

Rushikesh Kamalapurkar  
Patrick Walters · Joel Rosenfeld  
Warren Dixon

# Reinforcement Learning for Optimal Feedback Control

A Lyapunov-Based Approach

# **Communications and Control Engineering**

## **Series editors**

Alberto Isidori, Roma, Italy

Jan H. van Schuppen, Amsterdam, The Netherlands

Eduardo D. Sontag, Boston, USA

Miroslav Krstic, La Jolla, USA

**Communications and Control Engineering** is a high-level academic monograph series publishing research in control and systems theory, control engineering and communications. It has worldwide distribution to engineers, researchers, educators (several of the titles in this series find use as advanced textbooks although that is not their primary purpose), and libraries.

The series reflects the major technological and mathematical advances that have a great impact in the fields of communication and control. The range of areas to which control and systems theory is applied is broadening rapidly with particular growth being noticeable in the fields of finance and biologically-inspired control. Books in this series generally pull together many related research threads in more mature areas of the subject than the highly-specialised volumes of *Lecture Notes in Control and Information Sciences*. This series's mathematical and control-theoretic emphasis is complemented by *Advances in Industrial Control* which provides a much more applied, engineering-oriented outlook.

**Publishing Ethics:** Researchers should conduct their research from research proposal to publication in line with best practices and codes of conduct of relevant professional bodies and/or national and international regulatory bodies. For more details on individual ethics matters please see:

<https://www.springer.com/gp/authors-editors/journal-author/journal-author-help-desk/publishing-ethics/14214>.

More information about this series at <http://www.springer.com/series/61>

Rushikesh Kamalapurkar · Patrick Walters  
Joel Rosenfeld · Warren Dixon

# Reinforcement Learning for Optimal Feedback Control

A Lyapunov-Based Approach



Springer

Rushikesh Kamalapurkar  
Mechanical and Aerospace Engineering  
Oklahoma State University  
Stillwater, OK  
USA

Patrick Walters  
Naval Surface Warfare Center  
Panama City, FL  
USA

Joel Rosenfeld  
Electrical Engineering  
Vanderbilt University  
Nashville, TN  
USA

Warren Dixon  
Department of Mechanical  
and Aerospace Engineering  
University of Florida  
Gainesville, FL  
USA

ISSN 0178-5354

ISSN 2197-7119 (electronic)

Communications and Control Engineering

ISBN 978-3-319-78383-3

ISBN 978-3-319-78384-0 (eBook)

<https://doi.org/10.1007/978-3-319-78384-0>

Library of Congress Control Number: 2018936639

MATLAB® and Simulink® are registered trademarks of The MathWorks, Inc., 1 Apple Hill Drive, Natick, MA 01760-2098, USA, <http://www.mathworks.com>.

Mathematics Subject Classification (2010): 49-XX, 34-XX, 46-XX, 65-XX, 68-XX, 90-XX, 91-XX, 93-XX

© Springer International Publishing AG 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG part of Springer Nature

The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*To my nurturing grandmother, Mangala  
Vasant Kamalapurkar.*

—Rushikesh Kamalapurkar

*To my strong and caring grandparents.*  
—Patrick Walters

*To my wife, Laura Forest Gruss Rosenfeld,  
with whom I have set out on the greatest  
journey of my life.*

—Joel Rosenfeld

*To my beautiful son, Isaac Nathaniel Dixon.*  
—Warren Dixon

# Preface

Making the best possible decision according to some desired set of criteria is always difficult. Such decisions are even more difficult when there are time constraints and can be impossible when there is uncertainty in the system model. Yet, the ability to make such decisions can enable higher levels of autonomy in robotic systems and, as a result, have dramatic impacts on society. Given this motivation, various mathematical theories have been developed related to concepts such as optimality, feedback control, and adaptation/learning. This book describes how such theories can be used to develop optimal (i.e., the best possible) controllers/policies (i.e., the decision) for a particular class of problems. Specifically, this book is focused on the development of concurrent, real-time learning and execution of approximate optimal policies for infinite-horizon optimal control problems for continuous-time deterministic uncertain nonlinear systems.

The developed approximate optimal controllers are based on reinforcement learning-based solutions, where learning occurs through an actor–critic-based reward system. Detailed attention to control-theoretic concerns such as convergence and stability differentiates this book from the large body of existing literature on reinforcement learning. Moreover, both model-free and model-based methods are developed. The model-based methods are motivated by the idea that a system can be controlled better as more knowledge is available about the system. To account for the uncertainty in the model, typical actor–critic reinforcement learning is augmented with unique model identification methods. The optimal policies in this book are derived from dynamic programming methods; hence, they suffer from the curse of dimensionality. To address the computational demands of such an approach, a unique function approximation strategy is provided to significantly reduce the number of required kernels along with parallel learning through novel state extrapolation strategies.

The material is intended for readers that have a basic understanding of nonlinear analysis tools such as Lyapunov-based methods. The development and results may help to support educators, practitioners, and researchers with nonlinear systems/control, optimal control, and intelligent/adaptive control interests working in aerospace engineering, computer science, electrical engineering, industrial

engineering, mechanical engineering, mathematics, and process engineering disciplines/industries.

Chapter 1 provides a brief introduction to optimal control. Dynamic programming-based solutions to optimal control problems are derived, and the connections between the methods based on dynamic programming and the methods based on the calculus of variations are discussed, along with necessary and sufficient conditions for establishing an optimal value function. The chapter ends with a brief survey of techniques to solve optimal control problems. Chapter 2 includes a brief review of dynamic programming in continuous time and space. In particular, traditional dynamic programming algorithms such as policy iteration, value iteration, and actor–critic methods are presented in the context of continuous-time optimal control. The role of the optimal value function as a Lyapunov function is explained to facilitate online closed-loop optimal control. This chapter also highlights the problems and limitations of existing techniques, thereby motivating the development in this book. The chapter concludes with some historic remarks and a brief classification of the available dynamic programming techniques.

In Chap. 3, online adaptive reinforcement learning-based solutions are developed for infinite-horizon optimal control problems for continuous-time uncertain nonlinear systems. A novel actor–critic–identifier structure is developed to approximate the solution to the Hamilton–Jacobi–Bellman equation using three neural network structures. The actor and the critic neural networks approximate the optimal controller and the optimal value function, respectively, and a robust dynamic neural network identifier asymptotically approximates the uncertain system dynamics. An advantage of using the actor–critic–identifier architecture is that learning by the actor, critic, and identifier is continuous and concurrent, without requiring knowledge of system drift dynamics. Convergence is analyzed using Lyapunov-based adaptive control methods. The developed actor–critic method is extended to solve trajectory tracking problems under the assumption that the system dynamics are completely known. The actor–critic–identifier architecture is also extended to generate approximate feedback-Nash equilibrium solutions to  $N$ -player nonzero-sum differential games. Simulation results are provided to demonstrate the performance of the developed actor–critic–identifier method.

Chapter 4 introduces the use of an additional adaptation strategy called concurrent learning. Specifically, a concurrent learning-based implementation of model-based reinforcement learning is used to solve approximate optimal control problems online under a finite excitation condition. The development is based on the observation that, given a model of the system, reinforcement learning can be implemented by evaluating the Bellman error at any number of desired points in the state space. By exploiting this observation, a concurrent learning-based parameter identifier is developed to compensate for uncertainty in the parameters. Convergence of the developed policy to a neighborhood of the optimal policy is established using a Lyapunov-based analysis. Simulation results indicate that the developed controller can be implemented to achieve fast online learning without the addition of ad hoc probing signals as in Chap. 3. The developed model-based reinforcement learning method is extended to solve trajectory tracking problems for

uncertain nonlinear systems and to generate approximate feedback-Nash equilibrium solutions to  $N$ -player nonzero-sum differential games.

Chapter 5 discusses the formulation and online approximate feedback-Nash equilibrium solution for an optimal formation tracking problem. A relative control error minimization technique is introduced to facilitate the formulation of a feasible infinite-horizon total-cost differential graphical game. A dynamic programming-based feedback-Nash equilibrium solution to the differential graphical game is obtained via the development of a set of coupled Hamilton–Jacobi equations. The developed approximate feedback-Nash equilibrium solution is analyzed using a Lyapunov-based stability analysis to yield formation tracking in the presence of uncertainties. In addition to control, this chapter also explores applications of differential graphical games to monitoring the behavior of neighboring agents in a network.

Chapter 6 focuses on applications of model-based reinforcement learning to closed-loop control of autonomous vehicles. The first part of the chapter is devoted to online approximation of the optimal station keeping strategy for a fully actuated marine craft. The developed strategy is experimentally validated using an autonomous underwater vehicle, where the three degrees of freedom in the horizontal plane are regulated. The second part of the chapter is devoted to online approximation of an infinite-horizon optimal path-following strategy for a unicycle-type mobile robot. An approximate optimal guidance law is obtained through the application of model-based reinforcement learning and concurrent learning-based parameter estimation. Simulation results demonstrate that the developed method learns an optimal controller which is approximately the same as an optimal controller determined by an off-line numerical solver, and experimental results demonstrate the ability of the controller to learn the approximate solution in real time.

Motivated by computational issues arising in approximate dynamic programming, a function approximation method is developed in Chap. 7 that aims to approximate a function in a small neighborhood of a state that travels within a compact set. The development is based on the theory of universal reproducing kernel Hilbert spaces over the  $n$ -dimensional Euclidean space. Several theorems are introduced that support the development of this State Following (StaF) method. In particular, it is shown that there is a bound on the number of kernel functions required for the maintenance of an accurate function approximation as a state moves through a compact set. Additionally, a weight update law, based on gradient descent, is introduced where good accuracy can be achieved provided the weight update law is iterated at a high enough frequency. Simulation results are presented that demonstrate the utility of the StaF methodology for the maintenance of accurate function approximation as well as solving the infinite-horizon optimal regulation problem. The results of the simulation indicate that fewer basis functions are required to guarantee stability and approximate optimality than are required when a global approximation approach is used.

The authors would like to express their sincere appreciation to a number of individuals whose support made the book possible. Numerous intellectual discussions and research support were provided by all of our friends and colleagues in the Nonlinear Controls and Robotics Laboratory at the University of Florida, with particular thanks to Shubhendu Bhasin, Patryk Deptula, Huyen Dinh, Keith Dupree, Nic Fischer, Marcus Johnson, Justin Klotz, and Anup Parikh. Inspiration and insights for our work were provided, in part, through discussions with and/or reading foundational literature by Bill Hager, Michael Jury, Paul Robinson, Frank Lewis (the academic grandfather or great grandfather to several of the authors), Derong Liu, Anil Rao, Kyriakos Vamvoudakis, Richard Vinter, Daniel Liberzon, and Draguna Vrabie. The research strategies and breakthroughs described in this book would also not have been possible without funding support provided from research sponsors including: NSF award numbers 0901491 and 1509516, Office of Naval Research Grants N00014-13-1-0151 and N00014-16-1-2091, Prioria Robotics, and the Air Force Research Laboratory, Eglin AFB. Most importantly, we are eternally thankful for our families who are unwavering in their love, support, and understanding.

Stillwater, OK, USA  
Panama City, FL, USA  
Nashville, TN, USA  
Gainesville, FL, USA  
January 2018

Rushikesh Kamalapurkar  
Patrick Walters  
Joel Rosenfeld  
Warren Dixon

# Contents

<b>1 Optimal Control . . . . .</b>	1
1.1 Introduction . . . . .	1
1.2 Notation . . . . .	1
1.3 The Bolza Problem . . . . .	2
1.4 Dynamic Programming . . . . .	3
1.4.1 Necessary Conditions for Optimality . . . . .	3
1.4.2 Sufficient Conditions for Optimality . . . . .	5
1.5 The Unconstrained Affine-Quadratic Regulator . . . . .	5
1.6 Input Constraints . . . . .	7
1.7 Connections with Pontryagin's Maximum Principle . . . . .	9
1.8 Further Reading . . . . .	10
1.8.1 Numerical Methods . . . . .	10
1.8.2 Differential Games and Equilibrium Solutions . . . . .	11
1.8.3 Viscosity Solutions and State Constraints . . . . .	12
References . . . . .	13
<b>2 Approximate Dynamic Programming . . . . .</b>	17
2.1 Introduction . . . . .	17
2.2 Exact Dynamic Programming in Continuous Time and Space . . . . .	17
2.2.1 Exact Policy Iteration: Differential and Integral Methods . . . . .	18
2.2.2 Value Iteration and Associated Challenges . . . . .	22
2.3 Approximate Dynamic Programming in Continuous Time and Space . . . . .	22
2.3.1 Some Remarks on Function Approximation . . . . .	23
2.3.2 Approximate Policy Iteration . . . . .	24
2.3.3 Development of Actor-Critic Methods . . . . .	25
2.3.4 Actor-Critic Methods in Continuous Time and Space . . . . .	26
2.4 Optimal Control and Lyapunov Stability . . . . .	26

2.5	Differential Online Approximate Optimal Control . . . . .	28
2.5.1	Reinforcement Learning-Based Online Implementation . . . . .	29
2.5.2	Linear-in-the-Parameters Approximation of the Value Function . . . . .	30
2.6	Uncertainties in System Dynamics . . . . .	32
2.7	Persistence of Excitation and Parameter Convergence . . . . .	33
2.8	Further Reading and Historical Remarks . . . . .	34
	References . . . . .	37
<b>3</b>	<b>Excitation-Based Online Approximate Optimal Control . . . . .</b>	<b>43</b>
3.1	Introduction . . . . .	43
3.2	Online Optimal Regulation . . . . .	45
3.2.1	Identifier Design . . . . .	45
3.2.2	Least-Squares Update for the Critic . . . . .	49
3.2.3	Gradient Update for the Actor . . . . .	50
3.2.4	Convergence and Stability Analysis . . . . .	51
3.2.5	Simulation . . . . .	55
3.3	Extension to Trajectory Tracking . . . . .	59
3.3.1	Formulation of a Time-Invariant Optimal Control Problem . . . . .	59
3.3.2	Approximate Optimal Solution . . . . .	61
3.3.3	Stability Analysis . . . . .	63
3.3.4	Simulation . . . . .	67
3.4	<i>N</i> -Player Nonzero-Sum Differential Games . . . . .	73
3.4.1	Problem Formulation . . . . .	74
3.4.2	Hamilton–Jacobi Approximation Via Actor-Critic-Identifier . . . . .	75
3.4.3	System Identifier . . . . .	76
3.4.4	Actor-Critic Design . . . . .	80
3.4.5	Stability Analysis . . . . .	82
3.4.6	Simulations . . . . .	88
3.5	Background and Further Reading . . . . .	91
	References . . . . .	95
<b>4</b>	<b>Model-Based Reinforcement Learning for Approximate Optimal Control . . . . .</b>	<b>99</b>
4.1	Introduction . . . . .	99
4.2	Model-Based Reinforcement Learning . . . . .	101
4.3	Online Approximate Regulation . . . . .	103
4.3.1	System Identification . . . . .	103
4.3.2	Value Function Approximation . . . . .	104
4.3.3	Simulation of Experience Via Bellman Error Extrapolation . . . . .	105

4.3.4	Stability Analysis . . . . .	107
4.3.5	Simulation . . . . .	110
4.4	Extension to Trajectory Tracking . . . . .	118
4.4.1	Problem Formulation and Exact Solution . . . . .	118
4.4.2	Bellman Error . . . . .	119
4.4.3	System Identification . . . . .	120
4.4.4	Value Function Approximation . . . . .	121
4.4.5	Simulation of Experience . . . . .	122
4.4.6	Stability Analysis . . . . .	123
4.4.7	Simulation . . . . .	125
4.5	<i>N</i> -Player Nonzero-Sum Differential Games . . . . .	131
4.5.1	System Identification . . . . .	132
4.5.2	Model-Based Reinforcement Learning . . . . .	133
4.5.3	Stability Analysis . . . . .	135
4.5.4	Simulation . . . . .	140
4.6	Background and Further Reading . . . . .	144
	References . . . . .	145
<b>5</b>	<b>Differential Graphical Games . . . . .</b>	<b>149</b>
5.1	Introduction . . . . .	149
5.2	Cooperative Formation Tracking Control of Heterogeneous Agents . . . . .	151
5.2.1	Graph Theory Preliminaries . . . . .	151
5.2.2	Problem Formulation . . . . .	151
5.2.3	Elements of the Value Function . . . . .	153
5.2.4	Optimal Formation Tracking Problem . . . . .	153
5.2.5	System Identification . . . . .	158
5.2.6	Approximation of the Bellman Error and the Relative Steady-State Controller . . . . .	159
5.2.7	Value Function Approximation . . . . .	160
5.2.8	Simulation of Experience via Bellman Error Extrapolation . . . . .	161
5.2.9	Stability Analysis . . . . .	163
5.2.10	Simulations . . . . .	166
5.3	Reinforcement Learning-Based Network Monitoring . . . . .	180
5.3.1	Problem Description . . . . .	180
5.3.2	System Identification . . . . .	182
5.3.3	Value Function Approximation . . . . .	184
5.3.4	Stability Analysis . . . . .	188
5.3.5	Monitoring Protocol . . . . .	188
5.4	Background and Further Reading . . . . .	189
	References . . . . .	191

<b>6 Applications</b>	195
6.1 Introduction	195
6.2 Station-Keeping of a Marine Craft	196
6.2.1 Vehicle Model	196
6.2.2 System Identifier	198
6.2.3 Problem Formulation	200
6.2.4 Approximate Policy	203
6.2.5 Stability Analysis	205
6.2.6 Experimental Validation	207
6.3 Online Optimal Control for Path-Following	213
6.3.1 Problem Description	213
6.3.2 Optimal Control and Approximate Solution	215
6.3.3 Stability Analysis	215
6.3.4 Simulation Results	218
6.3.5 Experiment Results	220
6.4 Background and Further Reading	223
References	224
<b>7 Computational Considerations</b>	227
7.1 Introduction	227
7.2 Reproducing Kernel Hilbert Spaces	230
7.3 StaF: A Local Approximation Method	232
7.3.1 The StaF Problem Statement	232
7.3.2 Feasibility of the StaF Approximation and the Ideal Weight Functions	233
7.3.3 Explicit Bound for the Exponential Kernel	235
7.3.4 The Gradient Chase Theorem	237
7.3.5 Simulation for the Gradient Chase Theorem	240
7.4 Local Approximation for Efficient Model-Based Reinforcement Learning	242
7.4.1 StaF Kernel Functions	242
7.4.2 StaF Kernel Functions for Online Approximate Optimal Control	243
7.4.3 Analysis	246
7.4.4 Extension to Systems with Uncertain Drift Dynamics	252
7.4.5 Simulation	253
7.5 Background and Further Reading	260
References	261
<b>Appendix A: Supplementary Lemmas and Definitions</b>	265
<b>Index</b>	291

# Symbols

Lists of abbreviations and symbols used in definitions, lemmas, theorems, and the development in the subsequent chapters.

$\mathbb{R}$	Set of real numbers
$\mathbb{R}_{\geq (\leq) a}$	Set of real numbers greater (less) than or equal to $a$
$\mathbb{R} > (<) a$	Set of real numbers strictly greater (less) than $a$
$\mathbb{R}^n$	$n$ -dimensional real Euclidean space
$\mathbb{R}^{n \times m}$	The space of $n \times m$ matrices of real numbers
$\mathbb{C}^n$	$n$ -dimensional complex Euclidean space
$\mathcal{C}^n(\mathcal{D}_1, \mathcal{D}_2)$	The space of $n$ -times continuously differentiable functions with domain $\mathcal{D}_1$ and codomain $\mathcal{D}_2$ , and the domain and the codomain are suppressed when clear from the context
$I_n$	$n \times n$ Identity matrix
$\mathbf{0}_{n \times n}$	$n \times n$ Matrix of zeros
$\mathbf{1}_{n \times n}$	$n \times n$ Matrix of ones
$\text{diag}\{x_1, \dots, x_n\}$	Diagonal matrix with $x_1, \dots, x_n$ on the diagonal
$\in$	Belongs to
$\forall$	For all
$\subset$	Subset of
$\triangleq$	Equals by definition
$f : \mathcal{D}_1 \rightarrow \mathcal{D}_2$	A function $f$ with domain $\mathcal{D}_1$ and codomain $\mathcal{D}_2$
$\rightarrow$	Approaches
$\mapsto$	Maps to
$\Rightarrow$	Implies that
$*$	Convolution operator
$  \cdot  $	Absolute value
$\  \cdot \ $	Euclidean norm
$\  \cdot \ _F$	Frobenius norm, $\  \theta \ _F = \sqrt{\text{tr}(\theta^T \theta)}$
$\  \cdot \ _\infty$	Induced infinity norm

$\lambda_{\min}$	Minimum eigenvalue
$\lambda_{\max}$	Maximum eigenvalue
$\dot{x}, \ddot{x}, \dots, x^{(i)}$	First, second, ..., $i$ th time derivative of $x$
$\frac{\partial f(x,y,\dots)}{\partial y}$	Partial derivative of $f$ with respect to $y$
$\nabla_y f(x,y,\dots)$	Gradient of $f$ with respect to $y$
$\nabla f(x,y,\dots)$	Gradient of $f$ with respect to the first argument
$B_r$	The ball $x \in \mathbb{R}^n \mid \ x\  < r$
$B_r(y)$	The ball $x \in \mathbb{R}^n \mid \ x - y\  < r$
$\bar{A}$	Closure of a set $A$
$\text{int}(A)$	Interior of a set $A$
$\partial(A)$	Boundary of a set $A$
$\mathbf{1}_A$	Indicator function of a set $A$
$L_\infty(\mathcal{D}_1, \mathcal{D}_2)$	Space of uniformly essentially bounded functions with domain $\mathcal{D}_1$ and codomain $\mathcal{D}_2$ , and the domain and the codomain are suppressed when clear from the context
$\text{sgn}(\cdot)$	Vector and scalar signum function
$\text{tr}(\cdot)$	Trace of a matrix
$\text{vec}(\cdot)$	Stacks the columns of a matrix to form a vector
$\text{proj}(\cdot)$	A smooth projection operator
$[\cdot]^{\times}$	Skew-symmetric cross product matrix

# Chapter 1

## Optimal Control



### 1.1 Introduction

The ability to learn behaviors from interactions with the environment is a desirable characteristic of a cognitive agent. Typical interactions between an agent and its environment can be described in terms of actions, states, and rewards (or penalties). Actions executed by the agent affect the state of the system (i.e., the agent and the environment), and the agent is presented with a reward (or a penalty). Assuming that the agent chooses an action based on the state of the system, the behavior (or the policy) of the agent can be described as a map from the state-space to the action-space.

Desired behaviors can be learned by adjusting the agent-environment interaction through the rewards/penalties. Typically, the rewards/penalties are qualified by a cost. For example, in many applications, the correctness of a policy is often quantified in terms of the Lagrange cost and the Mayer cost. The Lagrange cost is the cumulative penalty accumulated along a path traversed by the agent and the Mayer cost is the penalty at the boundary. Policies with lower total cost are considered better and policies that minimize the total cost are considered optimal. The problem of finding the optimal policy that minimizes the total Lagrange and Meyer cost is known as the Bolza optimal control problem.

### 1.2 Notation

Throughout the book, unless otherwise specified, the domain of all the functions is assumed to be  $\mathbb{R}_{\geq 0}$ . Function names corresponding to state and control trajectories are reused to denote elements in the range of the function. For example, the notation  $u(\cdot)$  is used to denote the function  $u : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^m$ , the notation  $u$  is used to denote an arbitrary element of  $\mathbb{R}^m$ , and the notation  $u(t)$  is used to denote the value of the function  $u(\cdot)$  evaluated at time  $t$ . Unless otherwise specified, all the mathematical quantities are assumed to be time-varying, an equation of the form  $g(x) = f + h(y, t)$  is interpreted as  $g(x(t)) = f(t) + h(y(t), t)$  for all  $t \in \mathbb{R}_{\geq 0}$ , and a definition of the form  $g(x, y) \triangleq f(y) + h(x)$  for functions  $g : A \times B \rightarrow C$ ,  $f : B \rightarrow C$  and

$h : A \rightarrow C$  is interpreted as  $g(x, y) \triangleq f(y) + h(x)$ ,  $\forall (x, y) \in A \times B$ . The notation  $\overline{\|h\|}^\chi$  denotes  $\sup_{\xi \in \chi} \|h(\xi)\|$ , for a continuous function  $h : \mathbb{R}^n \rightarrow \mathbb{R}^k$  and a compact set  $\chi$ . When the compact set is clear from the context, the notation  $\overline{\|h\|}$  is utilized.

### 1.3 The Bolza Problem

Consider a controlled dynamical system described by the initial value problem

$$\dot{x}(t) = f(x(t), u(t), t), \quad x(t_0) = x_0, \quad (1.1)$$

where  $t_0$  is the initial time,  $x : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  denotes the system state and  $u : \mathbb{R}_{\geq t_0} \rightarrow U \subset \mathbb{R}^m$  denotes the control input, and  $U$  denotes the action-space.

To ensure local existence and uniqueness of Carathéodory solutions to (1.1), it is assumed that the function  $f : \mathbb{R}^n \times U \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is continuous with respect to  $t$  and  $u$ , and continuously differentiable with respect to  $x$ . Furthermore, the control signal,  $u(\cdot)$ , is restricted to be piecewise continuous. The assumptions stated here are sufficient but not necessary to ensure local existence and uniqueness of Carathéodory solutions to (1.1). For further discussion on existence and uniqueness of Carathéodory solutions, see [1, 2]. Further restrictions on the dynamical system are stated, when necessary, in subsequent chapters.

Consider a fixed final time optimal control problem where the optimality of a control policy is quantified in terms of a cost functional

$$J(t_0, x_0, u(\cdot)) = \int_{t_0}^{t_f} L(x(t; t_0, x_0, u(\cdot)), u(t), t) dt + \Phi(x_f), \quad (1.2)$$

where  $L : \mathbb{R}^n \times U \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$  is the Lagrange cost,  $\Phi : \mathbb{R}^n \rightarrow \mathbb{R}$  is the Mayer cost, and  $t_f$  and  $x_f \triangleq x(t_f)$  denote the final time and state, respectively. In (1.2), the notation  $x(t; t_0, x_0, u(\cdot))$  is used to denote a trajectory of the system in (1.1), evaluated at time  $t$ , under the controller  $u(\cdot)$ , starting at the initial time  $t_0$ , and with the initial state  $x_0$ . Similarly, for a given policy  $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^n$ , the short notation  $x(t; t_0, x_0, \phi(x(\cdot)))$  is used to denote a trajectory under the feedback controller  $u(t) = \phi(x(t; t_0, x_0, u(\cdot)))$ . Throughout the book, the symbol  $x$  is also used to denote generic initial conditions in  $\mathbb{R}^n$ . Furthermore, when the controller, the initial time, and the initial state are understood from the context, the shorthand  $x(\cdot)$  is used when referring to the entire trajectory, and the shorthand  $x(t)$  is used when referring to the state of the system at time  $t$ .

The two most popular approaches to solve Bolza problems are Pontryagin's maximum principle and dynamic programming. The two approaches are independent, both conceptually and in terms of their historic development. Both the approaches are developed on the foundation of calculus of variations, which has its origins in

Newton's Minimal Resistance Problem dating back to 1685 and Johann Bernoulli's Brachistochrone problem dating back to 1696. The maximum principle was developed by the Pontryagin school at the Steklov Institute in the 1950s [3]. The development of dynamic programming methods was simultaneously but independently initiated by Bellman at the RAND Corporation [4]. While Pontryagin's maximum principle results in optimal control methods that generate optimal state and control trajectories starting from a specific state, dynamic programming results in methods that generate optimal policies (i.e., they determine the optimal decision to be made at any state of the system).

Barring some comparative remarks, the rest of this monograph will focus on the dynamic programming approach to solve Bolza problems. The interested reader is directed to the books by Kirk [5], Bryson and Ho [6], Liberzon [7], and Vinter [8] for an in-depth discussion of Pontryagin's maximum principle.

## 1.4 Dynamic Programming

Dynamic programming methods generalize the Bolza problem. Instead of solving the fixed final time Bolza problem for particular values of  $t_0$ ,  $t_f$ , and  $x$ , a family of Bolza problems characterized by the cost functionals

$$J(t, x, u(\cdot)) = \int_t^{t_f} L(x(\tau; t, x, u(\cdot)), u(\tau), \tau) d\tau + \Phi(x_f) \quad (1.3)$$

is solved, where  $t \in [t_0, t_f]$ ,  $t_f \in \mathbb{R}_{\geq 0}$ , and  $x \in \mathbb{R}^n$ . A solution to the family of Bolza problems in (1.3) can be characterized using the optimal cost-to-go function (i.e., the optimal value function)  $V^* : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ , defined as

$$V^*(x, t) \triangleq \inf_{u_{[t, t_f]}} J(t, x, u(\cdot)), \quad (1.4)$$

where the notation  $u_{[t, \tau]}$  for  $\tau \geq t \geq t_0$  denotes the controller  $u(\cdot)$  restricted to the time interval  $[t, \tau]$ .

### 1.4.1 Necessary Conditions for Optimality

In the subsequent development, a set of necessary conditions for the optimality of the value function are developed based on Bellman's principle of optimality.

**Theorem 1.1** [7, p. 160] *The value function,  $V^*$ , satisfies the principle of optimality. That is, for all  $(x, t) \in \mathbb{R}^n \times [t_0, t_f]$ , and for all  $\Delta t \in (0, t_f - t]$ ,*

$$V^*(x, t) = \inf_{u_{[t, t+\Delta t]}} \left\{ \int_t^{t+\Delta t} L(x(\tau), u(\tau), \tau) d\tau + V^*(x(t + \Delta t), t + \Delta t) \right\}. \quad (1.5)$$

*Proof* Consider the function  $V : \mathbb{R}^n \times [t_0, t_f] \rightarrow \mathbb{R}$  defined as

$$V(x, t) \triangleq \inf_{u_{[t, t+\Delta t]}} \left\{ \int_t^{t+\Delta t} L(x(\tau), u(\tau), \tau) d\tau + V^*(x(t + \Delta t), t + \Delta t) \right\}.$$

Based on the definition in (1.4)

$$V(x, t) = \inf_{u_{[t, t+\Delta t]}} \left\{ \int_t^{t+\Delta t} L(x(\tau), u(\tau), \tau) d\tau + \inf_{u_{[t+\Delta t, t_f]}} J(t + \Delta t, x(t + \Delta t), u(\cdot)) \right\}.$$

Using (1.3) and combining the integrals,

$$V(x, t) = \inf_{u_{[t, t+\Delta t]}} \left\{ \inf_{u_{[t+\Delta t, t_f]}} J(t, x, u(\cdot)) \right\} \geq \inf_{u_{[t, t_f]}} J(t, x, u(\cdot)) = V^*(x, t). \quad (1.6)$$

Thus,  $V(x, t) \geq V^*(x, t)$ . On the other hand, by the definition of the infimum, for all  $\epsilon > 0$ , there exists a controller  $u_\epsilon(\cdot)$  such that

$$V^*(x, t) + \epsilon \geq J(t, x, u_\epsilon(\cdot)).$$

Let  $x_\epsilon$  denote the trajectory corresponding to  $u_\epsilon$ . Then,

$$\begin{aligned} J(t, x, u_\epsilon) &= \int_t^{t+\Delta t} L(x_\epsilon(\tau), u_\epsilon(\tau), \tau) d\tau + J(t + \Delta t, x_\epsilon(t + \Delta t), u_\epsilon), \\ &\geq \int_t^{t+\Delta t} L(x_\epsilon(\tau), u_\epsilon(\tau), \tau) d\tau + V(x_\epsilon(t + \Delta t), t + \Delta t) \geq V(x, t). \end{aligned}$$

Thus,  $V(x, t) \leq V^*(x, t)$ , which, along with (1.6), implies  $V(x, t) = V^*(x, t)$ .  $\square$

Under the assumption that  $V^* \in \mathcal{C}^1(\mathbb{R}^n \times [t_0, t_f], \mathbb{R})$ , the optimal value function can be shown to satisfy

$$0 = -\nabla_t V^*(x, t) - \inf_{u \in U} \{L(x, u, t) + \nabla_x V^{*T}(x, t) f(x, u, t)\},$$

for all  $t \in [t_0, t_f]$  and all  $x \in \mathbb{R}^n$ , with the boundary condition  $V^*(x, t_f) = \Phi(x)$ , for all  $x \in \mathbb{R}^n$ . In fact, the Hamilton–Jacobi–Bellman equation along with a Hamiltonian maximization condition completely characterize the solution to the family of Bolza problems.

### 1.4.2 Sufficient Conditions for Optimality

Theorem 1.2 presents necessary and sufficient conditions for a function to be the optimal value function.

**Theorem 1.2** *Let  $V^* \in C^1(\mathbb{R}^n \times [t_0, t_f], \mathbb{R})$  denote the optimal value function. Then, a function  $V : \mathbb{R}^n \times [t_0, t_f] \rightarrow \mathbb{R}$  is the optimal value function (i.e.,  $V(x, t) = V^*(x, t)$  for all  $(x, t) \in \mathbb{R}^n \times [t_0, t_f]$ ) if and only if:*

1.  $V \in C^1(\mathbb{R}^n \times [t_0, t_f], \mathbb{R})$  and  $V$  satisfies the Hamilton–Jacobi–Bellman equation

$$0 = -\nabla_t V(x, t) - \inf_{u \in U} \{L(x, u, t) + \nabla_x V^T(x, t) f(x, u, t)\}, \quad (1.7)$$

for all  $t \in [t_0, t_f]$  and all  $x \in \mathbb{R}^n$ , with the boundary condition  $V(x, t_f) = \Phi(x)$ , for all  $x \in \mathbb{R}^n$ .

2. For all  $x \in \mathbb{R}^n$ , there exists a controller  $u(\cdot)$ , such that the function  $V$ , the controller  $u(\cdot)$ , and the trajectory  $x(\cdot)$  of (1.1) under  $u(\cdot)$  with the initial condition  $x(t_0) = x$ , satisfy the equation

$$\begin{aligned} L(x(t), u(t), t) + \nabla_x V^T(x(t), t) f(x(t), u(t), t) \\ = \min_{\hat{u} \in U} \{L(x(t), \hat{u}, t) + \nabla_x V^T(x(t), t) f(x(t), \hat{u}, t)\}, \end{aligned} \quad (1.8)$$

for all  $t \in [t_0, t_f]$ .

*Proof* See [7, Sect. 5.1.4]. □

## 1.5 The Unconstrained Affine-Quadratic Regulator

The focus of this monograph is on unconstrained infinite-horizon total cost Bolza problems for nonlinear systems that are affine in the controller and cost functions that are quadratic in the controller. That is, optimal control problems where the system dynamics are of the form

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \quad (1.9)$$

where  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$  and  $g : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m}$  are locally Lipschitz functions, and the cost functional is of the form

$$J(t_0, x_0, u(\cdot)) = \int_{t_0}^{\infty} r(x(\tau; t_0, x_0, u(\cdot)), u(\tau)) d\tau, \quad (1.10)$$

where the local cost  $r : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}$  is defined as

$$r(x, u) \triangleq Q(x) + u^T R u, \quad (1.11)$$

where  $Q : \mathbb{R}^n \rightarrow \mathbb{R}$  is a positive definite function and  $R \in \mathbb{R}^{m \times m}$  is a symmetric positive definite matrix.

To ensure that the optimal control problem is well-posed, the minimization problem is constrained to the set of admissible controllers (see [9, Definition 1]), and the existence of at least one admissible controller is assumed. It is further assumed that the optimal control problem has a continuously differentiable value function. This assumption is valid for a large class of problems. For example, most unconstrained infinite horizon optimal control problems with smooth data have smooth value functions. However, there is a large class of relevant optimal control problems for which the assumption fails. For example, problems with bounded controls and terminal costs typically have nondifferentiable value functions. Dynamic programming-based solutions to such problems are characterized by viscosity solutions to the corresponding Hamilton–Jacobi–Bellman equation. For further details on viscosity solutions to Hamilton–Jacobi–Bellman equations, the reader is directed to [10] and [11].

Provided the aforementioned assumptions hold, the optimal value function is time-independent, That is,

$$V^*(x) \triangleq \inf_{u_{[t, \infty]}} J(t, x, u(\cdot)), \quad (1.12)$$

for all  $t \in \mathbb{R}_{\geq t_0}$ . Furthermore, the Hamiltonian minimization condition in (1.8) is satisfied by the controller  $u(t) = u^*(x(t))$ , where the policy  $u^* : \mathbb{R}^n \rightarrow \mathbb{R}^m$  is defined as

$$u^*(x) = -\frac{1}{2} R^{-1} g^T(x) (\nabla_x V^*(x))^T. \quad (1.13)$$

Hence, assuming that an optimal controller exists, a complete characterization of the solution to the optimal control problem can be obtained using the Hamilton–Jacobi–Bellman equation.

*Remark 1.3* While infinite horizon optimal control problems naturally arise in feedback control application where stability is of paramount importance, path planning applications often involve finite-horizon optimal control problems. The method of

dynamic programming has extensively been studied for finite horizon problems [12–20], although such problems are out of the scope of this monograph.

*Remark 1.4* The control-affine model in (1.9) is applicable to a wide variety of electro-mechanical systems. In particular, any linear system and any Euler-Lagrange nonlinear system that has a known and invertible inertia matrix can be modeled using a control-affine model. Examples include industrial manipulators, fully actuated autonomous underwater and air vehicles (where the range of operation does not include singular configurations), kinematic wheels, etc. Computation of the policy in (1.13) exploits the control-affine nature of the dynamics, and knowledge of the control effectiveness function,  $g$ , is required to implement the policy. The methods detailed in this monograph can be extended to systems with uncertain control effectiveness functions and to nonaffine systems (cf. [21–28]).

The following theorem fully characterizes solutions to optimal control problems for affine systems.

**Theorem 1.5** *For a nonlinear system described by (1.9),  $V^* \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R})$  is the optimal value function corresponding to the cost functional (1.10) if and only if it satisfies the Hamilton–Jacobi–Bellman equation*

$$r(x, u^*(x)) + \nabla_x V^*(x)(f(x) + g(x)u^*(x)) = 0, \quad \forall x \in \mathbb{R}^n, \quad (1.14)$$

with the boundary condition  $V(0) = 0$ . Furthermore, the optimal controller can be expressed as the state-feedback law  $u(t) = u^*(x(t))$ .

*Proof* For each  $x \in \mathbb{R}^n$  we have

$$\frac{\partial(r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u))}{\partial u} = 2u^T R + \nabla_x V^*(x)g(x).$$

hence,  $u = -\frac{1}{2}R^{-1}g^T(x)(\nabla_x V^*(x))^T = u^*(x)$  extremizes  $r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u)$ . Furthermore, the Hessian

$$\frac{\partial^2(r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u))}{\partial^2 u} = 2R$$

is positive definite. Hence,  $u = u^*(x)$  minimizes  $r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u)$ . Hence, Eq.(1.14) is equivalent to the conditions in (1.7) and (1.8).  $\square$

## 1.6 Input Constraints

The Bolza problem detailed in the previous section is an unconstrained optimal control problem. In practice, actuators are limited in the amount of control effort they can produce. Let  $u_i$  denote the  $i^{\text{th}}$  component of the control vector  $u$ . The

affine-quadratic formulation can be extended to systems with actuator constraints of the form  $|u_i(t)| \leq \bar{u}$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ ,  $\forall i = 1, \dots, m$  using a non-quadratic penalty function first introduced in [29].

Let  $\psi : \mathbb{R} \rightarrow \mathbb{R}$  be a strictly monotonically increasing continuously differentiable function such that  $\operatorname{sgn}(\psi(a)) = \operatorname{sgn}(a)$ ,  $\forall a \in \mathbb{R}$ , and  $|\psi(a)| \leq \bar{u}$  (e.g.,  $\psi(a) = \tanh(a)$ ). Consider a cost function of the form  $r(x, u) = Q(x) + U(u)$ , where

$$U(u) \triangleq 2 \sum_{i=1}^m r_i \left( \int_0^{u_i} \psi^{-1}(\xi) d\xi \right), \quad (1.15)$$

and  $r_i$  denotes the  $i^{\text{th}}$  diagonal element of the matrix  $R$ .

The following theorem characterizes the solutions to optimal control problems for affine systems with actuation constraints.

**Theorem 1.6** *For a nonlinear system described by (1.9),  $V^* \in \mathcal{C}^1(\mathbb{R}^n, \mathbb{R})$  is the optimal value function corresponding to the cost functional in (1.10), with the control penalty in (1.15), if and only if it satisfies the Hamilton–Jacobi–Bellman equation*

$$r(x, \phi(x)) + \nabla_x V^*(x)(f(x) + g(x)\phi(x)) = 0, \quad \forall x \in \mathbb{R}^n, \quad (1.16)$$

with the boundary condition  $V^*(0) = 0$ , where  $\phi(x) \triangleq -\psi\left(\frac{1}{2}R^{-1}g^T(x)(\nabla_x V^*(x))^T\right)$ . Furthermore, the optimal controller can be expressed as the state-feedback law  $u(t) = \bar{u}^*(x(t))$ , where

$$\bar{u}^*(x) \triangleq -\psi\left(\frac{1}{2}R^{-1}g^T(x)(\nabla_x V^*(x))^T\right).$$

*Proof* For each  $x \in \mathbb{R}^n$ ,

$$\frac{\partial(r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u))}{\partial u} = 2\psi^{-1}(u^T)R + \nabla_x V^*(x)g(x).$$

hence,  $u = -\psi\left(\frac{1}{2}R^{-1}g^T(x)(\nabla_x V^*(x))^T\right)$  extremizes  $r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u)$ . Furthermore, the Hessian is

$$\frac{\partial^2(r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u))}{\partial^2 u} = 2R \begin{bmatrix} \nabla_{u_1} \psi^{-1}(u_1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \nabla_{u_m} \psi^{-1}(u_m) \end{bmatrix}.$$

Provided the function  $\psi$  is strictly monotonically increasing, the Hessian is positive definite. Hence,  $u = -\psi\left(\frac{1}{2}R^{-1}g^T(x)(\nabla_x V^*(x))^T\right)$  minimizes  $r(x, u) + \nabla_x V^*(x)(f(x) + g(x)u)$ .  $\square$

## 1.7 Connections with Pontryagin's Maximum Principle

To apply Pontryagin's maximum principle to the unconstrained affine-quadratic regulator, define the Hamiltonian  $H : \mathbb{R}^n \times U \times \mathbb{R}^n \rightarrow \mathbb{R}$  as

$$H(x, u, p) = p^T (f(x) + g(x)u) - r(x, u).$$

Pontryagin's maximum principle provides the following necessary condition for optimality.

**Theorem 1.7.** [3, 5, 7] Let  $x^* : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  and  $u^* : \mathbb{R}_{\geq t_0} \rightarrow U$  denote the optimal state and control trajectories corresponding to the optimal control problem in Sect. 1.5. Then there exists a trajectory  $p^* : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  such that  $p^*(t) \neq 0$  for some  $t \in \mathbb{R}_{\geq t_0}$  and  $x^*$  and  $p^*$  satisfy the equations

$$\begin{aligned}\dot{x}^*(t) &= (\nabla_p H(x^*(t), u^*(t), p^*(t)))^T, \\ \dot{p}^*(t) &= -(\nabla_x H(x^*(t), u^*(t), p^*(t)))^T,\end{aligned}$$

with the boundary condition  $x^*(t_0) = x_0$ . Furthermore, the Hamiltonian satisfies

$$H(x^*(t), u^*(t), p^*(t)) \geq H(x^*(t), u, p^*(t)), \quad (1.17)$$

for all  $t \in \mathbb{R}_{\geq t_0}$  and  $u \in U$ , and

$$H(x^*(t), u^*(t), p^*(t)) = 0, \quad (1.18)$$

for all  $t \in \mathbb{R}_{\geq t_0}$ .

*Proof* See, e.g., [7, Sect. 4.2]. □

Under further assumptions on the state and the control trajectories, and on the functions  $f$ ,  $g$ , and  $r$ , the so-called *natural* transversality condition  $\lim_{t \rightarrow \infty} p(t) = 0$  can be obtained (cf. [30–32]). The natural transversality condition does not hold in general for infinite horizon optimal control problems. For some illustrative counterexamples and further discussion, see [30–35].

