ and $p = 1, \dots, 2n$.

Therefore we can use Equation (4.94) to write

$$x_{t_k} - x_{t_p} = x_{des_{t_k}} - x_{des_{t_p}} + \Gamma \tag{4.95}$$

where $|\gamma| \leq 2\delta_{tracking}$ for each γ an entry in the vector Γ .

- Let

$$\begin{aligned}
y'_{t+m} &= y_{t_1+m}, \\
y''_{t+m} &= y_{t_0+m}, \\
u'_t &= u_{t_1}, \\
u''_t &= u_{t_0}, \\
d &= 2\delta_{tracking} \text{ from Equations (4.92) and (4.74),} \\
D_l &= \delta - 2\delta_{tracking} \text{ from Equations (4.93) and (4.74),} \\
D_u &= \delta + 2\delta_{tracking} \text{ from Equations (4.92) and (4.74)}
\end{aligned}$$

for $m = -n + 1, \dots, n$. Therefore an application of Theorem 4.3.6 yields the following inequality.

$$|u_{t_1} - u_{t_0}| \geq \frac{1}{N_{fu}}(\delta - (2 + N_f(2n - 1))\delta_{tracking}). \quad (4.96)$$

- Let

$$\begin{aligned}
y'_{t+m} &= y_{t_k+m}, \\
y''_{t+m} &= y_{t_p+m}, \\
u'_t &= u_{t_k}, \\
u''_t &= u_{t_p}, \\
d_k &= \delta + 2\delta_{tracking} \text{ from Equations (4.92) and (4.74),}
\end{aligned}$$

for $m = -n + 1, \dots, n - 1$, $k = -n + 1, \dots, n - 1$ and $p = 0, \dots, 2n$. Therefore an application of Theorem 4.3.7 yields the following inequality.

$$|u_{t_k} - u_{t_p}| \leq 2nN_u(\delta + 2\delta_{tracking}) \quad (4.97)$$

Since Equation (4.75) gives us that $\delta \geq 2\delta_{tracking}$, Equation (4.97) becomes

$$|u_{t_k} - u_{t_p}| \leq 4nN_u\delta \quad (4.98)$$

Proof of A:) Define the $(2n \times 2n + 1)$ matrix

$$X = \begin{bmatrix} x_{t_0} - x_{t_0} & \cdots & x_{t_{2n}} - x_{t_0} \end{bmatrix}$$

Using Equation (4.95), X can be written as

$$\begin{aligned}
X &= \begin{bmatrix} x_{dest_0} - x_{dest_0} & \cdots & x_{dest_{2n}} - x_{dest_0} \end{bmatrix} + \Gamma \\
&= \delta \begin{bmatrix} 0[2n \times 1] & I[2n \times 2n] \end{bmatrix} + \Gamma \\
&\triangleq \delta \Lambda + \Gamma
\end{aligned} \tag{4.99}$$

where $I[\cdot]$ is the identity matrix, $0[\cdot]$ is a matrix of zeros and $|\gamma| \leq 2\delta_{tracking}$ for γ an entry of the matrix Γ .

Define the $(2n \times 2n)$ matrix

$$X_m = \begin{bmatrix} x_{t_0} - x_{t_m} & \cdots & x_{t_{m-1}} - x_{t_m}, & x_{t_{m+1}} - x_{t_m} & \cdots & x_{t_{2n}} - x_{t_m} \end{bmatrix}$$

Using Equation (4.95) X_m can be written

$$\begin{aligned}
X_m &= \begin{bmatrix} x_{dest_0} - x_{dest_m} & \cdots & x_{dest_{m-1}} - x_{dest_m}, & x_{dest_{m+1}} - x_{dest_m} & \cdots & x_{dest_{2n}} - x_{dest_m} \end{bmatrix} + \Gamma_m \\
&= \delta \begin{bmatrix} 0[m-1 \times 1] & I[m-1 \times m-1] & 0[m-1 \times n-m] \\ & -1 & \cdots & -1 \\ 0[n-m \times m-1] & & & I[n-m \times n-m] \end{bmatrix} + \Gamma_m \\
&\triangleq \delta \Lambda_m + \Gamma_m
\end{aligned}$$

where $|\gamma| \leq 2\delta_{tracking}$ for γ an entry of Γ_m . If we let λ_m be the $(m+1)_{th}$ row of Λ , the matrix defined in Equation (4.99), then we can write the matrix Λ_m as

$$\Lambda_m = \begin{bmatrix} \lambda_0 - \lambda_m & \cdots & \lambda_{m-1} - \lambda_m, & \lambda_{m+1} - \lambda_m & \cdots & \lambda_{2n} - \lambda_m \end{bmatrix}$$

Therefore, we can let

$$\begin{aligned}
\Theta &= \Lambda \\
\Theta_m &= \Lambda_m \\
\theta_k &= 0, \quad k = 0, 2, \dots, 2n \\
\theta_1 &= 1 \\
N &= 2n
\end{aligned}$$

and apply Equation (B.0.6) of Theorem B.0.6 to yield

$$\|\Lambda_m v\|_2 \geq \frac{1}{2n} \|v\|_2$$

for any $v \in \mathbf{R}^{2n}$. Since $\|\delta\Lambda_m v\|_2 = \delta \|\Lambda_m v\|_2$

$$\|\delta\Lambda_m v\|_2 \geq \frac{\delta}{2n} \|v\|_2 \quad (4.100)$$

Since $|\gamma_m| \leq 2\delta_{tracking}$ for each γ_m and entry of Γ_m , we can let

$$\begin{aligned} \Theta &= \delta\Lambda_m \\ \Psi &= \Gamma_m \\ \theta &= \frac{\delta}{2n} \text{ from Equation (4.100)} \\ \psi &= 2\delta_{tracking} \\ N &= 2n \end{aligned}$$

and apply Equation (B.2) of Theorem B.0.5 to get

$$\|X_m v\|_2 = \|\delta(\Lambda_m + \Gamma_m)v\|_2 \geq \left(\frac{\delta}{2n} - 4n\delta_{tracking} \right) \|v\|_2$$

for any $v \in \mathbf{R}^{2n}$. Equation (4.75) gives us that $\delta \geq 10n^2\delta_{tracking}$ Therefore

$$\|X_m v\|_2 = \|\delta(\Lambda_m + \Gamma_m)v\|_2 \geq \left(\frac{\delta}{2n} - 4n\delta_{tracking} \right) \|v\|_2 \geq n\delta_{tracking} \|v\|_2 > 0 \quad (4.101)$$

Since Equation (4.101) states $\|X_m v\|_2 > 0$, Lemma B.0.1 gives us that X_m is invertible. By Theorem B.0.4, X_0 invertible implies X' of Equation (4.78) invertible. So we have shown A:

Proof of B:) Now we let

$$\begin{aligned} A_m &= X_m \\ a_m &= x_{t_m} \\ \alpha_m &= n\delta_{tracking} \\ n &= 2n \end{aligned}$$

for $m = 0, \dots, 2n$. Therefore an application of Theorem A.0.3 yields that the mini-

mum width d_x of $\text{Co}(x_{t_0}, \dots, x_{t_{2n}})$ satisfies

$$d_x \geq \frac{1}{\sqrt{2n}} n \delta_{tracking}$$

Therefore, by Theorem A.0.2, $\text{Co}(x_{t_0}, \dots, x_{t_{2n}})$ contains a ball of radius r_x satisfying

$$\begin{aligned} r_x &\geq d_x \frac{\sqrt{2n+2}}{2n+1} \delta_{tracking} \\ &\geq \frac{n(n+1)}{2n+1} \delta_{tracking} \end{aligned}$$

Equation (4.80) is proven.

Proof of C:) Define the $(2n \times 2n + 1)$ matrix

$$Z = \begin{bmatrix} z_{t_0} - z_{t_0} & \cdots & z_{t_{2n}} - z_{t_0} \end{bmatrix}$$

Using Equations (4.94), Z can be written as

$$\begin{aligned} Z &= \begin{bmatrix} u_{t_0} - u_{t_0} & \cdots & u_{t_{2n}} - u_{t_0} \\ y_{dest_{t_0+n-1}} - y_{dest_{t_0+n-1}} & \cdots & y_{dest_{t_{2n}+n-1}} - y_{dest_{t_0+n-1}} \\ \vdots & & \vdots \\ y_{dest_{t_0-n+1}} - y_{dest_{t_0-n+1}} & \cdots & y_{dest_{t_{2n}-n+1}} - y_{dest_{t_0-n+1}} \end{bmatrix} + \Gamma \\ &= \delta \begin{bmatrix} \frac{u_{t_0} - u_{t_0}}{\delta} & \cdots & \frac{u_{t_{2n}} - u_{t_0}}{\delta} \\ 0[2n - 1 \times 2] & & I[2n - 1 \times 2n - 1] \end{bmatrix} + \Gamma \\ &\triangleq \delta \Lambda + \Gamma \end{aligned} \tag{4.102}$$

where $|\gamma| \leq 2\delta_{tracking}$ for γ an entry of Γ .

Define the $(2n \times 2n)$ matrix

$$Z_m = \begin{bmatrix} z_{t_0} - z_{t_m} & \cdots & z_{t_{m-1}} - z_{t_m}, & z_{t_{m+1}} - z_{t_m} & \cdots & z_{t_{2n}} - z_{t_m} \end{bmatrix}$$

Using Equation (4.94) Z_m can be written

$$\begin{aligned}
Z_m &= \begin{bmatrix} u_{t_0} & \cdots & u_{t_{m-1}} & u_{t_{m+1}} & \cdots & u_{t_{2n}} \\ y_{dest_{t_0+n-1}} & \cdots & y_{dest_{t_{m-1}+n-1}} & y_{dest_{t_{m+1}+n-1}} & \cdots & y_{dest_{t_{2n}+n-1}} \\ \vdots & & \vdots & \vdots & & \vdots \\ y_{dest_{t_0-n+1}} & \cdots & y_{dest_{t_{m-1}-n+1}} & y_{dest_{t_{m+1}-n+1}} & \cdots & y_{dest_{t_{2n}-n+1}} \end{bmatrix} \\
&- \begin{bmatrix} u_{t_m} & \cdots & u_{t_m} \\ y_{dest_{t_0+n-1}} & \cdots & y_{dest_{t_0+n-1}} \\ \vdots & & \vdots \\ y_{dest_{t_0-n+1}} & \cdots & y_{dest_{t_0-n+1}} \end{bmatrix} + \Gamma_m \\
&= \delta \begin{bmatrix} \frac{u_{t_0}-u_{t_m}}{\delta} & \cdots & \frac{u_{t_{m-1}}-u_{t_m}}{\delta} & \frac{u_{t_{m+1}}-u_{t_m}}{\delta} & \cdots & \frac{u_{t_{2n}}-u_{t_m}}{\delta} \\ 0[m-2 \times 1] & & & I[m-2 \times m-2] & & 0[m-2 \times n-m] \\ -1 & & & \cdots & & -1 \\ & & 0[n-m \times m-1] & & & I[n-m \times n-m] \end{bmatrix} + \Gamma_m \\
&\triangleq \delta \Lambda_m + \Gamma_m
\end{aligned}$$

where $|\gamma| \leq 2\delta_{tracking}$ for γ an entry of Γ_m . If we let λ_m be the $(m+1)_{th}$ row of Λ , the matrix defined in Equation (4.102), then we can write the matrix Λ_m as

$$\Lambda_m = \begin{bmatrix} \lambda_0 - \lambda_m & \cdots & \lambda_{m-1} - \lambda_m, & \lambda_{m+1} - \lambda_m & \cdots & \lambda_{2n} - \lambda_m \end{bmatrix}$$

Therefore, we can let

$$\begin{aligned}
\Theta &= \Lambda \\
\Theta_m &= \Lambda_m \\
\theta_k &= \frac{u_{t_k} - u_{t_0}}{\delta} \\
N &= 2n
\end{aligned}$$

and apply Equation (B.0.6) of Theorem B.0.6 to yield

$$\|\Lambda_m v\|_2 \geq \frac{\left| \frac{u_{t_0} - u_{t_1}}{\delta} \right|}{2n \max_{k=2, \dots, 2n} \left(1, \left| \frac{u_{t_0} - u_{t_k}}{\delta} \right|, \left| \frac{u_{t_1} - u_{t_k}}{\delta} \right| \right)} \|v\|_2$$

By applying Equation (4.96) and (4.98) we have

$$\|\Lambda_m v\|_2 \geq \frac{\frac{1}{N_{fu}} (\delta - (2 + N_f(2n - 1))\delta_{tracking})}{2n \max(\delta, 4nN_u\delta)} \|v\|_2$$

Since $N_u \geq 1$, we have that $2nN_u \geq 1$. Therefore

$$\|\Lambda_m v\|_2 \geq \frac{(\delta - (2 + N_f(2n - 1))\delta_{tracking})}{8n^2 N_u N_{fu} \delta} \|v\|_2$$

for any $v \in \mathbf{R}^{2n}$. Since $\|\delta\Lambda_m v\|_2 = \delta \|\Lambda_m v\|_2$

$$\|\delta\Lambda_m v\|_2 \geq \frac{(\delta - (2 + N_f(2n - 1))\delta_{tracking})}{8n^2 N_u N_{fu}} \|v\|_2 \quad (4.103)$$

Since $|\gamma_m| \leq 2\delta_{tracking}$ for each γ_m and entry of Γ_m , we can let

$$\begin{aligned} \Theta &= \delta\Lambda_m \\ \Psi &= \Gamma_m \\ \theta &= \frac{(\delta - (2 + N_f(2n - 1))\delta_{tracking})}{8n^2 N_u N_{fu}} \quad \text{from Equation (4.103)} \\ \psi &= 2\delta_{tracking} \\ N &= 2n \end{aligned}$$

and apply Equation (B.2) of Theorem B.0.5 to get

$$\|Z_m v\|_2 = \|\delta(\Lambda_m + \Gamma_m)v\|_2 \geq \left(\frac{(\delta - (2 + N_f(2n - 1))\delta_{tracking})}{8n^2 N_u N_{fu}} - 4n\delta_{tracking} \right) \|v\|_2 \quad (4.104)$$

for any $v \in \mathbf{R}^{2n}$. Equation (4.75) gives us that $\delta \geq (40N_u N_{fu} n^3 + N_f(2n - 1) + 2)\delta_{tracking}$. Substituting in Equation (4.104) yields

$$\|Z_m v\|_2 \geq n\delta_{tracking} > 0$$

Since $\|Z_m v\|_2 > 0$, Lemma B.0.1 gives us that Z_m is invertible. But by Theorem B.0.4, Z_0 invertible implies Z' of Equation (4.81) is invertible. So we have shown C:.