A quick comparison of Eq.(1.14) and (1.18) suggests that the optimal costate should satisfy

$$p^*(t) = -(\nabla_x V(x^*(t)))^T. \quad (1.19)$$

Differentiation of (1.19) with respect to time yields

$$\dot{p}^*(t) = -\nabla_x (\nabla_x V(x^*(t)))^T (f(x^*(t)) + g(x^*(t))u(t)).$$

Differentiation of (1.14) with respect to the state yields

$$(f(x^*) + g(x^*) u)^T \nabla_x (\nabla_x V(x^*))^T = -\nabla_x V(x^*) (\nabla_x f(x^*) + \nabla_x g(x^*) u) - \nabla_x r(x^*, u^*).$$

Provided the second derivatives are continuous, then  $\nabla_x (\nabla_x V(x^*))^T = (\nabla_x (\nabla_x V(x^*))^T)^T$ . Hence, the time derivative of the costate can be computed as

$$\begin{aligned}\dot{p}^*(t) &= -(\nabla_x (f(x^*(t)) + g(x^*(t)) u(t)))^T (\nabla_x V(x^*(t)))^T - (\nabla_x r(x^*(t), u^*(t)))^T, \\ &= -(\nabla_x H(x^*(t), u^*(t), p^*(t)))^T.\end{aligned}$$

Therefore, the expression of the costate in (1.19) satisfies Theorem 1.7. The relationship in (1.19) implies that the costate is the sensitivity of the optimal value function to changes in the system state trajectory. Furthermore, the Hamiltonian maximization conditions in (1.8) and (1.17) are equivalent. Dynamic programming and Pontryagin's maximum principle methods are therefore closely related. However, there are a few key differences between the two methods.

The solution in (1.13) obtained using dynamics programming is a feedback law. That is, dynamic programming can be used to generate a policy that can be used to close the control loop. Furthermore, once the Hamilton–Jacobi–Bellman equation is solved, the resulting feedback law is guaranteed to be optimal for any initial condition of the dynamical system. On the other hand, Pontryagin's maximum principle generates the optimal state, costate, and control trajectories for a given initial condition. The controller must be implemented in an open-loop manner. Furthermore, if the initial condition changes, the optimal solution is no longer valid and the optimal control problem needs to be solved again.

Since dynamic programming generates a feedback law, it provides much more information than the maximum principle. However, the added benefit comes at a heavy computational cost. To generate the optimal policy, the Hamilton–Jacobi–Bellman partial differential equation must be solved. In general, numerical methods to solve the Hamilton–Jacobi–Bellman equation grow exponentially in numerical complexity with increasing dimensionality. That is, dynamic programming suffers from the so-called Bellman's curse of dimensionality.

## 1.8 Further Reading

### 1.8.1 Numerical Methods

One way to develop optimal controllers for general nonlinear systems is to use numerical methods [5]. A common approach is to formulate the optimal control problem in terms of a Hamiltonian and then to numerically solve a two point boundary value problem for the state and co-state equations [36, 37]. Another approach is to cast the optimal control problem as a nonlinear programming problem via direct transcription and then solve the resulting nonlinear program [30, 38–42]. Numerical methods are offline, do not generally guarantee stability, or optimality, and are often

open-loop. These issues motivate the desire to find an analytical solution. Developing analytical solutions to optimal control problems for linear systems is complicated by the need to solve an algebraic Riccati equation or a differential Riccati equation. Developing analytical solutions for nonlinear systems is even further complicated by the sufficient condition of solving a Hamilton–Jacobi–Bellman partial differential equation, where an analytical solution may not exist in general. If the nonlinear dynamics are exactly known, then the problem can be simplified at the expense of optimality by solving an algebraic Riccati equations through feedback-linearization methods (cf. [43–47]).

Alternatively, some investigators temporarily assume that the uncertain system could be feedback-linearized, solve the resulting optimal control problem, and then use adaptive/learning methods to asymptotically learn the uncertainty [48–51] (i.e., asymptotically converge to the optimal controller). The nonlinear optimal control problem can also be solved using inverse optimal control [52–61] by circumventing the need to solve the Hamilton–Jacobi–Bellman equation. By finding a control Lyapunov function, which can be shown to also be a value function, an optimal controller can be developed that optimizes a derived cost. However, since the cost is derived rather than specified by mission/task objectives, this approach is not explored in this monograph. Optimal control-based algorithms such as state dependent Riccati equations [62–65] and model-predictive control [66–72] have been widely utilized for control of nonlinear systems. However, both state dependent Riccati equations and model-predictive control are inherently model-based. Furthermore, due to nonuniqueness of state dependent linear factorization in state dependent Riccati equations-based techniques, and since the optimal control problem is solved over a small prediction horizon in model-predictive control, they generally result in suboptimal policies. Furthermore, model-predictive control approaches are computationally intensive, and closed-loop stability of state dependent Riccati equations-based methods is generally impossible to establish a priori and has to be established through extensive simulation.

### 1.8.2 Differential Games and Equilibrium Solutions

A multitude of relevant control problems can be modeled as multi-input systems, where each input is computed by a player, and each player attempts to influence the system state to minimize its own cost function. In this case, the optimization problem for each player is coupled with the optimization problem for other players. Hence, in general, an optimal solution in the usual sense does not exist for such problems, motivating the formulation of alternative optimality criteria.

Differential game theory provides solution concepts for many multi-player, multi-objective optimization problems [73–75]. For example, a set of policies is called a Nash equilibrium solution to a multi-objective optimization problem if none of the players can improve their outcome by changing their policy while all the other players abide by the Nash equilibrium policies [76]. Thus, Nash equilibrium solutions

provide a secure set of strategies, in the sense that none of the players have an incentive to diverge from their equilibrium policy. Hence, Nash equilibrium has been a widely used solution concept in differential game-based control techniques. For an in-depth discussion on Nash equilibrium solutions to differential game problems, see Chaps. 3 and 4.

Differential game theory is also employed in multi-agent optimal control, where each agent has its own decentralized objective and may not have access to the entire system state. In this case, graph theoretic models of the information structure are utilized in a differential game framework to formulate coupled Hamilton–Jacobi equations (c.f. [77]). Since the coupled Hamilton–Jacobi equations are difficult to solve, reinforcement learning is often employed to get an approximate solution. Results such as [77, 78] indicate that adaptive dynamic programming can be used to generate approximate optimal policies online for multi-agent systems. For an in-depth discussion on the use of graph theoretic models of information structure in a differential game framework, see Chap. 5

### 1.8.3 Viscosity Solutions and State Constraints

A significant portion of optimal control problems of practical importance require the solution to satisfy state constraints. For example, autonomous vehicles operating in complex contested environments are required to observe strict static (e.g., due to policy or mission objectives or known obstacles/structures in the environment) and dynamic (e.g., unknown and then sensed obstacles, moving obstacles) no-entry zones. The value functions corresponding to optimal control problems with state constraints are generally not continuously differentiable, and may not even be differentiable everywhere. Hence, for these problems, the Hamilton–Jacobi–Bellman equation fails to admit classical solutions, and alternative solution concepts are required. A naive generalization would be to require a function to satisfy the Hamilton–Jacobi–Bellman equation almost everywhere. However, the naive generalization is not useful for optimal control since such generalized solutions are often unrelated to the value function of the corresponding optimal control problem.

An appropriate notion of generalized solutions to the Hamilton–Jacobi–Bellman equation, called viscosity solutions, was developed in [10]. It has been established that under the condition that the value function is continuous, it is a solution to the Hamilton–Jacobi–Bellman equation. Some uniqueness results are also available under further assumptions on the value function. For a detailed treatment of viscosity solutions to Hamilton–Jacobi–Bellman equations, see [79].

Various methods have been developed to approximate viscosity solutions to Hamilton–Jacobi–Bellman equations [79–81]; however, these methods are offline, require knowledge of the system dynamics, and are computationally expensive. Online computation of approximate classical solutions to the Hamilton–Jacobi–Bellman equation is achieved through dynamic programming methods. Dynamic programming methods in continuous state and time rely on a differential [82] or an

integral [83] formulation of the temporal difference error (called the Bellman error). The corresponding reinforcement learning algorithms are generally designed to minimize the Bellman error. Since such minimization yields estimates of generalized solutions, but not necessarily viscosity solutions, to the Hamilton–Jacobi–Bellman equation, reinforcement learning in continuous time and space for optimal control problems with state constraints has largely remained an open area of research.

## References

1. Carathéodory C (1918) Vorlesungen über reelle Funktionen. Teubner
2. Coddington EA, Levinson N (1955) Theory of ordinary differential equations. McGraw-Hill
3. Pontryagin LS, Boltyanskii VG, Gamkrelidze RV, Mishchenko EF (1962) The mathematical theory of optimal processes. Interscience, New York
4. Bellman R (1954) The theory of dynamic programming. Technical report, DTIC Document
5. Kirk D (2004) Optimal control theory: an introduction. Dover, Mineola
6. Bryson AE, Ho Y (1975) Applied optimal control: optimization, estimation, and control. Hemisphere Publishing Corporation
7. Liberzon D (2012) Calculus of variations and optimal control theory: a concise introduction. Princeton University Press
8. Vinter R (2010) Optimal control. Springer Science & Business Media
9. Beard R, Saridis G, Wen J (1997) Galerkin approximations of the generalized Hamilton–Jacobi–Bellman equation. *Automatica* 33:2159–2178
10. Crandall M, Lions P (1983) Viscosity solutions of Hamilton–Jacobi equations. *Trans Am Math Soc* 277(1):1–42
11. Bardi M, Dolcetta I (1997) Optimal control and viscosity solutions of Hamilton–Jacobi–Bellman equations. Springer
12. Cimen T, Banks SP (2004) Global optimal feedback control for general nonlinear systems with nonquadratic performance criteria. *Syst Control Lett* 53(5):327–346
13. Cheng T, Lewis FL, Abu-Khalaf M (2007) A neural network solution for fixed-final time optimal control of nonlinear systems. *Automatica* 43(3):482–490
14. Cheng T, Lewis FL, Abu-Khalaf M (2007) Fixed-final-time-constrained optimal control of nonlinear systems using neural network HJB approach. *IEEE Trans Neural Netw* 18(6):1725–1737
15. Kar I, Adhyaru D, Gopal M (2009) Fixed final time optimal control approach for bounded robust controller design using Hamilton–Jacobi–Bellman solution. *IET Control Theory Appl* 3(9):1183–1195
16. Wang F, Jin N, Liu D, Wei Q (2011) Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with epsilon-error bound. *IEEE Trans Neural Netw* 22:24–36
17. Heydari A, Balakrishnan SN (2012) An optimal tracking approach to formation control of nonlinear multi-agent systems. In: Proceedings of AIAA guidance, navigation and control conference
18. Wang D, Liu D, Wei Q (2012) Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach. *Neurocomputing* 78(1):14–22
19. Zhao Q, Xu H, Jagannathan S (2015) Neural network-based finite-horizon optimal control of uncertain affine nonlinear discrete-time systems. *IEEE Trans Neural Netw Learn Syst* 26(3):486–499
20. Li C, Liu D, Li H (2015) Finite horizon optimal tracking control of partially unknown linear continuous-time systems using policy iteration. *IET Control Theory Appl* 9(12):1791–1801

21. Ge SS, Zhang J (2003) Neural-network control of nonaffine nonlinear system with zero dynamics by state and output feedback. *IEEE Trans Neural Netw* 14(4):900–918
22. Wang D, Liu D, Wei Q, Zhao D, Jin N (2012) Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. *Automatica* 48(8):1825–1832
23. Zhang X, Zhang H, Sun Q, Luo Y (2012) Adaptive dynamic programming-based optimal control of unknown nonaffine nonlinear discrete-time systems with proof of convergence. *Neurocomputing* 91:48–55
24. Liu D, Huang Y, Wang D, Wei Q (2013) Neural-network-observer-based optimal control for unknown nonlinear systems using adaptive dynamic programming. *Int J Control* 86(9):1554–1566
25. Bian T, Jiang Y, Jiang ZP (2014) Adaptive dynamic programming and optimal control of nonlinear nonaffine systems. *Automatica* 50(10):2624–2632
26. Yang X, Liu D, Wei Q, Wang D (2015) Direct adaptive control for a class of discrete-time unknown nonaffine nonlinear systems using neural networks. *Int J Robust Nonlinear Control* 25(12):1844–1861
27. Kiumarsi B, Kang W, Lewis FL (2016)  $H_{\infty}$  control of nonaffine aerial systems using off-policy reinforcement learning. *Unmanned Syst* 4(1):1–10
28. Song R, Wei Q, Xiao W (2016) Off-policy neuro-optimal control for unknown complex-valued nonlinear systems based on policy iteration. *Neural Comput Appl* 46(1):85–95
29. Lyashevskiy S, Meyer AU (1995) Control system analysis and design upon the Lyapunov method. In: Proceedings of the American control conference, vol 5, pp 3219–3223
30. Fahroo F, Ross IM (2008) Pseudospectral methods for infinite-horizon nonlinear optimal control problems. *J Guid Control Dyn* 31(4):927–936
31. Pickenhain S (2014) Hilbert space treatment of optimal control problems with infinite horizon. In: Bock GH, Hoang PX, Rannacher R, Schlöder PJ (eds) Modeling, simulation and optimization of complex processes - HPSC 2012: Proceedings of the fifth international conference on high performance scientific computing, 5–9 March 2012, Hanoi, Vietnam. Springer International Publishing, Cham, pp 169–182
32. Tauchnitz N (2015) The pontryagin maximum principle for nonlinear optimal control problems with infinite horizon. *J Optim Theory Appl* 167(1):27–48
33. Halkin H (1974) Necessary conditions for optimal control problems with infinite horizons. *Econometrica* pp 267–272
34. Aseev SM, Kryazhimskii A (2007) The pontryagin maximum principle and optimal economic growth problems. *Proc Steklov Inst Math* 257(1):1–255
35. Aseev SM, Veliov VM (2015) Maximum principle for infinite-horizon optimal control problems under weak regularity assumptions. *Proc Steklov Inst Math* 291(1):22–39
36. von Stryk O, Bulirsch R (1992) Direct and indirect methods for trajectory optimization. *Ann Oper Res* 37(1):357–373
37. Betts JT (1998) Survey of numerical methods for trajectory optimization. *J Guid Control Dyn* 21(2):193–207
38. Hargraves CR, Paris S (1987) Direct trajectory optimization using nonlinear programming and collocation. *J Guid Control Dyn* 10(4):338–342
39. Huntington GT (2007) Advancement and analysis of a gauss pseudospectral transcription for optimal control. Ph.D thesis, Department of Aeronautics and Astronautics, MIT
40. Rao AV, Benson DA, Darby CL, Patterson MA, Francolin C, Huntington GT (2010) Algorithm 902: GPOPS, A MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method. *ACM Trans Math Softw* 37(2):1–39
41. Darby CL, Hager WW, Rao AV (2011) An hp-adaptive pseudospectral method for solving optimal control problems. *Optim Control Appl Methods* 32(4):476–502
42. Garg D, Hager WW, Rao AV (2011) Pseudospectral methods for solving infinite-horizon optimal control problems. *Automatica* 47(4):829–837
43. Freeman R, Kokotovic P (1995) Optimal nonlinear controllers for feedback linearizable systems. In: Proceedings of the American control conference, pp 2722–2726

44. Lu Q, Sun Y, Xu Z, Mochizuki T (1996) Decentralized nonlinear optimal excitation control. *IEEE Trans Power Syst* 11(4):1957–1962
45. Nevistic V, Primbs JA (1996) Constrained nonlinear optimal control: a converse HJB approach. Technical report CIT-CDS 96-021, California Institute of Technology, Pasadena, CA 91125
46. Primbs JA, Nevistic V (1996) Optimality of nonlinear design techniques: A converse HJB approach. Technical report CIT-CDS 96-022, California Institute of Technology, Pasadena, CA 91125
47. Sekoguchi M, Konishi H, Goto M, Yokoyama A, Lu Q (2002) Nonlinear optimal control applied to STATCOM for power system stabilization. In: Proceedings of the IEEE/PES transmission and distribution conference and exhibition, pp 342–347
48. Kim Y, Lewis FL (2000) Optimal design of CMAC neural-network controller for robot manipulators. *IEEE Trans Syst Man Cybern Part C Appl Rev* 30(1):22–31
49. Kim Y, Lewis FL, Dawson D (2000) Intelligent optimal control of robotic manipulator using neural networks. *Automatica* 36(9):1355–1364
50. Dupree K, Patre P, Wilcox Z, Dixon WE (2008) Optimal control of uncertain nonlinear systems using rise feedback. In: Proceedings of the IEEE conference on decision and control, Cancun, Mexico, pp 2154–2159
51. Dupree K, Patre PM, Wilcox ZD, Dixon WE (2009) Optimal control of uncertain nonlinear systems using a neural network and rise feedback. In: Proceedings of the American control conference, St. Louis, Missouri, pp 361–366
52. Freeman RA, Kokotovic PV (1996) Robust nonlinear control design: state-space and lyapunov techniques. Birkhäuser, Boston
53. Fausz J, Chellaboina VS, Haddad W (1997) Inverse optimal adaptive control for nonlinear uncertain systems with exogenous disturbances. In: Proceedings of the IEEE conference on decision and control, pp 2654–2659
54. Li ZH, Krstic M (1997) Optimal design of adaptive tracking controllers for nonlinear systems. *Automatica* 33:1459–1473
55. Krstic M, Li ZH (1998) Inverse optimal design of input-to-state stabilizing nonlinear controllers. *IEEE Trans Autom Control* 43(3):336–350
56. Krstic M, Tsiotras P (1999) Inverse optimal stabilization of a rigid spacecraft. *IEEE Trans Autom Control* 44(5):1042–1049
57. Luo W, Chu YC, Ling KV (2005) Inverse optimal adaptive control for attitude tracking of spacecraft. *IEEE Trans Autom Control* 50(11):1639–1654
58. Dupree K, Johnson M, Patre PM, Dixon WE (2009) Inverse optimal control of a nonlinear Euler-Lagrange system, part ii: Output feedback. In: Proceedings of the IEEE conference on decision and control, Shanghai, China, pp 327–332
59. Dupree K, Patre PM, Johnson M, Dixon WE (2009) Inverse optimal adaptive control of a nonlinear Euler-Lagrange system: Part i. In: Proceedings of the IEEE conference on decision and control, Shanghai, China, pp 321–326
60. Johnson M, Hu G, Dupree K, Dixon WE (2009) Inverse optimal homography-based visual servo control via an uncalibrated camera. In: Proceedings of the IEEE conference on decision and control, Shanghai, China, pp 2408–2413
61. Wang Q, Sharma N, Johnson M, Gregory CM, Dixon WE (2013) Adaptive inverse optimal neuromuscular electrical stimulation. *IEEE Trans Cybern* 43:1710–1718
62. Cloutier JR (1997) State-dependent riccati equation techniques: an overview. In: Proceedings of the American control conference, 2:932–936
63. Çimen T (2008) State-dependent riccati equation (SDRE) control: a survey. In: Proceedings IFAC World Congress, pp 6–11
64. Cimen T (2010) Systematic and effective design of nonlinear feedback controllers via the state-dependent riccati equation (sdre) method. *Annu Rev Control* 34(1):32–51
65. Yucelen T, Sadahalli AS, Pourbohrat F (2010) Online solution of state dependent riccati equation for nonlinear system stabilization. In: Proceedings of the American control conference, pp 6336–6341

66. Garcia CE, Prett DM, Morari M (1989) Model predictive control: theory and practice - a survey. *Automatica* 25(3):335–348
67. Mayne D, Michalska H (1990) Receding horizon control of nonlinear systems. *IEEE Trans Autom Control* 35(7):814–824
68. Morari M, Lee J (1999) Model predictive control: past, present and future. *Comput Chem Eng* 23(4–5):667–682
69. Allgöwer F, Zheng A (2000) Nonlinear model predictive control, vol 26. Springer
70. Mayne D, Rawlings J, Rao C, Scokaert P (2000) Constrained model predictive control: Stability and optimality. *Automatica* 36:789–814
71. Camacho EF, Bordons C (2004) Model predictive control, vol 2. Springer
72. Grüne L, Pannek J (2011) Nonlinear model predictive control. Springer
73. Isaacs R (1999) Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization. Dover books on mathematics, Dover Publications
74. Tijs S (2003) Introduction to game theory. Hindustan Book Agency
75. Basar T, Olsder GJ (1999) Dynamic noncooperative game theory, 2nd edn. Classics in applied mathematics, SIAM
76. Nash J (1951) Non-cooperative games. *Ann Math* 2:286–295
77. Vamvoudakis KG, Lewis FL (2011) Policy iteration algorithm for distributed networks and graphical games. In: Proceedings of the IEEE conference decision control European control conference, pp 128–135
78. Vamvoudakis KG, Lewis FL, Hudas GR (2012) Multi-agent differential graphical games: online adaptive learning solution for synchronization with optimality. *Automatica* 48(8):1598–1611
79. Dolcetta IC (1983) On a discrete approximation of the hamilton-jacobi equation of dynamic programming. *Appl Math Optim* 10(1):367–377
80. Sethian JA (1999) Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Cambridge University Press
81. Osher S, Fedkiw R (2006) Level set methods and dynamic implicit surfaces, vol 153. Springer Science & Business Media
82. Doya K (2000) Reinforcement learning in continuous time and space. *Neural Comput* 12(1):219–245
83. Vrabie D, Lewis FL (2009) Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Netw* 22(3):237–246

# Chapter 2

## Approximate Dynamic Programming



### 2.1 Introduction

Dynamic programming techniques based on the principle of optimality have been extensively studied in literature (cf. [1–7]). The applicability of classical dynamic programming techniques like policy iteration and value iteration is limited by the curse of dimensionality and the need for model knowledge. Simulation-based reinforcement learning techniques such as Q-learning [4] and temporal difference learning [2, 8] avoid the need for exact model knowledge. However, these techniques require the states and the actions to be on finite sets. Even though the theory is developed for finite state spaces of any size, the implementation of simulation-based reinforcement learning techniques is feasible only if the size of the state space is small. Extensions of simulation-based reinforcement learning techniques to general state spaces or very large finite state spaces involve parametric approximation of the policy, where the decision space is reduced to a finite dimensional vector space, and only a few (finite) weights are tuned to obtain the value function. Such algorithms have been studied in depth for systems with countable state and action-spaces under the name of neuro-dynamic programming (cf. [6, 8–14] and the references therein). Extensions of these techniques to general state spaces and continuous time-domains is challenging and has recently become an active area of research. The rest of this chapter focuses on the development of dynamic programming methods for continuous-time systems with continuous state-spaces.

### 2.2 Exact Dynamic Programming in Continuous Time and Space

A unifying characteristic of dynamic programming based methods is the use of a (state or action) value function. A state value function, as defined in the previous chapter is a map from the state space to the reals that assigns each state its value

(i.e., the total optimal cost-to-go when the system is started in that state). An action value function (generally referred to as the  $Q$ -function) is a map from the Cartesian product of the state space and the action-space to positive real numbers. The  $Q$ -function assigns each state-action pair,  $(s, a)$ , a value (i.e., the total optimal cost when the action  $a$  is performed in the state  $s$ , and the optimal policy is followed thereafter). Another unifying characteristic of dynamic programming based methods is the interaction of policy evaluation and policy improvement. Policy evaluation (also referred to as the prediction problem) refers to the problem of finding the (state or action) value function for a given arbitrary policy. Policy improvement refers to the problem of construction of a new policy that improves the original policy. The family of approximate optimal control methods that can be viewed as an interaction between policy evaluation and policy improvement is referred to as generalized policy iteration. Almost all dynamic programming-based approximate optimal control methods can be described as generalized policy iteration [8].

For the Bolza problem in Sect. 1.5, policy evaluation amounts to finding a solution to the generalized Hamilton–Jacobi–Bellman equation (first introduced in [15])

$$r(x, \phi(x)) + \nabla_x V(x)(f(x) + g(x)\phi(x)) = 0, \quad (2.1)$$

for a fixed policy  $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$ . The policy improvement step amounts to finding a solution to the minimization problem

$$\phi(x) = \arg \min_u (r(x, u) + \nabla_x V(x)(f(x) + g(x)u)),$$

for a fixed value function  $V : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$ . Since the system dynamics are affine in control, the policy improvement step reduces to the simple assignment

$$\phi(x) = -\frac{1}{2} R^{-1} g^T(x) \nabla_x V^T(x).$$

### 2.2.1 Exact Policy Iteration: Differential and Integral Methods

The policy iteration algorithm, also known as the successive approximation algorithm alternates between policy improvement and policy evaluation. The policy iteration algorithm was first developed by Bellman in [16], and a policy improvement theorem was provided by Howard in [17]. In Algorithm 2.1, a version of the policy iteration algorithm (cf. [15]) is presented for systems with continuous state space, where  $\phi^{(0)} : \mathbb{R}^n \rightarrow \mathbb{R}^m$  denotes an initial admissible policy (i.e., a policy that results in a finite cost, starting from any initial condition), and  $V^{(i)} : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  and  $\phi^{(i)} : \mathbb{R}^n \rightarrow \mathbb{R}^m$  denote the value function and the policy obtained in the  $i^{\text{th}}$  iteration. Provided the initial policy is admissible, policy iteration generates a sequence of policies and value functions that asymptotically approach the optimal policy and the optimal

value function. Furthermore, each policy in the sequence is at least as good as the previous policy, which also implies that each policy in the sequence is admissible. For a proof of convergence, see [18].

---

**Algorithm 2.1** Policy Iteration

---

```

while  $V^{(i)} \neq V^{(i-1)}$  do
    solve  $r(x, \phi^{(i-1)}(x)) + \nabla_x V^{(i)}(x) (f(x) + g(x) \phi^{(i-1)}(x)) = 0$  with  $V^{(i)}(0) = 0$  to
    compute  $V^{(i)}$ 
     $\phi^{(i)}(x) \leftarrow -\frac{1}{2} R^{-1} g^T(x) (\nabla_x V^{(i)}(x))^T$ 
     $i \leftarrow i + 1$ 
end while

```

---

Knowledge of the system dynamics (i.e., the functions  $f$  and  $g$ ) is required to implement policy iteration. Policy iteration can be implemented without the knowledge of system dynamics using an integral approach. Since the term  $\nabla_x V(x) (f(x) + g(x) \phi(x))$  is the time derivative of  $V$  along the trajectories of the system (1.9) under the policy  $\phi$ , the generalized Hamilton–Jacobi–Bellman equation can be integrated over the interval  $[\tau, \tau + T]$ , for some constant  $T \in \mathbb{R}_{>0}$  to yield

$$V(x) = V(x(\tau + T)) - \int_{\tau}^{\tau+T} r(x(t), \phi(x(t))) dt, \quad (2.2)$$

where the shorthand  $x(t)$  is utilized to denote  $x(t; \tau, x, \phi(x(\cdot)))$ , that is, a trajectory of the system in (1.9) under the feedback controller  $u = \phi(x)$  such that  $x(\tau) = x$ .

**Definition 2.1** A function  $\tilde{V} : \mathbb{R}^n \rightarrow \mathbb{R}_{>0}$  is called a solution to the integral generalized Hamilton–Jacobi–Bellman equation in (2.2) if  $\forall x \in \mathbb{R}^n$  and  $\forall \tau \in \mathbb{R}_{\geq t_0}$ ,  $\tilde{V}$  satisfies (2.2).

Note that since the dynamics, the policy, and the cost function are time-independent, to establish  $\tilde{V}$  as a solution to the integral generalized Hamilton–Jacobi–Bellman equation, it is sufficient to check (2.2) for  $\tau = t_0$  (or for any other arbitrary value of  $\tau$ ). Algorithm 2.2, first developed in [19] details a technique that utilizes the integral generalized Hamilton–Jacobi–Bellman equation to implement policy iteration without the knowledge of the drift dynamics,  $f$ . In Algorithm 2.2, the shorthand  $x^{(i)}(t)$  is utilized to denote  $x(t; \tau, x, \phi^{(i)}(x(\cdot)))$ . The equivalence of differential and integral policy iteration is captured in the following theorem.

**Theorem 2.2** For a given admissible policy  $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$ , if the integral generalized Hamilton–Jacobi–Bellman equation in (2.2) admits a continuously differentiable solution, then the solutions to the generalized Hamilton–Jacobi–Bellman equation in (2.1) and the integral generalized Hamilton–Jacobi–Bellman equation coincide and are unique.

**Algorithm 2.2** Integral Policy Iteration

---

```

while  $V^{(i)} \neq V^{(i-1)}$  do
    solve  $V^{(i)}(x) = -\int_{\tau}^{\tau+T} r\left(x^{(i-1)}(t), \phi^{(i-1)}(x^{(i-1)}(t))\right)dt + V^{(i)}(x^{(i-1)}(\tau+T))$  with
     $V^{(i)}(0) = 0$  to compute  $V^{(i)}$ 
     $\phi^{(i)}(x) \leftarrow -\frac{1}{2}R^{-1}g^T(x)(\nabla_x V^{(i)}(x))^T$ 
     $i \leftarrow i + 1$ 
end while

```

---

*Proof* The following proof is a slight modification of the argument presented in [19]. Under suitable smoothness assumptions and provided the policy  $\phi$  is admissible, the generalized Hamilton–Jacobi–Bellman equation in (2.1) admits a unique continuously differentiable solution [15]. Let  $\tilde{V} \in C^1(\mathbb{R}^n, \mathbb{R})$  be the solution to the generalized Hamilton–Jacobi–Bellman equation with the boundary condition  $\tilde{V}(0) = 0$ . For an initial condition  $x \in \mathbb{R}^n$ , differentiation of  $\tilde{V}(x(\cdot))$  with respect to time yields

$$\dot{\tilde{V}}(x(t)) = \nabla_x \tilde{V}(x(t))(f(x(t)) + g(x(t))\phi(x(t))).$$

Using the generalized Hamilton–Jacobi–Bellman equation,

$$\dot{\tilde{V}}(x(t)) = -r(x(t), \phi(x(t))).$$

Integrating the above expression over the interval  $[\tau, \tau+T]$  for some  $\tau \in \mathbb{R}_{\geq t_0}$  yields the integral generalized Hamilton–Jacobi–Bellman equation in (2.2). Thus, any solution to the generalized Hamilton–Jacobi–Bellman equation is also a solution to the integral generalized Hamilton–Jacobi–Bellman equation. To establish the other direction, let  $\tilde{V} \in C^1(\mathbb{R}^n, \mathbb{R})$  be a solution to the generalized Hamilton–Jacobi–Bellman equation and let  $\bar{V} \in C^1(\mathbb{R}^n, \mathbb{R})$  be a different solution to the integral generalized Hamilton–Jacobi–Bellman equation with the boundary conditions  $\tilde{V}(0) = 0$  and  $\bar{V}(0) = 0$ . Consider the time-derivative of the difference  $\tilde{V} - \bar{V}$ :

$$\dot{\tilde{V}}(x(t)) - \dot{\bar{V}}(x(t)) = \nabla_x \tilde{V}(x(t))(f(x(t)) + g(x(t))\phi(x(t))) - \dot{\bar{V}}(x(t)).$$

Since  $\tilde{V}$  is a solution to the generalized Hamilton–Jacobi–Bellman equation,

$$\dot{\tilde{V}}(x(t)) - \dot{\bar{V}}(x(t)) = -r(x(t), \phi(x(t))) - \dot{\bar{V}}(x(t)).$$

Integrating the above expression over the interval  $[\tau, \tau+T]$  and using  $x(\tau) = x$ ,

$$\int_{\tau}^{\tau+T} \left( \dot{\tilde{V}}(x(t)) - \dot{\bar{V}}(x(t)) \right) dt = - \int_{\tau}^{\tau+T} r(x(t), \phi(x(t))) dt - (\bar{V}(x(\tau+T)) - \bar{V}(x)).$$

Since  $\tilde{V}$  satisfies the integral generalized Hamilton–Jacobi–Bellman equation,

$$\int_{\tau}^{\tau+T} \left( \dot{\tilde{V}}(x(t)) - \dot{\tilde{V}}(x(t)) \right) dt = 0, \quad \forall \tau \in \mathbb{R}_{\geq t_0}.$$

Hence, for all  $x \in \mathbb{R}^n$  and for all  $\tau \in \mathbb{R}_{\geq t_0}$ ,  $\tilde{V} - \bar{V}$  is a constant along the trajectory  $x(t)$ ,  $t \in [\tau, \tau + T]$ , with the initial condition  $x(\tau) = x$ . Hence, using the time-independence of the dynamics, the policy, and the cost function, it can be concluded that  $\tilde{V} - \bar{V}$  is a constant along every trajectory  $x : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  of the system in (1.9) under the controller  $u(t) = \phi(x(t))$ . Since  $\tilde{V}(0) - \bar{V}(0) = 0$  it can be concluded that  $\tilde{V}(x(t)) - \bar{V}(x(t)) = 0$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ , provided the trajectory  $x(\cdot)$  passes through the origin (i.e.,  $x(t) = 0$  for some  $t \in \mathbb{R}^n$ ).

In general, a trajectory of a dynamical system need not pass through the origin. For example, consider  $\dot{x}(t) = -x(t)$ ,  $x(0) = 1$ . However, since the policy  $\phi$  is admissible, every trajectory of the system in (1.9) under the controller  $u(t) = \phi(x(t))$  asymptotically goes to zero, leading to the following claim.

*Claim* Provided  $\phi$  is admissible,  $\tilde{V} - \bar{V}$  is a constant along the trajectories of the system in (1.9) starting from every initial condition under the controller  $u(t) = \phi(x(t))$ , and the functions  $\tilde{V}$  and  $\bar{V}$  are continuous, then

$$\tilde{V}(x) = \bar{V}(x), \quad \forall x \in \mathbb{R}^n.$$

*Proof (Proof of Claim)* For the sake of contradiction, let  $|\tilde{V}(x^*) - \bar{V}(x^*)| > \epsilon$  for some  $\epsilon \in \mathbb{R}_{>0}$  and some  $x^* \in \mathbb{R}^n$ . Since  $\tilde{V} - \bar{V}$  is a constant, it can be concluded that

$$|\tilde{V}(x(t; t_0, x^*, \phi(x(\cdot)))) - \bar{V}(x(t; t_0, x^*, \phi(x(\cdot))))| > \epsilon, \quad \forall t \in \mathbb{R}_{\geq t_0}.$$

Since  $\phi$  is admissible,  $\lim_{t \rightarrow \infty} \|x(t; t_0, x^*, \phi(x(\cdot)))\| = 0$ . Since  $\tilde{V} - \bar{V}$  is a continuous function,  $\lim_{t \rightarrow \infty} |\tilde{V}(x(t; t_0, x^*, \phi(x(\cdot)))) - \bar{V}(x(t; t_0, x^*, \phi(x(\cdot))))| = 0$ . Hence, there exists a constant  $T$  such that

$$|\tilde{V}(x(T; t_0, x^*, \phi(x(\cdot)))) - \bar{V}(x(T; t_0, x^*, \phi(x(\cdot))))| < \epsilon,$$

which is a contradiction. Since the constants  $\epsilon$  and  $x^*$  were arbitrarily selected, the proof of the claim is complete.  $\square$

The claim implies that the solutions to the integral generalized Hamilton–Jacobi–Bellman equations are unique, and hence, the proof of the theorem is complete.  $\square$

### 2.2.2 Value Iteration and Associated Challenges

The policy iteration algorithm and the integral policy iteration algorithm both require an initial admissible policy. The requirement of a initial admissible policy can be circumvented using value iteration. Value iteration and its variants are popular generalized policy iteration algorithms for discrete-time systems owing to the simplicity of their implementation. In discrete time, value iteration algorithms work by turning Bellman's recurrence relation (the discrete time counterpart of the Hamilton–Jacobi–Bellman equation) into an update rule [3, 8, 20–22]. One example of a discrete-time value iteration algorithm is detailed in Algorithm 2.3. In Algorithm 2.3, the system dynamics are described by the difference equation  $x(k+1) = f(x(k)) + g(x(k))u(k)$ , where the objective is to minimize the total cost  $J(x(\cdot), u(\cdot)) = \sum_{k=0}^{\infty} r(x(k), u(k))$ , and  $V^{(0)} : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  denotes an arbitrary initialization. A key strength of value iteration over policy iteration is that the initialization  $V^{(0)}$  does not need to be a Lyapunov function or a value function corresponding to any admissible policy. An arbitrary initialization such as  $V^{(0)}(x) = 0, \forall x \in \mathbb{R}^n$  is acceptable. Hence, to implement value iteration, knowledge of an initial admissible policy is not needed. As a result, unlike policy iteration, the functions  $V^{(i)}$  generated by value iteration are not guaranteed to be value functions corresponding to admissible policies, and similarly, the policies  $\phi^{(i)}$  are not guaranteed to be admissible. However, it can be shown that the sequences  $V^{(i)}$  and  $\phi^{(i)}$  converge to the optimal value function,  $V^*$ , and the optimal policy,  $u^*$ , respectively [23–25]. An offline value iteration-like algorithm that relies on Pontryagin's maximum principle is developed in [26–28] where a single neural network is utilized to approximate the relationship between the state and the costate variables. Value iteration algorithms for continuous-time linear systems are presented in [29–32]. For nonlinear systems, an implementation of Q-learning is presented in [33]; however, closed-loop stability of the developed controller is not analyzed.

---

**Algorithm 2.3** Discrete time value iteration

---

```

while  $V^{(i)} \neq V^{(i-1)}$  do
     $\phi^{(i)}(x) \leftarrow -\frac{1}{2} R^{-1} g^T(x) (\nabla_x V^{(i-1)}(x))^T$ 
     $V^{(i)}(x) \leftarrow r(x, \phi^{(i)}(x)) + V^{(i-1)}(f(x) + g(x) \phi^{(i)}(x))$ 
     $i \leftarrow i + 1$ 
end while

```

---

## 2.3 Approximate Dynamic Programming in Continuous Time and Space

For systems with finite state and action-spaces, policy iteration and value iteration are established as effective tools for optimal control synthesis. However, in continuous state-space systems, both policy iteration and value iteration suffer from

Bellman's curse of dimensionality, (i.e., they become computationally intractable as the size of the state space grows). The need for excessive computation can be realistically sidestepped if one seeks to obtain an approximation to the optimal value function instead of the exact optimal value function (i.e., approximate dynamic programming). To obtain an approximation to the optimal value function using policy iteration, the generalized Hamilton–Jacobi–Bellman equation must be solved approximately in each iteration. Several methods to approximate the solutions to the generalized Hamilton–Jacobi–Bellman equation have been studied in the literature. The generalized Hamilton–Jacobi–Bellman equation can be solved numerically using perturbation techniques [34–37], finite difference [38–40] and finite element [41–44] techniques, or using approximation methods such as Galerkin projections [45, 46]. This monograph focuses on approximate dynamic programming algorithms that approximate the classical policy iteration and value iteration algorithms by using a parametric approximation of the policy or the value function (cf. [47–49]). The central idea is that if the policy or the value function can be parameterized with sufficient accuracy using a small number of parameters, the optimal control problem reduces to an approximation problem in the parameter space.

### 2.3.1 Some Remarks on Function Approximation

In this chapter and in the rest of the book, the value function is approximated using a linear-in-the-parameters approximation scheme. The following characteristics of the approximation scheme can be established using the Stone–Weierstrass Theorem (see [49–51]).

**Property 2.3** *Let  $V \in C^1(\mathbb{R}^n, \mathbb{R})$ ,  $\chi \subset \mathbb{R}^n$  be compact,  $\bar{\epsilon} \in \mathbb{R}_{>0}$  be a constant, and let  $\{\sigma_i \in C^1(\mathbb{R}^n, \mathbb{R}) \mid i \in \mathbb{N}\}$  be a set of countably many uniformly bounded basis functions (cf. [52, Definition 2.1]). Then, there exists  $L \in \mathbb{N}$ , a set of basis functions  $\{\sigma_i \in C^1(\mathbb{R}^n, \mathbb{R}) \mid i = 1, 2, \dots, L\}$ , and a set of weights  $\{w_i \in \mathbb{R} \mid i = 1, 2, \dots, L\}$  such that  $\sup_{x \in \chi} (|V(x) - W^T \sigma(x)| + \|\nabla_x V(x) - W^T \nabla_x \sigma(x)\|) \leq \bar{\epsilon}$ , where  $\sigma \triangleq [\sigma_1 \dots \sigma_L]^T$  and  $W \triangleq [w_1 \dots w_L]^T$ .*

Property 2.3, also known as the Universal Approximation Theorem, states that a single layer neural network can simultaneously approximate a function and its derivative given a sufficiently large number of basis functions. Using Property 2.3, a continuously differentiable function can be represented as  $V(x) = W^T \sigma(x) + \epsilon(x)$ , where  $\epsilon : \mathbb{R}^n \rightarrow \mathbb{R}$  denotes the function approximation error. The function approximation error, along with its derivative can be made arbitrarily small by increasing the number of basis functions used in the approximation.

### 2.3.2 Approximate Policy Iteration

An example of an approximate policy iteration method is detailed in Algorithm 2.4. In Algorithm 2.4,  $\hat{V}^{(i)} : \mathbb{R}^{n+L} \rightarrow \mathbb{R}_{\geq 0}$  denotes the parametric approximation of the value function  $V$ ,  $W_c \in \mathbb{R}^L$  denotes the vector of ideal values of the unknown parameters, and  $\hat{W}_c^{(i)}$  denotes an estimate of  $W_c$ . The Bellman error corresponding to the policy  $\phi$ , denoted by  $\delta_\phi$  is defined as

$$\delta_\phi(x, \hat{W}_c) = r(x, \phi(x)) + \nabla_x \hat{V}(x, \hat{W}_c)(f(x) + g(x)\phi(x)).$$

---

**Algorithm 2.4** Approximate policy iteration

---

```

while  $\hat{W}_c^{(i)} \neq \hat{W}_c^{(i-1)}$  do
     $\hat{W}_c^{(i)} \leftarrow \arg \min_{\hat{W}_c \in \mathbb{R}^L} \int_{x \in \mathbb{R}^n} \left( \delta_{\phi^{(i-1)}}(\sigma, \hat{W}_c) \right)^2 d\sigma$ 
     $\phi^{(i)}(x) \leftarrow -\frac{1}{2} R^{-1} g^T(x) \left( \nabla_x V^{(i)}(x, \hat{W}_c^{(i)}) \right)^T$ 
     $i \leftarrow i + 1$ 
end while

```

---

Similar to Algorithms 2.2, 2.4 can be expressed in a model-free form using integration. However, the usefulness of Algorithm 2.4 (and its model-free form) is limited by the need to solve the minimization problem  $\hat{W}_c^{(i)} = \arg \min_{\hat{W}_c \in \mathbb{R}^L} \int_{x \in \mathbb{R}^n} \left( \delta_{\phi^{(i-1)}}(\sigma, \hat{W}_c) \right)^2 d\sigma$ , which is often intractable due to computational and information constraints. A more useful implementation of approximate policy iteration is detailed in Algorithm 2.5 in the model-based form, where the minimization is carried out over a specific trajectory instead of the whole state space [19, 53]. In Algorithm 2.5, the set  $\Omega_{x_0}^\phi \subset \mathbb{R}^n$  is defined as

$$\Omega_{x_0}^\phi \triangleq \{x \in \mathbb{R}^n \mid x(t; t_0, x_0, \phi(x(\cdot))) = x, \text{ for some } t \in \mathbb{R}_{\geq t_0}\}.$$

---

**Algorithm 2.5** Approximate generalized policy iteration

---

```

while  $\hat{W}_c^{(i)} \neq \hat{W}_c^{(i-1)}$  do
     $\hat{W}_c^{(i)} \leftarrow \arg \min_{\hat{W}_c \in \mathbb{R}^L} \int_{x \in \Omega_{x_0}^\phi} \left( \delta_{\phi^{(i-1)}}(\sigma, \hat{W}_c) \right)^2 d\sigma$ 
     $\phi^{(i)}(x) \leftarrow -\frac{1}{2} R^{-1} g^T(x) \left( \nabla_x V^{(i)}(x, \hat{W}_c^{(i)}) \right)^T$ 
     $i \leftarrow i + 1$ 
end while

```

---

Under suitable persistence of excitation conditions, Algorithm 2.5 can be shown to converge to a neighborhood of the optimal value function and the optimal policy [19,

[24](#), [25](#), [49](#), [53](#)]. However, the algorithm is iterative in nature, and unlike exact policy iteration, the policies  $\phi^{(i)}$  cannot generally be shown to be stabilizing. Hence, the approximate policy iteration algorithms, as stated, are not suitable for online learning and online optimal feedback control. To ensure system stability during the learning phase, a two-network approach is utilized, where in addition to the value function, the policy,  $\phi$ , is also approximated using a parametric approximation,  $\hat{u}(x, \hat{W}_a)$ . The critic learns the value of a policy by updating the weights  $\hat{W}_c$  and the actor improves the current policy by updating the weights  $\hat{W}_a$ .

### 2.3.3 Development of Actor-Critic Methods

The actor-critic (also known as adaptive-critic) architecture is one of the most widely used architectures to implement generalized policy iteration algorithms [\[1, 8, 54\]](#). Actor-critic algorithms are pervasive in machine learning and are used to learn the optimal policy online for finite-space discrete-time Markov decision problems [\[1, 3, 8, 14, 55\]](#). The idea of learning with a critic (or a trainer) first appeared in [\[56, 57\]](#) where the state-space was partitioned to make the computations tractable. Critic-based methods were further developed to learn optimal actions in sequential decision problems in [\[54\]](#). Actor-critic methods were first developed in [\[58\]](#) for systems with finite state and action-spaces, and in [\[1\]](#) for systems with continuous state and action-spaces using neural networks to implement the actor and the critic. An analysis of convergence properties of actor-critic methods was presented in [\[47, 59\]](#) for deterministic systems and in [\[14\]](#) for stochastic systems. For a detailed review of actor-critic methods, see [\[60\]](#).

Several methods have been investigated to tune the actor and the critic networks in the actor-critic methods described in the paragraph above. The actor can learn to directly minimize the estimated cost-to-go, where the estimate of the cost-to-go is obtained by the critic [\[1, 14, 55, 58, 60, 61\]](#). The actor can also be tuned to minimize the Bellman error (also known as the temporal-difference error) [\[62\]](#). The critic network can be tuned using the method of temporal differences [\[1, 2, 8, 11, 12, 14, 63\]](#) or using heuristic dynamic programming [\[3, 9, 20, 64–67\]](#) or its variants [\[55, 68, 69\]](#).

The iterative nature of actor-critic methods makes them particularly suitable for offline computation and for discrete-time systems, and hence, discrete-time approximate optimal control has been a growing area of research over the past decade [\[24, 70–80\]](#). The trajectory-based formulation in Algorithm 2.5 lends itself to an online solution approach using asynchronous dynamic programming, where the parameters are adjusted on-the-fly using input-output data. The concept of asynchronous dynamic programming can be further exploited to apply actor-critic methods online to continuous-time systems.

### 2.3.4 Actor-Critic Methods in Continuous Time and Space

Baird [81] proposed advantage updating as an extension of the Q-learning algorithm. Advantage updating can be implemented in continuous-time and provides faster convergence. A continuous-time formulation of actor-critic methods was first developed by Doya in [82]. In [82], the actor and the critic weights are tuned continuously using an adaptive update law designed as a differential equation. While no stability or convergence results are provided in [82], the developed algorithms can be readily utilized to simultaneously learn and utilize an approximate optimal feedback controller in real-time for nonlinear systems. A sequential (one network is tuned at a time) actor-critic method that does not require complete knowledge of the internal dynamics of the system is presented in [83]. Convergence properties of actor-critic methods for continuous-time systems where both the networks are concurrently tuned are examined in [84], and a Lyapunov-based analysis that concurrently examines convergence and stability properties of an online implementation of the actor-critic method is developed in [85]. The methods developed in this monograph are inspired by the algorithms in [82] and the analysis techniques in [85]. In the following section, a basic structure of online continuous-time actor-critic methods is presented. Recent literature on continuous-time actor-critic methods is cited throughout the following sections in comparative remarks.

## 2.4 Optimal Control and Lyapunov Stability

Obtaining an analytical solution to the Bolza problem is often infeasible if the system dynamics are nonlinear. Many numerical solution techniques are available to solve Bolza problems; however, numerical solution techniques require exact model knowledge and are realized via open-loop implementation of offline solutions. Open-loop implementations are sensitive to disturbances, changes in objectives, and changes in the system dynamics; hence, online closed-loop solutions of optimal control problems are sought-after. Inroads to solve an optimal control problem online can be made by looking at the value function. Under a given policy, the value function provides a map from the state space to the set of real numbers that measures the quality of a state. In other words, under a given policy, the value function evaluated at a given state is the cost accumulated when starting in the given state and following the given policy. Under general conditions, the policy that drives the system state along the steepest negative gradient of the optimal value function turns out to be the optimal policy; hence, online optimal control design relies on computation of the optimal value function.

In online closed-loop approximate optimal control, the value function has an even more important role to play. Not only does the value function provide the optimal policy, but the value function is also a Lyapunov function that establishes global asymptotic stability of the closed-loop system.

**Theorem 2.4** Consider the affine dynamical system in (1.9). Let  $V^* : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  be the optimal value function corresponding to the affine-quadratic optimal control problem in (1.10). Assume further that  $f(0) = 0$  and that the control effectiveness matrix  $g(x)$  is full rank for all  $x \in \mathbb{R}^n$ . Then, the closed-loop system under the optimal controller  $u(t) = u^*(x(t))$  is asymptotically stable.

*Proof* Since  $f(0) = 0$ , it follows that when  $x(t_0) = 0$ , the controller that yields the lowest cost is  $u(t) = 0$ ,  $\forall t$ . Hence,  $V^*(0) = 0$ , and since the optimal controller is given by  $u(t) = u^*(x(t)) = -\frac{1}{2}R^{-1}g^T(x(t))\nabla_x V^*(x(t))$ , and  $g$  is assumed to be full rank, it can be concluded that  $\nabla_x V^*(0) = 0$ . Furthermore, if  $x \neq 0$ , it follows that  $V^*(x) \neq 0$ . Hence, the function  $V^*$  is a candidate Lyapunov function, and  $x = 0$  is an equilibrium point of the closed-loop dynamics

$$\dot{x}(t) = f(x(t)) + g(x(t))u^*(x(t)).$$

The time derivative of  $V^*$  along the trajectories of (1.9) is given by

$$\dot{V}^*(t) = \nabla_x V^*(x(t))(f(x(t)) + g(x(t))u(t)).$$

when the optimal controller is used,  $u(t) = u^*(x(t))$ . Hence,

$$\dot{V}^*(t) = \nabla_x V^*(x(t))(f(x(t)) + g(x(t))u^*(x(t))).$$

Since the optimal value function satisfies the Hamilton–Jacobi–Bellman equation, Theorem (1.2) implies that

$$\dot{V}^*(t) = -r(x(t), u^*(x(t))) \leq -Q(x(t))$$

Since the function  $Q$  is positive definite by design, [86, Theorem 4.2] can be invoked to conclude that the equilibrium point  $x = 0$  is asymptotically stable.  $\square$

The utility of Theorem 2.4 as a tool to analyze optimal controllers is limited because for nonlinear systems, analytical or exact numerical computation of the optimal controller is often intractable. Hence, one often works with approximate value functions and approximate optimal controllers. Theorem 2.4 provides a powerful tool for the analysis of approximate optimal controllers because the optimal policy is inherently robust to approximation (for an in-depth discussion regarding robustness of the optimal policy, see [87, 88]). That is, the optimal value function can also be used as a candidate Lyapunov function to establish practical stability (that is, uniform ultimate boundedness) of the system in (1.9) under controllers that are close to or asymptotically approach a neighborhood of the optimal controller. The rest of the discussion in this chapter focuses on the methodology employed in the rest of this monograph to generate an approximation of the optimal controller. Over the years, many different approximate optimal control methods have been developed for various classes of systems. For a brief discussion about alternative methods, see Sect. 2.8.

## 2.5 Differential Online Approximate Optimal Control

In an approximate actor-critic-based solution, the optimal value function  $V^*$  is replaced by a parametric estimate  $\hat{V}(x, \hat{W}_c)$  and the optimal policy  $u^*$  by a parametric estimate  $\hat{u}(x, \hat{W}_a)$  where  $\hat{W}_c \in \mathbb{R}^L$  and  $\hat{W}_a \in \mathbb{R}^L$  denote vectors of estimates of the ideal parameters.

Substituting the estimates  $\hat{V}$  and  $\hat{u}$  for  $V^*$  and  $u^*$  in (1.14), respectively, a residual error  $\delta : \mathbb{R}^n \times \mathbb{R}^L \times \mathbb{R}^L \rightarrow \mathbb{R}$ , called the Bellman error, is defined as

$$\delta(x, \hat{W}_c, \hat{W}_a) \triangleq \nabla_x \hat{V}(x, \hat{W}_c) (f(x) + g(x) \hat{u}(x, \hat{W}_a)) + r(x, \hat{u}(x, \hat{W}_a)). \quad (2.3)$$

The use of two separate sets of weight estimates  $\hat{W}_a$  and  $\hat{W}_c$  is motivated by the fact that the Bellman error is linear with respect to the critic weight estimates and nonlinear with respect to the actor weight estimates. Use of a separate set of weight estimates for the value function facilitates least-squares-based adaptive updates.

To solve the optimal control problem, the critic aims to find a set of parameters  $\hat{W}_c$  and the actor aims to find a set of parameters  $\hat{W}_a$  such that  $\delta(x, \hat{W}_c, \hat{W}_a) = 0$ , and  $\hat{u}(x, \hat{W}_a) = -\frac{1}{2} R^{-1} g^T(x) (\nabla \hat{V}(x, \hat{W}_c))^T \forall x \in \mathbb{R}^n$ . Since an exact basis for value function approximation is generally not available, an approximate set of parameters that minimizes the Bellman error is sought. In particular, to ensure uniform approximation of the value function and the policy over an operating domain  $\mathcal{D} \subset \mathbb{R}^n$ , it is desirable to find parameters that minimize the error  $E_s : \mathbb{R}^L \times \mathbb{R}^L \rightarrow \mathbb{R}$  defined as

$$E_s(\hat{W}_c, \hat{W}_a) \triangleq \sup_{x \in \mathcal{D}} |\delta(x, \hat{W}_c, \hat{W}_a)|. \quad (2.4)$$

Hence, in an online implementation of the deterministic actor-critic method, it is desirable treat the parameter estimates  $\hat{W}_c$  and  $\hat{W}_a$  as time-varying and update them online to minimize the instantaneous error  $E_s(\hat{W}_c(t), \hat{W}_a(t))$  or the cumulative instantaneous error

$$E(t) \triangleq \int_0^t E_s(\hat{W}_c(\tau), \hat{W}_a(\tau)) d\tau, \quad (2.5)$$

while the system in (1.9) is being controlled using the control law  $u(t) = \hat{u}(x(t), \hat{W}_a(t))$ .

### 2.5.1 Reinforcement Learning-Based Online Implementation

Computation of the Bellman error in (2.3) and the integral error in (2.5) requires exact model knowledge. Furthermore, computation of the integral error in (2.5) is generally infeasible. Two prevalent approaches employed to render the control design robust to uncertainties in the system drift dynamics are integral reinforcement learning (cf. [7, 19, 89–92]) and state derivative estimation (cf. [93, 94]). This section focuses on state derivative estimation based methods. For further details on integral reinforcement learning, see Sect. 2.2.1.

State derivative estimation-based techniques exploit the fact that if the system model is uncertain, the critic can compute the Bellman error at each time instance using the state-derivative  $\dot{x}(t)$  as

$$\delta_t(t) \triangleq \nabla_x \hat{V} \left( x(t), \hat{W}_c(t) \right) \dot{x}(t) + r \left( x(t), \hat{u} \left( x(t), \hat{W}_a(t) \right) \right). \quad (2.6)$$

If the state-derivative is not directly measurable, an approximation of the Bellman error can be computed using a dynamically generated estimate of the state-derivative. Since (1.14) constitutes a necessary and sufficient condition for optimality, the Bellman error serves as an indirect measure of how close the critic parameter estimates  $\hat{W}_c(t)$  are to their ideal values; hence, in reinforcement learning literature, each evaluation of the Bellman error is interpreted as gained experience. In particular, the critic receives state-derivative-action-reward tuples  $(x(t), \dot{x}(t), u(t), r(x(t), u(t)))$  and computes the Bellman error using (2.6). The critic then performs a one-step update to the parameter estimates  $\hat{W}_c(t)$  based on either the instantaneous experience, quantified by the squared error  $\delta_t^2(t)$ , or the cumulative experience, quantified by the integral squared error

$$E_t(t) \triangleq \int_0^t \delta_t^2(\tau) d\tau, \quad (2.7)$$

using a steepest descent update law. The use of the cumulative squared error is motivated by the fact that in the presence of uncertainties, the Bellman error can only be evaluated along the system trajectory; hence,  $E_t(t)$  is the closest approximation to  $E(t)$  in (2.5) that can be computed using available information.

Intuitively, for  $E_t(t)$  to approximate  $E(t)$  over an operating domain, the state trajectory  $x(t)$  needs to visit as many points in the operating domain as possible. This intuition is formalized by the fact that the use of the approximation  $E_t(t)$  to update the critic parameter estimates is valid provided certain exploration conditions<sup>1</sup> are met. In reinforcement learning terms, the exploration conditions translate to the need for the critic to gain enough experience to learn the value function. The exploration

---

<sup>1</sup>The exploration conditions are detailed in the next section for a linear-in-the-parameters approximation of the value function.

conditions can be relaxed using experience replay (cf. [92]), where each evaluation of the Bellman error  $\delta_{int}$  is interpreted as gained experience, and these experiences are stored in a history stack and are repeatedly used in the learning algorithm to improve data efficiency; however, a finite amount of exploration is still required since the values stored in the history stack are also constrained to the system trajectory. Learning based on simulation of experience has also been investigated in results such as [95–100] for stochastic model-based reinforcement learning; however, these results solve the optimal control problem off-line in the sense that repeated learning trials need to be performed before the algorithm learns the controller and system stability during the learning phase is not analyzed.

While the estimates  $\hat{W}_c(\cdot)$  are being updated by the critic, the actor simultaneously updates the parameter estimates  $\hat{W}_a(\cdot)$  using a gradient-based approach so that the quantity  $\hat{u}(x(t), \hat{W}_a(t)) + \frac{1}{2}R^{-1}g^T(x(t))(\nabla_x \hat{V}(x(t), \hat{W}_c(t)))^T$  decreases. The weight updates are performed online and in real-time while the system is being controlled using the control law  $u(t) = \hat{u}(x(t), \hat{W}_a(t))$ . Naturally, it is difficult to guarantee stability during the learning phase. In fact, the use of two different sets parameters to approximate the value function and the policy is required solely for the purpose of maintaining stability during the learning phase.

### 2.5.2 Linear-in-the-Parameters Approximation of the Value Function

For feasibility of analysis, the optimal value function is approximated using a linear-in-the-parameters approximation

$$\hat{V}(x, \hat{W}_c) \triangleq \hat{W}_c^T \sigma(x), \quad (2.8)$$

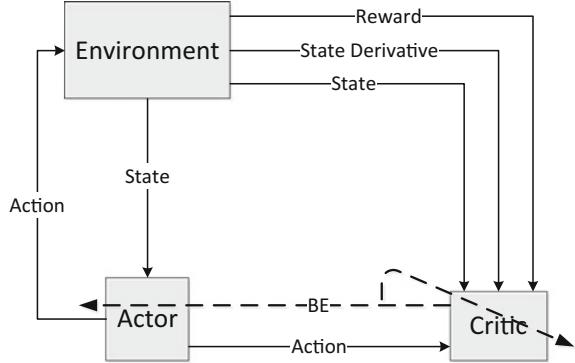
where  $\sigma : \mathbb{R}^n \rightarrow \mathbb{R}^L$  is a continuously differentiable nonlinear activation function such that  $\sigma(0) = 0$  and  $\nabla_x \sigma(0) = 0$ , and  $\hat{W}_c \in \mathbb{R}^L$ , where  $L$  denotes the number of unknown parameters in the approximation of the value function. Based on (1.13), the optimal policy is approximated using the linear-in-the-parameters approximation

$$\hat{u}(x, \hat{W}_a) \triangleq -\frac{1}{2}R^{-1}g(x)^T \nabla_x \sigma^T(x) \hat{W}_a. \quad (2.9)$$

The update law used by the critic to update the weight estimates is given by

$$\begin{aligned} \dot{\hat{W}}_c(t) &= -\eta_c \Gamma \frac{\omega(t)}{\rho(t)} \delta_t(t), \\ \dot{\Gamma}(t) &= \beta \Gamma(t) - \eta_c \Gamma(t) \frac{\omega(t) \omega^T(t)}{\rho^2(t)} \Gamma(t), \end{aligned} \quad (2.10)$$

**Fig. 2.1** The actor-critic architecture. The critic computes the Bellman error based on the state, the action, the reward, and the time-derivative of the state. The actor and the critic both improve their estimate of the value function using the Bellman error



where  $\omega(t) \triangleq \nabla_x \sigma(x(t)) \dot{x}(t) \in \mathbb{R}^L$  denotes the regressor vector,  $\rho(t) \triangleq 1 + v\omega^T(t) \Gamma(t) \omega(t) \in \mathbb{R}$ ,  $\eta_c, \beta, v \in \mathbb{R}_{>0}$  are constant learning gains,  $\bar{\Gamma} \in \mathbb{R}_{>0}$  is a saturation constant, and  $\Gamma$  is the least-squares gain matrix. The update law used by the actor to update the weight estimates is derived using a Lyapunov-based stability analysis, and is given by

$$\begin{aligned} \dot{\hat{W}}_a(t) = & \frac{\eta_c \nabla_x \sigma(x(t)) g(x(t)) R^{-1} g^T(x(t)) \nabla_x \sigma^T(x(t)) \hat{W}_a(t) \omega^T(t)}{4\rho(t)} \\ & - \eta_{a1} (\hat{W}_a(t) - \hat{W}_c(t)) - \eta_{a2} \hat{W}_a(t), \end{aligned} \quad (2.11)$$

where  $\eta_{a1}, \eta_{a2} \in \mathbb{R}_{>0}$  are constant learning gains. A block diagram of the resulting control architecture is in Fig. 2.1.

The stability analysis indicates that the sufficient exploration condition takes the form of a persistence of excitation condition that requires the existence of positive constants  $\underline{\psi}$  and  $T$  such that the regressor vector satisfies

$$\underline{\psi} I_L \leq \int_t^{t+T} \frac{\omega(\tau) \omega^T(\tau)}{\rho(\tau)} d\tau, \quad (2.12)$$

for all  $t \in \mathbb{R}_{\geq t_0}$ . The regressor is defined here as a trajectory indexed by time. It should be noted that different initial conditions result in different regressor trajectories; hence, the constants  $T$  and  $\underline{\psi}$  depend on the initial values of  $x(\cdot)$  and  $\hat{W}_a(\cdot)$ . Hence, the final result is generally not uniform in the initial conditions.

Let  $\tilde{W}_c(t) \triangleq W - \hat{W}_c(t)$  and  $\tilde{W}_a(t) \triangleq W - \hat{W}_a(t)$  denote the vectors of parameter estimation errors, where  $W \in \mathbb{R}^L$  denotes the constant vector of ideal parameters (see Property 2.3). Provided (2.12) is satisfied, and under sufficient conditions on the learning gains and the constants  $\underline{\psi}$  and  $T$ , the candidate Lyapunov function

$$V_L(x, \tilde{W}_c, \tilde{W}_a, t) \triangleq V^*(x) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1}(t) \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a$$

can be used to establish convergence of  $x(t)$ ,  $\tilde{W}_c(t)$ , and  $\tilde{W}_a(t)$  to a neighborhood of zero as  $t \rightarrow \infty$ , when the system in (1.9) is controlled using the control law

$$u(t) = \hat{u}\left(x(t), \hat{W}_a(t)\right), \quad (2.13)$$

and the parameter estimates  $\hat{W}_c(\cdot)$  and  $\hat{W}_a(\cdot)$  are updated using the update laws in (2.10) and (2.11), respectively.

## 2.6 Uncertainties in System Dynamics

The use of the state derivative to compute the Bellman error in (2.6) is advantageous because it is easier to obtain a dynamic estimate of the state derivative than it is to identify the system dynamics. For example, consider the high-gain dynamic state derivative estimator

$$\begin{aligned} \dot{\hat{x}}(t) &= g(x(t)) u(t) + k\tilde{x}(t) + \mu(t), \\ \dot{\mu}(t) &= (k\alpha + 1)\tilde{x}(t), \end{aligned} \quad (2.14)$$

where  $\hat{x}(t) \in \mathbb{R}^n$  is an estimate of the state derivative,  $\tilde{x}(t) \triangleq x - \hat{x}(t)$  is the state estimation error, and  $k, \alpha \in \mathbb{R}_{>0}$  are identification gains. Using (2.14), the Bellman error in (2.6) can be approximated by  $\hat{\delta}_t$  as

$$\hat{\delta}_t(t) = \nabla_{\hat{x}} \hat{V}\left(x(t), \hat{W}_c(t)\right) \dot{\hat{x}}(t) + r\left(x(t), \hat{u}\left(x(t), \hat{W}_a(t)\right)\right).$$

The critic can then learn the critic weights by using an approximation of cumulative experience, quantified using  $\hat{\delta}_t$  instead of  $\delta_t$  in (2.10), that is,

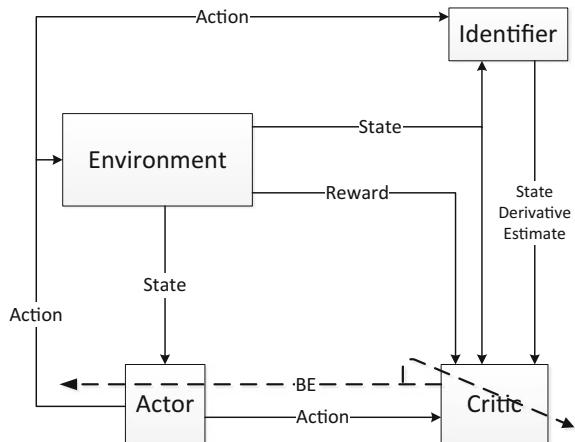
$$\hat{E}_t(t) = \int_0^t \hat{\delta}_t^2(\tau) d\tau. \quad (2.15)$$

Under additional sufficient conditions on the gains  $k$  and  $\alpha$ , the candidate Lyapunov function

$$V_L(x, \tilde{W}_c, \tilde{W}_a, \tilde{x}, x_f, t) \triangleq V^*(x) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1}(t) \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a + \frac{1}{2} \tilde{x}^T \tilde{x} + \frac{1}{2} x_f^T x_f,$$

where  $x_f(t) \triangleq \dot{\tilde{x}}(t) + \alpha\tilde{x}(t)$ , can be used to establish convergence of  $x(t)$ ,  $\tilde{W}_c(t)$ ,  $\tilde{W}_a(t)$ ,  $\tilde{x}(t)$ , and  $x_f(t)$  to a neighborhood of zero, when the system in (1.9) is

**Fig. 2.2** Actor-critic-identifier architecture. The critic uses estimates of the state derivative to compute the Bellman error



controlled using the control law (2.13). The aforementioned extension of the actor-critic method to handle uncertainties in the system dynamics using derivative estimation is known as the actor-critic-identifier architecture. A block diagram of the actor-critic-identifier architecture is presented in Fig. 2.2.

## 2.7 Persistence of Excitation and Parameter Convergence

In online implementations of reinforcement learning, the control policy derived from the approximate value function is used to control the system; hence, obtaining a good approximation of the value function is critical to the stability of the closed-loop system. Obtaining a good approximation of the value function online requires convergence of the unknown parameters to their ideal values. Hence, similar to adaptive control, the sufficient exploration condition manifests itself as a persistence of excitation condition when reinforcement learning is implemented online.

Parameter convergence has been a focus of research in adaptive control for several decades. It is common knowledge that least-squares and gradient descent-based update laws generally require persistence of excitation in the system state for convergence of the parameter estimates. Modification schemes such as projection algorithms,  $\sigma$ -modification, and  $e$ -modification are used to guarantee boundedness of parameter estimates and overall system stability; however, these modifications do not guarantee parameter convergence unless the persistence of excitation condition is satisfied [101–104].

In general, the controller in (2.13) does not ensure the persistence of excitation condition in (2.12). Thus, in an online implementation, an ad-hoc exploration signal is often added to the controller (cf. [8, 33, 105]). Since the exploration signal is not considered in the the stability analysis, it is difficult the ensure stability of the

online implementation. Moreover, the added probing signal causes large control effort expenditure and there is no means to know when it is sufficient to remove the probing signal. Chap. 4 addresses the challenges associated with the satisfaction of the condition in (2.12) via simulated experience and cumulative experience collected along the system trajectory.

## 2.8 Further Reading and Historical Remarks

Approximate optimal control has been an active topic of research since the seminal works of Bellman [106] and Pontryagin [107] in the 1950s. A comprehensive survey and classification of all the results dealing with approximate optimal control is out of the scope of this monograph. In the following, a brief (by no means exhaustive) classification of techniques based on Bellman's dynamic programming principle is presented. For a recent survey of approximate dynamic programming in deterministic systems, see [108]. Brief discussions on a few specific techniques directly related to the methodology used in this book are also presented. For a brief description of methods based on Pontryagin's maximum principle refer back to Sect. 1.8.

**On-Policy Versus Off-Policy Learning:** A generalized policy iteration technique is called on-policy if the data used to improve an estimate of the optimal policy is required to be collected using the same estimate. A generalized policy iteration technique is called off-policy if an estimate of the optimal policy can be improved using data collected using another policy. For example methods such as policy iteration, value iteration, heuristic dynamic programming, adaptive-critic methods [1, 14, 55, 59, 61], SARSA (cf. [109, 110]) are on-policy, whereas methods such as  $Q$ -learning [111] and  $R$ -learning [112] are off-policy. The distinction between on-policy and off-policy methods is important because most online generalized policy iteration methods require exploration for convergence, whereas the on-policy condition requires exploitation, hence leading to the exploration versus exploitation conflict. Off-policy methods avoid the exploration versus exploitation conflict since an arbitrary exploring policy can be used to facilitate learning.

Approximation of solutions of reward-maximization problems using indirect feedback generated by a critic network was first investigated in [57]. Critic-based methods were further developed to solve a variety of optimal control problems [1, 4, 20, 54], for example, heuristic dynamic programming [20], adaptive critic elements [1], and Q-learning [111]. A common theme among the aforementioned techniques is the use of two neuron-like elements, an actor element that is responsible for generating control signals and a critic element that is responsible for evaluation of the control signals generated by the actor (except Q-learning, which is implemented with just one neuron-like element that combines the information about the policy and the value function [4, 111]). The most useful feature of critic based methods is that they can be implemented online in real time.

**Policy Iteration, Value Iteration, and Policy Gradient:** Dynamic programming methods have traditionally been classified into three distinct schemes: policy

iteration, value iteration, and policy gradient. Policy iteration methods start with a stabilizing policy, find the value function corresponding to that policy (i.e., policy evaluation), and then update the policy to exploit the value function (i.e., policy improvement). A large majority of dynamic programming algorithms can be classified as policy iteration algorithms. For example, SARSA and the successive approximation methods developed in results such as [15, 16, 18, 21, 45, 46, 49, 79, 113–119] are policy iteration algorithms. In value iteration, starting from an arbitrary initial guess, the value function is directly improved by effectively combining the evaluation and the improvement phases into one single update. For example, algorithms such as  $Q$ -learning [111],  $R$ -learning [112], heuristic dynamic programming, action-dependent heuristic dynamic programming, dual heuristic programming, action-dependent dual heuristic programming, [9] and modern extensions of value iteration (see [6–8, 77] for a summary). Both policy iteration and value iteration are typically critic-only methods [60] and can be considered as special cases of generalized policy iteration [8, 21].

Policy gradient methods (also known as actor-only methods) are philosophically different from policy iteration and value iteration. In policy gradient methods, instead of approximating the value function, the policy is directly approximated by computing the gradient of the cost functional with respect to the unknown parameters in the approximation of the policy [120–123]. Modern policy gradient methods utilize an approximation of the value function to estimate the gradients, and are called actor-critic methods [14, 60, 124].

**Continuous-Time Versus Discrete-Time Methods:** For deterministic systems, reinforcement learning algorithms have been extended to solve finite and infinite-horizon discounted and total cost optimal regulation problems (cf. [24, 26, 48, 49, 70, 85, 91, 93, 105, 125]) under names such as adaptive dynamic programming or adaptive critic algorithms. The discrete/iterative nature of the approximate dynamic programming formulation lends itself naturally to the design of discrete-time optimal controllers [24, 26, 27, 70–75, 79, 126], and the convergence of algorithms for dynamic programming-based reinforcement learning controllers is studied in results such as [47, 59, 61, 72]. Most prior work has focused on convergence analysis for discrete-time systems, but some continuous examples are available [15, 19, 45, 47, 49, 81, 82, 84, 85, 105, 127–129]. For example, in [81] advantage updating was proposed as an extension of the  $Q$ -learning algorithm which could be implemented in continuous time and provided faster convergence. The result in [82] used a Hamilton–Jacobi–Bellman framework to derive algorithms for value function approximation and policy improvement, based on a continuous version of the temporal difference error. An Hamilton–Jacobi–Bellman framework was also used in [47] to develop a stepwise stable iterative approximate dynamic programming algorithm for continuous input-affine systems with an input-quadratic performance measure. Based on the successive approximation method first proposed in [15], an adaptive optimal control solution is provided in [45], where a Galerkin’s spectral method is used to approximate the solution to the generalized Hamilton–Jacobi–Bellman equation. A least-squares-based successive approximation solution to the generalized Hamilton–Jacobi–Bellman equation is provided in [49], where an neural network is trained

offline to learn the solution to the generalized Hamilton–Jacobi–Bellman equation. Another continuous formulation is proposed in [84].

**Online Versus Offline Learning:** A generalized policy iteration technique is called online if the learning laws are iterative in nature. That is, input-output data can be sequentially utilized while the system is running, to incrementally update the optimal policy. Convergence of the approximated policy to the optimal policy is typically obtained asymptotically over time. In contrast, methods that require expensive batch computational operations on large recorded datasets need to be implemented offline. Convergence of the approximate policy to the optimal policy is typically obtained in an iterative manner as the number of iterations goes to infinity. For example, methods such as heuristic dynamic programming and dual heuristic programming [20], adaptive critic methods [1], asynchronous dynamic programming [130, 131],  $Q$ –learning [111], and  $R$ –learning [112] are online generalized policy iteration methods, whereas methods such as successive approximation [15, 16, 18, 45, 132], policy iteration [16, 17] and value iteration [8, 21], single network adaptive critic [26, 27] are offline generalized policy iteration algorithms. It should be noted that the distinction between online and offline generalized policy iteration algorithms is becoming less pronounced as computers are getting faster and it is now possible to execute in real time many algorithms that were thought to be computationally infeasible. The distinction between online and offline techniques is further blurred by the receding horizon approach used in model-predictive control that enables techniques that were previously classified as offline to be utilized for real time feedback control.

**Model-Based Versus Model-Free Methods:** The ultimate objective of reinforcement learning is to learn a controller for a system just by observing its behavior, and the rewards that correspond to the behavior. Two different approaches are used to attack the problem. Model-based (also known as indirect) approaches utilize the observations about the behavior of the system (i.e., input and output measurements) to build a model of the system, and then utilize the model to learn the controller. For example, one of the early dynamic programming methods, the heuristic dynamic programming algorithm developed by Werbos, is a model-based reinforcement learning technique [20]. Different variations of model-based algorithms are developed in [95–100, 133, 134], and in some cases, are shown to outperform model-free algorithms [135]. Arguably the most successful implementation of model-based reinforcement learning is the work of Ng et al. [97, 136] regarding acrobatic autonomous helicopter maneuvers. However, apprenticeship and multiple iterations of offline training is required to learn the maneuvers. Since Bellman’s recurrence relation in discrete-time is inherently model-free, the bulk of research on reinforcement learning in discrete-time systems has been focused on model-free reinforcement learning. For example, while the classical formulation of dynamic programming (policy iteration and value iteration) requires a model, reinforcement learning-based implementations of dynamic programming, such as  $Q$ –learning,  $R$ –learning , SARSA, action-dependent heuristic dynamic programming and action-dependent dual heuristic programming [9] are model-free.

In continuous-time systems, to solve the generalized Hamilton–Jacobi–Bellman equation or the Hamilton–Jacobi–Bellman equation, either a model of the system

or measurements of the state-derivative are needed. Hence, early developments in dynamic programming for continuous-time systems required exact model knowledge [15, 45, 47, 49, 81, 82, 84, 85, 105]. Hence, the term model-based reinforcement learning is sometimes erroneously used to describe algorithms that require exact model knowledge. Motivated by the need to accommodate uncertainties, much of the recent research on continuous-time reinforcement learning has focused on model-free methods [19, 31, 53, 93, 119, 128, 129, 137–140].

## References

1. Barto A, Sutton R, Anderson C (1983) Neuron-like adaptive elements that can solve difficult learning control problems. *IEEE Trans Syst Man Cybern* 13(5):834–846
2. Sutton R (1988) Learning to predict by the methods of temporal differences. *Mach Learn* 3(1):9–44
3. Werbos P (1990) A menu of designs for reinforcement learning over time. *Neural Netw Control* 67–95
4. Watkins C, Dayan P (1992) Q-learning. *Mach Learn* 8(3):279–292
5. Bellman RE (2003) Dynamic programming. Dover Publications, Inc, New York
6. Bertsekas D (2007) Dynamic programming and optimal control, vol 2, 3rd edn. Athena Scientific, Belmont, MA
7. Lewis FL, Vrabie D, Syrmos VL (2012) Optimal control, 3rd edn. Wiley, Hoboken
8. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge
9. Werbos PJ (1992) Approximate dynamic programming for real-time control and neural modeling. In: White DA, Sorge DA (eds) *Handbook of intelligent control: neural, fuzzy, and adaptive approaches*, vol 15. Nostrand, New York, pp 493–525
10. Bertsekas D, Tsitsiklis J (1996) Neuro-dynamic programming. Athena Scientific, Nashua
11. Tsitsiklis JN, Van Roy B (1997) An analysis of temporal-difference learning with function approximation. *IEEE Trans Autom Control* 42(5):674–690
12. Tsitsiklis JN, Roy BV (1999) Average cost temporal-difference learning. *Automatica* 35(11):1799–1808
13. Tsitsiklis J (2003) On the convergence of optimistic policy iteration. *J Mach Learn Res* 3:59–72
14. Konda V, Tsitsiklis J (2004) On actor-critic algorithms. *SIAM J Control Optim* 42(4):1143–1166
15. Leake R, Liu R (1967) Construction of suboptimal control sequences. *SIAM J Control* 5:54
16. Bellman R (1957) Dynamic programming, 1st edn. Princeton University Press, Princeton
17. Howard R (1960) Dynamic programming and Markov processes. Technology Press of Massachusetts Institute of Technology (Cambridge)
18. Saridis G, Lee C (1979) An approximation theory of optimal control for trainable manipulators. *IEEE Trans Syst Man Cyber* 9(3):152–159
19. Vrabie D, Lewis FL (2009) Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Netw* 22(3):237–246
20. Werbos PJ (1977) Advanced forecasting methods for global crisis warning and models of intelligence. *Gen Syst Yearb* 22:25–38
21. Puterman ML, Shin MC (1978) Modified policy iteration algorithms for discounted markov decision problems. *Manag Sci* 24(11):1127–1137
22. Bertsekas DP (1987) Dynamic programming: deterministic and stochastic models. Prentice-Hall, Englewood Cliffs
23. Lincoln B, Rantzer A (2006) Relaxing dynamic programming. *IEEE Trans Autom Control* 51(8):1249–1260

24. Al-Tamimi A, Lewis FL, Abu-Khalaf M (2008) Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *IEEE Trans Syst Man Cybern Part B Cybern* 38:943–949
25. Heydari A (2014) Revisiting approximate dynamic programming and its convergence. *IEEE Trans Cybern* 44(12):2733–2743
26. Padhi R, Unnikrishnan N, Wang X, Balakrishnan S (2006) A single network adaptive critic (SNAC) architecture for optimal control synthesis for a class of nonlinear systems. *Neural Netw* 19(10):1648–1660
27. Heydari A, Balakrishnan S (2013) Finite-horizon control-constrained nonlinear optimal control using single network adaptive critics. *IEEE Trans Neural Netw Learn Syst* 24(1):145–157
28. Heydari A, Balakrishnan SN (2013) Fixed-final-time optimal control of nonlinear systems with terminal constraints. *Neural Netw* 48:61–71
29. Lee JY, Park JB, Choi YH (2013) On integral value iteration for continuous-time linear systems. In: Proceedings of the American control conference, pp 4215–4220
30. Jha SK, Bhasin S (2014) On-policy q-learning for adaptive optimal control. In: Proceedings of the IEEE symposium on adaptive dynamic programming and reinforcement learning, pp 1–6
31. Palanisamy M, Modares H, Lewis FL, Aurangzeb M (2015) Continuous-time q-learning for infinite-horizon discounted cost linear quadratic regulator problems. *IEEE Trans Cybern* 45(2):165–176
32. Bian T, Jiang ZP (2015) Value iteration and adaptive optimal control for linear continuous-time systems. In: Proceedings of the IEEE international conference on cybernetics and intelligent systems, IEEE conference on robotics, automation and mechatronics, pp 53–58
33. Mehta P, Meyn S (2009) Q-learning and pontryagin’s minimum principle. In: Proceedings of the IEEE conference on decision and control, pp 3598–3605
34. Al’Brekh E (1961) On the optimal stabilization of nonlinear systems. *J Appl Math Mech* 25(5):1254–1266
35. Lukes DL (1969) Optimal regulation of nonlinear dynamical systems. *SIAM J Control* 7(1):75–100
36. Nishikawa Y, Sannomiya N, Itakura H (1971) A method for suboptimal design of nonlinear feedback systems. *Automatica* 7(6):703–712
37. Garrard WL, Jordan JM (1977) Design of nonlinear automatic flight control systems. *Automatica* 13(5):497–505
38. Dolcetta IC (1983) On a discrete approximation of the hamilton-jacobi equation of dynamic programming. *Appl Math Optim* 10(1):367–377
39. Falcone M, Ferretti R (1994) Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations. *Numer Math* 67(3):315–344
40. Bardi M, Dolcetta I (1997) Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations. Springer, Berlin
41. Gonzalez R (1985a) On deterministic control problems: an approximation procedure for the optimal cost i. The stationary problem. *SIAM J Control Optim* 23(2):242–266
42. Gonzalez R, Rofman E (1985b) On deterministic control problems: an approximation procedure for the optimal cost ii. The nonstationary case. *SIAM J Control Optim* 23(2):267–285
43. Falcone M (1987) A numerical approach to the infinite horizon problem of deterministic control theory. *Appl Math Optim* 15(1):1–13
44. Kushner HJ (1990) Numerical methods for stochastic control problems in continuous time. *SIAM J Control Optim* 28(5):999–1048
45. Beard R, Saridis G, Wen J (1997) Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation. *Automatica* 33:2159–2178
46. Beard RW, McLain TW (1998) Successive Galerkin approximation algorithms for nonlinear optimal and robust control. *Int J Control* 71(5):717–743
47. Murray J, Cox C, Lendaris G, Saeks R (2002) Adaptive dynamic programming. *IEEE Trans Syst Man Cybern Part C Appl Rev* 32(2):140–153

48. Abu-Khalaf M, Lewis FL (2002) Nearly optimal HJB solution for constrained input systems using a neural network least-squares approach. In: Proceedings of the IEEE conference on decision and control, Las Vegas, NV, pp 943–948
49. Abu-Khalaf M, Lewis FL (2005) Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach. *Automatica* 41(5):779–791
50. Hornik K, Stinchcombe M, White H (1990) Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Netw* 3(5):551–560
51. Hornick K (1991) Approximation capabilities of multilayer feedforward networks. *Neural Netw* 4:251–257
52. Sadegh N (1993) A perceptron network for functional identification and control of nonlinear systems. *IEEE Trans Neural Netw* 4(6):982–988
53. Bian T, Jiang Y, Jiang ZP (2014) Adaptive dynamic programming and optimal control of nonlinear nonaffine systems. *Automatica* 50(10):2624–2632
54. Widrow B, Gupta N, Maitra S (1973) Punish/reward: Learning with a critic in adaptive threshold systems. *IEEE Trans Syst Man Cybern* 3(5):455–465
55. Prokhorov DV, Wunsch IDC (1997) Adaptive critic designs. *IEEE Trans Neural Netw* 8:997–1007
56. Fu KS (1964) Learning control systems. In: Tou JT, Wilcox RH (eds) Computing and information science, collected papers on learning, adaptation and control in information systems. Spartan Books, Washington, pp 318–343
57. Fu KS (1969) Learning control systems. In: Tou JT (ed) Advances in information systems science, vol 1. Springer. US, Boston, pp 251–292
58. Witten IH (1977) An adaptive optimal controller for discrete-time markov environments. *Inf Control* 34(4):286–295
59. Liu X, Balakrishnan S (2000) Convergence analysis of adaptive critic based optimal control. In: Proceedings of the American control conference, vol 3
60. Grondman I, Buşoniu L, Lopes GA, Babuška R (2012) A survey of actor-critic reinforcement learning: standard and natural policy gradients. *IEEE Trans Syst Man Cybern Part C Appl Rev* 42(6):1291–1307
61. Prokhorov D, Santiago R, Wunsch D (1995) Adaptive critic designs: a case study for neuro-control. *Neural Netw* 8(9):1367–1372
62. Fuselli D, De Angelis F, Boaro M, Squartini S, Wei Q, Liu D, Piazza F (2013) Action dependent heuristic dynamic programming for home energy resource scheduling. *Int J Electr Power Energy Syst* 48:148–160
63. Miller WT, Sutton R, Werbos P (1990) Neural networks for control. MIT Press, Cambridge
64. Werbos P (1987) Building and understanding adaptive systems: a statistical/numerical approach to factory automation and brain research. *IEEE Trans Syst Man Cybern* 17(1):7–20
65. Werbos PJ (1989) Back propagation: past and future. Proceedings of the international conference on neural network 1:1343–1353
66. Werbos PJ (1990) Backpropagation through time: what it does and how to do it. *Proc IEEE* 78(10):1550–1560
67. Werbos P (2000) New directions in ACDs: keys to intelligent control and understanding the brain. Proceedings of the IEEE-INNS-ENNS international joint conference on neural network 3:61–66
68. Si J, Wang Y (2001) On-line learning control by association and reinforcement. *IEEE Trans Neural Netw* 12(2):264–276
69. Yang L, Enns R, Wang YT, Si J (2003) Direct neural dynamic programming. In: Stability and control of dynamical systems with applications. Springer, Berlin, pp 193–214
70. Balakrishnan S (1996) Adaptive-critic-based neural networks for aircraft optimal control. *J Guid Control Dynam* 19(4):893–898
71. Lendaris G, Schultz L, Shannon T (2000) Adaptive critic design for intelligent steering and speed control of a 2-axle vehicle. In: International joint conference on neural network, pp 73–78