Proof of D:) Now we let

$$\begin{aligned} A_m &= Z_m \\ a_m &= z[j]_m \\ \alpha_m &= n\delta_{tracking} \\ n &= 2n \end{aligned}$$

for $m = 0, \dots, 2n$. Therefore an application of Theorem A.0.3 yields that the minimum width d_z of $\text{Co}(z_{t_0}, \dots, z_{t_{2n}})$ satisfies

$$d_z \geq \frac{n}{\sqrt{2n}} \delta_{tracking}$$

Therefore, by Theorem A.0.2, $\text{Co}(z_{t_0}, \dots, z_{t_{2n}})$ contains a ball of radius r_z satisfying

$$\begin{aligned} r_z &\geq d_z \frac{\sqrt{2n+2}}{2n+1} \delta_{tracking} \\ &= \frac{n(n+1)}{2n+1} \delta_{tracking} \end{aligned}$$

So Equation (4.83) is shown.

Proof of E:) We first show that G_x of Equation (4.84) is convex. Let $x', x'' \in G_x$. Then, by the definition of G_x the vectors x', x'' can be written

$$\begin{aligned} x' &= x_{dest_0} + \begin{bmatrix} g'_1 \\ \vdots \\ g'_{2n} \end{bmatrix} \\ x'' &= x_{dest_0} + \begin{bmatrix} g''_1 \\ \vdots \\ g''_{2n} \end{bmatrix} \end{aligned}$$

where $g'_i, g''_i \in [-\delta - 2\delta_{tracking}, \delta + 2\delta_{tracking}]$ for each $i = 1, \dots, 2n$. By Definition A.0.1, $G_x(x[j]_{des_0})$ is convex if each $x = \lambda x' + (1 - \lambda)x''$ is in G_x for $\lambda \in [0, 1]$.

$$\begin{aligned} x &= \lambda x' + (1 - \lambda)x'' \\ &= x_{dest_0} + \lambda \begin{bmatrix} g'_1 \\ \vdots \\ g'_{2n} \end{bmatrix} + (1 - \lambda) \begin{bmatrix} g''_1 \\ \vdots \\ g''_{2n} \end{bmatrix} \\ &\triangleq x_{dest_0} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix} \end{aligned}$$

where $g_i = \lambda g'_i + (1 - \lambda)g''_i$ for $i = 1, \dots, 2n$. Since $\lambda \in [0, 1]$, we have that each g_i lies in the interval between g'_i and g''_i . Therefore, $g_i \in [-(\delta + 2\delta_{tracking}), \delta + 2\delta_{tracking}]$. It follows from Equation (4.84) that $x \in G_x$. Therefore, the set G_x is convex. Now we show that G_x contains each of $x_{t_0}, \dots, x_{t_{2n}}$. Since

$$x_{t_k} = \begin{bmatrix} y_{t_k+n} \\ \vdots \\ y_{t_k-n+1} \end{bmatrix}$$

from Equation (4.73) and

$$\begin{aligned} |y_{t_k+m} - y_{des_{t_k+m}}| &\leq |y_{des_{t_k+m}} - y_{des_{t_k+m}}| + |y_{t_k+m} - y_{des_{t_k+m}}| \\ &\leq \delta + \delta_{tracking} \end{aligned} \quad (4.105)$$

from Equation (4.91), we have that

$$y_{t_k+m} = y_{des_{t_0+m}} + g \quad (4.106)$$

for $m = -n + 1, \dots, n$, where

$$g \in [-(\delta + \delta_{tracking}), \delta + \delta_{tracking}] \subset [-(\delta + 2\delta_{tracking}), (\delta + 2\delta_{tracking})]$$

Therefore, by Equation (4.84), $x_{t_k} \in G_x$ for $k = 0, \dots, 2n$. Since, in addition, G_x is convex, Definition A.0.2 gives us that

$$\text{Co}(x_{t_0}, \dots, x_{t_{2n}}) \subseteq G_x$$

Let $x', x'' \in G_x$, with

$$x' = \begin{bmatrix} y'_{t+n} \\ \vdots \\ y'_{t-n+1} \end{bmatrix}, \quad x'' = \begin{bmatrix} y''_{t+n} \\ \vdots \\ y''_{t-n+1} \end{bmatrix}$$

By Equation (4.84)

$$|y'_{t+m} - y''_{t+m}| \leq 2(\delta + 2\delta_{tracking}) \quad (4.107)$$

for $m = -n + 1, \dots, n$. Therefore

$$\begin{aligned} \|x' - x''\|_2 &= \sqrt{\sum_{m=-n+1}^n (y'_{t+m} - y''_{t+m})^2} \\ &\leq \sqrt{2n(2(\delta + 2\delta_{tracking}))^2} \\ &= \sqrt{2n}2(\delta + 2\delta_{tracking}) \end{aligned}$$

and Equation (4.86) is shown.

Proof of F:) The methods to prove the results of F: are similar to those used above in order to show the results of E:. Consequently, the argument for the convexity of G_z is omitted.

From Equation (4.98)

$$|u_{t_k} - u_{t_0}| \leq 4nN_u\delta$$

Therefore

$$u_{t_k} = u_{t_0} + g \quad (4.108)$$

where $g \in [-4nN_u\delta, 4nN_u\delta] \subseteq [-4nN_u(\delta + 3\delta_{tracking}), 4nN_u(\delta + 3\delta_{tracking})]$. From Equation (4.106), (4.108) and (4.87) we have that G_z contains each of $z_{t_0}, \dots, z_{t_{2n}}$. Since, in addition, $G_z(x_{dest_0})$ is convex, Definition A.0.2 gives us that

$$\text{Co}(z_{t_0}, \dots, z_{t_{2n}}) \subseteq G_z$$

Let $z', z'' \in G_z$, with

$$z' = \begin{bmatrix} u'_t \\ y'_{t+n-1} \\ \vdots \\ y'_{t-n+1} \end{bmatrix}, \quad z'' = \begin{bmatrix} u''_t \\ y''_{t+n-1} \\ \vdots \\ y''_{t-n+1} \end{bmatrix}$$

By Equation (4.87),

$$|u'_t - u''_t| \leq 8nN_u(\delta + 3\delta_{tracking}) \quad (4.109)$$

Using Equations (4.107) and (4.109) and $N_u > 1$ we have

$$\begin{aligned} \|z' - z''\|_2 &= \sqrt{(u'_t - u''_t)^2 + \sum_{m=-n+1}^{n-1} (y'_{t+m} - y''_{t+m})^2} \\ &\leq \sqrt{2n(8nN_u(\delta + 3\delta_{tracking}))^2} \\ &= \sqrt{2n}8nN_u(\delta + 3\delta_{tracking}) \end{aligned}$$

and Equation (4.89) is shown.

Let

$$x = \begin{bmatrix} y_{t+n-1} \\ \vdots \\ y_{t-n+1} \end{bmatrix} \in G_x(x[j]_{des_0})$$

By Equation (4.84)

$$|y_{t+m} - y_{des_{t_0+m}}| \leq \delta + 2\delta_{tracking} \quad (4.110)$$

for $m = -n + 1, \dots, n$.

Let

$$\begin{aligned} y'_{t+m} &= y_{t+m}, \\ y''_{t+m} &= y_{des_{t_0+m}}, \\ u'_t &= U(x), \\ u''_t &= U(x_{des_{t_0}}), \\ d_k &= \delta + 2\delta_{tracking} \quad \text{from Equation (4.110)} \end{aligned}$$

for $m = -n + 1, \dots, n - 1$. Therefore an application of Theorem 4.3.7 yields the following inequality.

$$|U(x) - U(x_{des_{t_0}})| \leq 2nN_u(\delta + 2\delta_{tracking}) \quad (4.111)$$

Let

$$\begin{aligned}
y'_{t+m} &= y_{t_0+m}, \\
y''_{t+m} &= y_{des_{t_0+m}}, \\
u'_t &= U(x_{t_0}), \\
u''_t &= U(x_{des_{t_0}}), \\
d_k &= \delta_{tracking} \text{ from Equation (4.91)}
\end{aligned}$$

for $m = -n + 1, \dots, n - 1$. Therefore an application of Theorem 4.3.7 yields the following inequality.

$$|U(x_{t_0}) - U(x_{des_{t_0}})| \leq 2nN_u\delta_{tracking} \quad (4.112)$$

From Equations (4.111) and (4.112)

$$|U(x) - U(x_{des_{t_0}})| \leq 2nN_u(\delta + 3\delta_{tracking})$$

Therefore

$$U(x) = U(x_{t_0}) + g = u_{t_0} + g \quad (4.113)$$

where $g \in [-2nN_u(\delta + 3\delta_{tracking}), 2nN_u(\delta + 3\delta_{tracking})]$. From Equations (4.87), (4.110) and (4.113) we have Equation (4.90).

□

Theorem 4.3.9 *The set*

$$D = \left\{ x = x_{des_{t_0}} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix} \text{ where } g_i \in [-\delta, \delta] \text{ } i = 1, \dots, 2n \right\} \quad (4.114)$$

contains $x_{des_{t_0}}, \dots, x_{des_{t_{2n}}}$.

Proof: From Equation (4.74) we have that

$$y_{des_{t_k+m}} = y_{des_{t_0+m}} + \delta \quad (4.115)$$

for $k = m + n, m = -n + 1, \dots, n$ and

$$y_{dest_{k+m}} = y_{dest_{t_0+m}} \quad (4.116)$$

for $k \neq m + n$. Therefore we can generalize Equations (4.115) and (4.116) to

$$x_{dest_k} = \begin{bmatrix} y_{dest_{k+n}} \\ \vdots \\ y_{dest_{k-n+1}} \end{bmatrix} = \begin{bmatrix} y_{dest_{t_0+n}} \\ \vdots \\ y_{dest_{t_0-n+1}} \end{bmatrix} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix} = x_{dest_{t_0}} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix}$$

where $|g_1|, \dots, |g_{2n}| \leq \delta$ Therefore, it is clear from Equation 4.114 that $x_{dest_k} \in D'$.

□

4.3.4 Approximation Theorems

In the previous section, we specified a set of desired samples \mathbf{X}_{des_j} in Equation (4.77) and then showed, through Theorem 4.3.8 that certain useful properties of the associated acquired samples were satisfied.

Now we would like to specify many such sets, and use the acquired samples to generate a new, better controller.

First note that as a consequence of the continuity of f and U , and the continuity of their first and second derivatives, we have that there exist L_x and L_z such that

$$\left| \frac{\partial^2 f(\lambda z + (1 - \lambda)z')}{\partial \lambda^2} \right| \leq L_z \|z - z'\|_2^2 \quad (4.117)$$

for all $z, z' \in D_y^{2n-1} \times D_u$, and

$$\left| \frac{\partial^2 U(\lambda x + (1 - \lambda)x')}{\partial \lambda^2} \right| \leq L_x \|x - x'\|_2^2 \quad (4.118)$$

for all $x, x' \in D_y^{2n}$.

Second, due to repeated references to Assumption 4.3.1, it is repeated here for convenience

Assumption 4.3.1 *There exist known functions \hat{U}_i and \hat{f}_i , and closed interval $\hat{D}_{y,i}$ satisfying*

$$\hat{D}_{y,i} \subseteq \hat{D}_{y,N} \quad (4.119)$$

such that

$$\left| \hat{U}_i(y_{des_{t+n}}, w_t) - U(y_{des_{t+n}}, w_t) \right| \leq \delta_{u,i} \quad (4.120)$$

$$\left| \hat{f}_i(u_t, w_t) - f(u_t, w_t) \right| \leq \delta_{f,i} \quad (4.121)$$

for all $y_{des_{t+n}} \in \hat{D}_{y,i}$, $u_t \in \hat{D}_{u,i}(w_t)$, and $w_t \in \hat{D}_{y,i}^{2n-1}$, where $\hat{D}_{y,N}$ is as in Theorem 4.2.1 and

$$\hat{D}_{u,i}(w_t) = \{u_t = U(y_{t+n}, w_t) | y_{t+n} \in \hat{D}_{y,i}\} \quad (4.122)$$

Theorem 4.3.10 *Let System (4.1) and Assumption 4.2.1 be given. Let Assumption 4.3.1 hold for $i = k$ and*

$$\delta_{u,k} \leq \frac{1}{2C_u\Delta^2} \quad (4.123)$$

$$\delta_{f,k} \leq \frac{\gamma^2}{2C_f\Delta^2} \quad (4.124)$$

$$\max_{y,y' \in \hat{\mathbf{D}}_{y,k}} |y - y'| \geq (\alpha + 2)\Delta\delta_{u,k} \quad (4.125)$$

where

$$C_u = (2^{2n} - 1) \left(\frac{3L_x n(2n+1)\sqrt{2n}}{2\sqrt{n(n+1)}} \right) (\alpha + 2)^3 + 6nL_x(\alpha + 2)^2 \quad (4.126)$$

$$C_f = (2^{2n} - 1) \left(\frac{256n^3 L_z N_u^3 (2n+1)\sqrt{2n}}{\sqrt{n(n+1)}} \right) (\alpha + 3)^3 + 48n^3 L_z N_u^3 (\alpha + 3)^2 \quad (4.127)$$

$$\gamma = \frac{\delta_{f,k}}{\delta_{u,k}} \quad (4.128)$$

$$\Delta = N_{fu} + \frac{N_{fu}N_u}{N_f} ((N_f + 1)^{n-1} - 1)\gamma \quad (4.129)$$

$$\alpha = \max(10n^2, 40n^3 N_u N_{fu} + N_f(2n - 1) + 2) \quad (4.130)$$

and the terms N_u , N_{fu} and N_f are those satisfying Equation (4.42). Then there exist desired samples (Equation (4.77))

$$\mathbf{X}_{des,j} \quad j = 0, \dots, J \quad (4.131)$$

of the form

$$(x_{t_m}, u_{t_m}) \quad \text{of } U \quad (4.132)$$

$$(z_{t_m}, y_{t_m+n}) \quad \text{of } f \quad (4.133)$$

for $m = 0, \dots, (2n + 1)J + 2n$ that can be obtained using the Sampling Procedure in Section 4.3.3 and from which \hat{f}_{k+1} and \hat{U}_{k+1} can be constructed such that Assump-

tion 4.3.1 is satisfied for $i = k + 1$ with

$$\delta_{u,k+1} = \frac{\delta_{u,k}}{2} \quad (4.134)$$

$$\delta_{f,k+1} = \frac{\delta_{f,k}}{2} \quad (4.135)$$

$$\hat{\mathbf{D}}_{y,k+1} = \{y \mid |y - y'| \leq \delta_{tracking,k}, \ y' \in \hat{\mathbf{D}}_{y,k}\} \quad (4.136)$$

$$\delta_{tracking,k} = \Delta\delta_{u,k} \quad (4.137)$$

$$(4.138)$$

Proof:

We separate the proof into four distinct steps.