72. Ferrari S, Stengel R (2002) An adaptive critic global controller. *Proc Am Control Conf* 4:2665–2670
73. Han D, Balakrishnan S (2002) State-constrained agile missile control with adaptive-critic-based neural networks. *IEEE Trans Control Syst Technol* 10(4):481–489
74. He P, Jagannathan S (2007) Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. *IEEE Trans Syst Man Cybern Part B Cybern* 37(2):425–436
75. Dierks T, Thumati B, Jagannathan S (2009) Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. *Neural Netw* 22(5–6):851–860
76. Wang D, Liu D, Wei Q (2012) Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach. *Neurocomputing* 78(1):14–22
77. Zhang H, Liu D, Luo Y, Wang D (2013) Adaptive dynamic programming for control algorithms and stability. *Communications and control engineering*, Springer, London
78. Wei Q, Liu D (2013) Optimal tracking control scheme for discrete-time nonlinear systems with approximation errors. In: Guo C, Hou ZG, Zeng Z (eds) *Advances in neural networks - ISNN 2013*, vol 7952. Lecture notes in computer science. Springer, Berlin, pp 1–10
79. Liu D, Wei Q (2014) Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems. *IEEE Trans Neural Netw Learn Syst* 25(3):621–634
80. Yang X, Liu D, Wei Q, Wang D (2015) Direct adaptive control for a class of discrete-time unknown nonaffine nonlinear systems using neural networks. *Int J Robust Nonlinear Control* 25(12):1844–1861
81. Baird L (1993) Advantage updating. Technical report, Wright Lab, Wright-Patterson Air Force Base, OH
82. Doya K (2000) Reinforcement learning in continuous time and space. *Neural Comput* 12(1):219–245
83. Vrabie D, Pastravanu O, Abu-Khalaf M, Lewis FL (2009) Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica* 45(2):477–484
84. Hanselmann T, Noakes L, Zaknich A (2007) Continuous-time adaptive critics. *IEEE Trans Neural Netw* 18(3):631–647
85. Vamvoudakis KG, Lewis FL (2010) Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica* 46(5):878–888
86. Khalil HK (2002) *Nonlinear systems*, 3rd edn. Prentice Hall, Upper Saddle River
87. Wang K, Liu Y, Li L (2014) Visual servoing trajectory tracking of nonholonomic mobile robots without direct position measurement. *IEEE Trans Robot* 30(4):1026–1035
88. Wang D, Liu D, Zhang Q, Zhao D (2016) Data-based adaptive critic designs for nonlinear robust optimal control with uncertain dynamics. *IEEE Trans Syst Man Cybern Syst* 46(11):1544–1555
89. Vamvoudakis KG, Vrabie D, Lewis FL (2009) Online policy iteration based algorithms to solve the continuous-time infinite horizon optimal control problem. *IEEE symposium on adaptive dynamic programming and reinforcement learning*, IEEE, pp 36–41
90. Vrabie D, Vamvoudakis KG, Lewis FL (2009) Adaptive optimal controllers based on generalized policy iteration in a continuous-time framework. In: *Proceedings of the mediterranean conference on control and automation*, IEEE, pp 1402–1409
91. Vrabie D, Lewis FL (2010) Integral reinforcement learning for online computation of feedback nash strategies of nonzero-sum differential games. In: *Proceedings of the IEEE Conference on decision and control*, pp 3066–3071
92. Modares H, Lewis FL, Naghibi-Sistani MB (2014) Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica* 50(1):193–202
93. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92

94. Kamalapurkar R, Dinh H, Bhasin S, Dixon WE (2015) Approximate optimal trajectory tracking for continuous-time nonlinear systems. *Automatica* 51:40–48
95. Singh SP (1992) Reinforcement learning with a hierarchy of abstract models. In: AAAI national conference on artificial intelligence 92:202–207
96. Atkeson CG, Schaal S (1997) Robot learning from demonstration. *Int Conf Mach Learn* 97:12–20
97. Abbeel P, Quigley M, Ng AY (2006) Using inaccurate models in reinforcement learning. In: International conference on machine learning. ACM, New York, pp 1–8
98. Deisenroth MP (2010) Efficient reinforcement learning using Gaussian processes. KIT Scientific Publishing
99. Mitrovic D, Klanke S, Vijayakumar S (2010) Adaptive optimal feedback control with learned internal dynamics models. In: Sigaud O, Peters J (eds) From motor learning to interaction learning in robots, vol 264. Studies in computational intelligence. Springer, Berlin, pp 65–84
100. Deisenroth MP, Rasmussen CE (2011) Pilco: a model-based and data-efficient approach to policy search. In: International conference on machine learning, pp 465–472
101. Narendra KS, Annaswamy AM (1987) A new adaptive law for robust adaptive control without persistent excitation. *IEEE Trans Autom Control* 32:134–145
102. Narendra K, Annaswamy A (1989) Stable adaptive systems. Prentice-Hall Inc, Upper Saddle River
103. Sastry S, Bodson M (1989) Adaptive control: stability, convergence, and robustness. Prentice-Hall, Upper Saddle River
104. Ioannou P, Sun J (1996) Robust adaptive control. Prentice Hall, Upper Saddle River
105. Vrabie D, Abu-Khalaf M, Lewis F, Wang Y (2007) Continuous-time ADP for linear systems with partially unknown dynamics. In: Proceedings of the IEEE international symposium on approximate dynamic programming and reformulation learning, pp 247–253
106. Bellman R (1954) The theory of dynamic programming. Technical report, DTIC Document
107. Pontryagin LS, Boltyanskii VG, Gamkrelidze RV, Mishchenko EF (1962) The mathematical theory of optimal processes. Interscience, New York
108. Kiumarsi B, Vamvoudakis KG, Modares H, Lewis FL (to appear) Optimal and autonomous control using reinforcement learning: a survey. *IEEE Trans Neural Netw Learn Syst*
109. Rummery GA, Niranjan M (1994) On-line q-learning using connectionist systems, Technical report. University of Cambridge, Department of Engineering
110. Sutton R (1996) Generalization in reinforcement learning: successful examples using sparse coarse coding. In: Advances in neural information processing systems, pp 1038–1044
111. Watkins CJCH (1989) Learning from delayed rewards. PhD thesis, University of Cambridge England
112. Schwartz A (1993) A reinforcement learning method for maximizing undiscounted rewards. *Proc Int Conf Mach Learn* 298:298–305
113. Bradtke S, Ydstie B, Barto A (1994) Adaptive linear quadratic control using policy iteration. In: Proceedings of the American control conference, IEEE, pp 3475–3479
114. McLain T, Beard R (1998) Successive galerkin approximations to the nonlinear optimal control of an underwater robotic vehicle. In: Proceedings of the IEEE international conference on robotics and automation
115. Lawton J, Beard R, McLain T (1999) Successive Galerkin approximation of nonlinear optimal attitude. *Proc Am Control Conf* 6:4373–4377
116. Lawton J, Beard R (1998) Numerically efficient approximations to the Hamilton–Jacobi–Bellman equation. *Proc Am Control Conf* 1:195–199
117. Bertsekas D (2011) Approximate policy iteration: a survey and some new methods. *J Control Theory Appl* 9:310–335
118. Modares H, Lewis FL, Naghibi-Sistani MB (2013) Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks. *IEEE Trans Neural Netw Learn Syst* 24(10):1513–1525
119. Luo B, Wu HN, Huang T, Liu D (2014) Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design. *Automatica*

120. Williams RJ (1988) Toward a theory of reinforcement-learning connectionist systems. Technical report. NU-CCS-88-3, Northeastern University, College of Computer Science
121. Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach Learn* 8(3):229–256
122. Jaakkola T, Singh S, Jordan M (1995) Reinforcement learning algorithm for partially observable Markov decision problems. In: Advances in neural information processing systems, pp 345–352
123. Kimura H, Miyazaki K, Kobayashi S (1997) Reinforcement learning in pomdps with function approximation. *Proc Int Conf Mach Learn* 97:152–160
124. Sutton RS, McAllester DA, Singh SP, Mansour Y (2000) Policy gradient methods for reinforcement learning with function approximation. In: Solla SA, Leen TK, Müller K (eds) Advances in neural information processing systems, vol 12, MIT Press, pp 1057–1063
125. Vamvoudakis KG, Lewis FL (2009) Online synchronous policy iteration method for optimal control. In: Yu W (ed) Recent advances in intelligent control systems. Springer, Berlin, pp 357–374
126. Chen Z, Jagannathan S (2008) Generalized Hamilton-Jacobi-Bellman formulation -based neural network control of affine nonlinear discrete-time systems. *IEEE Trans Neural Netw* 19(1):90–106
127. Bhasin S, Sharma N, Patre P, Dixon WE (2010) Robust asymptotic tracking of a class of nonlinear systems using an adaptive critic based controller. In: Proceedings of the American control conference, Baltimore, MD, pp 3223–3228
128. Jiang Y, Jiang ZP (2012) Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica* 48(10):2699–2704
129. Yang X, Liu D, Wang D (2014) Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints. *Int J Control* 87(3):553–566
130. Barto AG, Bradtko SJ, Singh SP (1991) Real-time learning and control using asynchronous dynamic programming, Technical report. University of Massachusetts at Amherst, Department of Computer and Information Science
131. Bertsekas DP, Tsitsiklis JN (1989) Parallel and distributed computation: numerical methods. Prentice-Hall Inc, Englewood Cliffs
132. Bertsekas DP (1976) On error bounds for successive approximation methods. *IEEE Trans Autom Control* 21(3):394–396
133. Sutton RS (1990) Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In: Proceedings of the international conference on machine learning, pp 216–224
134. Sutton RS (1991) Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bull* 2(4):160–163
135. Atkeson CG, Santamaría JC (1997) A comparison of direct and model-based reinforcement learning. In: Proceedings of the international conference on robotics and automation, Citeseer
136. Abbeel P, Ng AY (2005) Exploration and apprenticeship learning in reinforcement learning. In: Proceedings of the international conference on machine learning. ACM, pp 1–8
137. Yang X, Liu D, Wei Q (2014) Online approximate optimal control for affine non-linear systems with unknown internal dynamics using adaptive dynamic programming. *IET Control Theory Appl* 8(16):1676–1688
138. Bian T, Jiang Y, Jiang ZP (2015) Decentralized adaptive optimal control of large-scale systems with application to power systems. *IEEE Trans Ind Electron* 62(4):2439–2447
139. Modares H, Lewis FL, Jiang ZP (2015)  $H_\infty$  tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. *IEEE Trans Neural Netw Learn Syst*
140. Song R, Lewis FL, Wei Q, Zhang HG, Jiang ZP, Levine D (2015) Multiple actor-critic structures for continuous-time optimal control using input-output data. *IEEE Trans Neural Netw Learn Syst* 26(4):851–865

# Chapter 3

## Excitation-Based Online Approximate Optimal Control



### 3.1 Introduction

The focus of this chapter is adaptive online approximate optimal control of uncertain nonlinear systems. The state-derivative-based method summarized in Sect. 2.6 is further developed in this chapter. In Sect. 3.2, a novel actor-critic-identifier architecture is developed to obviate the need to know the system drift dynamics via simultaneous learning of the actor, the critic, and the identifier. The actor-critic-identifier method utilizes a persistence of excitation-based online learning scheme, and hence is an indirect adaptive control approach to reinforcement learning. The idea is similar to the heuristic dynamic programming algorithm [1], where Werbos suggested the use of a model network along with the actor and critic networks. Because of the generality of the considered system and objective function, the developed solution approach can be used in a wide range of applications in different fields.

The actor and critic neural networks developed in this chapter use gradient and least-squares-based update laws, respectively, to minimize the Bellman error. The identifier dynamic neural network is a combination of a Hopfield-type [2] component and a novel RISE (Robust Integral of Sign of the Error) component. The Hopfield component of the dynamic neural network learns the system dynamics based on online gradient-based weight tuning laws, while the RISE term robustly accounts for the function reconstruction errors, guaranteeing asymptotic estimation of the state and the state derivative. Online estimation of the state derivative allows the actor-critic-identifier architecture to be implemented without knowledge of system drift dynamics; however, knowledge of the input gain matrix is required to implement the control policy. While the design of the actor and critic are coupled through the Hamilton–Jacobi–Bellman equation, the design of the identifier is decoupled from the actor and the critic components, and can be considered as a modular component in the actor-critic-identifier architecture. Convergence of the actor-critic-identifier algorithm and stability of the closed-loop system are analyzed using Lyapunov-based adaptive control methods, and a persistence of excitation condition is used to guarantee exponential convergence to a bounded region in the neighborhood of the optimal control and uniformly ultimately bounded stability of the closed-loop system.

In Sect. 3.3, the developed actor-critic-identifier architecture is extended to a class of trajectory tracking problems. Approximate dynamic programming has been investigated and used as a tool to approximately solve optimal regulation problems. For these problems, function approximation techniques can be used to approximate the value function because it is a time invariant function. In tracking problems, the tracking error, and hence, the value function, is a function of the state and an explicit function of time. Approximation techniques like neural networks are commonly used in approximate dynamic programming literature for value function approximation. However, neural networks can only approximate functions on compact domains. Since the time-interval in an infinite-horizon problem is not compact, temporal features of the value function cannot be effectively identified using a neural network.

In Sect. 3.3, the tracking error and the desired trajectory both serve as inputs to the neural network, leading to a different Hamilton–Jacobi–Bellman equation that yields an optimal controller with a time-varying feedback component. In particular, this chapter addresses the technical obstacles that result from the time-varying nature of the optimal control problem by using a system transformation to convert the problem into a time-invariant optimal control problem. The resulting value function is a time-invariant function of the transformed states, and hence, lends itself to approximation using a neural network. A Lyapunov-based analysis is used to establish uniformly ultimately bounded tracking and approximate optimality of the controller. Simulation results are presented to demonstrate the applicability of the developed technique. To gauge the performance of the proposed method, a comparison with a numerical optimal solution is also presented.

In Sect. 3.4, the actor-critic-identifier architecture is extended to solve a  $N$ -player nonzero-sum infinite-horizon differential game subject to continuous-time uncertain nonlinear dynamics. Classical optimal control problems in the Bernoulli form aim to find a single control input that minimizes a single cost functional under boundary constraints and dynamical constraints imposed by the system [3, 4]. Various control problems can be modeled as multi-input systems, where each input is computed by a player, and each player attempts to influence the system state to minimize its own cost function. In this case, the objective is to find a Nash equilibrium solution to the resulting differential game.

In general, Nash equilibria are not unique. For a closed-loop differential game (i.e., the control is a function of the state and time) with perfect information (i.e., all the players know the complete state history), there can be infinitely many Nash equilibria. If the policies are constrained to be feedback policies, the resulting equilibria are called (sub)game perfect Nash equilibria or feedback-Nash equilibria. The value functions corresponding to feedback-Nash equilibria satisfy a coupled system of Hamilton–Jacobi equations (see, e.g., [5–8]). In this chapter,  $N$ -actor and  $N$ -critic neural network structures are used to approximate the optimal control laws and the optimal value function set, respectively. The main traits of this online algorithm involve the use of approximate dynamic programming techniques and adaptive theory to determine the feedback-Nash equilibrium solution to the game in a manner that does not require full knowledge of the system dynamics and approximately solves the underlying set of coupled Hamilton–Jacobi–Bellman equations of the

game problem. For an equivalent nonlinear system, previous research makes use of offline procedures or requires full knowledge of the system dynamics to determine the Nash equilibrium. A Lyapunov-based stability analysis shows that uniformly ultimately bounded tracking for the closed-loop system is guaranteed for the proposed actor-critic-identifier architecture and a convergence analysis demonstrates that the approximate control policies converge to a neighborhood of the optimal solutions.

## 3.2 Online Optimal Regulation<sup>1</sup>

In this section, an online adaptive reinforcement learning-based solution is developed for the infinite-horizon optimal control problem for continuous-time uncertain nonlinear systems. Consider the control-affine nonlinear system in (1.9). Recall from Sect. 2.6 the approximation of the Bellman error given by

$$\hat{\delta}_t(t) = \nabla_x \hat{V} \left( x(t), \hat{W}_c(t) \right) \dot{\hat{x}}(t) + r \left( x(t), \hat{u} \left( x(t), \hat{W}_a(t) \right) \right). \quad (3.1)$$

The actor and the critic adjust the weights  $\hat{W}_a(\cdot)$  and  $\hat{W}_c(\cdot)$ , respectively, to minimize the approximate Bellman error. The identifier learns the derivatives  $\dot{\hat{x}}(\cdot)$  to minimize the error between the true Bellman error and its approximation. The following assumptions facilitate the development of update laws for the identifier, the critic, and the actor.

**Assumption 3.1** The functions  $f$  and  $g$  are twice continuously differentiable.

**Assumption 3.2** The input gain matrix  $g(x)$  is known and uniformly bounded for all  $x \in \mathbb{R}^n$  (i.e.,  $0 < \|g(x)\| \leq \bar{g}$ ,  $\forall x \in \mathbb{R}^n$ , where  $\bar{g}$  is a known positive constant).

### 3.2.1 Identifier Design

To facilitate the design of the identifier, the following restriction is placed on the control input.

**Assumption 3.3** The control input is bounded (i.e.,  $u(\cdot) \in \mathcal{L}_\infty$ ).

Using Assumption 3.2, Property 2.3, and the projection algorithm in (3.27), Assumption 3.3 holds for the control design  $u(t) = \hat{u} \left( x(t), \hat{W}_a(t) \right)$  in (2.9). Using Assumption 3.3, the dynamic system in (1.9), with control  $u(\cdot)$ , can be represented using a multi-layer neural network as

---

<sup>1</sup>Parts of the text in this section are reproduced, with permission, from [9], ©2013, Elsevier.

$$\dot{x}(t) = F_u(x(t), u(t)) \triangleq W_f^T \sigma_f(V_f^T x(t)) + \varepsilon_f(x(t)) + g(x(t)) u(t), \quad (3.2)$$

where  $W_f \in \mathbb{R}^{(L_f+1) \times n}$ ,  $V_f \in \mathbb{R}^{n \times L_f}$  are the unknown ideal neural network weights,  $\sigma_f : \mathbb{R}^{L_f} \rightarrow \mathbb{R}^{L_f+1}$  is the neural network activation function, and  $\varepsilon_f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is the function reconstruction error. The following multi-layer dynamic neural network identifier is used to approximate the system in (3.2)

$$\dot{\hat{x}}(t) = \hat{F}_u(x(t), \hat{x}(t), u(t)) \triangleq \hat{W}_f^T(t) \sigma_f(\hat{V}_f^T(t) \hat{x}(t)) + g(x(t)) u(t) + \mu(t), \quad (3.3)$$

where  $\hat{x} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the dynamic neural network state,  $\hat{W}_f : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_f+1 \times n}$  and  $\hat{V}_f : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{n \times L_f}$  are weight estimates, and  $\mu : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  denotes the RISE feedback term defined as [10, 11]

$$\mu(t) \triangleq k\tilde{x}(t) - k\tilde{x}(0) + v(t), \quad (3.4)$$

where  $\tilde{x}(t) \triangleq x(t) - \hat{x}(t) \in \mathbb{R}^n$  is the identification error, and  $v(t) \in \mathbb{R}^n$  is a Filippov solution [12] to the initial value problem

$$\dot{v}(t) = (k\alpha + \gamma)\tilde{x}(t) + \beta_1 \text{sgn}(\tilde{x}(t)), \quad v(0) = 0,$$

where  $k, \alpha, \gamma, \beta_1 \in \mathbb{R}$  are positive constant control gains. The identification error dynamics can be written as

$$\begin{aligned} \dot{\tilde{x}}(t) &= \tilde{F}_u(x(t), \hat{x}(t), u(t)) \\ &= W_f^T \sigma_f(V_f^T x(t)) - \hat{W}_f^T(t) \sigma_f(\hat{V}_f^T(t) \hat{x}(t)) + \varepsilon_f(x(t)) - \mu(t), \end{aligned} \quad (3.5)$$

where  $\tilde{F}_u(x, \hat{x}, u) \triangleq F_u(x, u) - \hat{F}_u(x, \hat{x}, u) \in \mathbb{R}^n$ . A filtered identification error is defined as

$$e_f(t) \triangleq \dot{\tilde{x}}(t) + \alpha\tilde{x}(t). \quad (3.6)$$

Taking the time derivative of (3.6) and using (3.5) yields

$$\begin{aligned} \dot{e}_f(t) &= W_f^T \nabla_{V_f^T x} \sigma_f(V_f^T x(t)) V_f^T \dot{x}(t) - \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f(\hat{V}_f^T(t) \hat{x}(t)) \hat{V}_f^T(t) \dot{\hat{x}}(t) \\ &\quad - \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f(\hat{V}_f^T(t) \hat{x}(t)) \dot{\hat{V}}_f^T(t) \hat{x}(t) - \hat{W}_f^T(t) \sigma_f(\hat{V}_f^T(t) \hat{x}(t)) \\ &\quad + \dot{\varepsilon}_f(x(t), \dot{x}(t)) - k e_f(t) - \gamma \tilde{x}(t) - \beta_1 \text{sgn}(\tilde{x}(t)) + \alpha \dot{\tilde{x}}(t). \end{aligned} \quad (3.7)$$

Based on (3.7) and the subsequent stability analysis, the weight update laws for the dynamic neural network are designed as

$$\dot{\hat{W}}_f(t) = \text{proj} \left( \Gamma_{wf} \nabla_{V_f^T x} \sigma_f(\hat{V}_f^T(t) \hat{x}(t)) \hat{V}_f^T(t) \dot{\hat{x}}(t) \tilde{x}^T(t) \right),$$

$$\dot{\hat{V}}_f(t) = \text{proj} \left( \Gamma_{vf} \dot{\hat{x}}(t) \tilde{x}^T(t) \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \right), \quad (3.8)$$

where the projection operator is used to bound the weight estimates such that  $\|\hat{W}_f(t)\|, \|\hat{V}_f(t)\| \leq \bar{W}_f$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ ,  $\bar{W}_f \in \mathbb{R}_{>0}$  is a constant, and  $\Gamma_{vf} \in \mathbb{R}^{L_f+1 \times L_f+1}$  and  $\Gamma_{vf} \in \mathbb{R}^{n \times n}$  are positive constant adaptation gain matrices. The expression in (3.7) can be rewritten as

$$\dot{e}_f(t) = \tilde{N}(t) + N_{B1}(t) + \hat{N}_{B2}(t) - k e_f(t) - \gamma \tilde{x}(t) - \beta_1 \text{sgn}(\tilde{x}(t)), \quad (3.9)$$

where the auxiliary signals,  $\tilde{N} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$ ,  $N_{B1} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$ , and  $\hat{N}_{B2} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  are defined as

$$\begin{aligned} \tilde{N}(t) &\triangleq \alpha \dot{\hat{x}}(t) - \hat{W}_f^T(t) \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) - \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \dot{\hat{V}}_f^T(t) \hat{x}(t) \\ &\quad + \frac{1}{2} W_f^T \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\tilde{x}}(t) \\ &\quad + \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) V_f^T \dot{\tilde{x}}(t), \end{aligned} \quad (3.10)$$

$$\begin{aligned} N_{B1}(t) &\triangleq W_f^T \nabla_{V_f^T x} \sigma_f \left( V_f^T x(t) \right) V_f^T \dot{x}(t) - \frac{1}{2} W_f^T \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{x}(t) \\ &\quad - \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) V_f^T(t) \dot{x}(t) + \dot{e}_f(x(t), \dot{x}(t)), \end{aligned} \quad (3.11)$$

$$\begin{aligned} \hat{N}_{B2}(t) &\triangleq \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\tilde{x}}(t) \\ &\quad + \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \tilde{V}_f^T(t) \dot{\hat{x}}(t), \end{aligned} \quad (3.12)$$

where  $\tilde{W}_f(t) \triangleq W_f - \hat{W}_f(t)$  and  $\tilde{V}_f(t) \triangleq V_f - \hat{V}_f(t)$ . To facilitate the subsequent stability analysis, an auxiliary term  $N_{B2} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is defined by replacing  $\dot{\hat{x}}(t)$  in  $\hat{N}_{B2}(t)$  by  $\dot{x}(t)$ , and  $\tilde{N}_{B2} \triangleq \hat{N}_{B2} - N_{B2}$ . The terms  $N_{B1}$  and  $N_{B2}$  are grouped as  $N_B \triangleq N_{B1} + N_{B2}$ .

Provided  $x(t) \in \chi$ , where  $\chi \subset \mathbb{R}^n$  is a compact set containing the origin, using Assumption 3.2, Property 2.3, (3.6), (3.8), (3.11), and (3.12), the following bounds can be obtained

$$\|\tilde{N}(t)\| \leq \rho_1(\|z(t)\|) \|z(t)\|, \quad (3.13)$$

$$\|N_{B1}(t)\| \leq \zeta_1, \quad \|N_{B2}(t)\| \leq \zeta_2, \quad (3.14)$$

$$\|\dot{N}_B(t)\| \leq \zeta_3 + \zeta_4 \rho_2(\|z(t)\|) \|z(t)\|, \quad (3.14)$$

$$\|\dot{\tilde{x}}^T(t) \tilde{N}_{B2}(t)\| \leq \zeta_5 \|\tilde{x}(t)\|^2 + \zeta_6 \|e_f(t)\|^2, \quad (3.15)$$

$\forall t \in \mathbb{R}_{\geq t_0}$ , where  $\rho_1, \rho_2 : \mathbb{R} \rightarrow \mathbb{R}$  are positive strictly increasing functions arising from the Mean Value Theorem (see [13]),  $z \triangleq \begin{bmatrix} \tilde{x}^T & e_f^T \end{bmatrix}^T \in \mathbb{R}^{2n}$ , and  $\zeta_i \in \mathbb{R}$ ,  $i = 1, \dots, 6$  are positive constants. To facilitate the analysis, assume temporarily that  $x(t) \in \chi$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ . Let the auxiliary signal  $y : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{2n+2}$  be defined as

$$y(t) \triangleq \begin{bmatrix} \tilde{x}^T(t) & e_f^T(t) & \sqrt{P(t)} & \sqrt{Q(t)} \end{bmatrix}^T. \quad (3.16)$$

In (3.16), the auxiliary signal  $Q : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is defined as

$$Q(t) \triangleq \frac{1}{4}\alpha \left[ \text{tr}(\tilde{W}_f^T(t) \Gamma_{wf}^{-1} \tilde{W}_f(t)) + \text{tr}(\tilde{V}_f^T(t) \Gamma_{vf}^{-1} \tilde{V}_f(t)) \right],$$

where the auxiliary function  $P : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the Filippov solution [12] to the differential equation

$$\dot{P}(t) = -L(z(t), t), \quad P(t_0) = \beta_1 \sum_{i=1}^n |\tilde{x}_i(t_0)| - \tilde{x}^T(t_0) N_B(t_0). \quad (3.17)$$

In (3.17), the auxiliary function  $L : \mathbb{R}^{2n} \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is defined as

$$L(z, t) \triangleq e_f^T(N_{B1}(t) - \beta_1 \text{sgn}(\tilde{x}(t))) + \dot{\tilde{x}}^T(t) N_{B2}(t) - \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}(t)\|, \quad (3.18)$$

where  $\beta_1, \beta_2 \in \mathbb{R}$  are selected according to the sufficient conditions

$$\beta_1 > \max(\zeta_1 + \zeta_2, \zeta_1 + \frac{\zeta_3}{\alpha}), \quad \beta_2 > \zeta_4 \quad (3.19)$$

to ensure  $P(t) \geq 0$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$  (see Appendix A.1.1).

Let  $\mathcal{D} \subset \mathbb{R}^{2n+2}$  be an open and connected set defined as  $\mathcal{D} \triangleq \{y \in \mathbb{R}^{2n+2} \mid \|y\| < \inf(\rho^{-1}([2\sqrt{\lambda\eta}, \infty)))\}$ , where  $\lambda$  and  $\eta$  are defined in Appendix A.1.2. Let  $\overline{\mathcal{D}}$  be the compact set  $\overline{\mathcal{D}} \triangleq \{y \in \mathbb{R}^{2n+2} \mid \|y\| \leq \inf(\rho^{-1}([2\sqrt{\lambda\eta}, \infty)))\}$ . Let  $V_I : \mathcal{D} \rightarrow \mathbb{R}$  be a positive-definite, locally Lipschitz, regular function defined as

$$V_I(y) \triangleq \frac{1}{2}e_f^T e_f + \frac{1}{2}\gamma \tilde{x}^T \tilde{x} + P + Q. \quad (3.20)$$

The candidate Lyapunov function in (3.20) satisfies the inequalities

$$U_1(y) \leq V_I(y) \leq U_2(y), \quad (3.21)$$

where  $U_1(y), U_2(y) \in \mathbb{R}$  are continuous positive definite functions defined as

$$U_1 \triangleq \frac{1}{2} \min(1, \gamma) \|y\|^2 \quad U_2 \triangleq \max(1, \gamma) \|y\|^2.$$

Additionally, let  $\mathcal{S} \subset \mathcal{D}$  denote a set defined as  $\mathcal{S} \triangleq \{y \in \mathcal{D} \mid \rho(\sqrt{2U_2(y)}) < 2\sqrt{\lambda\eta}\}$  and let

$$\dot{y}(t) = h(y(t), t) \quad (3.22)$$

represent the closed-loop differential equations in (3.5), (3.8), (3.9), and (3.17), where  $h : \mathbb{R}^{2n+2} \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{2n+2}$  denotes the right-hand side of the closed-loop error signals.

**Theorem 3.4** *For the system in (1.9), the identifier developed in (3.3) along with the weight update laws in (3.8) ensures asymptotic identification of the state and its derivative, in the sense that all Filippov solutions to (3.22) that satisfy  $y(t_0) \in \mathcal{S}$ , are bounded, and further satisfy*

$$\lim_{t \rightarrow \infty} \|\tilde{x}(t)\| = 0, \quad \lim_{t \rightarrow \infty} \|\dot{\tilde{x}}(t)\| = 0,$$

*provided the control gains  $k$  and  $\gamma$  are selected sufficiently large based on the initial conditions of the states and satisfy the sufficient conditions*

$$\gamma > \frac{\zeta_5}{\alpha}, \quad k > \zeta_6, \quad (3.23)$$

*where  $\zeta_5$  and  $\zeta_6$  are introduced in (3.15), and  $\beta_1$ , and  $\beta_2$ , introduced in (3.18), are selected according to the sufficient conditions in (3.19).*

*Proof* See Appendix A.1.2. □

### 3.2.2 Least-Squares Update for the Critic

For online implementation, a normalized recursive formulation of the least-squares algorithm is developed for the critic update law as

$$\dot{\hat{W}}_c(t) = -k_c \Gamma(t) \frac{\omega(t)}{1 + v\omega^T(t) \Gamma(t) \omega(t)} \hat{\delta}_t(t), \quad (3.24)$$

where  $\omega : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^L$ , defined as  $\omega(t) \triangleq \nabla_x \sigma(x(t)) \hat{F}_u(x(t), \hat{x}(t), \hat{u}(x(t), \hat{W}_a(t)))$  is the critic neural network regressor vector,  $v, k_c \in \mathbb{R}$  are constant positive gains, and  $\Gamma : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L \times L}$  is a symmetric estimation gain matrix generated using the initial value problem

$$\dot{\Gamma}(t) = -k_c \Gamma(t) \frac{\omega(t) \omega^T(t)}{1 + v\omega^T(t) \Gamma(t) \omega(t)} \Gamma(t); \quad \Gamma(t_r^+) = \Gamma(0) = \bar{\gamma} I_L, \quad (3.25)$$

where  $t_r^+$  is the resetting time at which  $\lambda_{\min}\{\Gamma(t)\} \leq \underline{\gamma}, \bar{\gamma} > \underline{\gamma} > 0$ . The covariance resetting ensures that  $\Gamma(\cdot)$  is positive-definite for all time and prevents its value from becoming arbitrarily small in some directions, thus avoiding slow adaptation in some directions. From (3.25), it is clear that  $\dot{\Gamma}(t) \leq 0$ , which means that  $\Gamma(\cdot)$  can be bounded as

$$\underline{\gamma} \mathbf{I}_L \leq \Gamma(t) \leq \bar{\gamma} \mathbf{I}_L, \quad \forall t \in \mathbb{R}_{\geq t_0}. \quad (3.26)$$

### 3.2.3 Gradient Update for the Actor

The actor update, like the critic update in Sect. 3.2.2, is based on the minimization of the Bellman error  $\hat{\delta}_t$ . However, unlike the critic weights, the actor weights appear nonlinearly in  $\hat{\delta}_t$ , making it problematic to develop a least-squares update law. Hence, a gradient update law is developed for the actor which minimizes the squared Bellman error. The gradient-based update law for the actor neural network is given by

$$\begin{aligned} \dot{\hat{W}}_a(t) = & \text{proj} \left\{ -\frac{2k_{a1}}{\sqrt{1 + \omega^T(t)\omega(t)}} \cdot \right. \\ & \left( \hat{W}_c^T(t) \nabla_x \sigma(x(t)) \frac{\partial \hat{F}_u(x(t), \hat{x}(t), \hat{u}(x(t), \hat{W}_a(t)))}{\partial \hat{u}} \frac{\partial \hat{u}(x(t), \hat{W}_a(t))}{\partial \hat{W}_a} \right)^T \hat{\delta}_t(t) \\ & - \frac{4k_{a1}}{\sqrt{1 + \omega^T(t)\omega(t)}} \left( \frac{\partial \hat{u}(x(t), \hat{W}_a(t))}{\partial \hat{W}_a} \right)^T R \hat{u}(x(t), \hat{W}_a(t)) \hat{\delta}_t(t) \\ & \left. - k_{a2}(\hat{W}_a(t) - \hat{W}_c(t)) \right\} \end{aligned} \quad (3.27)$$

where  $G(x) \triangleq g(x) R^{-1} g(x)^T$ ,  $k_{a1}, k_{a2} \in \mathbb{R}$  are positive adaptation gains,  $\frac{1}{\sqrt{1 + \omega^T(t)\omega(t)}}$  is the normalization term, and the last term in (3.27) is added for stability (based on the subsequent stability analysis). Using the identifier developed in (3.3), the actor weight update law can now be simplified as

$$\begin{aligned} \dot{\hat{W}}_a(t) = & \text{proj} \left\{ -\frac{k_{a1}}{\sqrt{1 + \omega^T(t)\omega(t)}} \nabla_x \sigma(x(t)) G \nabla_x \sigma^T(x(t)) (\hat{W}_a(t) - \hat{W}_c(t)) \hat{\delta}_t(t) \right. \\ & \left. - k_{a2}(\hat{W}_a(t) - \hat{W}_c(t)) \right\}. \end{aligned} \quad (3.28)$$

The projection operator ensures that  $\|\hat{W}_a(t)\| \leq \bar{W}$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ , where  $\bar{W} \in \mathbb{R}_{>0}$  is a positive constant such that  $\|W\| \leq \bar{W}$ . For notational brevity, let  $B_{\bar{W}}$  denote the set  $\{w \in \mathbb{R}^L \mid \|w\| \leq 2\bar{W}\}$ .

*Remark 3.5* A recursive least-squares update law with covariance resetting is developed for the critic in (3.24), which exploits the fact that the critic weights appear linearly in the Bellman error  $\hat{\delta}_t(\cdot)$ . This is in contrast to the modified Levenberg–Marquardt algorithm in [14] which is similar to the normalized gradient update law. The actor update law in (3.27) also differs in the sense that the update law in [14] is purely motivated by the stability analysis whereas the proposed actor update law is based on the minimization of the Bellman error with an additional term for stability. Heuristically, the difference in the update law development could lead to improved performance in terms of faster convergence of the actor and critic weights, as seen from the simulation results in Sect. 3.2.5.

### 3.2.4 Convergence and Stability Analysis

Using the Hamilton–Jacobi–Bellman equation, the unmeasurable form of the Bellman error can be written as

$$\begin{aligned}\hat{\delta}_t &= \hat{W}_c^T \omega - W_c^T \nabla_x \sigma F_{u^*} + \hat{u}^T R \hat{u} - u^{*T} R u^* - \nabla_x \epsilon F_{u^*}, \\ &= -\tilde{W}_c^T \omega - W^T \nabla_x \sigma \tilde{F}_{\hat{u}} + \frac{1}{4} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a - \frac{1}{4} \nabla_x \epsilon G \nabla_x \epsilon^T - \nabla_x \epsilon F_{u^*},\end{aligned}\quad (3.29)$$

The dynamics of the critic weight estimation error  $\tilde{W}_c$  can now be developed by substituting (3.29) into (3.24) as

$$\begin{aligned}\dot{\tilde{W}}_c &= -k_c \Gamma \psi \psi^T \tilde{W}_c + k_c \Gamma \frac{\omega}{1 + v \omega^T \Gamma \omega} \left[ -W^T \nabla_x \sigma \tilde{F}_{\hat{u}} \right. \\ &\quad \left. + \frac{1}{4} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a - \frac{1}{4} \nabla_x \epsilon G \nabla_x \epsilon^T - \nabla_x \epsilon F_{u^*} \right],\end{aligned}\quad (3.30)$$

where  $\psi \triangleq \frac{\omega}{\sqrt{1 + v \omega^T \Gamma \omega}} \in \mathbb{R}^N$  is the normalized critic regressor vector, bounded as

$$\|\psi\| \leq \frac{1}{\sqrt{v \gamma}}, \quad \forall t \in \mathbb{R}_{\geq t_0} \quad (3.31)$$

where  $\gamma$  is introduced in (3.26). The error system in (3.30) can be represented by the following perturbed system

$$\dot{\tilde{W}}_c = \Omega_{nom} + \Delta_{per}, \quad (3.32)$$

where  $\Omega_{nom}(\tilde{W}_c, t) \triangleq -k_c \Gamma(t) \psi(t) \psi^T(t) \tilde{W}_c \in \mathbb{R}^N$ , denotes the nominal system, and  $\Delta_{per} \triangleq k_c \Gamma \frac{\omega}{1 + v \omega^T \Gamma \omega} \left[ -W^T \nabla_x \sigma \tilde{F}_{\hat{u}} + \frac{1}{4} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a - \frac{1}{4} \nabla_x \epsilon G \nabla_x \epsilon^T - \nabla_x \epsilon F_{u^*} \right] \in \mathbb{R}^N$  denotes the perturbation. Using Theorem 2.5.1 in [15], the nominal system

$$\dot{\tilde{W}}_c = -k_c \Gamma \psi \psi^T \tilde{W}_c \quad (3.33)$$

is globally exponentially stable, if the bounded signal  $\psi$  is uniformly persistently exciting [16, 17] over the compact set  $\chi \times \bar{\mathcal{D}} \times \mathbf{B}_{\bar{W}}$ , such that

$$\int_t^{t+\delta} \psi(\tau) \psi(\tau)^T d\tau \geq \mu_1 \mathbf{I}_L \quad \forall t \geq t_0,$$

for some positive constants  $\mu_1, \delta \in \mathbb{R}$ . Since  $\mathcal{Q}_{nom}$  is continuously differentiable and the Jacobian  $\nabla_{\tilde{W}_c} \mathcal{Q}_{nom} = -k_c \Gamma \psi \psi^T$  is bounded for the exponentially stable system in (3.33), the converse Lyapunov Theorem 4.14 in [18] can be used to show that there exists a function  $V_c : \mathbb{R}^N \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$ , which satisfies the following inequalities

$$\begin{aligned} c_1 \|\tilde{W}_c\|^2 &\leq V_c(\tilde{W}_c, t) \leq c_2 \|\tilde{W}_c\|^2 \\ \nabla_t V_c + \nabla_{\tilde{W}_c} V_c \mathcal{Q}_{nom}(\tilde{W}_c, t) &\leq -c_3 \|\tilde{W}_c\|^2 \\ \|\nabla_{\tilde{W}_c} V_c\| &\leq c_4 \|\tilde{W}_c\|, \end{aligned} \quad (3.34)$$

for some positive constants  $c_1, c_2, c_3, c_4 \in \mathbb{R}$ . Using Property 2.3, Assumptions 3.2 and 3.3, the projection bounds in (3.27), the fact that  $F_{u^*}$  is bounded over compact sets, and provided the conditions of Theorem 3.4 hold (required to prove that  $t \mapsto \tilde{F}_{\hat{u}}(x(t), \hat{x}(t), \hat{u}(x(t), \hat{W}_a(t))) \in \mathcal{L}_\infty$ ), the following bounds can be developed:

$$\begin{aligned} \|\tilde{W}_a\| &\leq \kappa_1, \quad \|\nabla_x \sigma G \nabla_x \sigma^T\| \leq \kappa_2, \\ \left\| \frac{1}{4} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a - \frac{1}{4} \nabla_x \epsilon G \nabla_x \epsilon^T - W^T \nabla_x \sigma \tilde{F}_{\hat{u}} - \nabla_x \epsilon F_{u^*} \right\| &\leq \kappa_3, \\ \left\| \frac{1}{2} W^T \nabla_x \sigma G \nabla_x \epsilon^T + \frac{1}{2} \nabla_x \epsilon G \nabla_x \epsilon^T + \frac{1}{2} W^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a + \frac{1}{2} \nabla_x \epsilon G \nabla_x \sigma^T \right\| &\leq \kappa_4, \end{aligned} \quad (3.35)$$

where  $\kappa_1, \kappa_2, \kappa_3, \kappa_4 \in \mathbb{R}$  are computable positive constants.

**Theorem 3.6** *If Assumptions 3.1–3.3 hold, the regressor  $\psi \triangleq \frac{\omega}{\sqrt{1+\omega^T \Gamma \omega}}$  is uniformly persistently exciting, and provided (3.19), (3.23), and the following sufficient gain condition is satisfied*

$$\frac{c_3}{k_{a1}} > \kappa_1 \kappa_2, \quad (3.36)$$

where  $k_{a1}, c_3, \kappa_1, \kappa_2$  are introduced in (3.27), (3.34), and (3.35), then the controller in (1.9), the actor-critic weight update laws in (3.24)–(3.25) and (3.28), and the identifier in (3.3) and (3.8), guarantee that the state of the system  $x$ , and the actor-critic weight estimation errors  $\tilde{W}_a$  and  $\tilde{W}_c$  are uniformly ultimately bounded.

*Proof* To investigate the stability of (1.9) with control  $\hat{u}$ , and the perturbed system in (3.32), consider  $V_L : \mathcal{X} \times \mathbb{R}^N \times \mathbb{R}^N \times [0, \infty) \rightarrow \mathbb{R}$  as the continuously differentiable, positive-definite Lyapunov function candidate defined as

$$V_L(x, \tilde{W}_c, \tilde{W}_a, t) \triangleq V^*(x) + V_c(\tilde{W}_c, t) + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a,$$

where  $V^*$  (i.e., the optimal value function) is the candidate Lyapunov function for (1.9), and  $V_c$  is the Lyapunov function for the exponentially stable system in (3.33). Since  $V^*$  is continuously differentiable and positive-definite, there exist class  $\mathcal{K}$  functions (see [18, Lemma 4.3]), such that

$$\alpha_1(\|x\|) \leq V^*(x) \leq \alpha_2(\|x\|) \quad \forall x \in \mathbb{R}^n. \quad (3.37)$$

Using (3.34) and (3.37),  $V_L(x, \tilde{W}_c, \tilde{W}_a, t)$  can be bounded as

$$\alpha_1(\|x\|) + c_1 \left\| \tilde{W}_c \right\|^2 + \frac{1}{2} \left\| \tilde{W}_a \right\|^2 \leq V_L(x, \tilde{W}_c, \tilde{W}_a, t) \leq \alpha_2(\|x\|) + c_2 \left\| \tilde{W}_c \right\|^2 + \frac{1}{2} \left\| \tilde{W}_a \right\|^2,$$

which can be written as

$$\alpha_3(\|\tilde{z}\|) \leq V_L(\tilde{z}, t) \leq \alpha_4(\|\tilde{z}\|) \quad \forall \tilde{z} \in \mathbb{R}^{n+2L},$$

where  $\tilde{z} \triangleq [x \ \tilde{W}_c \ \tilde{W}_a]^T \in \mathbb{R}^{n+2N}$ , and  $\alpha_3$  and  $\alpha_4$  are class  $\mathcal{K}$  functions. Taking the time derivative of  $V_L(\cdot)$  yields

$$\dot{V}_L = \frac{\partial V^*}{\partial x} f + \frac{\partial V^*}{\partial x} g \hat{u} + \frac{\partial V_c}{\partial t} + \frac{\partial V_c}{\partial \tilde{W}_c} \mathcal{Q}_{nom} + \frac{\partial V_c}{\partial \tilde{W}_c} \Delta_{per} - \tilde{W}_a^T \dot{\hat{W}}_a, \quad (3.38)$$

where the time derivative of  $V^*$  is taken along the trajectories of the system (1.9) with control  $\hat{u}$ , and the time derivative of  $V_c$  is taken along the along the trajectories of the perturbed system (3.32). To facilitate the subsequent analysis, the Hamilton–Jacobi–Bellman equation in (1.14) is rewritten as  $\frac{\partial V^*}{\partial x} f = -\frac{\partial V^*}{\partial x} g u^* - Q - u^{*T} R u^*$ . Substituting for  $\frac{\partial V^*}{\partial x} f$  in (3.38), using the fact that  $\frac{\partial V^*}{\partial x} g = -2u^{*T} R$  from (1.13), and using (3.27) and (3.34), (3.38) can be upper bounded as

$$\begin{aligned} \dot{V}_L &\leq -Q - u^{*T} R u^* - c_3 \left\| \tilde{W}_c \right\|^2 + c_4 \left\| \tilde{W}_c \right\| \left\| \Delta_{per} \right\| + 2u^{*T} R(u^* - \hat{u}) \\ &\quad + k_{a2} \tilde{W}_a^T (\hat{W}_a - \tilde{W}_c) + \frac{k_{a1}}{\sqrt{1 + \omega^T \omega}} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T (\hat{W}_a - \tilde{W}_c) \hat{\delta}_t. \end{aligned} \quad (3.39)$$

Substituting for  $u^*$ ,  $\hat{u}$ ,  $\hat{\delta}_t$ , and  $\Delta_{per}$  using (1.13), (2.9), (3.29), and (3.32), respectively, and substituting (3.26) and (3.31) into (3.39), yields

$$\begin{aligned}
\dot{V}_L &\leq -Q - c_3 \|\tilde{W}_c\|^2 - k_{a2} \|\tilde{W}_a\|^2 + \frac{1}{2} W^T \nabla_x \sigma G \nabla_x \epsilon^T \\
&\quad + \frac{1}{2} \nabla_x \epsilon G \nabla_x \epsilon^T + \frac{1}{2} W^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a + \frac{1}{2} \nabla_x \epsilon G \nabla_x \sigma^T \tilde{W}_a \\
&\quad + c_4 \frac{k_c \bar{\gamma}}{2\sqrt{\nu\gamma}} \left\| -W^T \nabla_x \sigma \tilde{F}_{\hat{u}} + \frac{1}{4} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a \right. \\
&\quad \left. - \frac{1}{4} \nabla_x \epsilon G \nabla_x \epsilon^T - \nabla_x \epsilon F_{u^*} \right\| \|\tilde{W}_c\| + k_{a2} \|\tilde{W}_a\| \|\tilde{W}_c\| \\
&\quad + \frac{k_{a1}}{\sqrt{1+\omega^T\omega}} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T (\tilde{W}_c - \tilde{W}_a) \left( -\tilde{W}_c^T \omega \right. \\
&\quad \left. - W^T \nabla_x \sigma \tilde{F}_{\hat{u}} + \frac{1}{4} \tilde{W}_a^T \nabla_x \sigma G \nabla_x \sigma^T \tilde{W}_a - \frac{1}{4} \nabla_x \epsilon G \nabla_x \epsilon^T - \nabla_x \epsilon F_{u^*} \right). \quad (3.40)
\end{aligned}$$

Using the bounds developed in (3.35), (3.40) can be further upper bounded as

$$\begin{aligned}
\dot{V}_L &\leq -Q - (c_3 - k_{a1}\kappa_1\kappa_2) \|\tilde{W}_c\|^2 - k_{a2} \|\tilde{W}_a\|^2 + k_{a1}\kappa_1^2\kappa_2\kappa_3 + \kappa_4 \\
&\quad + \left( \frac{c_4 k_c \bar{\gamma}}{2\sqrt{\nu\gamma}} \kappa_3 + k_{a1}\kappa_1\kappa_2\kappa_3 + k_{a1}\kappa_1^2\kappa_2 + k_{a2}\kappa_1 \right) \|\tilde{W}_c\|.
\end{aligned}$$

Provided  $c_3 > k_{a1}\kappa_1\kappa_2$ , completion of the square yields

$$\begin{aligned}
\dot{V}_L &\leq -Q - (1-\theta)(c_3 - k_{a1}\kappa_1\kappa_2) \|\tilde{W}_c\|^2 - k_{a2} \|\tilde{W}_a\|^2 + k_{a1}\kappa_1^2\kappa_2\kappa_3 + \kappa_4 \\
&\quad + \frac{1}{4\theta(c_3 - k_{a1}\kappa_1\kappa_2)} \left[ \frac{c_4 k_c \bar{\gamma}}{2\sqrt{\nu\gamma}} \kappa_3 + k_{a1}\kappa_1\kappa_2\kappa_3 + k_{a1}\kappa_1^2\kappa_2 + k_{a2}\kappa_1 \right]^2, \quad (3.41)
\end{aligned}$$

where  $0 < \theta < 1$ . Since  $Q$  is positive definite, [18, Lemma 4.3] indicates that there exist class  $\mathcal{K}$  functions  $\alpha_5$  and  $\alpha_6$  such that

$$\begin{aligned}
\alpha_5(\|\tilde{z}\|) &\leq Q + (1-\theta)(c_3 - k_{a1}\kappa_1\kappa_2) \|\tilde{W}_c\|^2 + k_{a2} \|\tilde{W}_a\|^2 \\
&\leq \alpha_6(\|\tilde{z}\|) \quad \forall v \in \overline{B_s}.
\end{aligned} \quad (3.42)$$

Using (3.42), the expression in (3.41) can be further upper bounded as

$$\dot{V}_L \leq -\alpha_5(\|\tilde{z}\|) + k_{a1}\kappa_1^2\kappa_2\kappa_3 + \kappa_4 + \kappa_5,$$

where

$$\kappa_5 \triangleq \frac{1}{4\theta(c_3 - k_{a1}\kappa_1\kappa_2)} \left[ \frac{c_4 k_c \bar{\gamma}}{2\sqrt{\nu\gamma}} \kappa_3 + k_{a1}\kappa_1\kappa_2\kappa_3 + k_{a1}\kappa_1^2\kappa_2 + k_{a2}\kappa_1 \right]^2.$$

Hence,  $\dot{V}_L(t)$  is negative whenever  $\tilde{z}(t)$  lies outside the compact set

$$\Omega_{\tilde{z}} \triangleq \left\{ \tilde{z} : \|\tilde{z}\| \leq \alpha_5^{-1} (\kappa_5 + k_{a1}\kappa_1^2\kappa_2\kappa_3 + \kappa_4) \right\}.$$

Invoking [18, Theorem 4.18], it can be concluded that  $\tilde{z}(\cdot)$  is uniformly ultimately bounded. The bounds in (3.35) depend on the actor neural network approximation error  $\nabla_x \epsilon$ , which can be reduced by increasing the number of neurons,  $L$ , thereby reducing the size of the residual set  $\Omega_{\tilde{z}}$ . From Property 2.3, as the number of neurons of the actor and critic neural networks approaches infinity,  $\nabla_x \epsilon \rightarrow 0$ .  $\square$

Since  $c_3$  is a function of the critic adaptation gain  $k_c$ ,  $k_{a1}$  is the actor adaptation gain, and  $\kappa_1, \kappa_2$  are known constants, the sufficient gain condition in (3.36) can be easily satisfied.

*Remark 3.7* Since the actor, the critic, and the identifier are continuously updated, the developed reinforcement learning algorithm can be compared to fully optimistic policy iteration in machine learning literature [19]. Unlike traditional policy iteration where policy improvement is done after convergence of the policy evaluation step, fully optimistic policy iteration carries out policy evaluation and policy improvement after every state transition. Proving convergence of optimistic policy iteration is complicated and is an active area of research in machine learning [19, 20]. By considering an adaptive control framework, this result investigates the convergence and stability behavior of fully optimistic policy iteration in continuous-time.

*Remark 3.8* The persistence of excitation condition in Theorem 2 is equivalent to the exploration paradigm in reinforcement learning which ensures sufficient sampling of the state-space and convergence to the optimal policy [21].

### 3.2.5 Simulation

The following nonlinear system is considered

$$\dot{x} = \begin{bmatrix} -x_1 + x_2 \\ -0.5x_1 - 0.5x_2(1 - (\cos(2x_1) + 2)^2) \end{bmatrix} + \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix} u, \quad (3.43)$$

where  $x(t) \triangleq [x_1(t) \ x_2(t)]^T \in \mathbb{R}^2$  and  $u(t) \in \mathbb{R}$ . The state and control penalties are selected as

$$Q(x) = x^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x; \quad R = 1.$$

The optimal value function and optimal control for the system in (3.43) are known, and given by [14]

$$V^*(x) = \frac{1}{2}x_1^2 + x_2^2; \quad u^*(x) = -(\cos(2x_1) + 2)x_2.$$

The activation function for the critic neural network is selected with three neurons as

$$\sigma(x) = [x_1^2 \quad x_1x_2 \quad x_2^2]^T,$$

which yields the optimal weights  $W = [0.5 \ 0 \ 1]^T$ . The activation function for the identifier dynamic neural network is selected as a symmetric sigmoid with five neurons in the hidden layer.

*Remark 3.9* The choice of a good basis for the value function and control policy is critical for convergence. For a general nonlinear system, choosing a suitable basis can be a challenging problem without any prior knowledge about the system.

The identifier gains are selected as

$$k = 800, \quad \alpha = 300, \quad \gamma = 5, \quad \beta_1 = 0.2, \quad \Gamma_{wf} = 0.1I_6, \text{ and } \Gamma_{vf} = 0.1I_2,$$

and the gains for the actor-critic learning laws are selected as

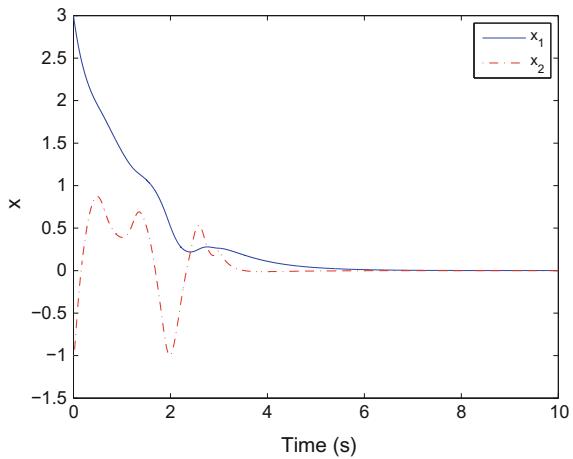
$$k_{a1} = 10, \quad k_{a2} = 50, \quad k_c = 20, \text{ and } \nu = 0.005.$$

The covariance matrix is initialized to  $\Gamma(0) = 5000I_3$ , all the neural network weights are randomly initialized in  $[-1, 1]$ , and the states are initialized to  $x(0) = [3, -1]$ . An implementation issue in using the developed algorithm is to ensure persistence of excitation of the critic regressor vector. Unlike linear systems, where persistence of excitation of the regressor translates to sufficient richness of the external input, no verifiable method exists to ensure persistence of excitation in nonlinear regulation problems. To ensure persistence of excitation qualitatively, a small exploratory signal consisting of sinusoids of varying frequencies,  $n(t) = \sin^2(t) \cos(t) + \sin^2(2t) \cos(0.1t) + \sin^2(-1.2t) \cos(0.5t) + \sin^5(t)$ , is added to the control  $u(t)$  for the first 3 s [14]. The proposed control algorithm is implemented using (2.9), (3.1), (3.3), (3.4), (3.24), (3.25), and (3.28). The evolution of states is shown in Fig. 3.1. The identifier approximates the system dynamics, and the state derivative estimation error is shown in Fig. 3.2.

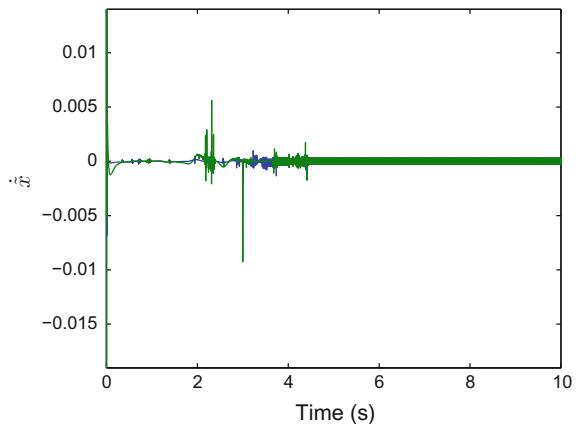
As compared to discontinuous sliding mode identifiers which require infinite bandwidth and exhibit chattering, the RISE-based identifier in (3.3) is continuous, and thus, mitigates chattering to a large extent, as seen in Fig. 3.2.

Persistence of excitation ensures that the weights converge close to their optimal values (i.e.,  $\hat{W}_c = [0.5004 \ 0.0005 \ 0.9999]^T \approx \hat{W}_a$ ) in approximately 2 s, as seen from the evolution of actor-critic weights in Figs. 3.3 and 3.4. The improved actor-critic weight update laws, based on minimization of the Bellman error, led to faster convergence of weights as compared to [14]. Errors in approximating the optimal value function and optimal control at steady state ( $t = 10$  s) are plotted against the states in Figs. 3.5 and 3.6, respectively.

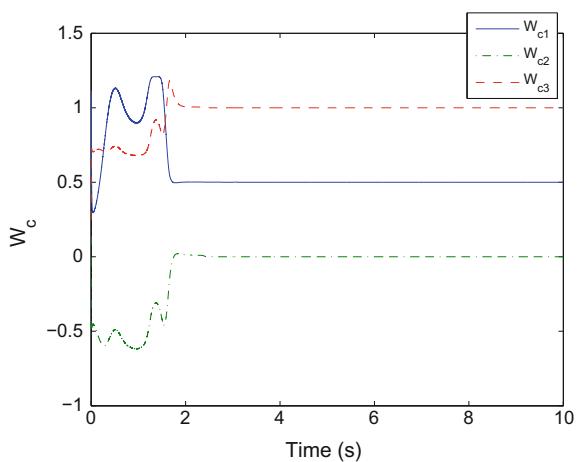
**Fig. 3.1** System states  $x(t)$  with persistently excited input for the first 3 s (reproduced with permission from [9], ©2013, Elsevier)



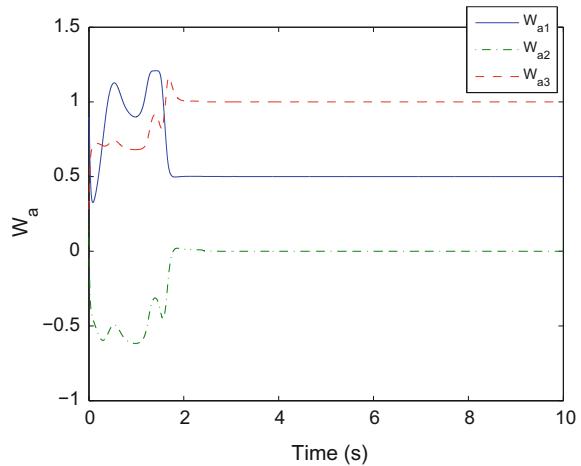
**Fig. 3.2** Error in estimating the state derivative  $\dot{x}(t)$  by the identifier (reproduced with permission from [9], ©2013, Elsevier)



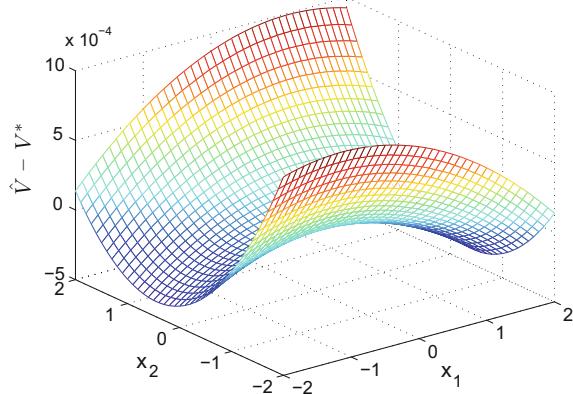
**Fig. 3.3** Convergence of critic weights  $\hat{W}_c(t)$  (reproduced with permission from [9], ©2013, Elsevier)



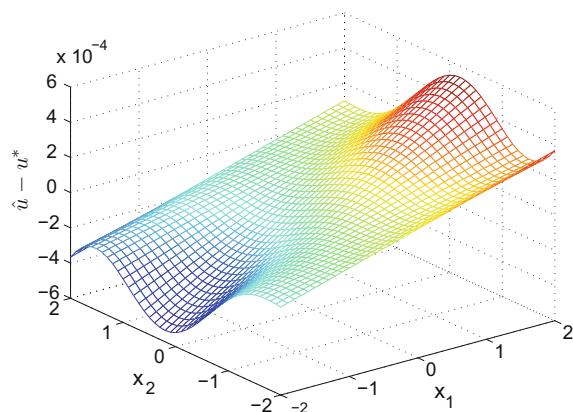
**Fig. 3.4** Convergence of actor weights  $\hat{W}_a(t)$  (reproduced with permission from [9], ©2013, Elsevier)



**Fig. 3.5** Error in approximating the optimal value function by the critic at steady state (reproduced with permission from [9], ©2013, Elsevier)



**Fig. 3.6** Error in approximating the optimal control by the actor at steady state (reproduced with permission from [9], ©2013, Elsevier)



### 3.3 Extension to Trajectory Tracking<sup>2</sup>

Approximate dynamic programming has been investigated and used as a method to approximately solve optimal regulation problems. However, the extension of this technique to optimal tracking problems for continuous-time nonlinear systems requires additional attention. The control development in this section uses a system transformation to convert the time-varying tracking problem into a time-invariant optimal control problem.

#### 3.3.1 Formulation of a Time-Invariant Optimal Control Problem

The control objective in this section is to track a bounded continuously differentiable signal  $x_d : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$ . To quantify this objective, a tracking error is defined as  $e(t) \triangleq x(t) - x_d(t)$ . The open-loop tracking error dynamics can then be expressed as

$$\dot{e}(t) = f(x(t)) + g(x(t))u(t) - \dot{x}_d(t). \quad (3.44)$$

The following assumptions are made to facilitate the formulation of an approximate optimal tracking controller.

**Assumption 3.10** The function  $g$  is bounded, the matrix  $g(x)$  has full column rank for all  $x \in \mathbb{R}^n$ , and the function  $g^+ : \mathbb{R}^n \rightarrow \mathbb{R}^{m \times n}$  defined as  $g^+ \triangleq (g^T g)^{-1} g^T$  is bounded and locally Lipschitz.

**Assumption 3.11** The desired trajectory is bounded such that  $\|x_d(t)\| \leq d, \forall t \in \mathbb{R}_{\geq t_0}$ , and there exists a locally Lipschitz function  $h_d : \mathbb{R}^n \rightarrow \mathbb{R}^n$  such that  $\dot{x}_d(t) = h_d(x_d(t))$  and  $g(x_d(t))g^+(x_d(t))(h_d(x_d(t)) - f(x_d(t))) = h_d(x_d(t)) - f(x_d(t)), \forall t \in \mathbb{R}_{\geq t_0}$ .

*Remark 3.12* Assumptions 3.10 and 3.11 can be eliminated if a discounted cost optimal tracking problem is considered instead of the total cost problem considered in this chapter. The discounted cost tracking problem considers a value function of the form  $V^*(\zeta) \triangleq \min_{u(\tau) \in U | \tau \in \mathbb{R}_{\geq t}} \int_t^\infty e^{\kappa(t-\tau)} r(\phi(\tau, t, \zeta, u(\cdot)), u(\tau)) d\tau$ , where  $\kappa \in \mathbb{R}_{>0}$  is a constant discount factor, and the control effort  $u$  is minimized instead of the control error  $\mu$ , introduced in (3.48). The control effort required for a system to perfectly track a desired trajectory is generally nonzero even if the initial system state is on the desired trajectory. Hence, in general, the optimal value function for a discounted cost problem does not satisfy  $V^*(0) = 0$ . In fact, the origin may not even

---

<sup>2</sup>Parts of the text in this section are reproduced, with permission, from [22], ©2015, Elsevier.

be a local minimum of the optimal value function. Online continuous-time reinforcement learning techniques are generally analyzed using the optimal value function as a candidate Lyapunov function. Since the origin is not necessarily a local minimum of the optimal value function for a discounted cost problem, it can not be used as a candidate Lyapunov function. Hence, to make the stability analysis tractable, a total-cost optimal control problem is considered in this section.

The steady-state control policy  $u_d : \mathbb{R}^n \rightarrow \mathbb{R}^m$  corresponding to the desired trajectory  $x_d$  is

$$u_d(x_d) = g^+(x_d)(h_d(x_d) - f(x_d)). \quad (3.45)$$

For notational brevity, let  $g_d^+(t) \triangleq g^+(x_d(t))$  and  $f_d(t) \triangleq f(x_d(t))$ . To transform the time-varying optimal control problem into a time-invariant optimal control problem, a new concatenated state  $\zeta : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{2n}$  is defined as [23]

$$\zeta \triangleq [e^T, x_d^T]^T. \quad (3.46)$$

Based on (3.44) and Assumption 3.11, the time derivative of (3.46) can be expressed as

$$\dot{\zeta}(t) = F(\zeta(t)) + G(\zeta(t))\mu(t), \quad (3.47)$$

where the functions  $F : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n}$ ,  $G : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n \times m}$ , and the control  $\mu : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^m$  are defined as

$$\begin{aligned} F(\zeta) &\triangleq \begin{bmatrix} f(e + x_d) - h_d(x_d) + g(e + x_d)u_d(x_d) \\ h_d(x_d) \end{bmatrix}, \\ G(\zeta) &\triangleq \begin{bmatrix} g(e + x_d) \\ 0 \end{bmatrix}, \quad \mu(t) \triangleq u(t) - u_d(x_d(t)). \end{aligned} \quad (3.48)$$

Local Lipschitz continuity of  $f$  and  $g$ , the fact that  $f(0) = 0$ , and Assumption 3.11 imply that  $F(0) = 0$  and  $F$  is locally Lipschitz.

The objective of the optimal control problem is to design a controller  $\mu(\cdot)$  that minimizes the cost functional

$$J(\zeta, \mu(\cdot)) \triangleq \int_{t_0}^{\infty} r_t(\zeta(\tau; t, \zeta, \mu(\cdot)), \mu(\tau)) d\tau, \quad (3.49)$$

subject to the dynamic constraints in (3.47) and  $r_t : \mathbb{R}^{2n} \times \mathbb{R}^m \rightarrow \mathbb{R}_{\geq 0}$  is the local cost defined as

$$r_t(\zeta, \mu) \triangleq Q_t(\zeta) + \mu^T R \mu. \quad (3.50)$$

In (3.50),  $R \in \mathbb{R}^{m \times m}$  is a positive definite symmetric matrix of constants. For ease of exposition, let the function  $Q_t : \mathbb{R}^{2n} \rightarrow \mathbb{R}_{\geq 0}$  in (3.50) be defined as  $Q_t(\zeta) \triangleq \zeta^T \bar{Q} \zeta$ , where  $\bar{Q} \in \mathbb{R}^{2n \times 2n}$  is a constant matrix defined as

$$\bar{Q} \triangleq \begin{bmatrix} Q & \mathbf{0}_{n \times n} \\ \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} \end{bmatrix}. \quad (3.51)$$

In (3.51),  $Q \in \mathbb{R}^{n \times n}$  is a constant positive definite symmetric matrix with a minimum eigenvalue  $\underline{q} \in \mathbb{R}_{>0}$ . Similar to Sect. 4.4, the developed technique can be easily generalized to local cost functions where  $Q_t(\zeta) \triangleq Q(e)$  for any continuous positive definite function  $Q : \mathbb{R}^n \rightarrow \mathbb{R}$ .

### 3.3.2 Approximate Optimal Solution

Assuming that a minimizing policy exists and that the optimal value function  $V^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}_{\geq 0}$  defined as

$$V^*(\zeta) \triangleq \min_{\mu(\tau)|\tau \in \mathbb{R}_{\geq t}} \int_t^\infty r_t(\zeta(\tau; t, \zeta, \mu(\cdot)), \mu(\tau)) d\tau \quad (3.52)$$

is continuously differentiable, the Hamilton–Jacobi–Bellman equation for the optimal control problem can be written as

$$H^* = \nabla_\zeta V^*(\zeta) (F(\zeta) + G(\zeta) \mu^*(\zeta)) + r_t(\zeta, \mu^*(\zeta)) = 0, \quad (3.53)$$

for all  $\zeta$ , with the boundary condition  $V^*(0) = 0$ , where  $H^*$  denotes the Hamiltonian, and  $\mu^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}^m$  denotes the optimal policy. For the local cost in (3.50) and the dynamics in (3.47), the optimal controller is given by  $\mu(t) = \mu^*(\zeta(t))$ , where  $\mu^*$  is the optimal policy given by

$$\mu^*(\zeta) = -\frac{1}{2} R^{-1} G^T(\zeta) (\nabla_\zeta V^*(\zeta))^T. \quad (3.54)$$

Using Property 2.3, the value function,  $V^*$ , can be represented using a neural network with  $L$  neurons as

$$V^*(\zeta) = W^T \sigma(\zeta) + \epsilon(\zeta), \quad (3.55)$$

where  $W \in \mathbb{R}^L$  is the constant ideal weight matrix bounded above by a known positive constant  $\bar{W} \in \mathbb{R}$  in the sense that  $\|W\| \leq \bar{W}$ ,  $\sigma : \mathbb{R}^{2n} \rightarrow \mathbb{R}^L$  is a bounded continuously differentiable nonlinear activation function, and  $\epsilon : \mathbb{R}^{2n} \rightarrow \mathbb{R}$  is the function reconstruction error [24, 25].

Using (3.54) and (3.55) the optimal policy can be expressed as

$$\mu^*(\zeta) = -\frac{1}{2} R^{-1} G^T(\zeta) (\nabla_\zeta \sigma^T(\zeta) W + \nabla_\zeta \epsilon^T(\zeta)). \quad (3.56)$$

Based on (3.55) and (3.56), the neural network approximations to the optimal value function and the optimal policy are given by

$$\begin{aligned}\hat{V}(\zeta, \hat{W}_c) &= \hat{W}_c^T \sigma(\zeta), \\ \hat{\mu}(\zeta, \hat{W}_a) &= -\frac{1}{2} R^{-1} G^T(\zeta) \nabla_{\zeta} \sigma^T(\zeta) \hat{W}_a,\end{aligned}\quad (3.57)$$

where  $\hat{W}_c \in \mathbb{R}^L$  and  $\hat{W}_a \in \mathbb{R}^L$  are estimates of the ideal neural network weights  $W$ .

The controller for the concatenated system is then designed as  $\mu(t) = \hat{\mu}(\zeta(t), \hat{W}_a(t))$ . The controller for the original system is obtained from (3.45), (3.48), and (3.57) as

$$u(t) = -\frac{1}{2} R^{-1} G^T(\zeta(t)) \nabla_{\zeta} \sigma^T(\zeta(t)) \hat{W}_a(t) + g_d^+(t) (h_d(x_d(t)) - f_d(t)). \quad (3.58)$$

Using the approximations  $\hat{\mu}$  and  $\hat{V}$  for  $\mu^*$  and  $V^*$  in (3.53), respectively, the error between the approximate and the optimal Hamiltonian (i.e., the Bellman error,  $\delta : \mathbb{R}^n \times \mathbb{R}^L \rightarrow \mathbb{R}$ ), is given in a measurable form by

$$\delta(\zeta, \hat{W}_c, \hat{W}_a) \triangleq \nabla \hat{V}(\zeta, \hat{W}_c) (F(\zeta) + G(\zeta) \hat{\mu}(\zeta, \hat{W}_a)) + r_t(\zeta, \hat{\mu}(\zeta, \hat{W}_a)). \quad (3.59)$$

The critic weights are updated to minimize  $\int_0^t \delta_t^2(\rho) d\rho$  using a normalized least-squares update law with an exponential forgetting factor as [26]

$$\dot{\hat{W}}_c(t) = -k_c \Gamma(t) \frac{\omega(t)}{1 + v \omega^T(t) \Gamma \omega(t)} \delta_t(t), \quad (3.60)$$

$$\dot{\Gamma}(t) = -k_c \left( -\lambda \Gamma(t) + \Gamma(t) \frac{\omega(t) \omega^T(t)}{1 + v \omega^T(t) \Gamma(t) \omega(t)} \Gamma(t) \right), \quad (3.61)$$

where  $\delta_t$  is the evaluation of the Bellman error along the system trajectories (i.e.,  $\delta_t(t) = \delta(\zeta(t), \hat{W}_a(t))$ ),  $v, k_c \in \mathbb{R}$  are constant positive adaptation gains,  $\omega : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^L$  is defined as  $\omega(t) \triangleq \nabla_{\zeta} \sigma(\zeta(t)) (F(\zeta(t)) + G(\zeta(t)) \hat{\mu}(\zeta(t), \hat{W}_a(t)))$ , and  $\lambda \in (0, 1)$  is the constant forgetting factor for the estimation gain matrix  $\Gamma \in \mathbb{R}^{L \times L}$ . The least-squares approach is motivated by faster convergence. With minor modifications to the stability analysis, the result can also be established for a gradient descent update law. The actor weights are updated to follow the critic weights as

$$\dot{\hat{W}}_a(t) = -k_{a1} (\hat{W}_a(t) - \hat{W}_c(t)) - k_{a2} \hat{W}_a(t), \quad (3.62)$$

where  $k_{a1}, k_{a2} \in \mathbb{R}$  are constant positive adaptation gains. The least-squares approach can not be used to update the actor weights because the Bellman error is a nonlinear function of the actor weights.

The following assumption facilitates the stability analysis using persistence of excitation.

**Assumption 3.13** The regressor  $\psi : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^L$ , defined as  $\psi(t) \triangleq \frac{\omega(t)}{\sqrt{1 + v\omega^T(t)\Gamma(t)\omega(t)}}$ , is persistently exciting (i.e., there exist  $T$ ,  $\underline{\psi} > 0$  such that  $\underline{\psi}I_L \leq \int_t^{t+T} \psi(\tau)\psi(\tau)^T d\tau$ ).

Using Assumption 3.13 and [26, Corollary 4.3.2],

$$\underline{\varphi}I_L \leq \Gamma(t) \leq \bar{\varphi}I_L, \quad \forall t \in \mathbb{R}_{\geq 0} \quad (3.63)$$

where  $\bar{\varphi}, \underline{\varphi} \in \mathbb{R}$  are constants such that  $0 < \underline{\varphi} < \bar{\varphi}$ . Since the evolution of  $\psi$  is dependent on the initial values of  $\zeta$  and  $\hat{W}_a$ , the constants  $\bar{\varphi}$  and  $\underline{\varphi}$  depend on the initial values of  $\zeta$  and  $\hat{W}_a$ . Based on (3.63), the regressor can be bounded as

$$\|\psi(t)\| \leq \frac{1}{\sqrt{v\underline{\varphi}}}, \quad \forall t \in \mathbb{R}_{\geq 0}. \quad (3.64)$$

### 3.3.3 Stability Analysis

Using (3.53), (3.59), and (3.60), an unmeasurable form of the Bellman error can be written as

$$\delta_t = -\tilde{W}_c^T \omega + \frac{1}{4} \tilde{W}_a^T \mathcal{G}_\sigma \tilde{W}_a + \frac{1}{4} \nabla_\zeta \epsilon \mathcal{G} \nabla_\zeta \epsilon^T + \frac{1}{2} W^T \nabla_\zeta \sigma \mathcal{G} \nabla_\zeta \epsilon^T - \nabla_\zeta \epsilon F, \quad (3.65)$$

where  $\mathcal{G} \triangleq GR^{-1}G^T$  and  $\mathcal{G}_\sigma \triangleq \nabla_\zeta \sigma GR^{-1}G^T \nabla_\zeta \sigma^T$ . The weight estimation errors for the value function and the policy are defined as  $\tilde{W}_c \triangleq W - \hat{W}_c$  and  $\tilde{W}_a \triangleq W - \hat{W}_a$ , respectively. Using (3.65), the weight estimation error dynamics for the value function are

$$\begin{aligned} \dot{\tilde{W}}_c &= k_c \Gamma \frac{\omega}{1 + v\omega^T \Gamma \omega} \left( \frac{\tilde{W}_a^T \mathcal{G}_\sigma \tilde{W}_a}{4} + \frac{\nabla_\zeta \epsilon \mathcal{G} \nabla_\zeta \epsilon^T}{4} + \frac{W^T \nabla_\zeta \sigma \mathcal{G} \nabla_\zeta \epsilon^T}{2} - \nabla_\zeta \epsilon F \right) \\ &\quad - k_c \Gamma \psi \psi^T \tilde{W}_c, \end{aligned} \quad (3.66)$$

where  $\psi \triangleq \frac{\omega}{\sqrt{1 + v\omega^T \Gamma \omega}} \in \mathbb{R}^L$  is the regressor vector.

Before stating the main result of the section, three supplementary technical lemmas are stated. To facilitate the discussion, let  $\mathcal{Y} \in \mathbb{R}^{2n+2L}$  be a compact set, and let  $\mathcal{Z}$  denote the projection of  $\mathcal{Y}$  onto  $\mathbb{R}^{n+2L}$ . Using the universal approximation property of neural networks, on the compact set defined by the projection of  $\mathcal{Y}$  onto  $\mathbb{R}^{2n}$ ,

the neural network approximation errors can be bounded such that  $\sup |\epsilon(\zeta)| \leq \bar{\epsilon}$  and  $\sup \|\nabla_\zeta \epsilon(\zeta)\| \leq \bar{\epsilon}$ , where  $\bar{\epsilon} \in \mathbb{R}$  is a positive constants, and there exists a positive constant  $L_F \in \mathbb{R}$  such that  $\sup \|F(\zeta)\| \leq L_F \|\zeta\|$ . Instead of using the fact that locally Lipschitz functions on compact sets are Lipschitz, it is possible to bound the function  $F$  as  $\|F(\zeta)\| \leq \rho(\|\zeta\|) \|\zeta\|$ , where  $\rho : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  is non-decreasing. This approach is feasible and results in additional gain conditions. To aid the subsequent stability analysis, Assumptions 3.10 and 3.11 are used to develop the bounds

$$\begin{aligned} & \left\| \left( \frac{\nabla_\zeta \epsilon}{4} + \frac{W^T \nabla_\zeta \sigma}{2} \right) \mathcal{G} \nabla_\zeta \epsilon^T \right\| + \bar{\epsilon} L_F \|x_d\| \leq \iota_1, \quad \|\mathcal{G}_\sigma\| \leq \iota_2, \\ & \|\nabla_\zeta \epsilon \mathcal{G} \nabla_\zeta \epsilon^T\| \leq \iota_3, \quad \left\| \frac{1}{2} W^T \mathcal{G}_\sigma + \frac{1}{2} \nabla_\zeta \epsilon \mathcal{G} \nabla_\zeta \sigma^T \right\| \leq \iota_4, \\ & \left\| \frac{1}{4} \nabla_\zeta \epsilon \mathcal{G} \nabla_\zeta \epsilon^T + \frac{1}{2} W^T \nabla_\zeta \sigma \mathcal{G} \nabla_\zeta \epsilon^T \right\| \leq \iota_5, \end{aligned} \quad (3.67)$$

on the compact set defined by the projection of  $\mathcal{Y}$  onto  $\mathbb{R}^{2n}$ , where  $\iota_1, \iota_2, \iota_3, \iota_4, \iota_5 \in \mathbb{R}$  are positive constants.

### Supporting Lemmas

The contribution in the previous section was the development of a transformation that enables the optimal policy and the optimal value function to be expressed as a time-invariant function of  $\zeta$ . The use of this transformation presents a challenge in the sense that the optimal value function, which is used as the Lyapunov function for the stability analysis, is not a positive definite function of  $\zeta$ , because the matrix  $\bar{Q}$  is positive semi-definite. In this section, this technical obstacle is addressed by exploiting the fact that the time-invariant optimal value function  $V^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}$  can be interpreted as a time-varying map  $V_t^* : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ , such that

$$V_t^*(e, t) = V^*\left(\begin{bmatrix} e \\ x_d(t) \end{bmatrix}\right) \quad (3.68)$$

for all  $e \in \mathbb{R}^n$  and for all  $t \in \mathbb{R}_{\geq 0}$ . Specifically, the time-invariant form facilitates the development of the approximate optimal policy, whereas the equivalent time-varying form can be shown to be a positive definite and decrescent function of the tracking error. In the following, Lemma 3.14 is used to prove that  $V_t^* : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$  is positive definite and decrescent, and hence, a candidate Lyapunov function.

**Lemma 3.14** *Let  $\overline{B}_a$  denote a closed ball around the origin with the radius  $a \in \mathbb{R}_{>0}$ . The optimal value function  $V_t^* : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$  satisfies the following properties*

$$V_t^*(e, t) \geq \underline{v}(\|e\|), \quad (3.69a)$$

$$V_t^*(0, t) = 0, \quad (3.69b)$$

$$V_t^*(e, t) \leq \bar{v}(\|e\|), \quad (3.69c)$$

$\forall t \in \mathbb{R}_{\geq 0}$  and  $\forall e \in \overline{B}_a$  where  $\underline{v} : [0, a] \rightarrow \mathbb{R}_{\geq 0}$  and  $\bar{v} : [0, a] \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions.

*Proof* See Appendix A.1.4. □

Lemmas 3.15 and 3.16 facilitate the stability analysis by establishing bounds on the error signal.

**Lemma 3.15** Let  $Z \triangleq [e^T \tilde{W}_c^T \tilde{W}_a^T]^T$ , and suppose that  $Z(\tau) \in \mathcal{Z}, \forall \tau \in [t, t+T]$ . The neural network weights and the tracking errors satisfy

$$-\inf_{\tau \in [t, t+T]} \|e(\tau)\|^2 \leq -\varpi_0 \sup_{\tau \in [t, t+T]} \|e(\tau)\|^2 + \varpi_1 T^2 \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2 + \varpi_2, \quad (3.70)$$

$$\begin{aligned} -\inf_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2 &\leq -\varpi_3 \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2 + \varpi_4 \inf_{\tau \in [t, t+T]} \|\tilde{W}_c(\tau)\|^2 L \\ &\quad + \varpi_5 \sup_{\tau \in [t, t+T]} \|e(\tau)\|^2 + \varpi_6, \end{aligned} \quad (3.71)$$

where the constants  $\varpi_0 - \varpi_6$  are defined in Appendix A.1.5.

*Proof* See Appendix A.1.5.  $\square$

**Lemma 3.16** Let  $Z \triangleq [e^T \tilde{W}_c^T \tilde{W}_a^T]^T$ , and suppose that  $Z(\tau) \in \mathcal{Z}, \forall \tau \in [t, t+T]$ . The critic weights satisfy

$$-\int_t^{t+T} \|\tilde{W}_c^T \psi\|^2 d\tau \leq -\underline{\psi} \varpi_7 \|\tilde{W}_c\|^2 + \varpi_8 \int_t^{t+T} \|e\|^2 d\tau + 3\iota_2^2 \int_t^{t+T} \|\tilde{W}_a(\sigma)\|^4 d\sigma + \varpi_9 T,$$

where the constants  $\varpi_7 - \varpi_9$  are defined in Appendix A.1.6.

*Proof* See Appendix A.1.6.  $\square$

### Gain Conditions and Gain Selection

The following section details sufficient gain conditions derived based on a stability analysis performed using the candidate Lyapunov function  $V_L : \mathbb{R}^{n+2L} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$  defined as  $V_L(Z, t) \triangleq V_t^*(e, t) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a$ . Using Lemma 3.14 and (3.63),

$$\underline{v}_l(\|Z\|) \leq V_L(Z, t) \leq \overline{v}_l(\|Z\|), \quad (3.72)$$

$\forall Z \in \overline{\mathbf{B}_b}$ ,  $\forall t \in \mathbb{R}_{\geq 0}$ , where  $\underline{v}_l : [0, b] \rightarrow \mathbb{R}_{\geq 0}$  and  $\overline{v}_l : [0, b] \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions.

To facilitate the discussion, define  $k_{a12} \triangleq k_{a1} + k_{a2}$ ,  $Z \triangleq [e^T \tilde{W}_c^T \tilde{W}_a^T]^T$ ,  $\iota \triangleq \frac{(k_{a2}\overline{W}+\iota_4)^2}{k_{a12}} + 2k_c(\iota_1)^2 + \frac{1}{4}\iota_3$ ,  $\varpi_{10} \triangleq \frac{\varpi_6 k_{a12} + 2\varpi_2 q + k_c \varpi_9}{8} + \iota$ , and  $\varpi_{11} \triangleq \frac{1}{16} \min(k_c \underline{\psi} \varpi_7, 2\varpi_0 q T, \varpi_3 k_{a12} T)$ . Let  $Z_0 \in \mathbb{R}_{\geq 0}$  denote a known constant bound on the initial condition such that  $\|Z(t_0)\| \leq Z_0$ , and let

$$\overline{Z} \triangleq \underline{v}_l^{-1} \left( \overline{v}_l \left( \max \left( Z_0, \sqrt{\frac{\varpi_{10} T}{\varpi_{11}}} \right) \right) + \iota T \right). \quad (3.73)$$

The sufficient gain conditions for the subsequent Theorem 3.17 are given by

$$\begin{aligned} k_{a12} &> \max \left( k_{a1} \xi_2 + \frac{k_c \iota_2}{4} \sqrt{\frac{\bar{Z}}{\nu \varphi}}, 3k_c \iota_2^2 \bar{Z} \right), \\ \xi_1 &> 2\bar{\epsilon} L_F, \quad k_c > \frac{k_{a1}}{\lambda \underline{\gamma} \xi_2}, \quad \underline{\psi} > \frac{2\varpi_4 k_{a12}}{k_c \varpi_7} T, \\ \underline{q} &> \max \left( \frac{\varpi_5 k_{a12}}{\varpi_0}, \frac{1}{2} k_c \varpi_8, k_c L_F \bar{\epsilon} \xi_1 \right), \\ T &< \min \left( \frac{1}{\sqrt{6L} k_{a12}}, \frac{\nu \varphi}{\sqrt{6L} k_c \bar{\varphi}}, \frac{1}{2\sqrt{n} L_F}, \sqrt{\frac{k_{a12}}{6L k_{a12} + 8\underline{q} \varpi_1}} \right), \end{aligned} \quad (3.74)$$

where  $\xi_1, \xi_2 \in \mathbb{R}$  are known adjustable positive constants. Similar conditions on  $\underline{\psi}$  and  $T$  can be found in persistence of excitation-based adaptive control in the presence of bounded or Lipschitz uncertainties (cf. [27, 28]). Furthermore, the compact set  $\mathcal{Z}$  satisfies the sufficient condition

$$\bar{Z} \leq r, \quad (3.75)$$

where  $r \triangleq \frac{1}{2} \sup_{z,y \in \mathcal{Z}} \|z - y\|$  denotes the radius of  $\mathcal{Z}$ . Since the Lipschitz constant and the bounds on neural network approximation error depend on the size of the compact set  $\mathcal{Z}$ , the constant  $\bar{Z}$  depends on  $r$ . Hence, feasibility of the sufficient condition in (3.75) is not apparent. Algorithm A.1 in the appendix details an iterative gain selection process to ensure satisfaction of the sufficient condition in (3.75). The main result of this section can now be stated as follows.