1: Choose \mathbf{X}_{des_j} and show that

$$x_{dest_{(2n+1)j+m}} \in \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k}). \quad (4.139)$$

for $j = 0, \dots, J$ and $m = -n + 1, \dots, n$. This result lets us apply the results of Theorem 4.3.8 to each \mathbf{X}_{des_j} .

2: Show that

$$\bigcup_{j=0, \dots, J} G_{x_j} \subseteq \hat{\mathbf{D}}_{y,k+1}^{2n} \quad (4.140)$$

3: From the acquired samples we can generate a function \hat{U}_{k+1} satisfying

$$\left| \hat{U}_{k+1}(x) - U(x) \right| \leq \delta_{u,k+1} \quad (4.141)$$

for all $x \in \hat{\mathbf{D}}_{y,k+1}^{2n}$ (see Assumption (4.3.1)).

4: From the acquired samples we can generate a function \hat{f}_{k+1} satisfying

$$\left| \hat{f}_{k+1}(z) - f(z) \right| \leq \frac{\delta_{f,k}}{2} \quad (4.142)$$

for

$$z = \begin{bmatrix} u \\ w_t \end{bmatrix} \in \hat{\mathbf{D}}_{u,k+1}(w_t) \times \hat{\mathbf{D}}_{y,k+1}^{2n-1}$$

Proof:

Proof of 1:) Since Equation (4.125) holds, $\hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})$ (see Definition 4.3.3) satisfies

$$\max_{y,y' \in \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})} |y - y'| \geq (\alpha + 2)\Delta\delta_{u,k} - 2\delta_{tracking,k} \quad (4.143)$$

From Equation (4.137) it follows that Equation (4.143) can be written as

$$\max_{y,y' \in \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})} |y - y'| \geq \delta \quad (4.144)$$

where $\delta = \alpha\Delta\delta_{u,k}$. This implies that the set $\hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})^{2n}$ is a $2n$ -cube whose sides have a minimum length δ . From Equation (4.114) it is clear that D_0 is a $2n$ -cube with side length at most δ . Since D_0 is translated an amount x_{dest_0} , where x_{dest_0} is arbitrary, we can choose x_{dest_0} such that $D_0 \subseteq \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})$ [17]. Furthermore, we can find

$$x_{dest_{(2n+1)j}}$$

for $j = 0, \dots, J$ such that the union of the D_j is exactly equal to $\hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})^{2n}$ for a finite J , that is

$$\bigcup_{j=0,\dots,J} D_j = \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})^{2n} \quad (4.145)$$

That is, we have chosen

$$\mathbf{X}_{des_j} \quad j = 0, \dots, J \quad (4.146)$$

such that Equation (4.145) holds. It also follows that Equation (4.139) holds.

Proof of 2:) Let $x \in \hat{\mathbf{D}}_{y,k+1}^{2n}$. From Equation (4.136) it follows that there exists an

⁷In the statement and proof of Theorem 4.3.8 we assumed $j = 0$ for simplicity. For general j , let the G_x , G_z and D of Equations (4.84), (4.87) and (4.114) associated with \mathbf{X}_{des_j} , be G_{x_j} , G_{z_j} and D_j respectively.

$x' \in \hat{\mathbf{D}}_{y,k}^{2n}$ such that

$$x_m = x'_m + \gamma_m$$

where z_i denotes the i_{th} element of a vector z , and $\gamma_m \in [-\delta_{tracking,k}, \delta_{tracking,k}]$. We note from Definition 4.3.3 that x'_m can be expressed as

$$x'_m = x''_m + \gamma'_m \quad \gamma'_m \in [-\delta_{tracking,k}, \delta_{tracking,k}]$$

and $x''_m \in \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})$. Therefore

$$x_m = x''_m + \gamma''_m \tag{4.147}$$

for a $\gamma''_m \in [-2\delta_{tracking,k}, 2\delta_{tracking,k}]$. Since $x''_m \in \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})$ for each $m = 1, \dots, 2n$,

$$x'' \in \hat{\mathbf{D}}'_{y,k}(\delta_{tracking,k})^{2n} \tag{4.148}$$

Therefore, from Equation (4.145), $x'' \in D_j$ for some j . From the definition of D_j (Equation (4.114)), we have that

$$x'' = x_{dest_{(2n+1)j}} + \begin{bmatrix} g''_1 \\ \vdots \\ g''_{2n} \end{bmatrix} \tag{4.149}$$

for some $g''_1, \dots, g''_{2n} \in [-\delta, \delta]$. Combining Equations (4.147), (4.148) and (4.149)

$$x = x_{dest_{(2n+1)j}} + \begin{bmatrix} \gamma''_1 \\ \vdots \\ \gamma''_{2n} \end{bmatrix} + \begin{bmatrix} g'_1 \\ \vdots \\ g'_{2n} \end{bmatrix} = x_{dest_{(2n+1)j}} + \begin{bmatrix} g_1 \\ \vdots \\ g_{2n} \end{bmatrix}$$

for some $g_1, \dots, g_{2n} \in [-\delta - 2\delta_{tracking,k}, \delta + 2\delta_{tracking,k}]$. Therefore, from the definition of G_{x_j} in Equation (4.84), it follows that $x \in G_{x_j}$. Therefore Equation (4.140) holds.

Proof of 3:) For each $j = 0, \dots, J$, let

$$\begin{aligned}
\mathbf{Z}_{2n} &= x_{t_{(2n+1)j}}, \dots, x_{t_{(2n+1)j+2n}} \\
p_0, \dots, p_{2n} &= u_{t_{(2n+1)j}}, \dots, u_{t_{(2n+1)j+2n}} \\
g &= U \\
\hat{g}_j &= \hat{g} \\
\mathbf{G} &= G_{x_j} \\
N &= 2n
\end{aligned}$$

Now we would like to apply the results of Corollary 4.3.2 with the definitions above. In order to do so we must show

$$\begin{bmatrix} x_{t_{(2n+1)j}} & \cdots & x_{t_{(2n+1)j+2n}} \\ 1 & \cdots & 1 \end{bmatrix} \text{ is invertible} \quad (4.150)$$

and find

- (i) d_u such that Equation (4.35) is satisfied for $\mathbf{G} = G_{x_j}$.
- (ii) d such that Equation (4.28) is satisfied for $m = 2n$.
- (iii) ϵ_1 such that Equation (4.21) is satisfied for $z_i = x_{t_{(2n+1)j+i}}$.
- (iv) L_1 such that Equation (4.23) is satisfied.
- (v) d_l such that Equation (4.33) is satisfied for $N = 2n$ and $\mathbf{G} = G_{x_j}$.

Equation (4.150) follows directly from Equation (4.78).

(i) Since we let $G = G_{x_j}$ a choice of

$$d_u = \sqrt{2n}(2\delta + 4\delta_{tracking_k})$$

and Equation (4.86) imply Equation (4.35)

(ii) Equation (4.85) in Theorem (4.3.8) implies that

$$G_{x_j} \subseteq \text{Co}(x_{t_{(2n+1)j}}, \dots, x_{t_{(2n+1)j+2n}}).$$

and Equation (4.86) holds. Therefore, Equation (4.28) is satisfied by letting

$$d = \sqrt{2n}(2\delta + 4\delta_{tracking_k})$$

(iii) Since our measurements are perfect (see Sampling Procedure), $\epsilon_1 = 0$, which satisfies Equation (4.21).

(iv) Equation (4.118) implies that for $L_1 = L_x$ Equation (4.23) holds.

(v) Equation (4.79) implies that Equation (4.33) holds for

$$d_l = \frac{\sqrt{n(n+1)}}{2n+1}$$

Therefore, the results of Corollary (4.3.2) hold. Using the notations in from Equations (4.126) and (4.129)

$$|\hat{g}_j(x) - U(x)| \leq C_u \Delta^2 \delta_{u,k}^2 \quad (4.151)$$

for each $x \in G_{x_j}$. We now choose \hat{U}_{k+1} as follows:

$$\hat{U}_{k+1}(x) = \hat{g}_{\min_j}(x) \text{ where } x \in G_{x_j} \quad (4.152)$$

Since Equation (4.140) holds, $\hat{U}_{k+1}(x)$ is defined for all $x \in \hat{\mathbf{D}}_{y,k+1}^{2n}$. Using Equations (4.123) and (4.134), Equation (4.151) can be rewritten as

$$|\hat{U}_{k+1}(x) - U(x)| \leq \delta_{u,k+1}$$

for all $x \in \hat{\mathbf{D}}_{y,k+1}^{2n}$.

Proof of 4:) For each $j = 0, \dots, J$, let

$$\begin{aligned}
\mathbf{Z}_{2n} &= z_{t_{(2n+1)j}}, \dots, z_{t_{(2n+1)j+2n}} \\
p_0, \dots, p_{2n} &= y_{t_{(2n+1)j+n}}, \dots, y_{t_{(2n+1)j+2n+n}} \\
g &= f \\
\hat{g}_j &= \hat{g} \\
\mathbf{G} &= G_{z_j} \\
N &= 2n
\end{aligned}$$

Now we would like to apply the results of Corollary 4.3.2 with the definitions above. In order to do so we must show

$$\begin{bmatrix} z_{t_{(2n+1)j}} & \cdots & z_{t_{(2n+1)j+2n}} \\ 1 & \cdots & 1 \end{bmatrix} \text{ is invertible} \quad (4.153)$$

and find

- (i) d_u such that Equation (4.35) is satisfied for $\mathbf{G} = G_{z_j}$.
- (ii) d such that Equation (4.28) is satisfied for $m = 2n$.
- (iii) ϵ_1 such that Equation (4.21) is satisfied for $z_i = z_{t_{(2n+1)j+i}}$.
- (iv) L_1 such that Equation (4.23) is satisfied.
- (v) d_l such that Equation (4.33) is satisfied for $N = 2n$ and $\mathbf{G} = G_{z_j}$.

Equation (4.153) follows directly from Equation (4.81).

(i) Since we let $G = G_{z_j}$ a choice of

$$d_u = 8nN_u\sqrt{2n}(\delta + 3\delta_{tracking_k})$$

and Equation (4.89) imply Equation (4.35).

(ii) Equation (4.88) in Theorem 4.3.8 implies that

$$G_{z_j} \subseteq \text{Co}(z_{t_{(2n+1)j}}, \dots, z_{t_{(2n+1)j+2n}}).$$

and Equation (4.89) holds. Therefore, Equation (4.28) is satisfied by letting

$$d = 8nN_u\sqrt{2n}(\delta + 3\delta_{tracking_k})$$

(iii) Since our measurements are perfect (see Sampling Procedure), $\epsilon_1 = 0$, which satisfies Equation (4.21).

(iv) Equation (4.117) implies that for $L_1 = L_z$ Equation (4.23) holds.

(v) Equation (4.82) implies that Equation (4.33) holds for

$$d_l = \frac{\sqrt{n(n+1)}}{2n+1}$$

Therefore, the results of Corollary 4.3.2 hold. Using the notations in Equations (4.127), (4.128) and (4.129)

$$|\hat{g}_j(z) - f(z)| \leq \frac{C_f \Delta^2}{\gamma^2} \delta_{f,k}^2 \quad (4.154)$$

for each $z \in G_{z_j}$. We now choose \hat{f}_{k+1} as follows:

$$\hat{f}_{k+1}(z) = \hat{g}_{\min_j}(z), \text{ where } z \in G_{z_j} \quad (4.155)$$

Using Equations (4.124) and (4.135), Equation (4.154) can be rewritten as

$$|\hat{f}_{k+1}(z) - U(z)| \leq \delta_{f,k+1} \quad (4.156)$$

for all $z \in \bigcup_{j=0,\dots,J} G_{z_j}$. Let

$$z = \begin{bmatrix} u_t \\ w_t \end{bmatrix} \in \hat{\mathbf{D}}_{u,k+1}(w_t) \times \hat{\mathbf{D}}_{y,k+1}^{2n-1}$$

Therefore, by Equation (4.122), z can be written

$$z = \begin{bmatrix} U(y_{t+n}, w_t) \\ w_t \end{bmatrix}$$

where $w_t \in \hat{\mathbf{D}}_{y,k+1}^{2n-1}$ and $y_{t+n} \in \hat{\mathbf{D}}_{y,k+1}$. Therefore,

$$x = \begin{bmatrix} y_{t+n} \\ w_t \end{bmatrix} \in \hat{\mathbf{D}}_{y,k+1}^{2n}$$

From Equation (4.140) $x \in G_{x_j}$ for some j . Thus Equation (4.90) holds and $z \in G_{z_j}$.

Thus Equation (4.156) holds for all

$$z = \begin{bmatrix} u \\ w_t \end{bmatrix} \in \hat{\mathbf{D}}_{u,k+1}(w_t) \times \hat{\mathbf{D}}_{y,k+1}^{2n-1}$$

and Part 4: is shown.