**Theorem 3.17** *Provided that the sufficient conditions in (3.74) and (3.75) are satisfied and Assumptions 3.10–3.13 hold, the controller in (3.58) and the update laws in (3.60)–(3.62) guarantee that the tracking error is ultimately bounded, and the error  $t \mapsto \|\hat{\mu}(\zeta(t), \hat{W}_a(t)) - \mu^*(\zeta(t))\|$  is ultimately bounded.*

*Proof* The time derivative of  $V_L$  is

$$\dot{V}_L = \nabla_\zeta V^* F + \nabla_\zeta V^* G \hat{\mu} + \tilde{W}_c^T \Gamma^{-1} \dot{\tilde{W}}_c - \tilde{W}_a^T \dot{\hat{W}}_a - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \dot{\Gamma} \Gamma^{-1} \tilde{W}_c.$$

Provided the sufficient conditions in (3.74) are satisfied, (3.60), (3.64)–(3.67), and the facts that  $\nabla_\zeta V^* F = -\nabla_\zeta V^* G \mu^* - r(\zeta, \mu^*)$  and  $\nabla_\zeta V^* G = -2\mu^{*T} R$  yield

$$\dot{V}_L \leq -\frac{q}{2} \|e\|^2 - \frac{1}{8} k_c \left\| \tilde{W}_c^T \psi \right\|^2 - \frac{k_{a12}}{4} \left\| \tilde{W}_a \right\|^2 + \iota. \quad (3.76)$$

The inequality in (3.76) is valid provided  $Z(t) \in \mathcal{Z}$ .

Integrating (3.76), using the facts that  $-\int_t^{t+T} \|e(\tau)\|^2 d\tau \leq -T \inf_{\tau \in [t, t+T]} \|e(\tau)\|^2$  and  $-\int_t^{t+T} \|\tilde{W}_a(\tau)\|^2 d\tau \leq -T \inf_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2$ , Lemmas 3.15 and 3.16, and the gain conditions in (3.74) yields

$$V_L(Z(t+T), t+T) - V_L(Z(t), t) \leq -\frac{k_c \psi \varpi \gamma}{16} \|\tilde{W}_c(t)\|^2 - \frac{\varpi_0 q T}{8} \|e(t)\|^2 + \varpi_{10} T - \frac{\varpi_3 k_{a12} T}{16} \|\tilde{W}_a(t)\|^2,$$

provided  $Z(\tau) \in \mathcal{Z}$ ,  $\forall \tau \in [t, t+T]$ . Thus,  $V_L(Z(t+T), t+T) - V_L(Z(t), t) < 0$  provided  $\|Z(t)\| > \sqrt{\frac{\varpi_{10} T}{\varpi_{11}}}$  and  $Z(\tau) \in \mathcal{Z}$ ,  $\forall \tau \in [t, t+T]$ . The bounds on the Lyapunov function in (3.72) yield  $V_L(Z(t+T), t+T) - V_L(Z(t), t) < 0$  provided  $V_L(Z(t), t) > \bar{v}_l \left( \sqrt{\frac{\varpi_{10} T}{\varpi_{11}}} \right)$  and  $Z(\tau) \in \mathcal{Z}$ ,  $\forall \tau \in [t, t+T]$ .

Since  $Z(t_0) \in \mathcal{Z}$ , (3.76) can be used to conclude that  $\dot{V}_L(Z(t_0), t_0) \leq \iota$ . The sufficient condition in (3.75) ensures that  $\underline{v}_l^{-1}(V_L(Z(t_0), t_0) + \iota T) \leq r$ ; hence,  $Z(t) \in \mathcal{Z}$ ,  $\forall t \in [t_0, t_0 + T]$ . If  $V_L(Z(t_0), t_0) > \bar{v}_l \left( \sqrt{\frac{\varpi_{10} T}{\varpi_{11}}} \right)$ , then  $Z(t) \in \mathcal{Z}$ ,  $\forall t \in [t_0, t_0 + T]$  implies  $V_L(Z(t_0 + T), t_0 + T) - V_L(Z(t_0), t_0) < 0$ ; hence,  $\underline{v}_l^{-1}(V_L(Z(t_0 + T), t_0 + T) + \iota T) \leq r$ . Thus,  $Z(t) \in \mathcal{Z}$ ,  $\forall t \in [t_0 + T, t_0 + 2T]$ . Using mathematical induction, it can be shown that the system state is bounded such that  $\sup_{t \in [0, \infty)} \|Z(t)\| \leq r$  and ultimately bounded such that

$$\limsup_{t \rightarrow \infty} \|Z(t)\| \leq \underline{v}_l^{-1} \left( \bar{v}_l \left( \sqrt{\frac{\varpi_{10} T}{\varpi_{11}}} \right) + \iota T \right).$$

If the regressor  $\psi$  satisfies a stronger u-persistence of excitation assumption (cf. [16, 17]), the tracking error and the weight estimation errors can be shown to be uniformly ultimately bounded.  $\square$

### 3.3.4 Simulation

Simulations are performed on a two-link manipulator to demonstrate the ability of the presented technique to approximately optimally track a desired trajectory. The two link robot manipulator is modeled using Euler–Lagrange dynamics as

$$M \ddot{q} + V_m \dot{q} + F_d \dot{q} + F_s = u, \quad (3.77)$$

where  $q = [q_1 \ q_2]^T$  and  $\dot{q} = [\dot{q}_1 \ \dot{q}_2]^T$  are the angular positions in radian and the angular velocities in radian/s respectively. In (3.77),  $M \in \mathbb{R}^{2 \times 2}$  denotes the inertia matrix, and  $V_m \in \mathbb{R}^{2 \times 2}$  denotes the centripetal–Coriolis matrix given by  $M \triangleq$

$\begin{bmatrix} p_1 + 2p_3c_2 & p_2 + p_3c_2 \\ p_2 + p_3c_2 & p_2 \end{bmatrix}$ ,  $V_m \triangleq \begin{bmatrix} -p_3s_2\dot{q}_2 & -p_3s_2(\dot{q}_1 + \dot{q}_2) \\ p_3s_2\dot{q}_1 & 0 \end{bmatrix}$ , where  $c_2 = \cos(q_2)$ ,  $s_2 = \sin(q_2)$ ,  $p_1 = 3.473 \text{ kg.m}^2$ ,  $p_2 = 0.196 \text{ kg.m}^2$ ,  $p_3 = 0.242 \text{ kg.m}^2$ , and  $F_d = \text{diag}[5.3, 1.1] \text{ Nm.s}$  and  $F_s(\dot{q}) = [8.45 \tanh(\dot{q}_1), 2.35 \tanh(\dot{q}_2)]^T \text{ Nm}$  are the models for the static and the dynamic friction, respectively.

The objective is to find a policy  $\mu$  that ensures that the state  $x \triangleq [q_1, q_2, \dot{q}_1, \dot{q}_2]^T$  tracks the desired trajectory  $x_d(t) = [0.5 \cos(2t), 0.33 \cos(3t), -\sin(2t), -\sin(3t)]^T$ , while minimizing the cost in (3.49), where  $Q = \text{diag}[10, 10, 2, 2]$ . Using (3.45)–(3.48) and the definitions

$$\begin{aligned} f &\triangleq \left[ x_3, x_4, \left( M^{-1}(-V_m - F_d) \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} - F_s \right)^T \right]^T, \\ g &\triangleq \left[ [0, 0]^T, [0, 0]^T, (M^{-1})^T \right]^T, \\ g_d^+ &\triangleq \left[ [0, 0]^T, [0, 0]^T, M(x_d) \right], \\ h_d &\triangleq [x_{d3}, x_{d4}, -4x_{d1}, -9x_{d2}]^T, \end{aligned} \quad (3.78)$$

the optimal tracking problem can be transformed into the time-invariant form in (3.48).

In this effort, the basis selected for the value function approximation is a polynomial basis with 23 elements given by

$$\begin{aligned} \sigma(\zeta) = \frac{1}{2} & \left[ \zeta_1^2 \zeta_2^2 \zeta_1 \zeta_3 \zeta_1 \zeta_4 \zeta_2 \zeta_3 \zeta_2 \zeta_4 \zeta_1^2 \zeta_2^2 \zeta_1^2 \zeta_5^2 \right. \\ & \left. \zeta_1^2 \zeta_6^2 \zeta_1^2 \zeta_7^2 \zeta_1^2 \zeta_8^2 \zeta_2^2 \zeta_5^2 \zeta_2^2 \zeta_6^2 \zeta_2^2 \zeta_7^2 \zeta_2^2 \zeta_8^2 \zeta_3^2 \zeta_5^2 \right. \\ & \left. \zeta_3^2 \zeta_6^2 \zeta_3^2 \zeta_7^2 \zeta_3^2 \zeta_8^2 \zeta_4^2 \zeta_5^2 \zeta_4^2 \zeta_6^2 \zeta_4^2 \zeta_7^2 \zeta_4^2 \zeta_8^2 \right]^T. \end{aligned} \quad (3.79)$$

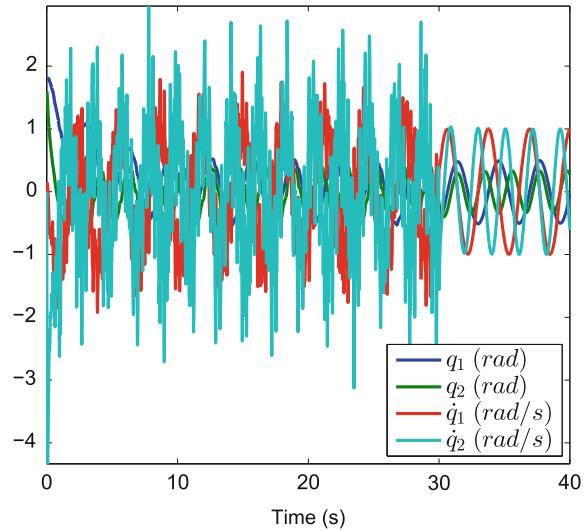
The control gains are selected as  $k_{a1} = 5$ ,  $k_{a2} = 0.001$ ,  $k_c = 1.25$ ,  $\lambda = 0.001$ , and  $\nu = 0.005$ . The initial conditions are  $x(0) = [1.8 \ 1.6 \ 0 \ 0]^T$ ,  $\hat{W}_c(0) = 10 \times \mathbf{1}_{23 \times 1}$ ,  $\hat{W}_a(0) = 6 \times \mathbf{1}_{23 \times 1}$ , and  $\Gamma(0) = 2000 \times I_{23}$ . To ensure persistence of excitation, a probing signal

$$p(t) = \begin{bmatrix} 2.55 \tanh(2t) \left( 20 \sin(\sqrt{232}\pi t) \cos(\sqrt{20}\pi t) \right. \\ \left. + 6 \sin(18e^2 t) + 20 \cos(40t)(21t) \right) \\ 0.01 \tanh(2t) \left( 20 \sin(\sqrt{132}\pi t) \cos(\sqrt{10}\pi t) \right. \\ \left. + 6 \cos(8et) + 20 \cos(10t) \cos(11t) \right) \end{bmatrix} \quad (3.80)$$

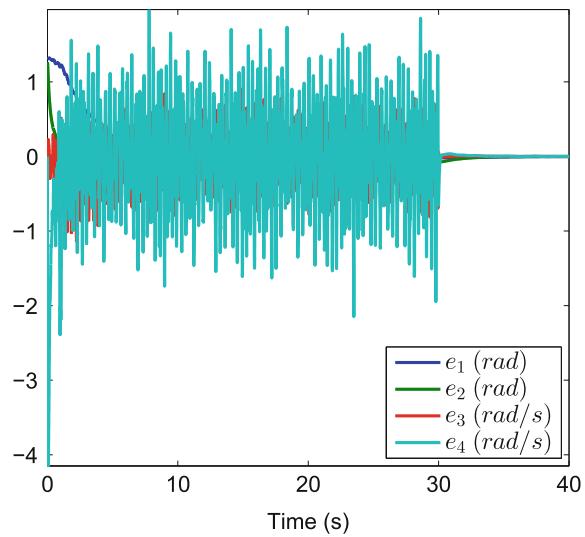
is added to the control signal for the first 30 s of the simulation [14].

It is clear from Figs. 3.7 and 3.8 that the system states are bounded during the learning phase and the algorithm converges to a stabilizing controller in the sense that the tracking errors go to zero when the probing signal is eliminated. Furthermore,

**Fig. 3.7** State trajectories with probing signal  
(reproduced with permission from [22], ©2015, Elsevier)



**Fig. 3.8** Error trajectories with probing signal  
(reproduced with permission from [22], ©2015, Elsevier)

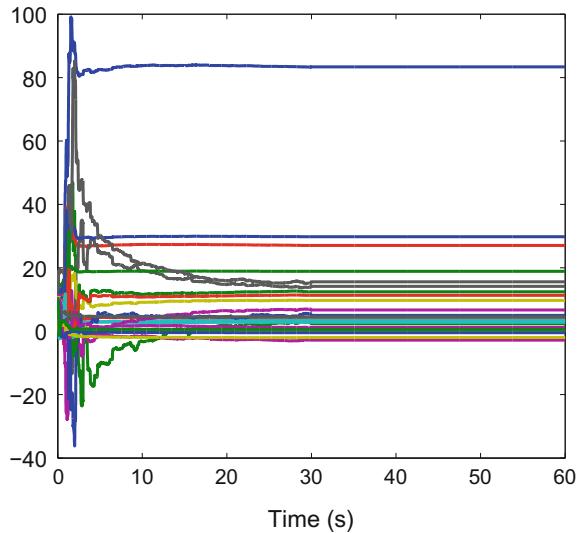


Figs. 3.9 and 3.10 shows that the weight estimates for the value function and the policy are bounded and they converge.

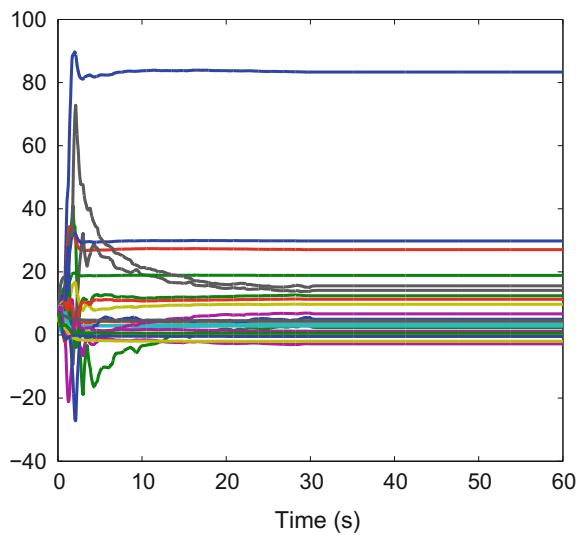
The neural network weights converge to the following values

$$\hat{W}_c = \hat{W}_a = [83.36 \ 2.37 \ 27.0 \ 2.78 \ -2.83 \ 0.20 \ 14.13 \\ 29.81 \ 18.87 \ 4.11 \ 3.47 \ 6.69 \ 9.71 \ 15.58 \ 4.97 \ 12.42 \\ 11.31 \ 3.29 \ 1.19 \ -1.99 \ 4.55 \ -0.47 \ 0.56]^T. \quad (3.81)$$

**Fig. 3.9** Evolution of critic weights (reproduced with permission from [22], ©2015, Elsevier)

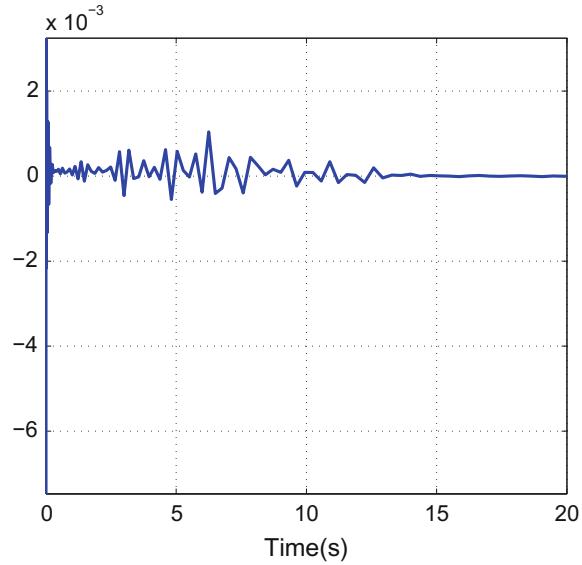


**Fig. 3.10** Evolution of actor weights (reproduced with permission from [22], ©2015, Elsevier)

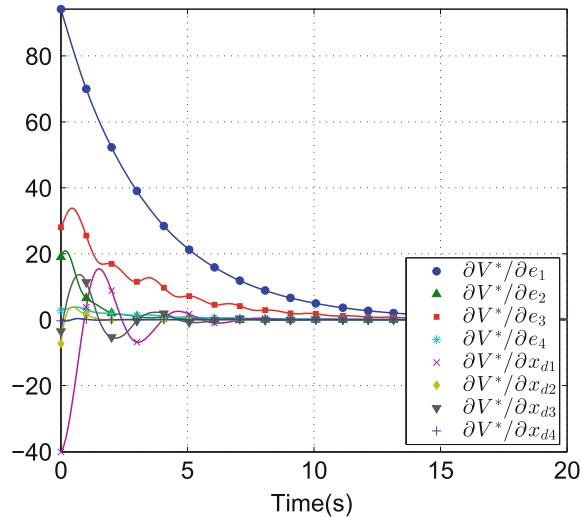


Note that the last sixteen weights that correspond to the terms containing the desired trajectories  $\zeta_5, \dots, \zeta_8$  are non-zero. Thus, the resulting value function  $V$  and the resulting policy  $\mu$  depend on the desired trajectory, and hence, are time-varying functions of the tracking error. Since the true weights are unknown, a direct comparison of the weights in (3.81) with the true weights is not possible. Instead, to gauge the performance of the presented technique, the state and the control trajectories obtained using the estimated policy are compared with those obtained using Radau-pseudospectral numerical optimal control computed using the GPOPS software [29].

**Fig. 3.11** Hamiltonian of the numerical solution computed using GPOPS (reproduced with permission from [22], ©2015, Elsevier)

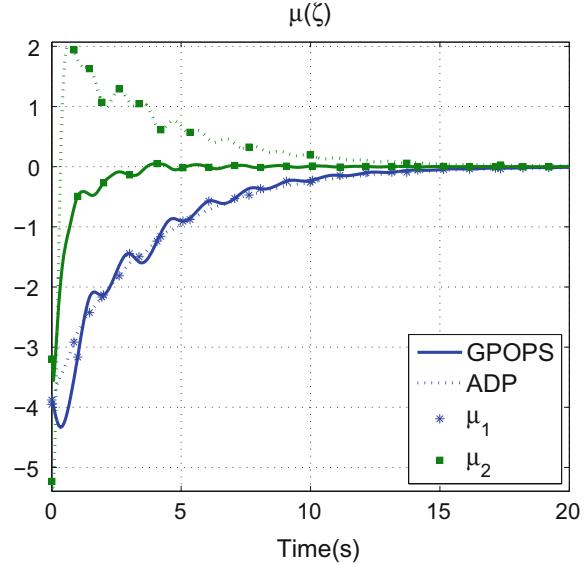


**Fig. 3.12** Costate of the numerical solution computed using GPOPS (reproduced with permission from [22], ©2015, Elsevier)

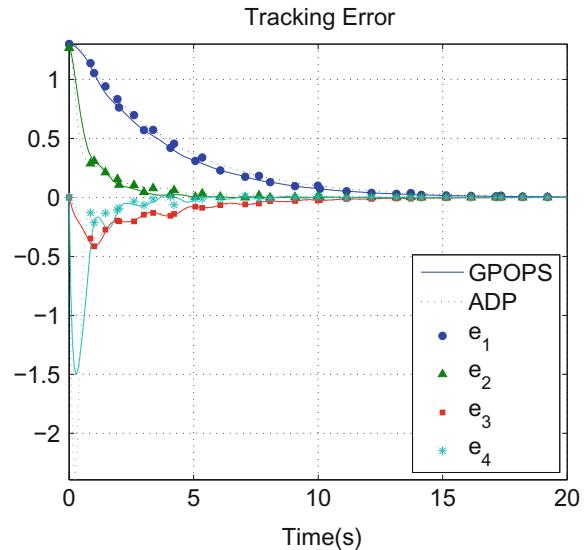


Since an accurate numerical solution is difficult to obtain for an infinite-horizon optimal control problem, the numerical optimal control problem is solved over a finite-horizon ranging over approximately five times the settling time associated with the slowest state variable. Based on the solution obtained using the proposed technique, the slowest settling time is estimated to be approximately twenty seconds. Thus, to approximate the infinite-horizon solution, the numerical solution is computed over a 100 second time horizon using 300 collocation points.

**Fig. 3.13** Control trajectories  $\mu(t)$  obtained from GPOPS and the developed technique (reproduced with permission from [22], ©2015, Elsevier)



**Fig. 3.14** Tracking error trajectories  $e(t)$  obtained from GPOPS and the developed technique (reproduced with permission from [22], ©2015, Elsevier)



As seen in Fig. 3.11, the Hamiltonian of the numerical solution is approximately zero. This supports the assertion that the optimal control problem is time-invariant. Furthermore, since the Hamiltonian is close to zero, the numerical solution obtained using GPOPS is sufficiently accurate as a benchmark to compare against the approximate dynamic programming-based solution obtained using the proposed technique. Note that in Fig. 3.12, the costate variables corresponding to the desired trajectories

are nonzero. Since these costate variables represent the sensitivity of the cost with respect to the desired trajectories, this further supports the assertion that the optimal value function depends on the desired trajectory, and hence, is a time-varying function of the tracking error.

Figures 3.13 and 3.14 show the control and the tracking error trajectories obtained from the developed technique (dashed lines) plotted alongside the numerical solution obtained using GPOPS (solid lines). The trajectories obtained using the developed technique are close to the numerical solution. The inaccuracies are a result of the facts that the set of basis functions in (3.79) is not exact, and the proposed method attempts to find the weights that generate the least total cost for the given set of basis functions. The accuracy of the approximation can be improved by choosing a more appropriate set of basis functions, or at an increased computational cost, by adding more basis functions to the existing set in (3.79). The total cost obtained using the numerical solution is found to be 75.42 and the total cost obtained using the developed method is found to be 84.31. Note that from Figs. 3.13 and 3.14, it is clear that both the tracking error and the control converge to zero after approximately 20 s, and hence, the total cost obtained from the numerical solution is a good approximation of the infinite-horizon cost.

### 3.4 N-Player Nonzero-Sum Differential Games<sup>3</sup>

In this section, an approximate online equilibrium solution is developed for an  $N$ -player nonzero-sum game subject to continuous-time nonlinear unknown dynamics and an infinite-horizon quadratic cost. A novel actor-critic-identifier structure is used, wherein a robust dynamic neural network is used to asymptotically identify the uncertain system with additive disturbances, and a set of critic and actor neural networks are used to approximate the value functions and equilibrium policies, respectively. The weight update laws for the actor neural networks are generated using a gradient-descent method, and the critic neural networks are generated by least-squares regression, which are both based on the modified Bellman error that is independent of the system dynamics. A Lyapunov-based stability analysis shows that uniformly ultimately bounded tracking is achieved and a convergence analysis demonstrates that the approximate control policies converge to a neighborhood of the optimal solutions. The actor, the critic, and the identifier structures are implemented in real-time, continuously, and simultaneously. Simulations on two and three player games illustrate the performance of the developed method.

---

<sup>3</sup>Parts of the text in this section are reproduced, with permission, from [30], ©2015, IEEE.

### 3.4.1 Problem Formulation

Consider a class of control-affine multi-input systems

$$\dot{x}(t) = f(x(t)) + \sum_{i=1}^N g_i(x(t)) u_i(t), \quad (3.82)$$

where  $x : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the state vector,  $u_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{m_i}$  are the control inputs, and  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  and  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m_i}$  are the drift and input matrices, respectively. Assume that  $g_1, \dots, g_N$ , and  $f$  are second order differentiable, and that  $f(0) = 0$  so that  $x = 0$  is an equilibrium point for the uncontrolled dynamics in (3.82). Let

$$U \triangleq \left\{ \{\phi_i : \mathbb{R}^n \rightarrow \mathbb{R}^{m_i}, i = 1, \dots, N\} \mid \{\phi_i, \dots, \phi_N\} \text{ is admissible for (3.82)} \right\}$$

be the set of all admissible tuples of feedback policies  $\phi_i : \mathbb{R}^n \rightarrow \mathbb{R}^{m_i}$  (cf. [6]). Let  $V_i^{\{\phi_1, \dots, \phi_N\}} : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  denote the value function of the  $i$ th player with respect to the feedback policies  $\{\phi_1, \dots, \phi_N\} \in U$ , defined as

$$V_i^{\{u_1, \dots, u_N\}}(x) = \int_t^\infty r_i(x(\tau; t, x), \phi_1(x(\tau; t, x)), \dots, \phi_N(x(\tau; t, x))) d\tau, \quad (3.83)$$

where  $x(\tau; t, x)$  for  $\tau \in [t, \infty)$  denotes the trajectory of (3.82) evaluated at time  $\tau$  obtained using the controllers  $u_i(\tau) = \phi_i(x(\tau; t, x))$ , starting from the initial time  $t$  and the initial condition  $x$ . In (3.83),  $r_i : \mathbb{R}^n \times \mathbb{R}^{m_1} \times \dots \times \mathbb{R}^{m_N} \rightarrow \mathbb{R}_{\geq 0}$  denotes the instantaneous cost defined as  $r_i(x, u_1, \dots, u_N) \triangleq Q_i(x) + \sum_{j=1}^N u_j^T R_{ij} u_j$ , where  $Q_i \in \mathbb{R}^{n \times n}$  and  $R_{ij} \in \mathbb{R}^{n \times n}$  are positive definite matrices. The control objective is to find an approximate feedback-Nash equilibrium solution to the infinite-horizon regulation differential game online. A feedback-Nash equilibrium solution is a tuple  $\{u_1^*, \dots, u_N^*\} \in U$  such that for all  $i \in \{1, \dots, N\}$ , for all  $x \in \mathbb{R}^n$ , the corresponding value functions satisfy

$$V_i^*(x) \triangleq V_i^{\{u_1^*, u_2^*, \dots, u_i^*, \dots, u_N^*\}}(x) \leq V_i^{\{u_1^*, u_2^*, \dots, \phi_i, \dots, u_N^*\}}(x)$$

for all  $\phi_i$  such that  $\{u_1^*, u_2^*, \dots, \phi_i, \dots, u_N^*\} \in U$ .

The exact closed-loop feedback-Nash equilibrium solution  $\{u_1^*, \dots, u_N^*\}$  can be expressed in terms of the value functions as [6, 7, 31, 32]

$$u_i^*(x) = -\frac{1}{2} R_{ii}^{-1} g_i^T(x) (\nabla_x V_i^*(x))^T, \quad (3.84)$$

where the value functions  $\{V_1^*, \dots, V_N^*\}$  satisfy the coupled Hamilton–Jacobi equations

$$\begin{aligned} 0 = & x^T Q_i x + \sum_{j=1}^N \frac{1}{4} \nabla_x V_j^*(x) G_{ij}(x) \left( \nabla_x V_j^*(x) \right)^T + \nabla_x V_i^*(x) f(x) \\ & - \frac{1}{2} \nabla_x V_i^*(x) \sum_{j=1}^N G_j(x) \left( \nabla_x V_j^*(x) \right)^T. \end{aligned} \quad (3.85)$$

In (3.85),  $G_j(x) \triangleq g_j(x) R_{jj}^{-1} g_j^T(x)$  and  $G_{ij}(x) \triangleq g_j(x) R_{jj}^{-1} R_{ij} R_{jj}^{-1} g_j^T(x)$ .

Computation of an analytical solution to the coupled nonlinear Hamilton–Jacobi equations in (3.85) is, in general, infeasible. Hence, an approximate solution is sought. Although nonzero-sum games contain non-cooperative components, the solution to each player’s coupled Hamilton–Jacobi equation in (3.85) requires knowledge of all the other player’s strategies in (3.84). The underlying assumption of rational opponents [33] is characteristic of differential game theory problems and it implies that the players share information, yet they agree to adhere to the equilibrium policy determined from the Nash game.

### 3.4.2 *Hamilton–Jacobi Approximation Via Actor-Critic-Identifier*

In this section, an actor-critic-identifier [9, 30] approximation architecture is used to solve the coupled nonlinear Hamilton–Jacobi equations in (3.85). The actor-critic-identifier architecture eliminates the need for exact model knowledge by using a dynamic neural network to robustly identify the system, a critic neural network to approximate the value function, and an actor neural network to find a control policy which minimizes the value functions. The following development focuses on the solution to a two player nonzero-sum game. The approach can easily be extended to the  $N$ –player game presented in Sect. 3.4.1. This section introduces the actor-critic-identifier architecture, and subsequent sections provide details of the design for the two player nonzero-sum game solution.

The optimal policies in (3.84) and the associated value functions  $V_i^*$  satisfy the Hamilton–Jacobi equations

$$r_i(x, u_1^*(x), \dots, u_N^*(x)) + \nabla_x V_i^*(x) F_u(x, u_1^*(x), \dots, u_N^*(x)) = 0, \quad (3.86)$$

where

$$F_u(x, u_1, \dots, u_N) \triangleq f(x) + \sum_{j=1}^N g_j(x) u_j \in \mathbb{R}^n. \quad (3.87)$$

Replacing the optimal Jacobian  $\nabla_x V_i^*$  and optimal control policies  $u_i^*$  by parametric estimates  $\nabla_x \hat{V}_i(x, \hat{W}_{ci})$  and  $\hat{u}_i(x, \hat{W}_{ai})$ , respectively, where  $\hat{W}_{ci}$  and  $\hat{W}_{ai}$  are the estimates of the unknown parameters, yields the Bellman error

$$\begin{aligned}\delta_i(x, \hat{W}_{ci}, \hat{W}_{a1}, \dots, \hat{W}_{aN}) &= r_i(x, \hat{u}_1(x, \hat{W}_{a1}), \dots, \hat{u}_N(x, \hat{W}_{aN})) \\ &\quad + \nabla_x \hat{V}_i(x, \hat{W}_{ci}) F_u(x, \hat{u}_1(x, \hat{W}_{a1}), \dots, \hat{u}_N(x, \hat{W}_{aN})).\end{aligned}\quad (3.88)$$

The approximate Hamiltonian in (3.88) is dependent on  $F_u$ , and hence, complete knowledge of the system. To overcome this limitation, an online system identifier replaces the system dynamics  $F_u$  with a parametric estimate  $\hat{F}_u$ , defined as  $\hat{F}_u(t) \triangleq \dot{\hat{x}}(t)$  where  $\hat{x}(\cdot)$  is an estimate of the state,  $x(\cdot)$ . Hence, the Bellman error in (3.88) is approximated at each time instance as

$$\hat{\delta}_i(x, \dot{\hat{x}}, \hat{W}_{ci}, \hat{W}_{a1}, \dots, \hat{W}_{aN}) = r_i(x, \hat{u}_1(x, \hat{W}_{a1}), \dots, \hat{u}_N(x, \hat{W}_{aN})) + \nabla_x \hat{V}_i(x, \hat{W}_{ci}) \dot{\hat{x}}.\quad (3.89)$$

The objective is to update the actors,  $\hat{u}_i$ , the critics,  $\hat{V}_i$ , and the identifier,  $\hat{F}_u$ , simultaneously, based on the minimization of the Bellman residual errors  $\hat{\delta}_i$ . All together, the actors, the critics, and the identifier constitute the actor-critic-identifier architecture. The update laws for the actors, the critics, and the identifiers are designed based on a Lyapunov-based analysis to ensure stability of the closed-loop system during the learning phase.

### 3.4.3 System Identifier

Consider the two-player case for the dynamics given in (3.82) as

$$\dot{x}(t) = f(x(t)) + g_1(x(t)) u_1(t) + g_2(x(t)) u_2(t), \quad x(t_0) = x_0, \quad (3.90)$$

where  $u_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{m_i}$  are the control inputs, and the state  $x : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is assumed to be available for feedback. The following assumptions about the system will be used in the subsequent development.

**Assumption 3.18** The input matrices  $g_1$  and  $g_2$  are known and bounded according to the inequalities  $\|g_1(x)\| \leq \bar{g}_1$  and  $\|g_2(x)\| \leq \bar{g}_2$ , for all  $x \in \mathbb{R}^n$ , where  $\bar{g}_1$  and  $\bar{g}_2$  are known positive constants.

**Assumption 3.19** The control inputs  $u_1(\cdot)$  and  $u_2(\cdot)$  are bounded (i.e.,  $u_1(\cdot), u_2(\cdot) \in \mathcal{L}_\infty$ ). This assumption facilitates the design of the state-derivative estimator, and is relaxed in Sect. 3.4.5.

Based on Property 2.3, the nonlinear system in (3.90) can be represented using a multi-layer neural network as

$$\begin{aligned}\dot{x}(t) &= W_f^T \sigma_f \left( V_f^T x(t) \right) + \epsilon_f(x(t)) + g_1(x(t)) u_1(t) + g_2(x(t)) u_2(t), \\ &\triangleq F_u(x(t), u_1(t), u_2(t)),\end{aligned}\quad (3.91)$$

where  $W_f \in \mathbb{R}^{L_f+1 \times n}$  and  $V_f \in \mathbb{R}^{n \times L_f}$  are unknown ideal neural network weight matrices with  $L_f \in \mathbb{N}$  representing the neurons in the output layers. In (3.91),  $\sigma_f : \mathbb{R}^{L_f} \rightarrow \mathbb{R}^{L_f+1}$  is the vector of basis functions, and  $\epsilon_f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is the function reconstruction error in approximating the function  $f$ . The proposed dynamic neural network used to identify the system in (3.90) is

$$\begin{aligned}\dot{\hat{x}}(t) &= \hat{W}_f^T(t) \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) + g_1(x(t)) u_1(t) + g_2(x(t)) u_2(t) + \mu(t), \\ &\triangleq \hat{F}_u(x(t), \hat{x}(t), u_1(t), u_2(t)),\end{aligned}\quad (3.92)$$

where  $\hat{x} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the state of the dynamic neural network,  $\hat{W}_f : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_f+1 \times n}$ ,  $\hat{V}_f : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{n \times L_f}$  are the estimates of the ideal weights of the neural networks, and  $\mu : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  denotes the RISE feedback term (cf. [34]) defined as

$$\mu(t) \triangleq k(\tilde{x}(t) - \tilde{x}(t_0)) + v(t), \quad (3.93)$$

where the measurable identification error  $\tilde{x} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is defined as

$$\tilde{x}(t) \triangleq x(t) - \hat{x}(t), \quad (3.94)$$

and  $v : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is a Filippov solution to the initial value problem

$$\dot{v}(t) = (k\alpha + \gamma)\tilde{x}(t) + \beta_1 \text{sgn}(\tilde{x}(t)), \quad v(t_0) = 0,$$

where  $k, \alpha, \gamma, \beta \in \mathbb{R}$  are positive constant gains, and  $\text{sgn}(\cdot)$  denotes a vector signum function.

The identification error dynamics are developed by taking the time derivative of (3.94) and substituting for (3.91) and (3.92) as

$$\dot{\tilde{x}} = W_f^T \sigma_f \left( V_f^T x(t) \right) - \hat{W}_f^T \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) + \epsilon_f(x(t)) - \mu(t). \quad (3.95)$$

To facilitate the subsequent analysis an auxiliary identification error is defined as

$$e_f(t) \triangleq \dot{\tilde{x}}(t) + \alpha \tilde{x}(t). \quad (3.96)$$

Taking the time derivative of (3.96) and using (3.95) yields

$$\begin{aligned}\dot{\epsilon}_f(t) &= W_f^T \nabla_{V_f^T x} \sigma_f \left( V_f^T x(t) \right) V_f^T \dot{x}(t) - \hat{W}_f^T(t) \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \\ &\quad - \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \dot{\hat{V}}_f^T(t) \hat{x}(t) - \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\hat{x}}(t) \\ &\quad + \dot{\epsilon}_f(x(t), \dot{x}(t)) - k\epsilon_f(t) - \gamma \tilde{x}(t) - \beta_1 \operatorname{sgn}(\tilde{x}(t)) + \alpha \dot{\tilde{x}}(t).\end{aligned}\quad (3.97)$$

The weight update laws for the dynamic neural network in (3.92) are developed based on the subsequent stability analysis as

$$\begin{aligned}\dot{\hat{W}}_f(t) &= \operatorname{proj} \left( \Gamma_{wf} \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\hat{x}}(t) \tilde{x}^T(t) \right), \\ \dot{\hat{V}}_f(t) &= \operatorname{proj} \left( \Gamma_{vf} \dot{\hat{x}}(t) \tilde{x}^T(t) \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \right),\end{aligned}\quad (3.98)$$

where  $\operatorname{proj}(\cdot)$  is a smooth projection operator [35, 36], and  $\Gamma_{wf} \in \mathbb{R}^{L_f+1 \times L_f+1}$ ,  $\Gamma_{vf} \in \mathbb{R}^{n \times n}$  are positive constant adaptation gain matrices. Adding and subtracting  $\frac{1}{2} W_f^T \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\hat{x}}(t) + \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) V_f^T \dot{\hat{x}}(t)$ , and grouping similar terms, the expression in (3.97) can be rewritten as

$$\dot{\epsilon}_f(t) = \tilde{N}(t) + N_{B1}(t) + \hat{N}_{B2}(t) - k\epsilon_f(t) - \gamma \tilde{x}(t) - \beta_1 \operatorname{sgn}(\tilde{x}(t)), \quad (3.99)$$

where the auxiliary signals,  $\tilde{N}$ ,  $N_{B1}$ , and  $\hat{N}_{B2} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  in (3.99) are defined as

$$\begin{aligned}\tilde{N}(t) &\triangleq \alpha \dot{\tilde{x}}(t) - \hat{W}_f^T(t) \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) - \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \dot{\hat{V}}_f^T(t) \hat{x}(t) \\ &\quad + \frac{1}{2} W_f^T \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\tilde{x}}(t) \\ &\quad + \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) V_f^T \dot{\tilde{x}}(t),\end{aligned}\quad (3.100)$$

$$\begin{aligned}N_{B1}(t) &\triangleq W_f^T \nabla_{V_f^T x} \sigma_f \left( V_f^T x(t) \right) V_f^T \dot{x}(t) - \frac{1}{2} W_f^T \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{x}(t) \\ &\quad - \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) V_f^T \dot{x}(t) + \dot{\epsilon}_f(x(t), \dot{x}(t)),\end{aligned}\quad (3.101)$$

$$\begin{aligned}\hat{N}_{B2}(t) &\triangleq \frac{1}{2} \tilde{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \hat{V}_f^T(t) \dot{\hat{x}}(t) \\ &\quad + \frac{1}{2} \hat{W}_f^T(t) \nabla_{V_f^T x} \sigma_f \left( \hat{V}_f^T(t) \hat{x}(t) \right) \tilde{V}_f^T(t) \dot{\hat{x}}(t).\end{aligned}\quad (3.102)$$

To facilitate the subsequent stability analysis, an auxiliary term  $N_{B2} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is defined by replacing  $\dot{\hat{x}}(t)$  in  $\hat{N}_{B2}(t)$  by  $\dot{x}(t)$ , and the mismatch between  $N_{B2}$  and  $\hat{N}_{B2}$  is defined as  $\tilde{N}_{B2} \triangleq \hat{N}_{B2} - N_{B2}$ . The terms  $N_{B1}$  and  $N_{B2}$  are grouped as  $N_B \triangleq N_{B1} + N_{B2}$ . Using Property 2.3, Assumption 3.18, (3.96), (3.98), (3.101), and (3.102) the following bounds can be obtained over the set  $\chi \times \mathbb{R}^{2n} \times \mathbb{R}^{(L_f+1) \times n} \times \mathbb{R}^{n \times L_f}$

$$\|\tilde{N}(t)\| \leq \rho_1(\|z(t)\|) \|z(t)\|, \quad \|N_{B1}(t)\| \leq \zeta_1, \quad \|N_{B2}(t)\| \leq \zeta_2, \quad (3.103)$$

$$\|\dot{N}_B(t)\| \leq \zeta_3 + \zeta_4 \rho_2(\|z(t)\|) \|z(t)\|, \quad (3.104)$$

$$\|\dot{\tilde{x}}^T(t) \tilde{N}_{B2}(t)\| \leq \zeta_5 \|\tilde{x}(t)\|^2 + \zeta_6 \|e_f(t)\|^2, \quad (3.105)$$

where  $z(t) \triangleq [\tilde{x}^T(t) \ e_f^T(t)]^T \in \mathbb{R}^{2n}$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$  and  $\rho_1, \rho_2 : \mathbb{R} \rightarrow \mathbb{R}$  are positive, strictly increasing functions, and  $\zeta_i \in \mathbb{R}$ ,  $i = 1, \dots, 6$  are positive constants. To facilitate the subsequent stability analysis, let the auxiliary signal  $y : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{2n+2}$  be defined as

$$y(t) \triangleq [\tilde{x}^T(t) \ e_f^T(t) \ \sqrt{P(t)} \ \sqrt{Q(t)}]^T, \quad \forall t \in \mathbb{R}_{\geq t_0} \quad (3.106)$$

where the auxiliary signal  $P : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the Filippov solution to the initial value problem [11]

$$\begin{aligned} \dot{P}(t) &= \beta_2 \rho_2(\|z(t)\|) \|z(t)\| \|\tilde{x}(t)\| - e_f^T(t) (N_{B1}(t) - \beta_1 \operatorname{sgn}(\tilde{x}(t))) - \dot{\tilde{x}}^T(t) N_{B2}(t), \\ P(t_0) &= \beta_1 \sum_{i=1}^n |\tilde{x}_i(t_0)| - \tilde{x}^T(t_0) N_B(t_0), \end{aligned} \quad (3.107)$$

where  $\beta_1, \beta_2 \in \mathbb{R}$  are selected according to the sufficient conditions

$$\beta_1 > \max(\zeta_1 + \zeta_2, \ \zeta_1 + \frac{\zeta_3}{\alpha}), \quad \beta_2 > \zeta_4, \quad (3.108)$$

such that  $P(t) \geq 0$  for all  $t \in [0, \infty)$  (see Appendix A.1.1). The auxiliary function  $Q : \mathbb{R}^{n(2L_f+1)} \rightarrow \mathbb{R}$  in (3.106) is defined as  $Q \triangleq \frac{1}{4} \alpha \left[ \operatorname{tr}(\tilde{W}_f^T \Gamma_{wf}^{-1} \tilde{W}_f) + \operatorname{tr}(\tilde{V}_f^T \Gamma_{vf}^{-1} \tilde{V}_f) \right]$ .

Let  $\mathcal{D} \subset \mathbb{R}^{2n+2}$  be the open and connected set defined as  $\mathcal{D} \triangleq \{y \in \mathbb{R}^{2n+2} \mid \|y\| < \inf(\rho^{-1}([2\sqrt{\lambda}\eta, \infty)))\}$ , where  $\lambda$  and  $\eta$  are defined in Appendix A.1.7. Let  $\overline{\mathcal{D}}$  be the compact set  $\overline{\mathcal{D}} \triangleq \{y \in \mathbb{R}^{2n+2} \mid \|y\| \leq \inf(\rho^{-1}([2\sqrt{\lambda}\eta, \infty)))\}$ . Let  $V_I : \mathcal{D} \rightarrow \mathbb{R}$  be a positive-definite, locally Lipschitz, regular function defined as

$$V_I(y) \triangleq \frac{1}{2} e_f^T e_f + \frac{1}{2} \gamma \tilde{x}^T \tilde{x} + P + Q. \quad (3.109)$$

The candidate Lyapunov function in (3.109) satisfies the inequalities

$$U_1(y) \leq V_I(y) \leq U_2(y), \quad (3.110)$$

where  $U_1(y), U_2(y) \in \mathbb{R}$  are continuous positive definite functions defined as

$$U_1 \triangleq \frac{1}{2} \min(1, \gamma) \|y\|^2 \quad U_2 \triangleq \max(1, \gamma) \|y\|^2.$$

Additionally, let  $\mathcal{S} \subset \mathcal{D}$  denote a set defined as  $\mathcal{S} \triangleq \{y \in \mathcal{D} \mid \rho(\sqrt{2U_2(y)}) < 2\sqrt{\lambda\eta}\}$ , and let

$$\dot{y}(t) = h(y(t), t) \quad (3.111)$$

represent the closed-loop differential equations in (3.95), (3.98), (3.99), and (3.107), where  $h(y, t) : \mathbb{R}^{2n+2} \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{2n+2}$  denotes the right-hand side of the the closed-loop error signals.

**Theorem 3.20** *For the system in (3.90), the identifier developed in (3.92) along with the weight update laws in (3.98) ensures asymptotic identification of the state and its derivative, in the sense that*

$$\lim_{t \rightarrow \infty} \|\tilde{x}(t)\| = 0, \quad \lim_{t \rightarrow \infty} \|\dot{\tilde{x}}(t)\| = 0,$$

provided Assumptions 3.18 and 3.19 hold, and the control gains  $k$  and  $\gamma$  are selected sufficiently large based on the initial conditions of the states, and satisfy the following sufficient conditions

$$\alpha\gamma > \zeta_5, \quad k > \zeta_6, \quad (3.112)$$

where  $\zeta_5$  and  $\zeta_6$  are introduced in (3.105), and  $\beta_1, \beta_2$  introduced in (3.107), are selected according to the sufficient conditions in (3.108).

*Proof* See Appendix A.1.7. □

### 3.4.4 Actor-Critic Design

Using Property 2.3 and (3.84), the optimal value function and the optimal controls can be represented by neural networks as

$$\begin{aligned} V_i^*(x) &= W_i^T \sigma_i(x) + \epsilon_i(x), \\ u_i^*(x) &= -\frac{1}{2} R_{ii}^{-1} g_i^T(x) (\nabla_x \sigma_i^T(x) W_i + \nabla_x \epsilon_i^T(x)), \end{aligned} \quad (3.113)$$

where  $W_i \in \mathbb{R}^{L_i}$  are unknown constant ideal neural network weights,  $L_i$  is the number of neurons,  $\sigma_i = [\sigma_{i1} \ \sigma_{i2} \dots \sigma_{iL_i}]^T : \mathbb{R}^n \rightarrow \mathbb{R}^{L_i}$  are smooth neural network activation functions, such that  $\sigma_i(0) = 0$  and  $\nabla_x \sigma_i(0) = 0$ , and  $\epsilon_i : \mathbb{R}^n \rightarrow \mathbb{R}$  are the function reconstruction errors.

Using Property 2.3, both  $V_i^*$  and  $\nabla_x V_i^*$  can be uniformly approximated by neural networks in (3.113) (i.e., as  $L_i \rightarrow \infty$ , the approximation errors  $\epsilon_i, \nabla_x \epsilon_i \rightarrow 0$  for  $i = 1, 2$ , respectively). The critic  $\hat{V}$  and the actor  $\hat{u}$  approximate the optimal value function and the optimal controls in (3.113), and are given as

$$\begin{aligned}\hat{u}_i(x, \hat{W}_{ai}) &= -\frac{1}{2}R_{ii}^{-1}g_i^T(x)\nabla_x\sigma_i^T(x)\hat{W}_{ai}, \\ \hat{V}_i(x, \hat{W}_{ci}) &= \hat{W}_{ci}^T\sigma_i(x),\end{aligned}\quad (3.114)$$

where  $\hat{W}_{ci} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_i}$  and  $\hat{W}_{ai} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_i}$  are estimates of the ideal weights of the critic and actor neural networks, respectively. The weight estimation errors for the critic and actor are defined as  $\tilde{W}_{ci}(t) \triangleq W_i - \hat{W}_{ci}(t)$  and  $\tilde{W}_{ai}(t) \triangleq W_i - \hat{W}_{ai}(t)$  for  $i = 1, 2$ , respectively.

### Least-Squares Update for the Critic

The recursive formulation of the normalized least-squares algorithm is used to derive the update laws for the two critic weights as

$$\dot{\hat{W}}_{ci}(t) = -k_{ci}\Gamma_{ci}(t) \frac{\omega_i(t)}{1 + v_i\omega_i^T(t)\Gamma_{ci}(t)\omega_i(t)}\hat{\delta}_{ti}(t), \quad (3.115)$$

where  $\omega_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_i}$ , defined as  $\omega_i(t) \triangleq \nabla_x\sigma_i(x(t))\hat{F}_u(x(t), \hat{x}(t), \hat{u}_1(x(t), \hat{W}_{a1}(t)), u_2(x(t), \hat{W}_{a2}(t)))$  for  $i = 1, 2$ , is the critic neural network regressor vector,  $v_i, k_{ci} \in \mathbb{R}$  are constant positive gains and  $\hat{\delta}_{ti} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  denotes evaluation of the approximate Bellman error in (3.89), along the system trajectories, defined as  $\hat{\delta}_{ti} \triangleq \hat{\delta}(x(t), \hat{x}(t), \hat{W}_{ci}(t), \hat{W}_{a1}(t), \hat{W}_{a2}(t))$ . In (3.115),  $\Gamma_{ci} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_i \times L_i}$  for  $i = 1, 2$ , are symmetric estimation gain matrices generated by

$$\dot{\Gamma}_{ci}(t) = -k_{ci}\left(-\lambda_i\Gamma_{ci}(t) + \Gamma_{ci}(t)\frac{\omega_i(t)\omega_i^T(t)}{1 + v_i\omega_i^T(t)\Gamma_{ci}(t)\omega_i(t)}\Gamma_{ci}(t)\right), \quad (3.116)$$

where  $\lambda_1, \lambda_2 \in (0, 1)$  are forgetting factors. The use of forgetting factors ensures that  $\Gamma_{c1}$  and  $\Gamma_{c2}$  are positive-definite for all time and prevents arbitrarily small values in some directions, making adaptation in those directions very slow. Thus, the covariance matrices ( $\Gamma_{c1}, \Gamma_{c2}$ ) can be bounded as

$$\varphi_{11}\mathbf{I}_{L_1} \leq \Gamma_{c1}(t) \leq \varphi_{01}\mathbf{I}_{L_1}, \quad \varphi_{12}\mathbf{I}_{L_2} \leq \Gamma_{c2}(t) \leq \varphi_{02}\mathbf{I}_{L_2}. \quad (3.117)$$

### Gradient Update for the Actor

The actor update, like the critic update, is based on the minimization of the Bellman error. However, unlike the critic weights, the actor weights appear nonlinearly in the Bellman error, making it problematic to develop a least-squares update law. Hence, a gradient update law is developed for the actor which minimizes the squared Bellman error  $E_a(x, \hat{x}, \hat{W}_{c1}, \hat{W}_{c2}, \hat{W}_{a1}, \hat{W}_{a2}) \triangleq \sum_{i=1}^2 \hat{\delta}_i(x, \hat{x}, \hat{W}_{ci}, \hat{W}_{a1}, \hat{W}_{a2})$ . The actor neural networks are updated as

$$\dot{\hat{W}}_{ai}(t) = \text{proj} \left\{ -\frac{k_{ai1}}{\sqrt{1 + \omega_i^T(t)\omega_i(t)}} E'_{ai}(t) - k_{ai2}(\hat{W}_{ai}(t) - \hat{W}_{ci}(t)) \right\}, \quad (3.118)$$

where  $E'_{ai}(t) \triangleq \frac{\partial E_a(x(t), \dot{x}(t), \hat{W}_{c1}(t), \hat{W}_{c2}(t), \hat{W}_{a1}(t), \hat{W}_{a2}(t))}{\partial \hat{W}_{ai}}$ ,  $k_{ai1}, k_{ai2} \in \mathbb{R}$  are positive adaptation gains, and the smooth projection operator ensures that  $\|\hat{W}_{ia}(t)\| \leq \bar{W}$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ ,  $i = 1, 2$ , where  $\bar{W} \in \mathbb{R}_{>0}$  is a positive constant such that  $\|W\| \leq \bar{W}$  [35, 36].

The first term in (3.118) is normalized and the last term is added as feedback for stability (based on the subsequent stability analysis). For notational brevity, let  $B_{\bar{W}_i}$  denote the set  $\{w \in \mathbb{R}^{L_i} \mid \|w\| \leq 2\bar{W}\}$ .

### 3.4.5 Stability Analysis

The dynamics of the critic weight estimation errors  $\tilde{W}_{c1}$  and  $\tilde{W}_{c2}$  can be developed as

$$\begin{aligned} \dot{\tilde{W}}_{c1} &= k_{c1} \Gamma_{c1} \frac{\omega_1}{1 + \nu_1 \omega_1^T \Gamma_{c1} \omega_1} \left[ -\tilde{W}_{c1}^T \omega_1 - W_1^T \nabla_x \sigma_1 \tilde{F}_{\hat{u}} - u_1^{*T} R_{11} u_1^* - \epsilon'_{1v} F_{u^*} + \hat{u}_1^T R_{11} \hat{u}_1 \right. \\ &\quad \left. + W_1^T \nabla_x \sigma_1 (g_1(\hat{u}_1 - u_1^*) + g_2(\hat{u}_2 - u_2^*)) - u_2^{*T} R_{12} u_2^* + \hat{u}_2^T R_{12} \hat{u}_2 \right], \end{aligned}$$

and

$$\begin{aligned} \dot{\tilde{W}}_{c2} &= k_{c2} \Gamma_{c2} \frac{\omega_2}{1 + \nu_2 \omega_2^T \Gamma_{c2} \omega_2} \left[ -\tilde{W}_{c2}^T \omega_2 - W_2^T \nabla_x \sigma_2 \tilde{F}_{\hat{u}} - u_2^{*T} R_{22} u_2^* - \epsilon'_{2v} F_{u^*} + \hat{u}_2^T R_{22} \hat{u}_2 \right. \\ &\quad \left. + W_2^T \nabla_x \sigma_2 (g_1(\hat{u}_1 - u_1^*) + g_2(\hat{u}_2 - u_2^*)) - u_1^{*T} R_{21} u_1^* + \hat{u}_1^T R_{21} \hat{u}_1 \right]. \quad (3.119) \end{aligned}$$

Substituting for  $(u_1^*, u_2^*)$  and  $(\hat{u}_1, \hat{u}_2)$  from (3.113) and (3.114), respectively, in (3.119) yields

$$\begin{aligned} \dot{\tilde{W}}_{c1} &= -k_{c1} \Gamma_{c1} \psi_1 \psi_1^T \tilde{W}_{c1} + k_{c1} \Gamma_{c1} \frac{\omega_1}{1 + \nu_1 \omega_1^T \Gamma_{c1} \omega_1} \left[ -W_1^T \nabla_x \sigma_1 \tilde{F}_{\hat{u}} \right. \\ &\quad + \frac{1}{4} \tilde{W}_{a2}^T \nabla_x \sigma_2 G_{12} \nabla_x \sigma_2^T \tilde{W}_{a2} - \frac{1}{4} \nabla_x \epsilon_2 G_{12} \nabla_x \epsilon_2^T + \frac{1}{4} \tilde{W}_{a1}^T \nabla_x \sigma_1 G_1 \nabla_x \sigma_1^T \tilde{W}_{a1} \\ &\quad + \frac{1}{2} \left( \tilde{W}_{a2} \nabla_x \sigma_2 + \nabla_x \epsilon_2^T \right) (G_2 \nabla_x \sigma_1^T W_1 - G_{12} \nabla_x \sigma_2^T W_2) \\ &\quad \left. - \frac{1}{4} \nabla_x \epsilon_1 G_1 \nabla_x \epsilon_1^T - \nabla_x \epsilon_1 F_{u^*} \right], \end{aligned}$$

$$\begin{aligned}\dot{\tilde{W}}_{c2} = & -k_{c2}\Gamma_{c2}\psi_2\psi_2^T\tilde{W}_{c2} + k_{c2}\Gamma_{c2}\frac{\omega_2}{1+\nu_2\omega_2^T\Gamma_{c2}\omega_2}\left[-W_2^T\nabla_x\sigma_2\tilde{F}_{\hat{u}}\right. \\ & +\frac{1}{4}\tilde{W}_{a1}^T\nabla_x\sigma_1G_{21}\nabla_x\sigma_1^T\tilde{W}_{a1}-\frac{1}{4}\nabla_x\epsilon_1G_{21}\nabla_x\epsilon_1^T+\frac{1}{4}\tilde{W}_{a2}^T\nabla_x\sigma_2G_2\nabla_x\sigma_2^T\tilde{W}_{a2} \\ & \left.+\frac{1}{2}\left(\tilde{W}_{a1}\nabla_x\sigma_1+\nabla_x\epsilon_1^T\right)\left(G_1\nabla_x\sigma_2^TW_2-G_{21}\nabla_x\sigma_1^TW_1\right)-\frac{1}{4}\nabla_x\epsilon_2G_2\nabla_x\epsilon_2^T\right. \\ & \left.-\nabla_x\epsilon_2F_{u^*}\right],\end{aligned}\quad (3.120)$$

where  $\psi_i(t) \triangleq \frac{\omega_i(t)}{\sqrt{1+\nu_i\omega_i^T(t)\Gamma_{ci}(t)\omega_i(t)}} \in \mathbb{R}^{L_i}$  are the normalized critic regressor vectors for  $i = 1, 2$ , respectively, bounded as

$$\|\psi_1\| \leq \frac{1}{\sqrt{\nu_1\varphi_{11}}}, \quad \|\psi_2\| \leq \frac{1}{\sqrt{\nu_2\varphi_{12}}}, \quad (3.121)$$

where  $\varphi_{11}$  and  $\varphi_{12}$  are introduced in (3.117). The error systems in (3.120) can be represented as the following perturbed systems

$$\dot{\tilde{W}}_{c1} = \Omega_1 + \Lambda_{01}\Delta_1, \quad \dot{\tilde{W}}_{c2} = \Omega_2 + \Lambda_{02}\Delta_2, \quad (3.122)$$

where  $\Omega_i(\tilde{W}_{ci}, t) \triangleq -\eta_{ci}\Gamma_{ci}(t)\psi_i(t)\psi_i^T(t)\tilde{W}_{ci} \in \mathbb{R}^{L_i}$  denotes the nominal system,  $\Lambda_{0i} \triangleq \frac{\eta_{ci}\Gamma_{ci}\omega_i}{1+\nu_i\omega_i^T\Gamma_{ci}\omega_i}$  denotes the perturbation gain, and the perturbations  $\Delta_i \in \mathbb{R}^{L_i}$  are denoted as

$$\begin{aligned}\Delta_i = & \left[ -W_i^T\nabla_x\sigma_i\tilde{F}_{\hat{u}} + \frac{1}{4}\tilde{W}_{ai}^T\nabla_x\sigma_iG_i\nabla_x\sigma_i^T\tilde{W}_{ai} - \nabla_x\epsilon_iF_{u^*}\right. \\ & +\frac{1}{4}\tilde{W}_{ak}^T\nabla_x\sigma_kG_{ik}\nabla_x\sigma_k^T\tilde{W}_{ak}-\frac{1}{4}\nabla_x\epsilon_kG_{ik}\nabla_x\epsilon_k^T \\ & \left.-\frac{1}{4}\nabla_x\epsilon_iG_i\nabla_x\epsilon_i^T+\frac{1}{2}\left(\tilde{W}_{ak}\nabla_x\sigma_k+\nabla_x\epsilon_k^T\right)\left(G_k\nabla_x\sigma_i^TW_i-G_{ik}\nabla_x\sigma_k^TW_k\right)\right],\end{aligned}$$

where  $i = 1, 2$  and  $k = 3 - i$ . Using Theorem 2.5.1 in [15], it can be shown that the nominal systems

$$\dot{\tilde{W}}_{c1} = -k_{c1}\Gamma_{c1}\psi_1\psi_1^T\tilde{W}_{c1}, \quad \dot{\tilde{W}}_{c2} = -k_{c2}\Gamma_{c2}\psi_2\psi_2^T\tilde{W}_{c2}, \quad (3.123)$$

are exponentially stable if the bounded signals  $(\psi_1(t), \psi_2(t))$  are uniformly persistently exciting over the compact set  $\chi \times \bar{\mathcal{D}} \times \mathbf{B}_{\bar{W}_1} \times \mathbf{B}_{\bar{W}_2}$ , as [17]

$$\mu_{i2}\mathbf{I}_{L_i} \geq \int_{t_0}^{t_0+\delta_i} \psi_i(\tau)\psi_i(\tau)^T d\tau \geq \mu_{i1}\mathbf{I}_{L_i} \quad \forall t_0 \geq 0, i = 1, 2,$$

where  $\mu_{i1}, \mu_{i2}, \delta_i \in \mathbb{R}$  are positive constants independent of the initial conditions. Since  $\Omega_i$  is continuously differentiable in  $\tilde{W}_{ci}$  and the Jacobian  $\nabla_{\tilde{W}_{ci}} \Omega_i = -\eta_{ci} \Gamma_{ci} \psi_i \psi_i^T$  is bounded for the exponentially stable system (3.123) for  $i = 1, 2$ , the Converse Lyapunov Theorem 4.14 in [18] can be used to show that there exists a function  $V_c : \mathbb{R}^{L_i} \times \mathbb{R}^{L_i} \times [0, \infty) \rightarrow \mathbb{R}$ , which satisfies the following inequalities

$$\begin{aligned} c_{11} \left\| \tilde{W}_{c1} \right\|^2 + c_{12} \left\| \tilde{W}_{c2} \right\|^2 &\leq V_c(\tilde{W}_{c1}, \tilde{W}_{c2}, t), \\ V_c(\tilde{W}_{c1}, \tilde{W}_{c2}, t) &\leq c_{21} \left\| \tilde{W}_{c1} \right\|^2 + c_{22} \left\| \tilde{W}_{c2} \right\|^2, \\ -c_{31} \left\| \tilde{W}_{c1} \right\|^2 - c_{32} \left\| \tilde{W}_{c2} \right\|^2 &\geq \frac{\partial V_c}{\partial t} + \frac{\partial V_c}{\partial \tilde{W}_{c1}} \Omega_1(\tilde{W}_{c1}, t) + \frac{\partial V_c}{\partial \tilde{W}_{c2}} \Omega_2(\tilde{W}_{c2}, t), \\ \left\| \frac{\partial V_c}{\partial \tilde{W}_{c1}} \right\| &\leq c_{41} \left\| \tilde{W}_{c1} \right\|, \\ \left\| \frac{\partial V_c}{\partial \tilde{W}_{c2}} \right\| &\leq c_{42} \left\| \tilde{W}_{c2} \right\|, \end{aligned} \quad (3.124)$$

for some positive constants  $c_{1i}, c_{2i}, c_{3i}, c_{4i} \in \mathbb{R}$  for  $i = 1, 2$ . Using Property 2.3, Assumption 3.18, the projection bounds in (3.118), the fact that  $t \mapsto F_u(x(t), u_1^*(x(t)), u_2^*(x(t))) \in \mathcal{L}_\infty$  over compact sets (using (3.91)), and provided the conditions of Theorem 1 hold (required to prove that  $t \mapsto \tilde{F}_{\hat{u}}(x(t), \hat{x}(t), \hat{u}(x(t), \hat{W}_a(t))) \in \mathcal{L}_\infty$ ), the following bounds are developed to facilitate the subsequent stability proof

$$\begin{aligned} \iota_1 &\geq \left\| \tilde{W}_{a1} \right\|, & \iota_2 &\geq \left\| \tilde{W}_{a2} \right\|, \\ \iota_3 &\geq \left\| \nabla_x \sigma_1 G_1 \nabla_x \sigma_1^T \right\|, & \iota_4 &\geq \left\| \nabla_x \sigma_2 G_2 \nabla_x \sigma_2^T \right\|, \\ \iota_5 &\geq \left\| \Delta_1 \right\|; & \iota_6 &\geq \left\| \Delta_2 \right\|, \\ \iota_7 &\geq \frac{1}{4} \|G_1 - G_{21}\| \left\| \nabla_x V_1^* \right\|^2 + \frac{1}{4} \|G_2 - G_{12}\| \left\| \nabla_x V_2^* \right\|^2 \\ &\quad + \frac{1}{2} \left\| \nabla_x V_1^* (G_2 + G_1) \nabla_x V_2^{*T} \right\|, \\ \iota_8 &\geq \left\| -\frac{1}{2} (\nabla_x V_1^* - \nabla_x V_2^*) (G_1 \nabla_x \sigma_1^T W_{a1} - G_2 \nabla_x \sigma_2^T W_{a2}) \right. \\ &\quad \left. + \frac{1}{2} (\nabla_x V_1^* - \nabla_x V_2^*) (G_1 \nabla_x \sigma_1^T \tilde{W}_{a1} - G_2 \nabla_x \sigma_2^T \tilde{W}_{a2}) \right\|, \\ \iota_9 &\geq \left\| \nabla_x \sigma_1 G_{21} \nabla_x \sigma_1^T \right\|, & \iota_{10} &\geq \left\| \nabla_x \sigma_2 G_1 \nabla_x \sigma_1^T \right\|, \\ \iota_{11} &\geq \left\| \nabla_x \sigma_1 G_2 \nabla_x \sigma_2^T \right\|, & \iota_{12} &\geq \left\| \nabla_x \sigma_2 G_{12} \nabla_x \sigma_2^T \right\|, \end{aligned} \quad (3.125)$$

where  $\iota_j \in \mathbb{R}$  for  $j = 1, \dots, 12$  are computable positive constants.

**Theorem 3.21** *If Assumptions 3.18 and 3.19 hold, the regressors  $\psi_i$  for  $i = 1, 2$  are uniformly persistently exciting, and provided (3.108), (3.112), and the following sufficient gain conditions are satisfied*

$$\begin{aligned} c_{31} &> k_{a11}\iota_1\iota_3 + k_{a21}\iota_2\iota_{11}, \\ c_{32} &> k_{a21}\iota_2\iota_4 + k_{a11}\iota_1\iota_{10}, \end{aligned}$$

where  $k_{a11}$ ,  $k_{a21}$ ,  $c_{31}$ ,  $c_{32}$ ,  $\iota_1$ ,  $\iota_2$ ,  $\iota_3$ , and  $\iota_4$  are introduced in (3.118), (3.124), and (3.125), then the controller in (3.114), the actor-critic weight update laws in (3.115)–(3.116) and (3.118), and the identifier in (3.92) and (3.98), guarantee that the state of the system,  $x(\cdot)$ , and the actor-critic weight estimation errors,  $(\tilde{W}_{a1}(\cdot), \tilde{W}_{a2}(\cdot))$  and  $(\tilde{W}_{c1}(\cdot), \tilde{W}_{c2}(\cdot))$ , are uniformly ultimately bounded.

*Proof* To investigate the stability of (3.90) with control inputs  $\hat{u}_1$  and  $\hat{u}_2$ , and the perturbed system (3.122), consider  $V_L : \chi \times \mathbb{R}^{L_1} \times \mathbb{R}^{L_1} \times \mathbb{R}^{L_2} \times \mathbb{R}^{L_2} \times [0, \infty) \rightarrow \mathbb{R}$  as the continuously differentiable, positive-definite Lyapunov function candidate, given as

$$V_L(x, \tilde{W}_{c1}, \tilde{W}_{c2}, \tilde{W}_{a1}, \tilde{W}_{a2}, t) \triangleq V_1^*(x) + V_2^*(x) + V_c(\tilde{W}_{c1}, \tilde{W}_{c2}, t) + \frac{1}{2}\tilde{W}_{a1}^T \tilde{W}_{a1} + \frac{1}{2}\tilde{W}_{a2}^T \tilde{W}_{a2},$$

where  $V_i^*$  for  $i = 1, 2$  (the optimal value function for (3.90)), is the Lyapunov function for (3.90), and  $V_c$  is the Lyapunov function for the exponentially stable system in (3.123). Since  $(V_1^*, V_2^*)$  are continuously differentiable and positive-definite, [18, Lemma 4.3] implies that there exist class  $\mathcal{K}$  functions  $\alpha_1$  and  $\alpha_2$  defined on  $[0, r]$ , where  $B_r \subset \mathcal{X}$ , such that

$$\alpha_1(\|x\|) \leq V_1^*(x) + V_2^*(x) \leq \alpha_2(\|x\|), \quad \forall x \in B_r. \quad (3.126)$$

Using (3.124) and (3.126),  $V_L$  can be bounded as

$$\begin{aligned} &\alpha_1(\|x\|) + c_{11} \|\tilde{W}_{c1}\|^2 + c_{12} \|\tilde{W}_{c2}\|^2 + \frac{1}{2} \left( \|\tilde{W}_{a1}\|^2 + \|\tilde{W}_{a2}\|^2 \right) \leq V_L \\ &\leq \alpha_2(\|x\|) + c_{21} \|\tilde{W}_{c1}\|^2 + c_{22} \|\tilde{W}_{c2}\|^2 + \frac{1}{2} \left( \|\tilde{W}_{a1}\|^2 + \|\tilde{W}_{a2}\|^2 \right). \end{aligned}$$

which can be written as  $\alpha_3(\|w\|) \leq V_L(w, t) \leq \alpha_4(\|w\|)$ ,  $\forall w \in B_s$ , where  $w \triangleq [x^T \tilde{W}_{c1}^T \tilde{W}_{c2}^T \tilde{W}_{a1}^T \tilde{W}_{a2}^T]^T$ ,  $\alpha_3$  and  $\alpha_4$  are class  $\mathcal{K}$  functions defined on  $[0, s]$ , where  $B_s \subset \chi \times \mathbb{R}^{L_1} \times \mathbb{R}^{L_1} \times \mathbb{R}^{L_2} \times \mathbb{R}^{L_2}$  is a ball of radius  $s$  centered at the origin. Taking the time derivative of  $V_L$  yields

$$\begin{aligned}\dot{V}_L &= (\nabla_x V_1^* + \nabla_x V_2^*) (f + g_1 \hat{u}_1 + g_2 \hat{u}_2) + \frac{\partial V_c}{\partial t} + \frac{\partial V_c}{\partial \tilde{W}_{c1}} \Omega_1 + \frac{\partial V_c}{\partial \tilde{W}_{c1}} A_{01} \Delta_1 + \frac{\partial V_c}{\partial \tilde{W}_{c2}} \Omega_2 \\ &\quad + \frac{\partial V_c}{\partial \tilde{W}_{c2}} A_{02} \Delta_2 - \tilde{W}_{a1}^T \dot{\tilde{W}}_{a1} - \tilde{W}_{a2}^T \dot{\tilde{W}}_{a2},\end{aligned}\tag{3.127}$$

where the time derivatives of  $V_i^*$  for  $i = 1, 2$ , are taken along the trajectories of the system (3.90) with control inputs  $(\hat{u}_1, \hat{u}_2)$  and the time derivative of  $V_c$  is taken along the along the trajectories of the perturbed system (3.122). Using (3.86),  $\nabla_x V_i^* f = -\nabla_x V_i^* (g_1 u_1^* + g_2 u_2^*) - Q_i(x) - \sum_{j=1}^2 u_j^{*T} R_{ij} u_j^*$  for  $i = 1, 2$ . Substituting for the  $\nabla_x V_i^* f$  terms in (3.127), using the fact that  $\nabla_x V_i^* g_i = -2u_i^{*T} R_{ii}$  from (3.84), and using (3.118) and (3.124), (3.127) can be upper bounded as

$$\begin{aligned}\dot{V}_L &\leq -Q - u_1^{*T} (R_{11} + R_{21}) u_1^* - u_2^{*T} (R_{22} + R_{12}) u_2^* + 2u_1^{*T} R_{11} (u_1^* - \hat{u}_1) \\ &\quad + 2u_2^{*T} R_{22} (u_2^* - \hat{u}_2) + \nabla_x V_1^* g_2 (\hat{u}_2 - u_2^*) + \nabla_x V_2^* g_1 (\hat{u}_1 - u_1^*) \\ &\quad + c_{41} A_{01} \|\tilde{W}_{c1}\| \|\Delta_1\| - c_{31} \|\tilde{W}_{c1}\|^2 + c_{42} A_{02} \|\tilde{W}_{c2}\| \|\Delta_2\| - c_{32} \|\tilde{W}_{c2}\|^2 \\ &\quad + \tilde{W}_{a1}^T \left[ \frac{k_{a11}}{\sqrt{1 + \omega_1^T \omega_1}} \frac{\partial E_a}{\partial \tilde{W}_{a1}} + k_{a12} (\hat{W}_{a1} - \tilde{W}_{c1}) \right] \\ &\quad + \tilde{W}_{a2}^T \left[ \frac{k_{a21}}{\sqrt{1 + \omega_2^T \omega_2}} \frac{\partial E_a}{\partial \tilde{W}_{a2}} + k_{a22} (\hat{W}_{a2} - \tilde{W}_{c2}) \right],\end{aligned}\tag{3.128}$$

where  $Q \triangleq Q_1 + Q_2$ . Substituting for  $u_i^*$ ,  $\hat{u}_i$ , and  $\Delta_i$  for  $i = 1, 2$  using (3.84), (3.114), (3.119), and (3.122), respectively, and using (3.117) and (3.121) in (3.128), yields

$$\begin{aligned}\dot{V}_L &\leq \frac{1}{4} \|G_1 - G_{21}\| \|\nabla_x V_1^*\|^2 + \frac{1}{4} \|G_2 - G_{12}\| \|\nabla_x V_2^*\|^2 \\ &\quad + \frac{1}{2} \|\nabla_x V_1^* (G_1 + G_2) \nabla_x V_2^{*T}\| - Q \\ &\quad - \frac{1}{2} (\nabla_x V_1^* - \nabla_x V_2^*) (G_1 \nabla_x \sigma_1^T W_{a1} - G_2 \nabla_x \sigma_2^T W_{a2}) \\ &\quad + \frac{1}{2} (\nabla_x V_1^* - \nabla_x V_2^*) (G_1 \nabla_x \sigma_1^T \tilde{W}_{a1} - G_2 \nabla_x \sigma_2^T \tilde{W}_{a2}) \\ &\quad + c_{41} \frac{k_{c1} \varphi_{01}}{2\sqrt{\nu_1 \varphi_{11}}} \|\Delta_1\| \|\tilde{W}_{c1}\| - c_{31} \|\tilde{W}_{c1}\|^2 \\ &\quad + c_{42} \frac{k_{c2} \varphi_{02}}{2\sqrt{\nu_2 \varphi_{12}}} \|\Delta_2\| \|\tilde{W}_{c2}\| - c_{32} \|\tilde{W}_{c2}\|^2 \\ &\quad + k_{a12} \|\tilde{W}_{a1}\| \|\tilde{W}_{c1}\| + k_{a22} \|\tilde{W}_{a2}\| \|\tilde{W}_{c2}\| - k_{a12} \|\tilde{W}_{a1}\|^2 - k_{a22} \|\tilde{W}_{a2}\|^2\end{aligned}$$

$$\begin{aligned}
& + \frac{k_{a11}}{\sqrt{1 + \omega_1^T \omega_1}} \tilde{W}_{a1}^T \left( \left( \tilde{W}_{c1} - \tilde{W}_{a1} \right)^T \nabla_x \sigma_1 G_1 \nabla_x \sigma_1^T \left( -\tilde{W}_{c1}^T \omega_1 + \Delta_1 \right) \right. \\
& + \left( \tilde{W}_{a1}^T \nabla_x \sigma_1 G_{21} - \tilde{W}_{c2}^T \nabla_x \sigma_2 G_2 \right) \nabla_x \sigma_1^T \left( -\tilde{W}_{c2}^T \omega_2 + \Delta_2 \right) \\
& + \left. \left( W_1^T \nabla_x \sigma_1 G_{21} - W_2^T \nabla_x \sigma_2 G_1 \right) \nabla_x \sigma_1^T \left( -\tilde{W}_{c2}^T \omega_2 + \Delta_2 \right) \right) \\
& + \frac{k_{a21}}{\sqrt{1 + \omega_2^T \omega_2}} \tilde{W}_{a2}^T \left( \left( \tilde{W}_{c2} - \tilde{W}_{a2} \right)^T \nabla_x \sigma_2 G_2 \nabla_x \sigma_2^T \left( -\tilde{W}_{c2}^T \omega_2 + \Delta_2 \right) \right. \\
& + \left( \tilde{W}_{a2}^T \nabla_x \sigma_2 G_{12} - \tilde{W}_{c1}^T \nabla_x \sigma_1 G_2 \right) \nabla_x \sigma_2^T \left( -\tilde{W}_{c1}^T \omega_1 + \Delta_1 \right) \\
& \left. + \left( W_2^T \nabla_x \sigma_2 G_{12} - W_1^T \nabla_x \sigma_1 G_2 \right) \nabla_x \sigma_2^T \left( -\tilde{W}_{c1}^T \omega_1 + \Delta_1 \right) \right). \tag{3.129}
\end{aligned}$$

Using the bounds developed in (3.125), (3.129) can be further upper bounded as

$$\begin{aligned}
\dot{V}_L & \leq -Q - (c_{31} - k_{a11}\iota_1\iota_3 - k_{a21}\iota_2\iota_{11}) \left\| \tilde{W}_{c1} \right\|^2 - k_{a12} \left\| \tilde{W}_{a1} \right\|^2 \\
& - (c_{32} - k_{a21}\iota_2\iota_4 - k_{a11}\iota_1\iota_{10}) \left\| \tilde{W}_{c2} \right\|^2 + \sigma_2 \left\| \tilde{W}_{c2} \right\| - k_{a22} \left\| \tilde{W}_{a2} \right\|^2 \\
& + \sigma_1 \left\| \tilde{W}_{c1} \right\| + k_{a11}\iota_1 (\iota_1(\iota_3\iota_5 + \iota_6\iota_9) + \iota_6(\bar{W}_1\iota_9 + \bar{W}_2\iota_{10})) \\
& + k_{a21}\iota_2 (\iota_2(\iota_4\iota_6 + \iota_5\iota_{12}) + \iota_5(\bar{W}_1\iota_{11} + \bar{W}_2\iota_{12})) + \iota_7 + \iota_8,
\end{aligned}$$

where

$$\begin{aligned}
\sigma_1 & \triangleq \frac{c_{41}k_{c1}\varphi_{01}}{2\sqrt{\nu_1\varphi_{11}}} \iota_5 + k_{a11}(\iota_1\iota_3(\iota_1 + \iota_5)) + k_{a21}\iota_2(\iota_{11}(\iota_5 + \bar{W}_1) + \iota_{12}(\iota_2 + \bar{W}_2)) \\
& + k_{a12}\iota_1, \\
\sigma_2 & \triangleq \frac{c_{42}k_{c2}\varphi_{02}}{2\sqrt{\nu_2\varphi_{12}}} \iota_6 + k_{a21}(\iota_2\iota_4(\iota_2 + \iota_6)) + k_{a11}\iota_1(\iota_9(\iota_1 + \bar{W}_1) + \iota_{10}(\iota_6 + \bar{W}_2)) \\
& + k_{a22}\iota_2.
\end{aligned}$$

Provided  $c_{31} > k_{a11}\iota_1\iota_3 + k_{a21}\iota_2\iota_{11}$  and  $c_{32} > k_{a21}\iota_2\iota_4 + k_{a11}\iota_1\iota_{10}$ , completion of the squares yields

$$\begin{aligned}
\dot{V}_L & \leq -Q - k_{a22} \left\| \tilde{W}_{a2} \right\|^2 - k_{a12} \left\| \tilde{W}_{a1} \right\|^2 \\
& - (1 - \theta_1)(c_{31} - k_{a11}\iota_1\iota_3 - k_{a21}\iota_2\iota_{11}) \left\| \tilde{W}_{c1} \right\|^2 \\
& - (1 - \theta_2)(c_{32} - k_{a21}\iota_2\iota_4 - k_{a11}\iota_1\iota_{10}) \left\| \tilde{W}_{c2} \right\|^2
\end{aligned}$$

$$\begin{aligned}
& + k_{a11}\iota_1 (\iota_1(\iota_3\iota_5 + \iota_6\iota_9) + \iota_6(\bar{W}_1\iota_9 + \bar{W}_2\iota_{10})) \\
& + k_{a21}\iota_2 (\iota_2(\iota_4\iota_6 + \iota_5\iota_{12}) + \iota_5(\bar{W}_1\iota_{11} + \bar{W}_2\iota_{12})) \\
& + \frac{\sigma_1^2}{4\theta_1(c_{31} - k_{a11}\iota_1\iota_3 - k_{a21}\iota_2\iota_{11})} + \iota_7 \\
& + \frac{\sigma_2^2}{4\theta_2(c_{32} - k_{a21}\iota_2\iota_4 - k_{a11}\iota_1\iota_{10})} + \iota_8,
\end{aligned} \tag{3.130}$$

where  $\theta_1, \theta_2 \in (0, 1)$  are adjustable parameters. Since  $Q$  is positive definite, according to [18, Lemma 4.3], there exist class  $\mathcal{K}$  functions  $\alpha_5$  and  $\alpha_6$  such that

$$\alpha_5(\|w\|) \leq F(w) \leq \alpha_6(\|w\|) \quad \forall w \in B_s, \tag{3.131}$$

where

$$\begin{aligned}
F(w) = & Q + k_{a12} \left\| \tilde{W}_{a1} \right\|^2 + (1 - \theta_1)(c_{31} - k_{a11}\iota_1\iota_3 - k_{a21}\iota_2\iota_{11}) \left\| \tilde{W}_{c1} \right\|^2 \\
& + (1 - \theta_2)(c_{32} - k_{a21}\iota_2\iota_4 - k_{a11}\iota_1\iota_{10}) \left\| \tilde{W}_{c2} \right\|^2 + k_{a22} \left\| \tilde{W}_{a2} \right\|^2,
\end{aligned}$$

Using (3.131), the expression in (3.130) can be further upper bounded as  $\dot{V}_L \leq -\alpha_5(\|w\|) + \Upsilon$ , where

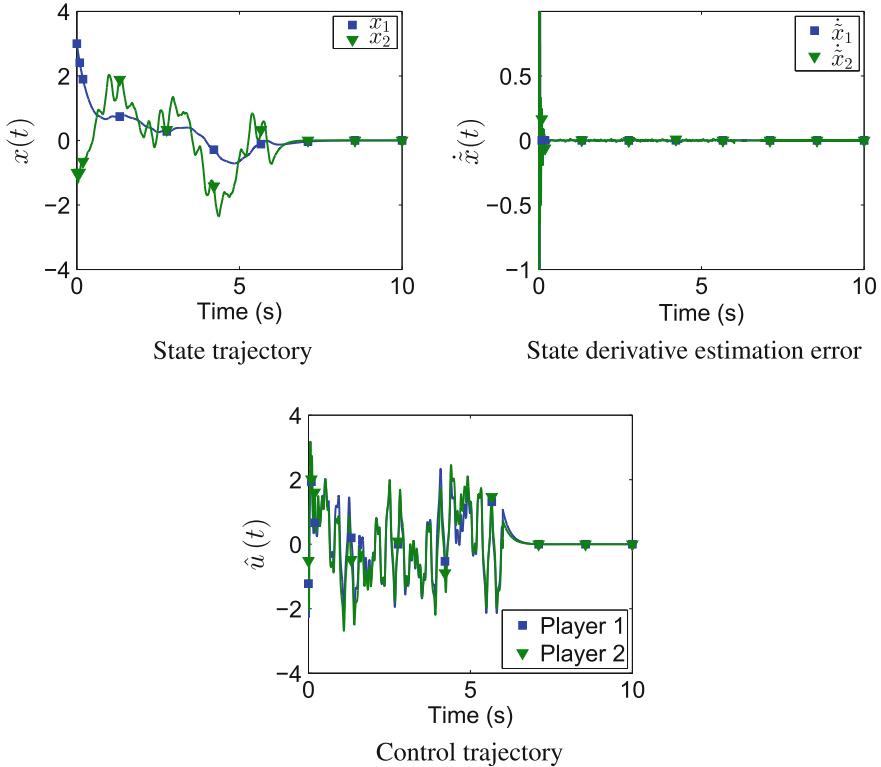
$$\begin{aligned}
\Upsilon = & k_{a11}\iota_1 (\iota_1(\iota_3\iota_5 + \iota_6\iota_9) + \iota_6(\bar{W}_1\iota_9 + \bar{W}_2\iota_{10})) + \frac{\sigma_1^2}{4\theta_1(c_{31} - k_{a11}\iota_1\iota_3 - k_{a21}\iota_2\iota_{11})} \\
& + k_{a21}\iota_2 (\iota_2(\iota_4\iota_6 + \iota_5\iota_{12}) + \iota_5(\bar{W}_1\iota_{11} + \bar{W}_2\iota_{12})) + \frac{\sigma_2^2}{4\theta_2(c_{32} - k_{a21}\iota_2\iota_4 - k_{a11}\iota_1\iota_{10})} \\
& + \iota_7 + \iota_8,
\end{aligned}$$

which proves that  $\dot{V}_L$  is negative whenever  $w$  lies outside the compact set  $\Omega_w \triangleq \{w : \|w\| \leq \alpha_5^{-1}(\Upsilon)\}$ , and hence,  $\|w(\cdot)\|$  is uniformly ultimately bounded, according to [18, Theorem 4.18].

### 3.4.6 Simulations

#### Two-Player Game with a Known Feedback-Nash Equilibrium Solution

The following two player non-zero sum game considered in [32, 37–39] has a known analytical solution, and hence is utilized in this section to demonstrate the performance of the developed technique. The system dynamics are given by  $\dot{x} = f(x) + g_1(x)u_1 + g_2(x)u_2$ , where



**Fig. 3.15** Evolution of the system states, state derivative estimates, and control signals for the two-player nonzero-sum game, with persistently excited input for the first six seconds (reproduced with permission from [30], ©2015, IEEE)

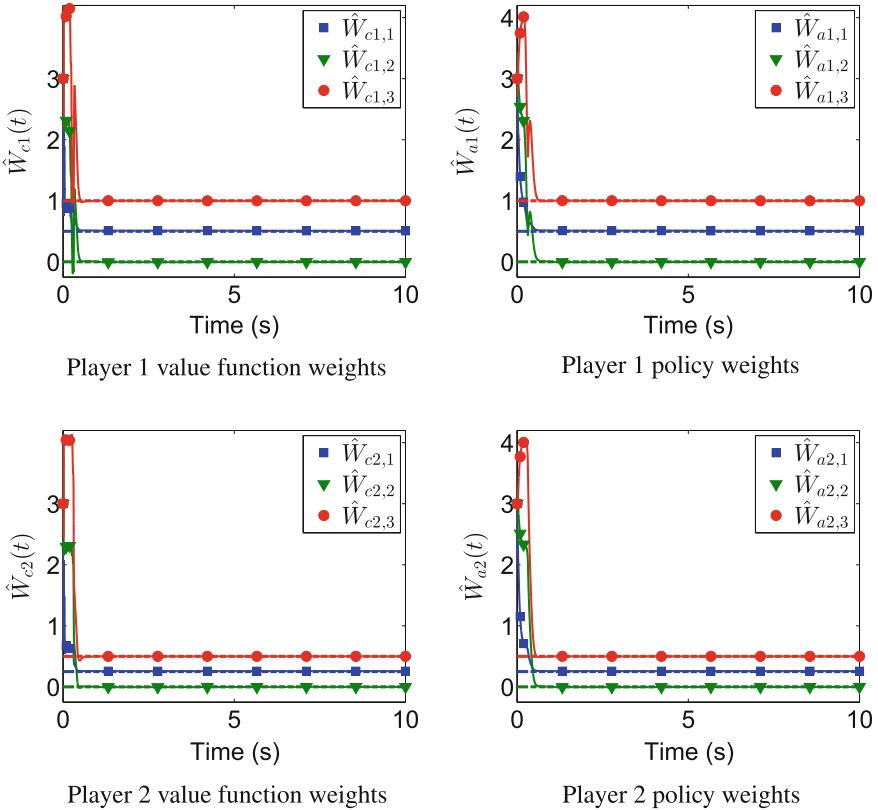
$$f(x) = \begin{bmatrix} (x_2 - 2x_1) \\ -\frac{1}{2}x_1 - x_2 + \frac{1}{4}x_2 (\cos(2x_1) + 2)^2 + \frac{1}{4}x_2 (\sin(4x_1^2) + 2)^2 \end{bmatrix}, \quad (3.132)$$

$$g_1(x) = [0 \ \cos(2x_1) + 2]^T, \quad (3.132)$$

$$g_2(x) = [0 \ \sin(4x_1^2) + 2]^T. \quad (3.133)$$

The objective is to design  $u_1$  and  $u_2$  to find a feedback-Nash equilibrium solution to the optimal control problem described by (3.83), where the local cost is given by  $r_i(x, u_i, u_j) = x^T Q_i x + u_i^T R_{ii} u_i + u_j^T R_{ij} u_j$ ,  $i = 1, 2$ , and  $j = 3 - i$ , where  $R_{11} = 2R_{22} = 2$ ,  $R_{12} = 2R_{21} = 2$ ,  $Q_1 = 2$ , and  $Q_2 = \mathbb{I}_2$ . The known analytical solutions for the optimal value functions of player 1 and player 2 are  $V_1^*(x) = \frac{1}{2}x_1^2 + x_2^2$ ,  $V_2^*(x) = \frac{1}{4}x_1^2 + \frac{1}{2}x_2^2$ , and the corresponding optimal control inputs are  $u_1^*(x) = -(\cos(2x_1) + 2)x_2$ ,  $u_2^*(x) = -\frac{1}{2}(\sin(4x_1^2) + 2)x_2$ .