□

4.3.5 Proof of Problem Statement

We now have all the tools to complete the proof of Theorem 4.2.1.

Proof:

Let

$$\begin{aligned}\delta_u &= \frac{1}{2C_u\Delta^2} \\ \delta_f &= \frac{\gamma^2}{2C_f\Delta^2} \\ C &= (\alpha + 2)\Delta\end{aligned}\tag{4.157}$$

where

$$\begin{aligned}C_u &= (2^{2n} - 1) \left(\frac{3L_x n(2n+1)\sqrt{2n}}{2\sqrt{n(n+1)}} \right) (\alpha + 2)^3 + 6nL_x(\alpha + 2)^2 \\ C_f &= (2^{2n} - 1) \left(\frac{256n^3 L_z N_u^3 (2n+1)\sqrt{2n}}{\sqrt{n(n+1)}} \right) (\alpha + 3)^3 + 48n^3 L_z N_u^3 (\alpha + 3)^2 \\ \gamma &= \frac{\delta_{f,k}}{\delta_{u,k}} \\ \Delta &= N_{fu} + \frac{N_{fu}N_u}{N_f} ((N_f + 1)^{n-1} - 1)\gamma \\ \alpha &= \max(10n^2, 40n^3 N_u N_{fu} + N_f(2n - 1) + 2)\end{aligned}$$

and N_u , N_{fu} and N_f are those satisfying Equation (4.42). In order to apply Theorem 4.3.10, we need to show that

$$\begin{aligned}\delta_{u,k} &\leq \delta_u \\ \delta_{f,k} &\leq \delta_f \\ \max_{y,y' \in \hat{\mathbf{D}}_{y,k}} |y - y'| &\geq C\delta_u\end{aligned}\tag{4.158}$$

where for any k , the quantities $\delta_{u,k}$, $\delta_{f,k}$ and $\hat{\mathbf{D}}_{y,k}$ satisfy

$$\begin{aligned}|\hat{U}_k(y_{t+n}, w_t) - U(y_{t+n}, w_t)| &\leq \delta_{u,k} \\ |\hat{f}_k(u_t, w_t) - f(u_t, w_t)| &\leq \delta_{f,k}\end{aligned}$$

for

$$\begin{aligned}x &\in \hat{\mathbf{D}}_{y,k}^{2n} \\ z &\in \hat{\mathbf{D}}_{y,k}^{2n-1} \times \hat{\mathbf{D}}_{u,k}(w_t) \\ \hat{\mathbf{D}}_{u,k}(w_t) &= \{u | u = U(y_{t+n}, w_t), y_{t+n} \in \hat{\mathbf{D}}_{y,k}\}\end{aligned}$$

In the statement of Theorem 4.2.1 Equations (4.158) is assumed to be satisfied for $k = 0$.

Assume that Equations (4.158) hold for $k = j$. Then, after acquiring the samples of Equation (4.146) and constructing the new approximations \hat{U}_{j+1} and \hat{f}_{j+1} of Equations (4.152) and (4.155), we have from Theorem 4.3.10 (Equations (4.134), (4.135) and (4.136)) that

$$\begin{aligned}\delta_{u,j+1} &= \frac{\delta_{u,j}}{2} \\ \delta_{f,j+1} &= \frac{\delta_{f,j}}{2} \\ \hat{\mathbf{D}}_{y,j+1} &\subset \hat{\mathbf{D}}_{y,j}\end{aligned}\tag{4.159}$$

Since Equations (4.158) hold for $k = j$, Equations (4.159) imply that they hold for $k = j + 1$ as well. Hence we have

$$\begin{aligned}\delta_{u,j+1} &\leq \delta_u \\ \delta_{f,j+1} &\leq \delta_f \\ \max_{y,y' \in \hat{\mathbf{D}}_{y,j+1}} |y - y'| &\geq C\delta_u\end{aligned}\tag{4.160}$$

Equations (4.160) implies that Theorem 4.3.10 can be applied once again, with $k = j + 1$.

Hence we have that the conditions of Theorem 4.3.10 hold for all $k \geq 0$ if they are valid for $k = 0$. Since the latter is assumed to be true in the statement of Theorem 4.2.1, we have that for all $k \geq 0$,

$$\begin{aligned}\delta_{u,k+1} &= \frac{\delta_{u,k}}{2} \\ \delta_{f,k+1} &= \frac{\delta_{f,k}}{2} \\ \hat{\mathbf{D}}_{y,k+1} &= \{y \mid |y - y'| \leq \delta_{tracking,k}, y' \in \hat{\mathbf{D}}_{y,k}\} \\ \delta_{tracking,k} &= a\delta_{u,k} + b\delta_{f,k}\end{aligned}\tag{4.161}$$

where a and b are given by the right hand side of Equation (4.49).

We make use of Equations (4.161) in two distinct phases. In the first phase we use these results to extend the boundaries of $\hat{\mathbf{D}}_{y,k}$ to $\hat{\mathbf{D}}_{y,N}$, the target set. In the second phase, we repeatedly apply the inequalities to reduce the errors $\delta_{u,0}$ and $\delta_{f,0}$ to $\delta_{u,N}$

and $\delta_{f,N}$ respectively, as described in Theorem 4.2.1

Phase 1: Since $\delta_{u,k+1}$ and $\delta_{f,k+1}$ are upper bounds satisfying the inequalities of Equations (4.120) and (4.121) with $k+1$, we can always redefine these terms as more conservative values as long as the conditions of Equations (4.158) are satisfied with k replaced by $k+1$. In particular, let

$$\begin{aligned}\delta_{u,k+1} &= \delta_{u,k} \\ \delta_{f,k+1} &= \delta_{f,k}\end{aligned}$$

Therefore it is clear from Equation 4.49 that $\delta_{tracking_{k+1}} = \delta_{tracking_k}$ and if we then apply Theorem 4.3.10 p times with initial errors of $\delta_{u,0}$ and $\delta_{f,0}$, we get

$$\begin{aligned}\delta_{u,p} &= \delta_{u,0} \\ \delta_{f,p} &= \delta_{f,0} \\ \delta_{tracking_p} &= \delta_{tracking_0}\end{aligned}\tag{4.162}$$

If we begin with set $\hat{\mathbf{D}}_{y,0}$, after p applications of Equation (4.161) we arrive at

$$\hat{\mathbf{D}}_{y,p} = \{y \mid |y - y'| \leq p\delta_{tracking,0}, \quad y \in \hat{\mathbf{D}}_{y,0}\}$$

Since each iteration extends the boundary of $\hat{\mathbf{D}}_{y,0}$ by a constant discrete number, we can iterate the application of Theorem 4.3.10 until $\hat{\mathbf{D}}_{y,p}$ contains $\hat{\mathbf{D}}_{y,N}$.

Phase 2: Referring to Equation (4.161), we note that $\hat{\mathbf{D}}_{y,k+1} \supset \hat{\mathbf{D}}_{y,k}$. Therefore, just as we chose conservative $\delta_{u,k+1}$ and $\delta_{f,k+1}$ in order to extend the boundaries of $\hat{\mathbf{D}}_{y,k}$ as we iterate the results of Theorem 4.3.10, we can chose $\hat{\mathbf{D}}_{y,k+1} = \hat{\mathbf{D}}_{y,k}$ as a more conservative choice in order to reduce the error with each iteration of the theorem, as long as Equations (4.158) are satisfied with k replaced by $k+1$. Similarly, since $\hat{\mathbf{D}}_{y,p} \supset \hat{\mathbf{D}}_{y,N}$, we redefine $\hat{\mathbf{D}}_{y,p} = \hat{\mathbf{D}}_{y,N}$. Therefore, Equation (4.161) becomes

$$\begin{aligned}\delta_{u,k+1} &= \frac{\delta_{u,k}}{2} \\ \delta_{f,k+1} &= \frac{\delta_{f,k}}{2} \\ \hat{\mathbf{D}}_{y,k+1} &= \{y \mid |y - y'| \leq \delta_{tracking,k}, \quad y' \in \hat{\mathbf{D}}_{y,k}\}\end{aligned}\tag{4.163}$$

with the initial conditions

$$\begin{aligned}\delta_{u,p} &= \delta_{u,0} \\ \delta_{f,p} &= \delta_{f,0} \\ \hat{\mathbf{D}}_{y,p} &= \hat{\mathbf{D}}_{y,N}\end{aligned}$$

After m iterations of Equation (4.161), we get

$$\begin{aligned}\delta_{u,p+m} &= \frac{\delta_{u,p}}{2^m} \\ \delta_{f,p+m} &= \frac{\delta_{f,p}}{2^m} \\ \hat{\mathbf{D}}_{y,p+m} &= \hat{\mathbf{D}}_{y,N}\end{aligned}$$

we can always chose m large enough such that

$$\begin{aligned}\delta_{u,p+m} &\leq \delta_{u,N} \\ \delta_{f,p+m} &\leq \delta_{f,N} \\ \hat{\mathbf{D}}_{y,p+m} &= \hat{\mathbf{D}}_{y,N}\end{aligned}\tag{4.164}$$

Let $N = p + m$. Since by $p + m$ iterations of sampling, generating \hat{f} and \hat{U} , and applying the results of Theorem 4.3.10, Equations (4.164) hold, we have proven Theorem 4.2.1.

Chapter 5

Implementation

5.1 Helicopter Dynamics

Helicopter dynamics include the helicopter body dynamics as well as main rotor and flybar dynamics. The main inputs to a standard helicopter configuration include throttle, collective, roll cyclic, pitch cyclic and rudder. Collective, roll cyclic and pitch cyclic refer to the pitch of the main rotor blades. Rudder refers to the thrust of the tail rotor. Refer to Figure 5-1 for the specific geometries. In order to simplify the dynamics, the helicopter model will be generated not in terms of these inputs, but in terms of the actions these inputs have on the helicopter. Thus actuator dynamics are ignored. These actions are the main rotor thrust magnitude and direction, and the tail rotor thrust. A further simplification includes the assumption that the helicopter is limited to motion in a vertical plane. Thus only the longitudinal dynamics are modeled.

The dynamics are derived in a continuous time state space form, and then a first order approximation of these dynamics is taken to provide the discrete time model used in simulation. A discrete time model is employed, and is the focus of the results presented in Chapter 4 because any learning controller is necessarily implemented on a digital computer. In addition, sampling must occur at a finite rate. In fact the period length of sampling, as specified by the sensors, is chosen as the time step for the discrete model. Thus the dynamics of the computer and the sensors are directly

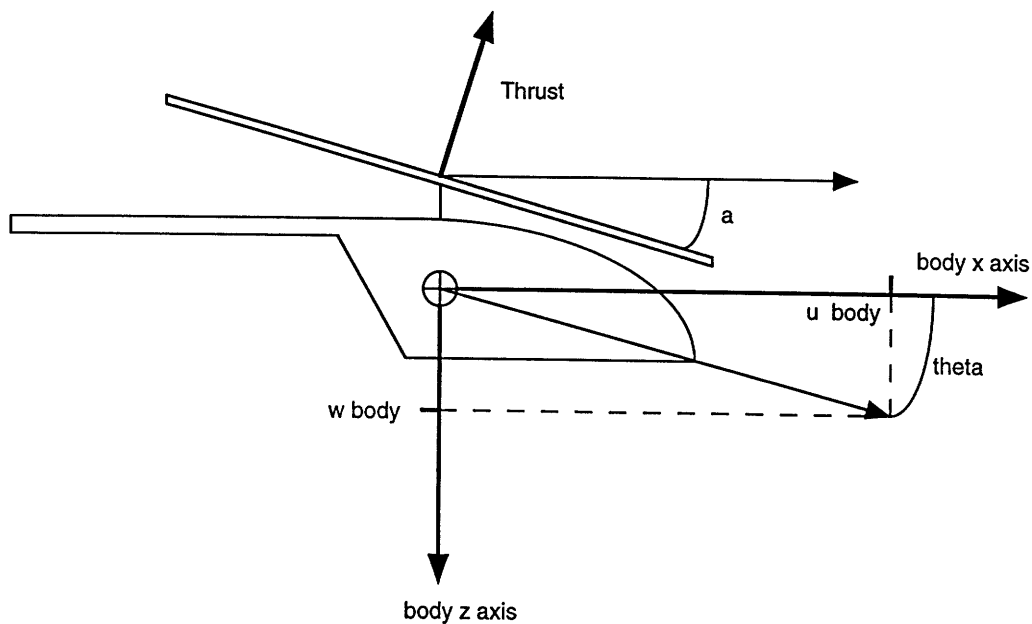


Figure 5-1: Helicopter Model

accounted for in the model.

The continuous time, longitudinal model, is presented in its entirety below.

$$\begin{aligned} \dot{u} &= \frac{F_{xb}}{m} - qw \\ \dot{w} &= \frac{F_{zb}}{m} - qu \\ \dot{q} &= \frac{M_\theta}{I_y} \\ \dot{\theta} &= q \end{aligned}$$

The terms F_{xb} , F_{zb} , and M_θ are functions of the states and inputs. The F terms are body axis forces and the M term is the pitching moment. Table 5.1 lists the states, the controls and their meanings.

The force and moment terms are written

u	forward velocity in the body frame
w	vertical velocity in the body frame
q	angular pitch rate
θ	pitch
a	main rotor flapping angle
T	thrust

Table 5.1: Longitudinal States and Controls

m	vehicle mass	0.70 slugs
I_y	Moment of inertia about y-axis	0.20 slugs- ft^2
ρ	atmospheric density	0.002377 slugs/ ft^2
$X_{uu,fus}$	Fuselage drag coefficient along x axis	-0.3 ft^2
$Z_{ww,fus}$	Fuselage drag coefficient along z axis	-1.3 ft^2
$Z_{ww,ht}$	Horizontal tail drag coefficient along z axis	-0.12 ft^2
x_{mr}	x position of main rotor from center of mass	0 ft
z_{mr}	z position of main rotor from center of mass	-0.688 ft
x_{fus}	Aerodynamic center of fuselage along x axis	0.17 ft
z_{fus}	Aerodynamic center of fuselage along z axis	0 ft
x_{ht}	Aerodynamic center of horizontal tail along x axis	-2.0 ft
z_{ht}	Aerodynamic center of horizontal tail along z axis	-0.12 ft

Table 5.2: Parameter Definitions and Values

$$\begin{aligned}
 F_{xb} &= -Ta + \frac{\rho}{2} X_{uu,fus} |u| u - mg \sin \theta \\
 F_{zb} &= -T + \frac{\rho}{2} (Z_{ww,fus} + Z_{ww,ht}) |w| w + mg \cos \theta \\
 M_{\theta} &= -Tx_{mr} + Taz_{mr} + \frac{\rho}{2} [(Z_{ww,fus}x_{fus} + Z_{ww,ht}x_{ht}) |w| w - X_{uu,fus} |u| uz_{fus}]
 \end{aligned}$$

Table 5.1 lists all the parameters and typical values for a small model helicopter.

The discrete time model can not be derived from the continuous model provided above. Since the sampling rate for the sensors on the actual helicopter is approximately 50 Hz, the time step Δt is chosen as .02 seconds. The time derivatives are

approximated by first order differences. For example

$$\dot{x}(t) \approx \frac{x(t + \Delta t) - x(t)}{\Delta t}$$

Making this approximation for the dynamics above, and solving for the states at time $t + \Delta t$ in terms of the states at time t yields.

$$\begin{aligned} u(t + \Delta t) &= \Delta t \left(\frac{F_{xb}(t)}{m} - q(t)w(t) \right) + u(t) \\ w(t + \Delta t) &= \Delta t \left(\frac{F_{zb}(t)}{m} - q(t)u(t) \right) + w(t) \\ q(t + \Delta t) &= \Delta t \left(\frac{M_\theta(t)}{I_y} \right) + q(t) \\ \theta(t + \Delta t) &= \Delta t q(t) + \theta(t) \end{aligned} \tag{5.1}$$

where

$$\begin{aligned} F_{xb}(t) &= -T(t)a(t) + \frac{\rho}{2}X_{uu, fus} |u(t)| u(t) - mg \sin \theta(t) \\ F_{zb}(t) &= -T(t) + \frac{\rho}{2}(Z_{ww, fus} + Z_{ww, ht}) |w(t)| w(t) + mg \cos \theta(t) \\ M_\theta(t) &= -T(t)x_{mr} + T(t)a(t)z_{mr} + \frac{\rho}{2}[(Z_{ww, fus}x_{fus} + Z_{ww, ht}x_{ht}) |w(t)| w(t) \\ &\quad - X_{uu, fus} |u(t)| u(t)z_{fus}] \end{aligned}$$

5.1.1 Zero Dynamics and Feedback Linearization

The strategies employed are all applications of feedback linearization, each making different assumptions about the extent of *a priori* knowledge of the model, and some employing current learning and adaptive strategies. Feedback linearization, though, can not be directly applied, since the zero dynamics of the system under the linearizing input are unstable. To see this, let us find the zero dynamics of the above system with outputs (u, w) . Recall that the zero dynamics refers to the behavior of the unobservable states when the feedback linearizing controller holds the outputs at an equilibrium point. Let us take this equilibrium point to be the origin. Solving for

the inputs (T, a) we get

$$T = mg \cos(\theta)$$

$$a = -\tan(\theta)$$

The zero dynamics are thus

$$q(t + \Delta t) = -x_{mr} mg \cos \theta - z_{mr} mg \sin \theta$$

$$\theta(t + \Delta t) = \Delta t q(t) + \theta(t)$$

The origin of the above system is clearly not asymptotically stable, since trajectories extend off into infinity. In fact, any pair of states treated as outputs would yield unstable zero dynamics.

To solve the problem of unstable zero dynamics, we chose as outputs (w, q) , the states which are most directly influenced by the inputs, and we wrap a stabilizing loop around the feedback linearized system which limits the $(w_{desired}, q_{desired})$ commands to those commands which maintain bounded signals and allows us to track the helicopter's earth frame velocities,

$$u_e(t) = u(t) \cos \theta(t) + w(t) \sin \theta(t)$$

$$w_e(t) = w(t) \cos \theta(t) - u(t) \sin \theta(t)$$

This outer loop is designed using linear methodologies, but is robust enough to be valid for a large set of commanded (u_e, w_e) . In fact, it is robust for velocities in excess of $100 \frac{ft}{s}$. Such velocities are quite fast for the small model helicopter under consideration. Now, let's go through the procedure of solving the control problem in the manner described. The first step is to find the feedback linearizing controller which takes any state to any desired (w, q) in one time step. This function is not

difficult to find, and the function exists in any set not containing $T = 0$.

$$\begin{aligned}
T(t) &= \frac{m}{\Delta t}(w(t) - w(t)_{des}) + m(g \cos \theta(t) - q(t)u(t)) \\
&\quad + \frac{\rho}{2}(Z_{ww, fus} + Z_{ww, ht}) |w(t)| w(t) \\
a(t) &= \frac{I_y}{T(t)z_{mr} \Delta t}(q(t)_{des} - q(t)) + \frac{x_{mr}}{z_{mr}} \\
&\quad + \frac{\rho}{2T(t)z_{mr}} X_{uu, fus} z_{fus} |u(t)| u(t) \\
&\quad - \frac{\rho}{2T(t)z_{mr}} (Z_{ww, fus} x_{fus} + Z_{ww, ht} x_{ht}) |w(t)| w(t)
\end{aligned}$$

Applying these solutions to the system, and letting our new inputs be (w_{des}, q_{des}) yields the new system

$$\begin{aligned}
w(t+1) &= w(t)_{des} \\
q(t+1) &= q(t)_{des} \\
u(t+\Delta t) &= \Delta t \left(\frac{F_{xb}(t)}{m} - q(t)w(t) \right) + u(t) \\
\theta(t+1) &= \Delta t q(t) \theta(t)
\end{aligned}$$

We proceed now by designing the outer loop which defines w_{des} and q_{des} in terms of the desired earth frame velocities. To do so, let us first linearize the transformed system about the hover condition. The helicopter is in hover when it is at zero pitch attitude and has no velocity or angular velocity. Therefore, hover is the origin of the state space. Carrying out the linearization yields.

$$\begin{bmatrix} u_e(t+1) \\ w_e(t+1) \\ q(t+1) \\ \theta(t+1) \end{bmatrix} = A \begin{bmatrix} u_e(t) \\ w_e(t) \\ q(t) \\ \theta(t) \end{bmatrix} + B \begin{bmatrix} w_{des} \\ q_{des} \end{bmatrix}$$

where

$$A = \begin{bmatrix} 1 & 0 & \frac{I_y}{mz_{mr}} & -\Delta t g \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \Delta t & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & -\frac{I_y}{mz_{mr}} \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

The feedback law

$$\begin{bmatrix} w(t)_{des} \\ q(t)_{des} \end{bmatrix} = K(u(t), w(t), q(t), \theta(t), w(t)_{e,des}, u(t)_{e,des})$$

where the newest control inputs are $w(t)_{e,des}$ and $u(t)_{e,des}$, was then selected as linear function stabilizing the linear system and providing qualitatively attractive transient responses. After some design effort, the function K for the particular parameter values given in Table 5.1 was chosen as

$$K(t) = \begin{bmatrix} 0.1w_e(t) + 0.9w(t)_{e,des} \\ 0.1q(t) + \frac{.4}{\Delta t} \left(\frac{u_e(t) - u(t)_{e,des}}{32.2} - \theta \right) \end{bmatrix}$$

This completes the derivation of the feedback linearizing controller and its stabilizing outer loop controller. The outer loop K will remain unchanged for each of the control strategies except the new learning strategy proposed in this thesis.

5.1.2 Simplified Model

These new learning strategy of this thesis requires that the zero dynamics of the feedback linearized system be asymptotically stable. This is clearly not the case for the complete system shown above. Therefore, a more tractable control problem, based on

a simplified model, This model assumes that the $T(t)$ transformation which linearizes the $w(t)$ dynamics is known, and any remaining design effort and uncertainty lies exclusively in the $q(t)$ dynamics. If we let $A(t) \triangleq T(t)a(t)$, then the new system can be written

$$\begin{aligned} w(t+1) &= w(t) + v(t) \\ q(t+1) &= q(t) + \frac{\Delta t}{I_y}(-.688 * A(t) + 2.59 \times 10^{-5} |w| w) \end{aligned} \tag{5.2}$$

This system has two inputs $(v(t), A(t))$ and two outputs $(w(t), q(t))$, and therefore has no zero dynamics. In addition, it is clear that the system is already input-output decoupled.

5.2 Control Applications

In addition to the new learning strategy of this thesis, results are presented for a few competing algorithms. These include the perfect feedback linearizing controller already developed in Section 5.1.1, as well as a feedback linearizing controller based on a linear approximation of the helicopter, an adaptive control strategy and a neural network based strategy.

5.2.1 Feedback Linearizing Controller Based on a Linear Model

As a second example, assume that the engineer designing the controller possesses significantly less knowledge about the helicopter dynamics. Assume that his/her knowledge is limited to the linearization of 5.1 about hover. Carrying out this linearization

yields

$$\begin{bmatrix} u(t+1) \\ w(t+1) \\ q(t+1) \\ \theta(t+1) \end{bmatrix} = A \begin{bmatrix} u(t) \\ w(t) \\ q(t) \\ \theta(t) \end{bmatrix} + B \begin{bmatrix} T(t) \\ a(t) \end{bmatrix}$$

$$\begin{bmatrix} w(t) \\ q(t) \end{bmatrix} = C \begin{bmatrix} u(t+1) \\ w(t+1) \\ q(t+1) \\ \theta(t+1) \end{bmatrix}$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & -\Delta t g \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \Delta t & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & -\Delta t g \\ -\frac{\Delta t}{m} & 0 \\ -\frac{\Delta t x_{mr}}{I_y} & \frac{\Delta t g m z_{mr}}{I_y} \\ 0 & 0 \end{bmatrix}$$

$$E = \begin{bmatrix} 0 \\ \Delta t g \\ 0 \\ 0 \end{bmatrix}$$

The input-output model can then be written.

$$\begin{bmatrix} w(t+1) \\ q(t+1) \end{bmatrix} = CA \begin{bmatrix} u(t) \\ w(t) \\ q(t) \\ \theta(t) \end{bmatrix} + CB \begin{bmatrix} T(t) \\ a(t) \end{bmatrix} + CE$$

If the matrix CB has an inverse, which it certainly does for the case we are considering, then the feedback linearizing transformation of the input is easy to find.

$$\begin{bmatrix} T(t+1) \\ a(t+1) \end{bmatrix} = (CB)^{-1} \left(\begin{bmatrix} w(t+1) \\ q(t+1) \end{bmatrix} - CA \begin{bmatrix} u(t) \\ w(t) \\ q(t) \\ \theta(t) \end{bmatrix} - CE \right)$$

An identical formulation can be found for the simplified system of Equation 5.2. In this case, the resulting transformation is

$$\begin{bmatrix} v(t+1) \\ A(t+1) \end{bmatrix} = (CB)^{-1} \left(\begin{bmatrix} w(t+1) \\ q(t+1) \end{bmatrix} - CA \begin{bmatrix} w(t) \\ q(t) \end{bmatrix} - CE \right)$$

where

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 \\ 0 & -\frac{\Delta t}{I_v} 0.688 \end{bmatrix}$$

$$E = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

5.2.2 Adaptive Strategy

As one might correctly assume, the performance of the feedback linearization controller based on the approximate linear model suffers because of the limited prior knowledge on which it is based. A designer can only expect good performance local to hover. One might not be able to obtain or stabilize about large velocities. A solution to this problem can be found in the Adaptive control literature. The designer wants a controller which holds the aircraft at some large forward velocity. He still assumes that the system is linear, but does not know what the linear model is. He would like to identify the model (the linearization about the prescribed forward velocity). To proceed, the designer treats the elements of the A , B and E tensors as parameters. Initial values are chosen, and the controller which is well defined by these parameters is used to fly the vehicle toward the prescribed velocities. Since the initial parameter values are not correct, there will be measurable discrepancies between the expected and actual trajectories of the helicopter. The basic idea behind adaptive control is to use these discrepancies to update the parameters. As a slight modification to this approach, the controller is differently parameterized. Such a parameterization is chosen because it takes on the form used for the neural network strategy described in the next section. This new form fixes the nominal controller, and adds the difference between the perfect controller and the initial nominal one. For the sake of generality, we let $x(t)$ be the state at time t , and $u(t)$ be the input.

$$U(t) = (C_{nom}B_{nom})^{-1}(y(t+1)_{des} - C_{nom}A_{nom}x(t) - E_{nom})\Theta z(t)$$

where Θ is the matrix of parameters, and

$$z(t) \triangleq \begin{bmatrix} y(t+1)_{des} \\ x(t) \\ 1 \end{bmatrix}$$

Initially, the value of Θ may not be known, so a guess of its value must be made. If we let this guess be $\hat{\Theta}$, then the controller is

$$\hat{U}(t) = (C_{nom}B_{nom})^{-1}(y(t+1)_{des} - C_{nom}A_{nom}x(t) - E_{nom})\hat{\Theta}(t)z(t)$$

Note that $\hat{\Theta}(t)$ is written as a function of time since it will be adapted as time progresses. The parameters $\hat{\Theta}(t)$ are updated from measurements of $U(t)$ and $z(t)$. If a linearly independent set of measurements is collected, a least squared rule can be applied. A different approach is taken in this paper. First, an error metric is defined.

$$e = (U(t) - \hat{U}(t))^T(U(t) - \hat{U}(t))$$

Now we find the gradient of this error with respect to the parameters.

$$\frac{\partial e}{\partial \hat{\Theta}} = -2(U(t) - \hat{U}(t))z^T$$

Finally, the parameters are made to evolve in the direction of the negative of the above gradient.

$$\hat{\Theta} = \hat{\Theta} + \rho(U(t) - \hat{U}(t))z^T$$

where ρ is the adaptation *rate*. The logic behind this choice is clear. The negative of the gradient describes the direction in $\hat{\Theta}$ which reduces e . If ρ is taken small, then error reduction can be guaranteed.

5.2.3 Neural Network Strategy

The neural network strategy proposed here is essentially the same as that proposed above. In fact, the parameterized controller can be written in nearly the same form.

$$\hat{U}(t) = (C_{nom}B_{nom})^{-1}(y(t+1)_{des} - C_{nom}A_{nom}x(t) - E_{nom})\hat{\Theta}(t)\beta(z(t))$$

The matrix Θ remains a matrix of parameters, and the definition of $z(t)$ has not changed. The difference is that $z(t)$ now acts through the vector function β of basis functions.

$$\beta(z(t)) = \begin{bmatrix} \beta_1(z(t)) \\ \vdots \\ \beta_N(z(t)) \end{bmatrix}$$

These basis functions, and their coefficients Θ are the structure referred to as the neural network. As mentioned in Chapter 1, a variety of choices of basis functions serve as Universal approximators. This implies that there exists a β and a Θ such that the difference between the perfect feedback linearizing controller $U(z(t))$ and the approximate neural network based controller $\hat{U}(z(t))$ is bounded over some compact set.

$$\|U(z(t)) - \hat{U}(z(t))\|_2 \leq \delta$$

for all $z(t) \in Z$, Z compact and any $\delta > 0$ and some Θ . The update rule is derived exactly as above for the adaptive controller. The parameters receive an additive term in the direction of the negative gradient of error with respect to the parameters.

$$e = (U(t) - \hat{U}(t))^T (U(t) - \hat{U}(t))$$

Now we find the gradient of this error with respect to the parameters.

$$\frac{\partial e}{\partial \hat{\Theta}} = -2(U(t) - \hat{U}(t))\beta(z)^T$$

Finally, the parameters are made to evolve in the direction of the negative of the above gradient.

$$\hat{\Theta} = \hat{\Theta} + \rho(U(t) - \hat{U}(t))\beta(z)^T$$

where ρ is the adaptation *rate*. Since the update rule may become unstable if training is continued even when the local controller error is smaller than the networks approximating capability, the parameters are not updated for small errors.

5.2.4 New Learning Strategy

Finally we reach the point where the primary results of this thesis are used to define a controller. As in the previous two sections, the algorithm here will be applied to the simplified system of Equation (5.2). In order to simplify the implementation of the learning algorithm in simulation, a simple change of variables is made. Let the new states be w and $r = q \times 10^3$.

$$\begin{aligned}w(t+1) &= w(t) + v(t) \\r(t+1) &= r(t) + \frac{\Delta t}{I_y}(-688 * A(t) + 2.59 \times 10^{-2} |w| w)\end{aligned}\tag{5.3}$$

In order to implement the algorithm of Chapter 4 we must find the constants N_{fu} , N_u , N_f , L_x and L_z satisfying Equations (4.42), (4.117) and (4.118). These are listed below

$$\begin{aligned}N_{fu} &= \frac{688\Delta t}{I_y} \\N_u &= \frac{I_y}{688\Delta t} \\N_f &= 1 \\L_x &= 386 \\L_z &= 2.59 \times 10^{-3}\end{aligned}$$

Note that since the delay of the system equals 1, we do not need an \hat{f} in order to generate the control of Equation (4.50). Therefore we only need to design an initial \hat{U}_0 . An obvious choice is the linear design of Section 5.2.1. To make the problem more challenging, let us also add some error to this \hat{U} . Simulations are made for various additive errors e .

$$\begin{aligned}
\hat{U}_0 &= \begin{bmatrix} v(t+1) \\ A(t+1) \end{bmatrix} \\
&= (CB)^{-1} \left(\begin{bmatrix} w(t+1)_{des} \\ r(t+1)_{des} \end{bmatrix} - A \begin{bmatrix} w(t) \\ r(t) \end{bmatrix} \right) + \begin{bmatrix} 0 \\ e \end{bmatrix} \begin{bmatrix} w(t+1)_{des} \\ r(t+1)_{des} \\ w(t) \\ r(t) \end{bmatrix}
\end{aligned}$$

where

$$\begin{aligned}
A &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
B &= \begin{bmatrix} 1 & 0 \\ 0 & -\frac{\Delta t}{I_y} 688 \end{bmatrix}
\end{aligned}$$

Note that the assumption of perfect knowledge of the $w(t)$ dynamics is made, reducing the problem to learning the correct $q(t)$ dynamics. Now particular $\delta_{u,0}$ must be found for all

$$x \triangleq \begin{bmatrix} w(t+1)_{des} \\ r(t+1)_{des} \\ w(t) \\ r(t) \end{bmatrix} \in \hat{\mathbf{D}}_{y,0}^4$$

where we let $\hat{\mathbf{D}}_{y,0} = [a, b]$ as our region of initial knowledge. This $\delta_{u,0}$ can be found as a function of the error vector e and the parameters a and b .

$$\delta_{u,0} = 4 \times 10^{-5} (b - a)^2 + |e|_{\infty} (b - a)$$

The only step remaining is to find values for a , b and δ_0 for which the algorithm can

be applied. In particular, Theorem 4.2.1 states that we must satisfy

$$\begin{aligned} b - a &\geq C\delta_{u,0} \\ \delta_{u,0} &\leq \delta_u \end{aligned} \tag{5.4}$$

Calculating the C and δ_u of Equations (4.157) for the above system of Equation (5.3) with $\Delta t = .02$ we get

$$\begin{aligned} C &= 136 \\ \delta_u &= .008 \end{aligned} \tag{5.5}$$

Any a , b and e such that Equations (5.4) are satisfied with C and δ_u of Equation (5.5) are sufficient.

Now all that remains is to define the error bounds we would like to achieve, and the set over which we would like these bounds to hold. This choice is made to guarantee that the controller is valid for commanded earth frame velocities on the order of one hundred feet per second. Let the conditions after N iterations of the algorithm be

$$\begin{aligned} \hat{\mathbf{D}}_{y,N} &= [-100, 100] \\ \delta_{u,N} &= \delta_u \end{aligned}$$

We therefore are not asking to improve error, just expand the region over which our controller is valid and guarantee that the new errors do not increase.

All the above definitions and conditions are sufficient to define the learning controller presented in Chapter 4.

Chapter 6

Results

There are several topics which must be addressed in order to gain a complete picture about the relative merits of each of the controllers presented in Chapter 5. The most important of these are the performance of the controller, the required design effort, and the computational and memory requirements needed to implement each controller. Table 6 provides a quick summary of these issues, and the following sections discuss them in greater depth.

6.1 Controller Performance

Let us first compare the performance of each of the controllers. The most important measure we have of controller performance is tracking error. Since the feedback lin-

i	Tracking Error (rad/s)	Performance Guarantees	Computational Effort	Required Prior Knowledge
Perfect	zero	yes	low	high
Linear	4×10^{-2}	no	low	low
Adaptive	10^{-6}	no	moderate	low
Neural	10^{-5}	no	high	low
Learning	10^{-6}	yes	high	low

Table 6.1: Relative Strengths of Various Control Strategies

earization algorithms apply only to the inner loop controller, the performance of each of the strategies is compared by comparing the intermediate signal $|q(t) - q(t)_{desired}|$. We must also insure that given the complete controller, including the outer stabilizing loop, also performs satisfactorily. Therefore plots of $(w_{earth} - w_{earth,desired})$ and $(u_{earth} - u_{earth,desired})$ are also presented. These errors are less significant since they contain little information about the performance of the inner loop. They only insure that the capabilities of the inner loop are sufficient to provide a complete stable system. For each simulation run, plots of controller error are also presented. Since it was shown in Theorem 4.3.5 that worst case tracking error is proportional to controller error (and vice versa), only tracking error will be referenced. The controller error is shown for completeness.

The simulations whose results are presented are for an extreme case where commanded velocities and artificially added controller errors are large. For cases of small velocities and errors, each of the controllers perform sufficiently well and the differences are nominal at best. The particular case chosen for simulation is for

$$\begin{aligned} u_{earth,desired} &= 100 \\ w_{earth,desired} &= 20 \\ e &= [.01 .01 .01 .01] \end{aligned}$$

The error term e has been described in Section 5.2.4. It is an artificially added error term to the nominal controllers employed by all but the perfect feedback linearizing controller.

Since all of the controllers are essentially approximations to the perfect feedback linearizing controller, it is clear that the benchmark for any performance judgment is the perfect feedback linearizing controller of Section 5.1.1. This is the case where commanded $q(t)_{des}$ is achieved exactly by the system and controller error is zero. The outer loop response for this case can be found in Figure 6-1. Be reminded that any errors result from the outer loop. Simulation results for the remaining cases are

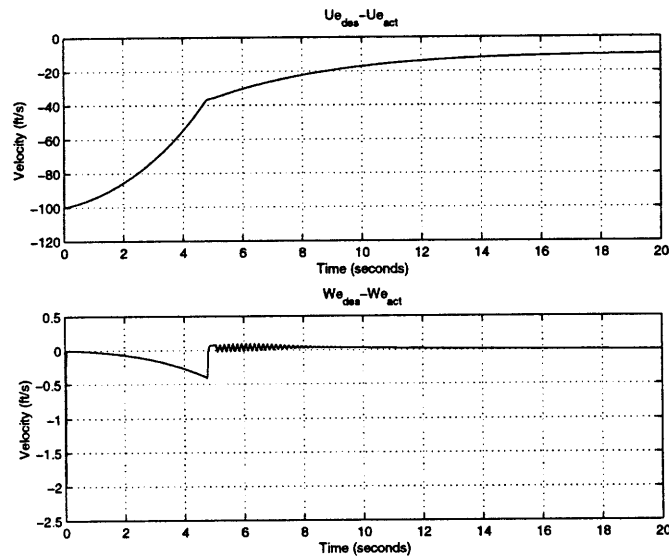


Figure 6-1: Perfect Feedback Linearizing Controller

included in Figures 6-2, 6-3, 6-4 and 6-5.

The first, and simplest alternate controller implemented on the helicopter is the feedback linearizing controller based on an approximate linear model of the system. This controller clearly suffers from difficulties not experienced by any of the other implementations. We can readily see from the plots of Figure 6-2 that tracking error is largest for this case. This is expected since no methods are employed to correct this error. In fact, we can easily calculate this error, as we have done in Section 5.2.4 as part of the implementation of the new strategy of Chapter 4. The error is a consequence of nonlinearities present in the perfect controller, as well as from the artificially introduced errors.

The adaptive controller attempts to compensate the errors by adding an additional term to the controller which is adapted as the system is run. It is clear that the adaptive controller can directly compensate for the artificially added errors (since they take the same form as the adaptive terms), but not the nonlinearities. The nonlinearities have a quadratic form which can not be represented by additive adaptive

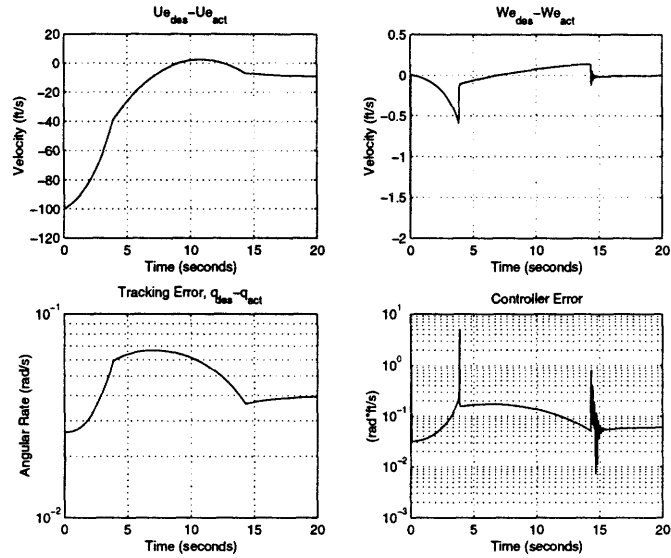


Figure 6-2: Linear Feedback Controller, Large Additive Errors

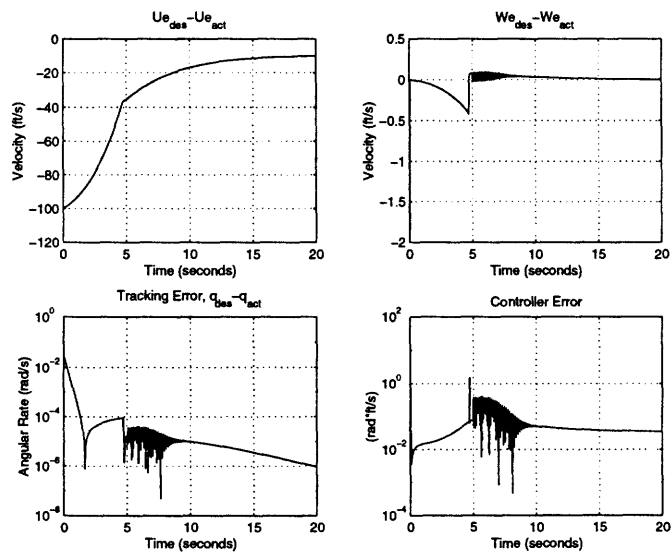


Figure 6-3: Adaptive Controller, Large Additive Errors

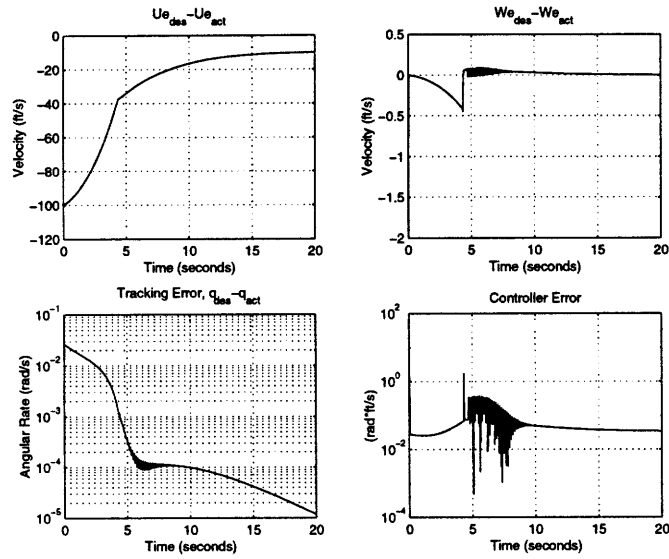


Figure 6-4: Neural Controller, Large Additive Errors

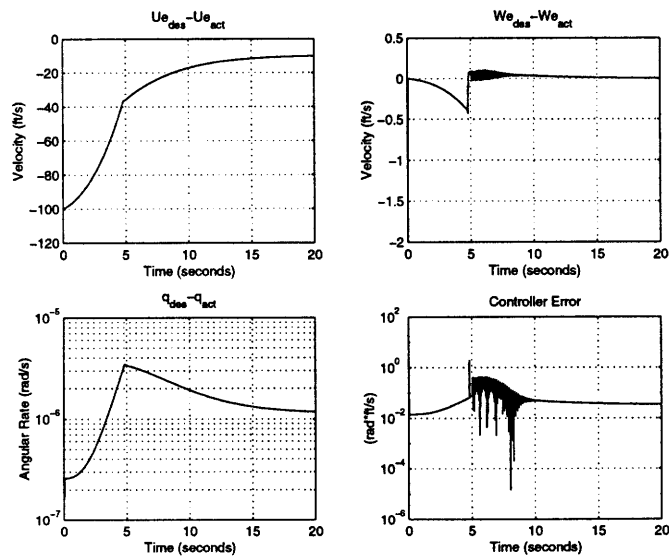


Figure 6-5: New Controller, Large Additive Errors

part of the controller. Remember, though, that the adaptive term is time varying, and as the system acquires a new equilibrium point, the adaptive term makes the controller look like the local linearized controller. Thus, although there are evident differences in the transients between this controller and the perfect controller, the system eventually settles down to a steady state with nearly perfect tracking. It is important to note, though, that if the difference between the perfect and the initial nominal controller were significant, we could not guarantee that the adaptive controller would not go unstable. To keep any instabilities from occurring, the adaptation rate is kept small. It is also interesting to note that the initial tracking errors (at time=0) are of the same magnitude of the errors of the linear controller, and decrease over time. The final tracking errors (Figure 6-3) are six orders of magnitude smaller than those of the uncompensated linear controller for the case of large additive errors (Figure 6-2).

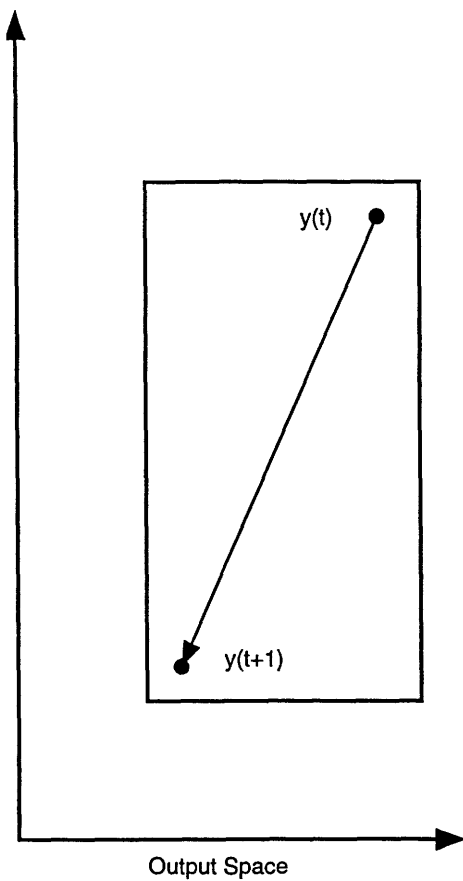
The neural controller suffers from the adaptive controller only in the complexity of the added adaptive term. The additive term chosen for in this implementation is a linear combination of *hat* shaped wavelets. The combination of these wavelets represent the product of linear splines. There was great difficulty in implementing this network based controller because the input to the controller takes for arguments $(w(t + 1)_{des}, q(t + 1)_{des}, w(t), q(t))$, thus requiring a large network. In fact, to reduce the network to a small enough number of wavelet such that the network was actually fast enough to run on a Spark 20 workstation (by fast enough we imply that the simulation could be run in less than a day), the resolution of the wavelets had to be reduced significantly. It was found that the approximating power of the network was only nominally better than that of the adaptive term in the adaptive controller. As a result, there is little difference in the tracking and approximating performance of the neural network controller as compared to the adaptive controller. Indeed, the final tracking error (Figure 6-5), although five orders of magnitude smaller than that of the uncompensated linear controller (Figure 6-2), is still larger than that of the adaptive controller (Figure 6-3). This occurs because the basis functions of the neural network are spatially localized. Therefore a basis' coefficient is not adapted till the basis function has a nonzero value when evaluated at the controllers arguments. Since the

controller takes different commands initially than later in the simulation, the initial training has little effect on the errors later on. This spatial localization, though, has a strength which outweighs this weakness. Once the controller has been trained about some input, $(w(t+1)_{des}, q(t+1)_{des}, w(t), q(t))$, the network does not become untrained. The adaptive controller, on the other hand, contains no local information. If adaptive parameters are trained about some particular input, they necessarily lose information about previous inputs.

Finally, we come to the controller presented in this thesis. Although this controller also suffers from the dimensional problems encountered by the neural controller, there are obvious ways to overcome them. In particular, we can more easily specify over which set we must acquire samples to perform the desired maneuver. For example, in the most general application of the learning algorithm, we learn a controller which can take us from any member of a particular set of states $(\hat{D}_{y,N}^{2n})$ to any output in the interval $\hat{D}_{y,N}$. If we do not intend to use this full capability, we can be more selective in what we learn. Therefore, for any given state we need only learn the controller which allows us to command to members of some subset of $\hat{D}_{y,N}$. In fact, the method employed in simulation is to learn a large number of local controllers whose domains overlap and cover $\hat{D}_{y,N}^{2n}$ as opposed to the global controller whose domain is $\hat{D}_{y,N}^{2n}$. We must remember, though, that commanded outputs must be limited to those over which we have trained. This limitation is represented graphically in Figure 6-6.

Once the controller is trained, flight simulations can begin. We can see in the simulations (Figure 6-5) that the tracking performance is significantly better than any of the other controllers. Its tracking error is approximately six orders of magnitude better than those of the linear controller. In addition, this small error is maintained throughout the simulation, since training had been completed before the simulations start. The tracking error for the adaptive and neural controllers are initially large.

In this case, we can reach any output in the boxed region from any other output in the region



In this case, we have two overlapping boxed regions. To move from a point in region 1 to a point in region 2 we must include an intermediate move to a point common to the two regions

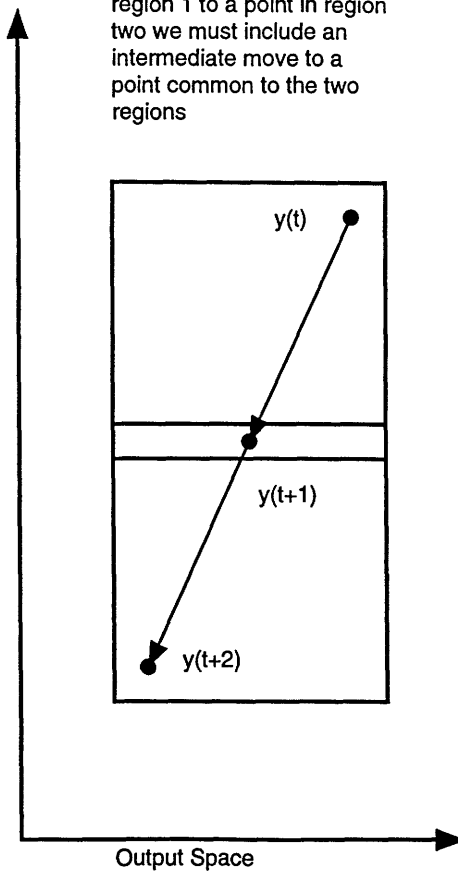


Figure 6-6: Limitation of New Controller

6.2 Computational Effort

The required computational effort is where the greatest weakness occurs in the proposed algorithm. With each additional input argument, the required number of samples increases by an order of magnitude. In particular, the number of required samples grows by the relation MN^{2n} , where n is the order of the system and M and N are constants depending on \mathbf{D}_k and δ_k .

As mentioned above, the neural network based algorithm also suffers from this fault. Within this fault, each of the two algorithms have their individual strengths and weaknesses.

Much work has been done to make neural networks compact. Individual papers propose techniques to dynamically change the structure of the network in order to minimize its size and complexity. In fact, the network employed in this paper uses a hierarchical structure which increases the resolution of the network in the case of large errors, and reduces the resolution if it is found error bounds can be satisfied with fewer basis functions. The algorithm proposed in Chapter 4 fixes the resolution (density of samples). No method is proposed to make local changes in the resolution to reduce the number of samples required. This amounts to a weakness in the proposed algorithm.

A relative strength of the proposed algorithm is mentioned in the previous section. As mentioned it is easy to restrict the domain over which the samples are taken. If the domain is chosen wisely, the size of the set of samples can be reduced by orders of magnitude. In this paper, the domain was restricted to a region local to a line in the four dimensional input space. Thereby, the order of the number of samples is reduced by three. This represents to a strength of the algorithm.

6.3 Design Effort

The amount of required design effort is meant to be the strength of the proposed algorithm. This effort is only nominally greater than the effort required to find a local

linear model of the system to which the algorithm is applied. The only additional work is the implementation of the algorithm. The entire algorithm is implemented with the three simple and short Matlab functions presented in the appendix. The perfect controller, of course, requires significantly more modeling effort. The network based controller also requires additional effort. This includes the design and implementation of the network. A basic network is not exceptionally difficult to implement. The multi-resolution network of this paper, though, does displace several times more lines of code than that for the algorithm of Chapter 4.

Chapter 7

Final Comments and Further Work

7.1 Final Comments and Further Work

It is evident from the results that on a small simple system, the learning control algorithm of this paper provides performance comparable to other modern control algorithms, and can guarantee that performance through a rigorous proof.

But, given the current state of computer technology, the implementation of the proposed algorithm on a large, high dimensional system may not yet be practical. In addition, the consequences of process and sensor noise have not been addressed. It is clear that the introduction of either source of noise will have a significant impact. Currently, the only method of compensating for process noise is to model the source of the noise itself. For such a solution to work, the interfering process must be time invariant. Such an assumption is usually false. Sensor noise can be taken into account in the arguments of Chapter 4, but may severely limit the guarantees provided in the proof. In fact, careful examination would suggest that given large errors, the methods of the proof would provide no guarantees at all.

Despite the above challenges, the proposed controller is a step in the right direction, and with additional work can be extended to include a larger set of real world systems. When computers become capable of implementing the above algorithm on

large systems, and they will, this learning strategy could prove effective.

Appendix A

Convex Hulls

Many definitions and theorems provided in this appendix are drawn from [20] and [6]. A few are original.

Definition A.0.1 *A set $\mathbf{A} \in \mathbf{R}^n$ is convex if $x, y \in \mathbf{A}$ implies that*

$$\{\lambda x + (1 - \lambda)y : 0 \leq \lambda \leq 1\} \subseteq \mathbf{A}.$$

Definition A.0.2 *The convex hull $Co(\mathbf{A})$ of a set \mathbf{A} in \mathbf{R}^n is the intersection of all convex sets in \mathbf{R}^n containing \mathbf{A} [20, p. 54]. Alternately, $Co(\mathbf{A})$ is the smallest convex set containing \mathbf{A} [20, p. 54]. More precisely, if \mathbf{C} is any convex set in \mathbf{R}^n containing \mathbf{A} then $Co(\mathbf{A}) \subseteq \mathbf{C}$.*

Theorem A.0.1 [20, p. 55]

Let $a_1, \dots, a_m \in \mathbf{R}^n$. Then

$$\begin{aligned} Co(\{a_1, \dots, a_m\}) = \{ & \lambda_1 a_1 + \dots + \lambda_m a_m : \\ & \lambda_1, \dots, \lambda_m \geq 0, \\ & \lambda_1 + \dots + \lambda_m = 1\} \end{aligned}$$

Corollary A.0.1 *Let $a_1, \dots, a_m \in \mathbb{R}^n$. Then each $a \in \text{Co}(\{a_1, \dots, a_m\})$ can be written in the form*

$$a = \lambda a' + (1 - \lambda)a_m \text{ where } a' \in \text{Co}(\{a_1, \dots, a_{m-1}\}), \quad 0 \leq \lambda \leq 1. \quad (\text{A.1})$$

Proof: From Theorem (A.0.1) each $a \in \text{Co}(\{a_1, \dots, a_m\})$ can be written

$$a = \lambda_1 a_1 + \dots + \lambda_m a_m \quad (\text{A.2})$$

for

$$\lambda_1 + \dots + \lambda_m = 1, \text{ where } \lambda_1, \dots, \lambda_m \geq 0. \quad (\text{A.3})$$

We consider two cases: (1) $\lambda_m = 1$ and (2) $\lambda_m \neq 1$.

Case (1): Since $\lambda_m = 1$, $\lambda_i = 0$ for $i = 1, \dots, m - 1$. Hence, a can be expressed as

$$a = a_m = \lambda a' + (1 - \lambda)a_m,$$

where $\lambda = 0$ and $a' \in \text{Co}(\{a_1, \dots, a_{m-1}\})$, which proves Equation (A.1).

Case (2): **Case $\lambda_m \neq 1$:** Letting $\lambda = 1 - \lambda_m$ and

$$a' = \frac{\lambda_1}{\lambda} a_1 + \dots + \frac{\lambda_{m-1}}{\lambda} a_{m-1},$$

Equation (A.2) can be manipulated to yield

$$a = \lambda a' + (1 - \lambda)a_m$$

To prove Equation (A.1) it remains to show that

$$a' \in \text{Co}(\{a_1, \dots, a_{m-1}\}). \quad (\text{A.4})$$

By Theorem A.0.1, Equation (A.4) is true if

$$\frac{\lambda_1}{\lambda} + \dots + \frac{\lambda_{m-1}}{\lambda} = 1 \text{ and} \quad (\text{A.5})$$

$$\frac{\lambda_1}{\lambda}, \dots, \frac{\lambda_{m-1}}{\lambda} \geq 0 \quad (\text{A.6})$$

Equation (A.6) is true since $\lambda \geq 0$ and each $\lambda_i \geq 0$. Equation (A.5) follows from Equation (A.3) by noting that $\lambda_m = 1 - \lambda$.

□

Definition A.0.3 Let $A = \text{Co}(a_0, \dots, a_n) \subset \mathbf{R}^n$. The minimum width of A is the minimum distance

$$\min_{i=0, \dots, n} \|b_i - a_i\|_2$$

where

$$b_i = \operatorname{argmin}_{b \in \text{Co}(a_0, \dots, a_{i-1}, a_{i+1}, \dots, a_n)} \|b - a_i\|_2$$

Theorem A.0.2 [6, p. 112] If a closed bounded convex set $A \subset \mathbf{R}^n$ has minimum width d , then A contains a ball of radius¹

$$\begin{aligned} r &\geq \frac{d}{2\sqrt{n}}, \quad n \text{ odd} \\ r &\geq \frac{d\sqrt{n+2}}{2(n+1)}, \quad n \text{ even} \end{aligned}$$

Theorem A.0.3 Suppose we are given $n + 1$ points a_0, \dots, a_n in \mathbf{R}^n . Define the matrices

$$A_i = \begin{bmatrix} a_0 - a_i & \cdots & a_{i-1} - a_i & a_{i+1} - a_i & \cdots & a_n - a_i \end{bmatrix} \quad (\text{A.7})$$

where each A_i satisfies

$$\|A_i v\|_2 \geq \alpha_i \|v\|_2 \quad (\text{A.8})$$

for some α_i and all $v \in \mathbf{R}^n$. Then the minimum width d of $\text{Co}(a_0, \dots, a_n)$ satisfies

$$d \geq \frac{\min_{i=0, \dots, n} \alpha_i}{\sqrt{n}} \quad (\text{A.9})$$

¹a ball in \mathbf{R}^n is defined in Definition 4.3.1

Proof: Begin by noting Definition A.0.3. Since each b_i as defined in Definition A.0.3 lies in the set $\text{Co}(a_0, \dots, a_{i-1}, a_{i+1}, \dots, a_n)$, using Theorem A.0.1 we can write b_i as the sum

$$b_i = \lambda_0 a_0 + \dots + \lambda_{i-1} a_{i-1} + \lambda_{i+1} a_{i+1} + \dots + \lambda_n a_n \quad (\text{A.10})$$

where

$$0 \leq \lambda_i \leq 1 \text{ and } \sum_i \lambda_i = 1, \quad i = 0, \dots, i-1, i+1, \dots, n.$$

Letting

$$\lambda = \left[\lambda_0 \quad \dots \quad \lambda_{i-1} \quad \lambda_{i+1} \quad \dots \quad \lambda_n \right]^T$$

we can write Equation (A.10) in matrix form as

$$b_i = \left[a_0 \quad \dots \quad a_{i-1} \quad a_{i+1} \quad \dots \quad a_n \right] \lambda \quad (\text{A.11})$$

Since $\sum_i \lambda_i = 1$ we can also write

$$A_i = \left[a_i \quad \dots \quad a_i \right] \lambda \quad (\text{A.12})$$

Therefore, using Equations (A.7), (A.11) and (A.12)

$$b_i - a_i = A_i \lambda \quad (\text{A.13})$$

Therefore, noting Definition A.0.3 and Equations (A.10) and (A.13), the minimum width d is the minimum of

$$\|A_i \lambda\|_2$$

over all $i = 0, \dots, n$ and all λ satisfying Equation (A.9). We proceed by finding a lower bound for $\|A_i \lambda\|_2$ and thus also for d . Using $\|\lambda\|_2 \geq \frac{1}{\sqrt{n}} \|\lambda\|_1 = \frac{1}{\sqrt{n}}$ and Equation (A.8) we have

$$d = \min_{i, \lambda} \|A_i \lambda\|_2 \geq \min_{i, \lambda} (\alpha_i \|\lambda\|_2) \geq \min_i \alpha_i \min_{\lambda} \|\lambda\|_2 \geq \frac{\min_{i=0, \dots, n} \alpha_i}{\text{sqrtn}}$$



Appendix B

Matrix Algebra

Theorem B.0.4 *Assume the real $N \times N$ matrix*

$$\Theta = \begin{bmatrix} \theta_1 - \theta_0 & \dots & \theta_N - \theta_0 \end{bmatrix} \tag{B.1}$$

has an inverse. Then the matrix

$$\Psi = \begin{bmatrix} \theta_0 & \dots & \theta_N \\ 1 & \dots & 1 \end{bmatrix}$$

also has an inverse.

Proof: We can show Ψ has an inverse by equivalently showing that the columns of Ψ span \mathbf{R}^{N+1} . Choose some arbitrary $\psi \in \mathbf{R}^{N+1}$. Let the first N elements of ψ be $\psi_1 \in \mathbf{R}^N$ and the last element be $\psi_2 \in \mathbf{R}$. Since Θ is invertible, ψ_1 can be written as a linear combination of the columns of Θ . Therefore, there exist scalars a_1, \dots, a_N

such that

$$\begin{aligned}
\psi &= \begin{bmatrix} \psi_1 \\ \psi_2 \end{bmatrix} \\
&= a_1 \begin{bmatrix} \theta_1 - \theta_0 \\ 0 \end{bmatrix} + \cdots + a_N \begin{bmatrix} \theta_N - \theta_0 \\ 0 \end{bmatrix} + \psi_2 \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \\
&= a_1 \begin{bmatrix} \theta_1 \\ 1 \end{bmatrix} + \cdots + a_N \begin{bmatrix} \theta_N \\ 1 \end{bmatrix} + (\psi_2 - a_1 - \cdots - a_N) \begin{bmatrix} \theta_0 \\ 1 \end{bmatrix}
\end{aligned}$$

Since we have expressed ψ as a combination of the columns of Ψ and ψ was chosen arbitrarily, Ψ^{-1} exists. □

Theorem B.0.5 *Let Θ and Ψ be two $N \times N$ real matrices. Suppose*

$$\|\Theta x\|_2 \geq \theta \|x\|_2$$

for some $\theta > 0$ and all $x \in \mathbf{R}^N$. Let the absolute value of any element of Ψ be less than or equal to ψ . Then

$$\|(\Theta + \Psi)x\|_2 \geq (\theta - N\psi) \|x\|_2 \tag{B.2}$$

Proof: Using properties of matrix norms [4] and Equation (B.4)

$$\begin{aligned}
\|(\Theta + \Psi)x\|_2 &\geq \|\Theta x\|_2 - \|\Psi x\|_2 \\
&\geq \theta \|x\|_2 - \text{trace}(\Psi^T \Psi)^{\frac{1}{2}} \|x\|_2 \\
&\geq (\theta - N\psi) \|x\|_2
\end{aligned} \tag{B.3}$$

□

Lemma B.0.1 *Given the assumptions of Theorem B.0.5, if $\theta > N\psi$ then $\Theta + \Psi$ is invertible.*

Proof: Since $\theta > N\psi$, Equation (B.3) yields

$$\|(\Theta + \Psi)x\|_2 > 0$$

for any $x \in \mathbf{R}^N$, x not the zero vector. This condition is sufficient and necessary for the invertibility of $\Theta + \Psi$.

□

Theorem B.0.6 *Let an $N \times N + 1$ matrix Θ take the form*

$$\Theta = \begin{bmatrix} \theta_0 & \cdots & \theta_N \\ 0 & 0 & \\ \vdots & \vdots & I[N \times N] \\ 0 & 0 & \end{bmatrix}$$

Let the $(m_1)_{th}$ column of Θ be $\theta[m]$. Define the $N + 1$ matrices

$$\Theta_m = \begin{bmatrix} \theta[0] - \theta[m] & \cdots & \theta[m-1] - \theta[m] & \alpha[m+1] - \alpha[m] & \cdots & \alpha[N] - \alpha[m] \end{bmatrix}$$

for any $m = 0, \dots, N$. If $\theta_0 \neq \theta_1$, then

$$\|\Theta_m v\|_2 \geq \frac{|\theta_0 - \theta_1|}{N\theta} \|v\|_2 \quad (\text{B.4})$$

for any $v \in \mathbf{R}^N$ where

$$\theta = \max_{k=2, \dots, N} (1, |\theta_0 - \theta_k|, |\theta_1 - \theta_k|) \quad (\text{B.5})$$

Proof: The inverse of Θ_m is easily calculated as

$$\Theta_m^{-1} = \frac{1}{\theta_0 - \theta_1} \begin{bmatrix} 1 & \theta_1 - \theta_2 & \dots & \theta_1 - \theta_N \\ -1 & \theta_2 - \theta_0 & \dots & \theta_N - \theta_0 \\ 0 & I[m - 2 \times m - 2] & 0 & 0[m - 2 \times N - m] \\ \vdots & & \vdots & \\ 0 & 0[N - m \times m - 2] & 0 & I[N - m] \end{bmatrix}$$

for $m = 2, \dots, N$. For the cases $m = 0$ and $m = 1$ we have

$$\Theta_0^{-1} = \frac{1}{\theta_1 - \theta_0} \begin{bmatrix} 1 & \theta_0 - \theta_2 & \dots & \theta_N - \theta_2 \\ 0 & & I[N - 1 \times N - 1] & \end{bmatrix}$$

$$\Theta_1^{-1} = \frac{1}{\theta_0 - \theta_1} \begin{bmatrix} 1 & \theta_1 - \theta_2 & \dots & \theta_1 - \theta_N \\ 0 & & I[N - 1 \times N - 1] & \end{bmatrix}$$

Since the two norm of an $N \times N$ matrix is less than or equal to N times the maximum absolute value of an element of the matrix

$$\|\Theta_m^{-1}\|_2 \leq N \max_{k=2, \dots, N} \left(1, \left| \frac{\theta_0 - \theta_k}{\theta_0 - \theta_1} \right|, \left| \frac{\theta_1 - \theta_k}{\theta_0 - \theta_1} \right| \right) \leq N \frac{\max_{k=2, \dots, N} (1, |\theta_0 - \theta_k|, |\theta_1 - \theta_k|)}{|\theta_0 - \theta_1|} \quad (\text{B.6})$$

Therefore using properties of matrix norms [4].

$$\|v\|_2 = \|\Theta_m^{-1} \Theta_m v\|_2 \leq \|\Theta_m^{-1}\|_2 \|\Theta_m v\|_2$$

Solving for $\|\Theta_m v\|_2$ yields

$$\|\Theta_m v\|_2 \geq \frac{\|v\|_2}{\|\Theta_m^{-1}\|_2}$$

Applying Equation (B.6) yields Equation (B.4).

□

Bibliography

- [1] Fred Brauer and John A. Nohel. *The Qualitative Theory of Ordinary Differential Equations*. Dover Publications, Inc., New York, 1969.
- [2] Fu-Chuang Chen and Hassan K. Khalil. Adaptive control of a class of nonlinear discrete-time systems using neural networks. *IEEE Transactions on Automatic Control*, 40(5):791–801, May 1995.
- [3] G. Cybenko. Approximation by superposition fo a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314, 1989.
- [4] M. Dahleh and G. Verghese. 6.241: Dynamic systems. Technical report, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1997. Course notes for 6.241.
- [5] I. Daubechies. *Ten Lectures on Wavelets*. SIAM, Philadelphia, PA, 1992.
- [6] H. G. Eggleston. *Convexity*. Cambridge University Press, New York, 1958.
- [7] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.
- [8] H.Shim, T. John Koo, F. Hoffmann, and S. Sastrya. A comprehensive study on control design of autonomous helicopter. Technical report, University of California at Berkeley, Robotics and Intelligent Machines Laboratory, Berkeley, California, 1998.

- [9] Liang Jin and Madan M. Gupta. Fast neural learning and control of discrete-time nonlinear systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(3):478–488, March 1995.
- [10] Hassan K. Khalil. *Nonlinear Systems*. Prentice-Hall, Upper Saddle River, NJ, second edition, 1996.
- [11] Lesse Leitner, Anthony Calise, and J. V. R. Prasad. Analysis of adaptive neural networks. *Journal of Guidance, Control, and Dynamics*, 20(5):972–979, September-October 1997.
- [12] I. J. Leonttaritis and S. A. Billings. Input-output parametric models for nonlinear systems Part I: deterministic nonlinear systems. *International Journal of Control*, 41(2):303–328, 1985.
- [13] S. Monaco and D. Normand-Cyrot. Minimum-phase nonlinear discrete-time systems and feedback stabilization. *Proceedings of the 26th Conference on Decision and Control*, December 1987.
- [14] James F. Montgomery and George A. Bekey. Learning helicopter control through "teaching by showing". Technical report, University of Southern California, Institute for Robotics and Intelligent Systems, Department of Computer Science, Los Angeles, California, 1998.
- [15] K.S. Narendra and A.M. Annaswamy. *Stable Adaptive Systems*. Prentice-Hall, Englewood Cliffs, N.J., 1989.
- [16] J. Park and I.W. Sandberg. Universal approximation using radial-basis function networks. *Neural Computation*, 3:246–257, 1991.
- [17] Murray H. Protter and Jr. Charles B. Morrey. *A First Course in Real Analysis*. Springer-Verlag, New York, second edition, 1991.
- [18] H. L. Royden. *Real Analysis*. The Macmillan Company, New York, 1963.

- [19] R. M. Sanner and J.-J.E. Slotine. Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks*, 3(6):823–863, 1992.
- [20] Roger Webster. *Convexity*. Oxford University Press, Oxford, 1994.
- [21] Ssu-Hsin Yu and Anuradha M. Annaswamy. Adaptive control of nonlinear dynamic systems using θ -adaptive neural networks. *Automatica*, 33(11):1975–1995, 1997.

5733-12