To implement the developed technique, the activation function for critic neural networks are selected as  $\sigma_i(x) = [x_1^2 \ x_1 x_2 \ x_2^2]^T$ ,  $i = 1, 2$ , while the activation



**Fig. 3.16** Convergence of actor and critic weights for player 1 and player 2 in the nonzero-sum game (reproduced with permission from [30], ©2015, IEEE)

function for the identifier dynamic neural network is selected as a symmetric sigmoid with 5 neurons in the hidden layer. The identifier gains are selected as  $k = 300$ ,  $\alpha = 200$ ,  $\gamma = 5$ ,  $\beta_1 = 0.2$ ,  $\Gamma_{wf} = 0.1I_6$ , and  $\Gamma_{vf} = 0.1I_2$ , and the gains of the actor-critic learning laws are selected as  $k_{a11} = k_{a12} = 10$ ,  $k_{a21} = k_{a22} = 20$ ,  $k_{c1} = 50$ ,  $k_{c2} = 10$ ,  $v_1 = v_2 = 0.001$ , and  $\lambda_1 = \lambda_2 = 0.03$ . The covariance matrix is initialized to  $\Gamma(0) = 5000I_3$ , the neural network weights for state derivative estimator are randomly initialized with values between  $[-1, 1]$ , the weights for the actor and the critic are initialized to  $[3, 3, 3]^T$ , the state estimates are initialized to zero, and the states are initialized to  $x(0) = [3, -1]$ . Similar to results such as [9, 14, 32, 39, 40], a small amplitude exploratory signal (noise) is added to the control to excite the states for the first six seconds of the simulation, as seen from the evolution of states and control in Fig. 3.15. The identifier approximates the system dynamics, and the state derivative estimation error is shown in Fig. 3.15. The time histories of the critic neural network weights and the actor neural network weights are given in Fig. 3.16, where solid lines denote the weight estimates and dotted lines denote the true values of the weights. Persistence of excitation ensures that the weights converge to their

known ideal values in less than five seconds of simulation. The use of two separate neural networks facilitates the design of least-squares-based update laws in (3.115). The least-squares-based update laws result in a performance benefit over single neural network-based results such as [40], where the convergence of weights is obtained after about 250 s of simulation.

### Three Player Game

To demonstrate the performance of the developed technique in the multi-player case, the two player simulation is augmented with another actor. The resulting dynamics are  $\dot{x} = f(x) + g_1(x)u_1 + g_2(x)u_2 + g_3(x)u_3$ , where

$$f(x) = \begin{bmatrix} (x_2 - 2x_1) \\ \left( -\frac{1}{2}x_1 - x_2 + \frac{1}{4}x_2(\cos(2x_1) + 2)^2 + \frac{1}{4}x_2(\sin(4x_1^2) + 2)^2 \right. \\ \left. + \frac{1}{4}x_2(\cos(4x_1^2) + 2)^2 \right) \end{bmatrix},$$

$$g_3(x) = [0 \quad \cos(4x_1^2) + 2]^T, \quad (3.134)$$

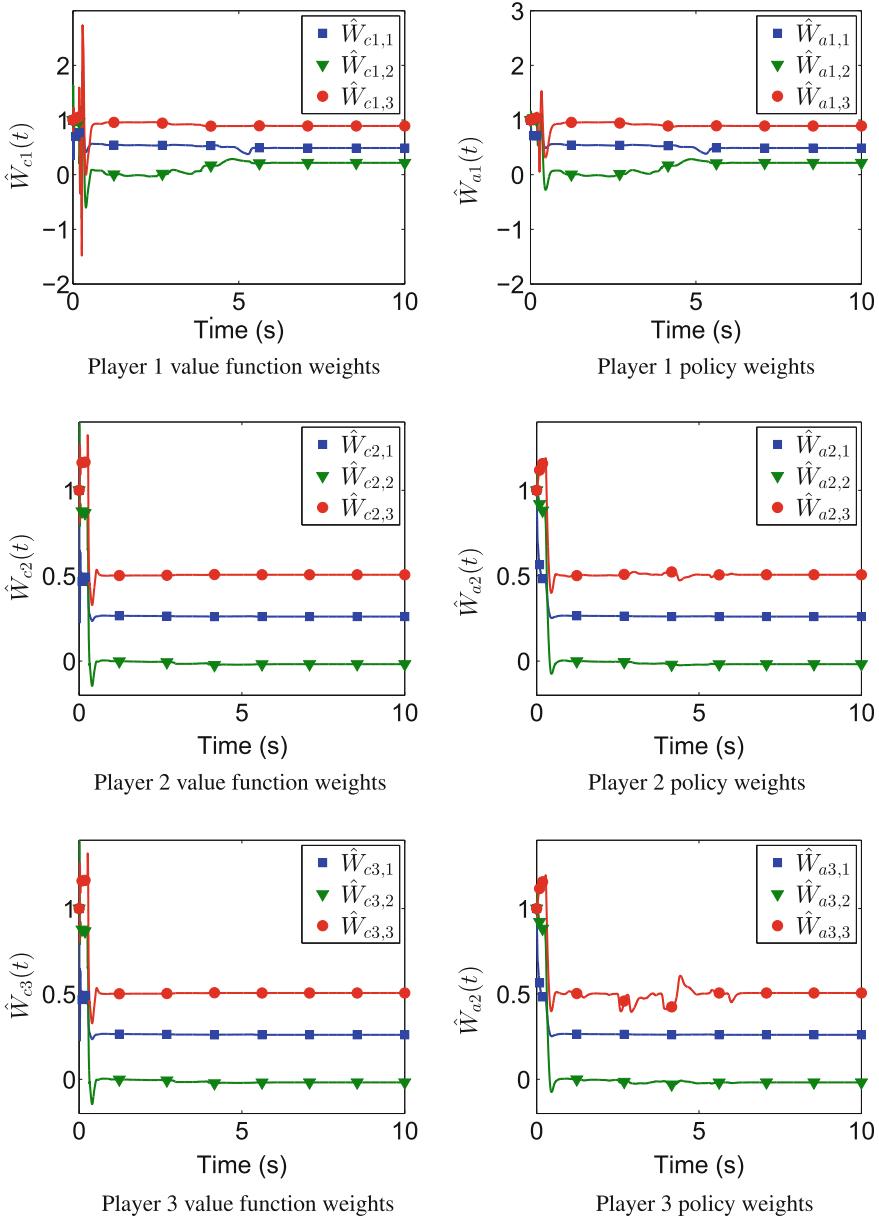
and  $g_1$  and  $g_2$  are the same as (3.133). Figure 3.17 demonstrates the convergence of the actor and the critic weights. Since the feedback-Nash equilibrium solution is unknown for the dynamics in (3.134), the obtained weights are not compared against their true values. Figure 3.18 demonstrates the regulation of the system states and the state derivative estimation error to the origin, and the boundedness of the control signals.

*Remark 3.22* An implementation issue in using the developed algorithm as well as results such as [9, 14, 32, 39, 40] is to ensure persistence of excitation of the critic regressor vector. Unlike linear systems, where persistence of excitation of the regressor translates to the sufficient richness of the external input, no verifiable method exists to ensure persistence of excitation in nonlinear systems. In this simulation, a small amplitude exploratory signal consisting of a sum of sines and cosines of varying frequencies is added to the control to ensure persistence of excitation qualitatively, and convergence of critic weights to their optimal values is achieved. The exploratory signal  $n(t)$ , designed using trial and error, is present in the first six seconds of the simulation and is given by

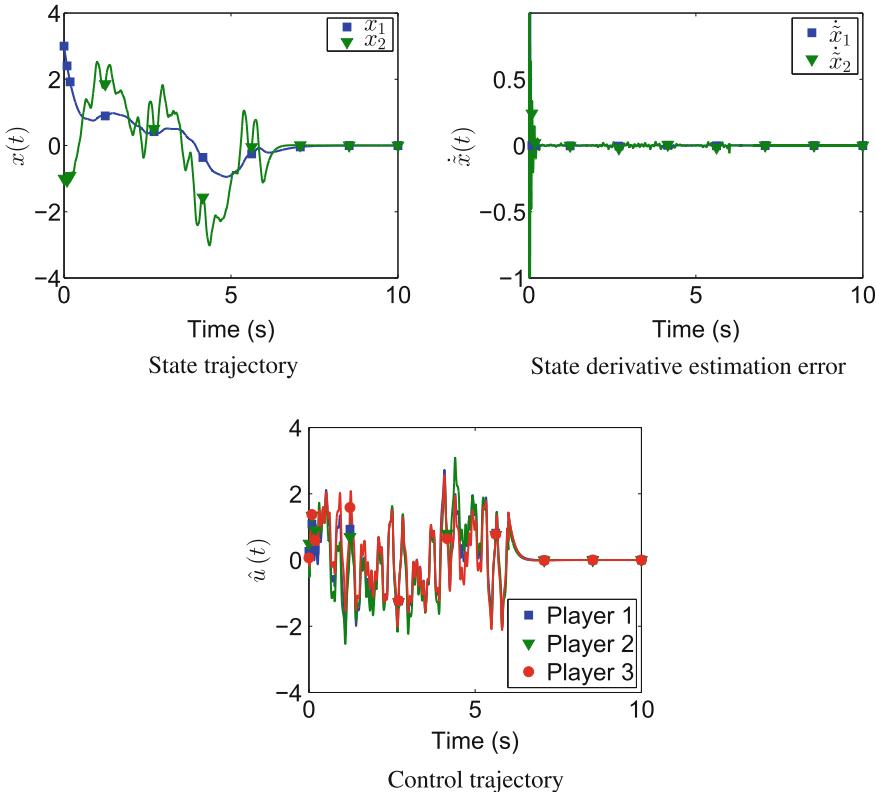
$$n(t) = \sin(5\pi t) + \sin(\epsilon t) + \sin^5(t) + \cos^5(20t) + \sin^2(-1.2t)\cos(0.5t).$$

## 3.5 Background and Further Reading

Reinforcement learning-based techniques have been developed to approximately solve optimal control problems for continuous-time and discrete-time deterministic systems in results such as [9, 14, 41–47] for set-point regulation, and [23, 48–52]



**Fig. 3.17** Convergence of actor and critic weights for the three-player nonzero-sum game (reproduced with permission from [30], ©2015, IEEE)



**Fig. 3.18** Evolution of the system states, state derivative estimates, and control signals for the three-player nonzero-sum game, with persistently excited input for the first six seconds (reproduced with permission from [30], ©2015, IEEE)

for trajectory tracking. Extension of approximate dynamic programming to systems with continuous time and state was pioneered by Doya [41] who used a Hamilton–Jacobi–Bellman framework to derive algorithms for value function approximation and policy improvement, based on a continuous-time version of the temporal difference error. Murray et al. [53] also used the Hamilton–Jacobi–Bellman framework to develop a *stepwise stable* iterative approximate dynamic programming algorithm for continuous-time input-affine systems with an input quadratic performance measure. In Beard et al. [54], Galerkin’s spectral method is used to approximate the solution to the generalized Hamilton–Jacobi–Bellman equation, and the solution is used to compute a stabilizing feedback controller offline. In [55], Abu-Khalaf and Lewis propose a least-squares successive approximation solution, where a neural network is trained offline to learn the generalized Hamilton–Jacobi–Bellman solution.

All of the aforementioned approaches are offline and/or require complete knowledge of system dynamics. One of the contributions in [56] is that only partial knowl-

edge of the system dynamics is required, and a hybrid continuous-time/discrete-time sampled data controller is developed based on policy iteration, where the feedback control operation of the actor occurs at faster time scale than the learning process of the critic. Vamvoudakis and Lewis [14] extended the idea by designing a model-based online algorithm called synchronous policy iteration, synchronous which involved synchronous, continuous-time adaptation of both actor and critic neural networks.

Over the past few years, research has focused on the development of robust [57, 58] and off-policy [59] approximate dynamic programming methods for near-optimal control of nonlinear systems.

For trajectory tracking, approximate dynamic programming approaches are presented in results such as [49, 60], where the value function, and the controller presented are time-varying functions of the tracking error. For discrete time systems, several approaches have been developed to address the tracking problem. Park et al. [61] use generalized back-propagation through time to solve a finite-horizon tracking problem that involves offline training of neural networks. An approximate dynamic programming-based approach is presented in [48] to solve an infinite-horizon optimal tracking problem where the desired trajectory is assumed to depend on the system states. Greedy heuristic dynamic programming based algorithms are presented in results such as [23, 62, 63] which transform the nonautonomous system into an autonomous system, and approximate convergence of the sequence of value functions to the optimal value function is established.

Recently results on near-optimal trajectory tracking include integral reinforcement learning [64], Q-learning [65], guaranteed cost [66], decentralized [67], robust [68], and event driven [69] approximate dynamic programming methods.

Generalization of reinforcement learning controllers to differential game problems is investigated in results such as [14, 32, 38, 39, 70–73]. Techniques utilizing Q-learning algorithms have been developed for a zero-sum game in [74]. An approximate dynamic programming procedure that provides a solution to the Hamilton–Jacobi–Isaacs equation associated with the two-player zero-sum nonlinear differential game is introduced in [70]. The approximate dynamic programming algorithm involves two iterative cost functions finding the upper and lower performance indices as sequences that converge to the saddle point solution to the game. The actor-critic structure required for learning the saddle point solution is composed of four action networks and two critic networks. The iterative approximate dynamic programming solution in [71] considers solving zero-sum differential games under the condition that the saddle point does not exist, and a mixed optimal performance index function is obtained under a deterministic mixed optimal control scheme when the saddle point does not exist. Another approximate dynamic programming iteration technique is presented in [72], in which the nonlinear quadratic zero-sum game is transformed into an equivalent sequence of linear quadratic zero-sum games to approximate an optimal saddle point solution. In [73], an integral reinforcement learning method is used to determine an online solution to the two player nonzero-sum game for a linear system without complete knowledge of the dynamics.

The synchronous policy iteration method in [38] is further generalized to solve the two-player zero-sum game problem in [39] and a multi-player nonzero-sum

game in [32] and [40] for nonlinear continuous-time systems with known dynamics. Furthermore, [75] presents a policy iteration method for an infinite-horizon two-player zero-sum Nash game with unknown nonlinear continuous-time dynamics.

Recent results also focus on the development of data-driven approximate dynamic programming methods for set-point regulation, trajectory tracking, and differential games to relax the persistence of excitation conditions. These methods are surveyed at the end of the next chapter.

## References

1. Werbos PJ (1992) Approximate dynamic programming for real-time control and neural modeling. In: White DA, Sorge DA (eds) *Handbook of intelligent control: neural, fuzzy, and adaptive approaches*, vol 15. Nostrand, New York, pp 493–525
2. Hopfield J (1984) Neurons with graded response have collective computational properties like those of two-state neurons. *Proc Nat Acad Sci USA* 81(10):3088
3. Kirk D (2004) *Optimal Control Theory: An Introduction*. Dover, Mineola, NY
4. Lewis FL, Vrabie D, Syrmos VL (2012) *Optimal Control*, 3rd edn. Wiley, Hoboken
5. Case J (1969) Toward a theory of many player differential games. *SIAM J Control* 7:179–197
6. Starr A, Ho CY (1969) Nonzero-sum differential games. *J Optim Theory App* 3(3):184–206
7. Starr A, Ho, (1969) Further properties of nonzero-sum differential games. *J Optim Theory App* 4:207–219
8. Friedman A (1971) *Differential games*. Wiley, Hoboken
9. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92
10. Xian B, Dawson DM, de Queiroz MS, Chen J (2004) A continuous asymptotic tracking control strategy for uncertain nonlinear systems. *IEEE Trans Autom Control* 49(7):1206–1211
11. Patre PM, MacKunis W, Kaiser K, Dixon WE (2008) Asymptotic tracking for uncertain dynamic systems via a multilayer neural network feedforward and RISE feedback control structure. *IEEE Trans Autom Control* 53(9):2180–2185
12. Filippov AF (1988) *Differential equations with discontinuous right-hand sides*. Kluwer Academic Publishers, Dordrecht
13. Kamalapurkar R, Rosenfeld JA, Klotz J, Downey RJ, Dixon WE (2014) Supporting lemmas for RISE-based control methods. [arXiv:1306.3432](https://arxiv.org/abs/1306.3432)
14. Vamvoudakis KG, Lewis FL (2010) Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica* 46(5):878–888
15. Sastry S, Bodson M (1989) *Adaptive control: stability, convergence, and robustness*. Prentice-Hall, Upper Saddle River
16. Panteley E, Loria A, Teel A (2001) Relaxed persistency of excitation for uniform asymptotic stability. *IEEE Trans Autom Control* 46(12):1874–1886
17. Loría A, Panteley E (2002) Uniform exponential stability of linear time-varying systems: revisited. *Syst Control Lett* 47(1):13–24
18. Khalil HK (2002) *Nonlinear systems*, 3rd edn. Prentice Hall, Upper Saddle River
19. Bertsekas D, Tsitsiklis J (1996) *Neuro-dynamic programming*. Athena Scientific, Belmont
20. Busoniu L, Babuska R, De Schutter B, Ernst D (2010) *Reinforcement learning and dynamic programming using function approximators*. CRC Press, Boca Raton
21. Sutton RS, Barto AG (1998) *Reinforcement learning: an introduction*. MIT Press, Cambridge
22. Kamalapurkar R, Dinh H, Bhasin S, Dixon WE (2015) Approximate optimal trajectory tracking for continuous-time nonlinear systems. *Automatica* 51:40–48

23. Zhang H, Wei Q, Luo Y (2008) A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy hdp iteration algorithm. *IEEE Trans Syst Man Cybern Part B Cybern* 38(4):937–942
24. Hornik K, Stinchcombe M, White H (1990) Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Netw* 3(5):551–560
25. Lewis FL, Selmic R, Campos J (2002) Neuro-fuzzy control of industrial systems with actuator nonlinearities. Society for Industrial and Applied Mathematics, Philadelphia
26. Ioannou P, Sun J (1996) Robust adaptive control. Prentice Hall, Upper Saddle River
27. Misovec KM (1999) Friction compensation using adaptive non-linear control with persistent excitation. *Int J Control* 72(5):457–479
28. Narendra K, Annaswamy A (1986) Robust adaptive control in the presence of bounded disturbances. *IEEE Trans Autom Control* 31(4):306–315
29. Rao AV, Benson DA, Darby CL, Patterson MA, Francolin C, Huntington GT (2010) Algorithm 902: GPOPS, A MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method. *ACM Trans Math Softw* 37(2):1–39
30. Johnson M, Kamalapurkar R, Bhasin S, Dixon WE (2015) Approximate n-player nonzero-sum game solution for an uncertain continuous nonlinear system. *IEEE Trans Neural Netw Learn Syst* 26(8):1645–1658
31. Basar T, Olsder GJ (1999) Dynamic noncooperative game theory. Classics in applied mathematics, 2nd edn. SIAM, Philadelphia
32. Vamvoudakis KG, Lewis FL (2011) Multi-player non-zero-sum games: online adaptive learning solution of coupled hamilton-jacobi equations. *Automatica* 47:1556–1569
33. Basar T, Bernhard P (2008)  $H^\infty$ -optimal control and related minimax design problems: a dynamic game approach, 2nd edn. Modern Birkhäuser Classics, Birkhäuser, Boston
34. Patre PM, Dixon WE, Makkar C, Mackunis W (2006) Asymptotic tracking for systems with structured and unstructured uncertainties. In: Proceedings of the IEEE conference on decision and control, San Diego, California, pp 441–446
35. Dixon WE, Behal A, Dawson DM, Nagarkatti S (2003) Nonlinear control of engineering systems: a lyapunov-based approach. Birkhauser, Boston
36. Krstic M, Kanellakopoulos I, Kokotovic PV (1995) Nonlinear and adaptive control design. Wiley, New York
37. Nevidistic V, Primbs JA (1996) Constrained nonlinear optimal control: a converse HJB approach. Technical report. CIT-CDS 96-021, California Institute of Technology, Pasadena, CA 91125
38. Vamvoudakis KG, Lewis FL (2009) Online synchronous policy iteration method for optimal control. In: Yu W (ed) Recent advances in intelligent control systems. Springer, Berlin, pp 357–374
39. Vamvoudakis KG, Lewis FL (2010) Online neural network solution of nonlinear two-player zero-sum games using synchronous policy iteration. In: Proceedings of the IEEE conference on decision and control
40. Zhang H, Cui L, Luo Y (2013) Near-optimal control for nonzero-sum differential games of continuous-time nonlinear systems using single-network ADP. *IEEE Trans Cybern* 43(1):206–216
41. Doya K (2000) Reinforcement learning in continuous time and space. *Neural Comput* 12(1):219–245
42. Chen Z, Jagannathan S (2008) Generalized Hamilton-Jacobi-Bellman formulation -based neural network control of affine nonlinear discrete-time systems. *IEEE Trans Neural Netw* 19(1):90–106
43. Dierks T, Thumati B, Jagannathan S (2009) Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. *Neural Netw* 22(5–6):851–860
44. Zhang H, Liu D, Luo Y, Wang D (2013) Adaptive dynamic programming for control algorithms and stability. Communications and control engineering, Springer, London
45. Liu D, Wei Q (2014) Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems. *IEEE Trans Neural Netw Learn Syst* 25(3):621–634

46. Modares H, Lewis FL, Naghibi-Sistani MB (2014) Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica* 50(1):193–202
47. Yang X, Liu D, Wang D (2014) Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints. *Int J Control* 87(3):553–566
48. Dierks T, Jagannathan S (2009) Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics. In: Proceedings of the IEEE conference on decision and control, Shanghai, CN, pp 6750–6755
49. Zhang H, Cui L, Zhang X, Luo Y (2011) Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method. *IEEE Trans Neural Netw* 22(12):2226–2236
50. Wei Q, Liu D (2013) Optimal tracking control scheme for discrete-time nonlinear systems with approximation errors. In: Guo C, Hou ZG, Zeng Z (eds) *Advances in neural networks - ISNN 2013*, vol 7952. Lecture notes in computer science. Springer, Berlin, pp 1–10
51. Kiumarsi B, Lewis FL, Modares H, Karimpour A, Naghibi-Sistani MB (2014) Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. *Automatica* 50(4):1167–1175
52. Qin C, Zhang H, Luo Y (2014) Online optimal tracking control of continuous-time linear systems with unknown dynamics by using adaptive dynamic programming. *Int J Control* 87(5):1000–1009
53. Murray J, Cox C, Lendaris G, Saeks R (2002) Adaptive dynamic programming. *IEEE Trans Syst Man Cybern Part C Appl Rev* 32(2):140–153
54. Beard R, Saridis G, Wen J (1997) Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation. *Automatica* 33:2159–2178
55. Abu-Khalaf M, Lewis FL (2005) Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach. *Automatica* 41(5):779–791
56. Vrabie D, Lewis FL (2009) Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Netw* 22(3):237–246
57. Wang K, Liu Y, Li L (2014) Visual servoing trajectory tracking of nonholonomic mobile robots without direct position measurement. *IEEE Trans Robot* 30(4):1026–1035
58. Wang D, Liu D, Zhang Q, Zhao D (2016) Data-based adaptive critic designs for nonlinear robust optimal control with uncertain dynamics. *IEEE Trans Syst Man Cybern Syst* 46(11):1544–1555
59. Li H, Liu D, Wang D (2014) Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics. *IEEE Trans Autom Sci Eng* 11(3):706–714
60. Dierks T, Jagannathan S (2010) Optimal control of affine nonlinear continuous-time systems. In: Proceedings of the American control conference, pp 1568–1573
61. Park YM, Choi MS, Lee KY (1996) An optimal tracking neuro-controller for nonlinear dynamic systems. *IEEE Trans Neural Netw* 7(5):1099–1110
62. Luo Y, Liang M (2011) Approximate optimal tracking control for a class of discrete-time non-affine systems based on GDHP algorithm. In: IWACI International Workshop on Advanced Computational Intelligence, pp 143–149
63. Wang D, Liu D, Wei Q (2012) Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach. *Neurocomputing* 78(1):14–22
64. Modares H, Lewis FL (2014) Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning. *Automatica* 50(7):1780–1792
65. Luo B, Liu D, Huang T, Wang D (2016) Model-free optimal tracking control via critic-only q-learning. *IEEE Trans Neural Netw Learn Syst* 27(10):2134–2144
66. Yang X, Liu D, Wei Q, Wang D (2016) Guaranteed cost neural tracking control for a class of uncertain nonlinear systems using adaptive dynamic programming. *Neurocomputing* 198:80–90
67. Zhao B, Liu D, Yang X, Li Y (2017) Observer-critic structure-based adaptive dynamic programming for decentralised tracking control of unknown large-scale nonlinear systems. *Int J Syst Sci* 48(9):1978–1989

68. Wang D, Liu D, Zhang Y, Li H (2018) Neural network robust tracking control with adaptive critic framework for uncertain nonlinear systems. *Neural Netw* 97:11–18
69. Vamvoudakis KG, Mojoodi A, Ferraz H (2017) Event-triggered optimal tracking control of nonlinear systems. *Int J Robust Nonlinear Control* 27(4):598–619
70. Wei Q, Zhang H (2008) A new approach to solve a class of continuous-time nonlinear quadratic zero-sum game using ADP. In: IEEE international conference on networking, sensing and control, pp 507–512
71. Zhang H, Wei Q, Liu D (2010) An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. *Automatica* 47:207–214
72. Zhang X, Zhang H, Luo Y, Dong M (2010) Iteration algorithm for solving the optimal strategies of a class of nonaffine nonlinear quadratic zero-sum games. In: Proceedings of the IEEE conference on decision and control, pp 1359–1364
73. Mellouk A (ed) (2011) Advances in reinforcement learning. InTech
74. Littman M (2001) Value-function reinforcement learning in markov games. *Cogn Syst Res* 2(1):55–66
75. Johnson M, Bhasin S, Dixon WE (2011) Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm. In: Proceedings of the IEEE conference on decision and control, pp 142–147

# Chapter 4

## Model-Based Reinforcement Learning for Approximate Optimal Control



### 4.1 Introduction

In reinforcement learning-based approximate online optimal control, the Hamilton–Jacobi–Bellman equation along with an estimate of the state derivative (cf. [1, 2]), or an integral form of the Hamilton–Jacobi–Bellman equation (cf. [3]) is utilized to approximately evaluate the Bellman error along the system trajectory. The Bellman error, evaluated at a point, provides an indirect measure of the quality of the estimated value function evaluated at that point. Therefore, the unknown value function parameters are updated based on evaluation of the Bellman error along the system trajectory. Such weight update strategies create two challenges for analyzing convergence. The system states need to satisfy the persistence of excitation condition, and the system trajectory needs to visit enough points in the state-space to generate a good approximation of the value function over the entire domain of operation. As in Chap. 3, these challenges are typically addressed in the related literature (cf. [2, 4–12]) by adding an exploration signal to the control input to ensure sufficient exploration of the domain of operation. However, no analytical methods exist to compute the appropriate exploration signal when the system dynamics are nonlinear.

The aforementioned challenges arise from the restriction that the Bellman error can only be evaluated along the system trajectories. In particular, the integral Bellman error is meaningful as a measure of the quality of the estimated value function only if it is evaluated along the system trajectories, and state derivative estimators can only generate numerical estimates of the state derivative along the system trajectories. Recently, [11] demonstrated that experience replay can be used to improve data efficiency in online approximate optimal control by reuse of recorded data. However, since the data needs to be recorded along the system trajectory, the system trajectory under the designed approximate optimal controller needs to provide enough excitation for learning. In general, such excitation is not available. As a result, the simulation results in [11] are generated using an added probing signal.

In this chapter and in results such as [13–22], a different approach is used to improve data efficiency by observing that if the system dynamics are known, the state derivative, and hence, the Bellman error can be evaluated at any desired point in the state-space. Unknown parameters in the value function can therefore be adjusted based on least-squares minimization of the Bellman error evaluated at any number of arbitrary points in the state-space. For example, in an infinite-horizon regulation problem, the Bellman error can be computed at points uniformly distributed in a neighborhood around the origin of the state-space. The results of this chapter indicate that convergence of the unknown parameters in the value function is guaranteed provided the selected points satisfy a rank condition. Since the Bellman error can be evaluated at any desired point in the state-space, sufficient exploration can be achieved by appropriately selecting the points to cover the domain of operation.

If the system dynamics are partially unknown, an approximation to the Bellman error can be evaluated at any desired point in the state-space based on an estimate of the system dynamics. If each new evaluation of the Bellman error along the system trajectory is interpreted as gaining experience via exploration, the use of a model to evaluate the Bellman error at an unexplored point in the state-space can be interpreted as a simulation of the experience. Learning based on simulation of experience has been investigated in results such as [23–28] for stochastic model-based reinforcement learning; however, these results solve the optimal control problem off-line in the sense that repeated learning trials need to be performed before the algorithm learns the controller, and system stability during the learning phase is not analyzed. This chapter and the results in [13–22] further the state of the art for non-linear control-affine plants with linearly parameterizable uncertainties in the drift dynamics by providing an online solution to deterministic infinite-horizon optimal regulation problems. In this chapter, a concurrent learning-based parameter estimator is developed to exponentially identify the unknown parameters in the system model, and the parameter estimates are used to implement simulation of experience by extrapolating the Bellman error.

In Sect. 4.4 (see also, [14, 22]), the model-based reinforcement learning method is extended to solve trajectory tracking problems for uncertain nonlinear systems. The technical challenges associated with the nonautonomous nature of the trajectory tracking problem are addressed in Chap. 3, where it is established that under a matching condition on the desired trajectory, the optimal trajectory tracking problem can be reformulated as a stationary optimal control problem. Since the value function associated with a stationary optimal control problem is time-invariant, it can be approximated using traditional function approximation techniques. The reformulation developed in Chap. 3 requires computation of the steady-state tracking controller, which depends on the system model; therefore, the development in Chap. 3 requires exact model knowledge. Obtaining an accurate estimate of the desired steady-state controller, and injecting the resulting estimation error in the stability analysis are the major technical challenges in extending the work in Chap. 3 to uncertain systems. In Sect. 4.4, an estimate of the steady-state controller is generated using concurrent learning-based system identifiers. The use of an estimate instead of the true steady-state controller results in additional approximation errors that can potentially cause

instability during the learning phase. This chapter analyzes the stability of the closed-loop system in the presence of the aforementioned additional approximation error. The error between the actual steady-state controller and its estimate is included in the stability analysis by examining the trajectories of the concatenated system under the implemented control signal. In addition to estimating the desired steady-state controller, the concurrent learning-based system identifier is also used to simulate experience by evaluating the Bellman error over unexplored areas of the state-space [14, 29, 30].

In Sect. 4.5 (see also, [16]), the model-based reinforcement learning method is extended to obtain an approximate feedback-Nash equilibrium solution to an infinite-horizon  $N$ -player nonzero-sum differential game online, without requiring persistence of excitation, for a nonlinear control-affine system with uncertain linearly parameterized drift dynamics. A system identifier is used to estimate the unknown parameters in the drift dynamics. The solutions to the coupled Hamilton–Jacobi equations and the corresponding feedback-Nash equilibrium policies are approximated using parametric universal function approximators. Based on estimates of the unknown drift parameters, estimates for the Bellman errors are evaluated at a set of pre-selected points in the state-space. The critic and the actor weights are updated using a concurrent learning-based least-squares approach to minimize the instantaneous Bellman errors and the Bellman errors evaluated at pre-selected points. Simultaneously, the unknown parameters in the drift dynamics are updated using a history stack of recorded data via a concurrent learning-based gradient descent approach. It is shown that under a condition milder than persistence of excitation, uniformly ultimately bounded convergence of the unknown drift parameters, the critic weights and the actor weights to their true values can be established. Simulation results demonstrate the effectiveness of the developed approximate solutions to infinite-horizon optimal regulation and tracking problems online for inherently unstable nonlinear systems with uncertain drift dynamics. The simulations also demonstrate that the developed method can be used to implement reinforcement learning without the addition of a probing signal.

## 4.2 Model-Based Reinforcement Learning

Consider the control-affine nonlinear dynamical system in (1.9) and recall the expression for the Bellman error in (2.3)

$$\delta(x, \hat{W}_c, \hat{W}_a) \triangleq \nabla_x \hat{V}(x, \hat{W}_c) \left( f(x) + g(x) \hat{u}(x, \hat{W}_a) \right) + r(x, \hat{u}(x, \hat{W}_a)).$$

To solve the optimal control problem, the critic aims to find a set of parameters  $\hat{W}_c$  and the actor aims to find a set of parameters  $\hat{W}_a$  that minimize the integral error  $E$ , introduced in (2.5). Computation of the error  $E$  requires evaluation of the Bellman error over the entire domain  $\mathcal{D}$ , which is generally infeasible. As a result, a derivative-

based evaluation of  $E$  along the system trajectories, denoted by  $E_t$  and introduced in (2.7), is utilized for learning in Chap. 3. Intuitively, the state trajectory,  $x$ , needs to visit as many points in the operating domain as possible for  $E_t$  to approximate  $E$  over an operating domain. This intuition is formalized by the fact that techniques in Chap. 3 require persistence of excitation to achieve convergence.

The persistence of excitation condition is relaxed in [11] to a finite excitation condition by using integral reinforcement learning along with experience replay, where each evaluation of the Bellman error along the system trajectory is interpreted as gained experience. These experiences are stored in a history stack and are repeatedly used in the learning algorithm to improve data efficiency. In this chapter, a different approach is used to circumvent the persistence of excitation condition. Given a model of the system and the current parameter estimates  $\hat{W}_c(t)$  and  $\hat{W}_a(t)$ , the Bellman error in (2.3) can be evaluated at any point  $x_i \in \mathbb{R}^n$ . That is, the critic can gain experience on how well the value function is estimated at any arbitrary point  $x_i$  in the state-space without actually visiting  $x_i$ . In other words, given a fixed state  $x_i$  and a corresponding planned action  $\hat{u}(x_i, \hat{W}_a)$ , the critic can use the dynamic model to simulate a visit to  $x_i$  by computing the state derivative at  $x_i$ . This results in simulated experience quantified by the Bellman error  $\delta_{ti}(t) = \delta(x_i, \hat{W}_c(t), \hat{W}_a(t))$ .

In the case where the drift dynamics,  $f$ , are uncertain, a parametric approximation  $\hat{f}(x, \hat{\theta})$  of the function  $f$ , where  $\hat{\theta}$  denotes the matrix of parameter estimates, can be utilized to approximate the Bellman error in (2.3) as

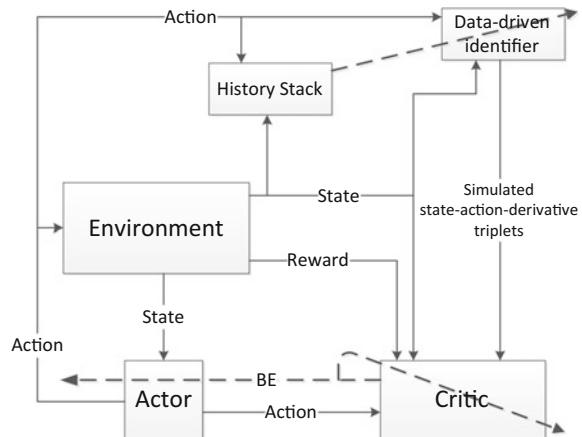
$$\hat{\delta}(x, \hat{W}_c, \hat{W}_a, \hat{\theta}) \triangleq \nabla_x \hat{V}(x, \hat{W}_c) (\hat{f}(x, \hat{\theta}) + g(x) \hat{u}(x, \hat{W}_a)) + r(x, \hat{u}(x, \hat{W}_a)). \quad (4.1)$$

Given current parameter estimates  $\hat{W}_c(t)$ ,  $\hat{W}_a(t)$ , and  $\hat{\theta}(t)$ , the approximate Bellman error in (4.1) can be evaluated at any point  $x_i \in \mathbb{R}^n$ . This results in simulated experience quantified by the Bellman error  $\hat{\delta}_{ti}(t) = \hat{\delta}(x_i, \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t))$ . The simulated experience can then be used along with gained experience by the critic to learn the value function. Motivation behind using simulated experience is that by selecting multiple (say  $N$ ) points, the error signal in (2.7) can be augmented to yield a heuristically better approximation  $\hat{E}_{ti}(t)$ , given by

$$\hat{E}_{ti}(t) \triangleq \int_0^t \left( \hat{\delta}_t^2(\tau) + \sum_{i=1}^N \hat{\delta}_{ti}^2(\tau) \right) d\tau,$$

for the desired error signal in (2.5). A block diagram of the simulation-based actor-critic-identifier architecture is presented in Fig. 4.1. For notational brevity, the dependence of all the functions on the system states and time is suppressed in the stability analysis subsections unless required for clarity of exposition.

**Fig. 4.1** Simulation-based actor-critic-identifier architecture. In addition to the state-action measurements, the critic also utilizes states, actions, and the corresponding state-derivatives to learn the value function



### 4.3 Online Approximate Regulation<sup>1</sup>

This section develops a data-driven implementation of model-based reinforcement learning to solve approximate optimal regulation problems online under a persistence of excitation-like rank condition. The development is based on the observation that, given a model of the system, reinforcement learning can be implemented by evaluating the Bellman error at any number of desired points in the state-space. In this result, a parametric system model is considered, and a concurrent learning-based parameter identifier is developed to compensate for uncertainty in the parameters. Uniformly ultimately bounded regulation of the system states to a neighborhood of the origin, and convergence of the developed policy to a neighborhood of the optimal policy are established using a Lyapunov-based analysis, and simulation results are presented to demonstrate the performance of the developed controller.

Online implementation of simulation of experience requires uniform online estimation of the function  $f$  using the parametric approximation  $\hat{f}(x, \hat{\theta})$  (i.e., the parameter estimates  $\hat{\theta}$  need to converge to their true values  $\theta$ ). In the following, a general Lyapunov-based characterization of a system identifier that achieves uniform approximation of  $f$  is developed based on recent ideas on data-driven parameter convergence in adaptive control (cf. [29–34]).

#### 4.3.1 System Identification

To facilitate online system identification, let  $f(x) = Y(x)\theta$  denote the linear parametrization of the function  $f$ , where  $Y : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times p_\theta}$  is the regression matrix,

<sup>1</sup>Parts of the text in this section are reproduced, with permission, from [18], ©2016, Elsevier.

and  $\theta \in \mathbb{R}^{p_\theta}$  is the vector of constant unknown parameters. Let  $\hat{\theta} \in \mathbb{R}^{p_\theta}$  be an estimate of the unknown parameter vector  $\theta$ . The following development assumes that an adaptive system identifier that satisfies conditions detailed in Assumption 4.1 is available. For completeness, a concurrent learning-based system identifier that satisfies Assumption 4.1 is presented in Appendix A.2.3.

**Assumption 4.1** A compact set  $\Theta \subset \mathbb{R}^{p_\theta}$  such that  $\theta \in \Theta$  is known a priori. The estimates  $\hat{\theta} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{p_\theta}$  are updated based on a switched update law of the form

$$\dot{\hat{\theta}}(t) = f_{\theta s}(\hat{\theta}(t), t), \quad (4.2)$$

$\hat{\theta}(t_0) = \hat{\theta}_0 \in \Theta$ , where  $s \in \mathbb{N}$  denotes the switching index and  $\{f_{\theta s} : \mathbb{R}^{p_\theta} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^{p_\theta}\}_{s \in \mathbb{N}}$  denotes a family of continuously differentiable functions. The dynamics of the parameter estimation error  $\tilde{\theta} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{p_\theta}$ , defined as  $\tilde{\theta}(t) \triangleq \theta - \hat{\theta}(t)$ , can be expressed as  $\dot{\tilde{\theta}}(t) = f_{\theta s}(\theta - \tilde{\theta}(t), t)$ . Furthermore, there exists a continuously differentiable function  $V_\theta : \mathbb{R}^{p_\theta} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  that satisfies

$$\underline{v}_\theta(\|\tilde{\theta}\|) \leq V_\theta(\tilde{\theta}, t) \leq \bar{v}_\theta(\|\tilde{\theta}\|), \quad (4.3)$$

$$\nabla_{\tilde{\theta}} V_\theta(\tilde{\theta}, t) (-f_{\theta s}(\theta - \tilde{\theta}, t)) + \nabla_t V_\theta(\tilde{\theta}, t) \leq -K \|\tilde{\theta}\|^2 + D \|\tilde{\theta}\|, \quad (4.4)$$

for all  $s \in \mathbb{N}$ ,  $t \in \mathbb{R}_{\geq t_0}$ , and  $\tilde{\theta} \in \mathbb{R}^{p_\theta}$ , where  $\underline{v}_\theta, \bar{v}_\theta : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions,  $K \in \mathbb{R}_{>0}$  is an adjustable parameter, and  $D \in \mathbb{R}_{>0}$  is a positive constant.

The subsequent analysis in Sect. 4.3.4 indicates that when a system identifier that satisfies Assumption 4.1 is employed to facilitate online optimal control, the ratio  $\frac{D}{K}$  needs to be sufficiently small to establish set-point regulation and convergence to optimality. Using an estimate  $\hat{\theta}$ , the Bellman error in (2.3) can be approximated by  $\hat{\delta} : \mathbb{R}^{n+2L+p} \rightarrow \mathbb{R}$  as

$$\hat{\delta}(x, \hat{W}_c, \hat{W}_a, \hat{\theta}) \triangleq \nabla_x \hat{V}(x, \hat{W}_c) (Y(x)\hat{\theta} + g(x)\hat{u}(x, \hat{W}_a)) + r(x, \hat{u}(x, \hat{W}_a)). \quad (4.5)$$

In the following, the approximate Bellman error in (4.5) is used to obtain an approximate solution to the Hamilton–Jacobi–Bellman equation in (1.14).

### 4.3.2 Value Function Approximation

Approximations to the optimal value function  $V^*$  and the optimal policy  $u^*$  are designed based on neural network-based representations. Given any compact set  $\chi \subset \mathbb{R}^n$  and a positive constant  $\bar{\epsilon} \in \mathbb{R}$ , the universal function approximation property of neural networks can be exploited to represent the optimal value function  $V^*$

as  $V^*(x) = W^T \sigma(x) + \epsilon(x)$ , for all  $x \in \chi$ , where  $W \in \mathbb{R}^L$  is the ideal weight matrix bounded above by a known positive constant  $\bar{W}$  in the sense that  $\|W\| \leq \bar{W}$ ,  $\sigma : \mathbb{R}^n \rightarrow \mathbb{R}^L$  is a continuously differentiable nonlinear activation function such that  $\sigma(0) = 0$  and  $\sigma'(0) = 0$ ,  $L \in \mathbb{N}$  is the number of neurons, and  $\epsilon : \mathbb{R}^n \rightarrow \mathbb{R}$  is the function reconstruction error such that  $\sup_{x \in \chi} |\epsilon(x)| \leq \bar{\epsilon}$  and  $\sup_{x \in \chi} |\nabla_x \epsilon(x)| \leq \bar{\epsilon}$ .

Based on the neural network representation of the value function, a neural network representation of the optimal controller is derived as  $u^*(x) = -\frac{1}{2} R^{-1} g^T(x) (\nabla_x \sigma^T(x) W + \nabla_x \epsilon^T(x))$ . The neural network approximations  $\hat{V} : \mathbb{R}^n \times \mathbb{R}^L \rightarrow \mathbb{R}$  and  $\hat{u} : \mathbb{R}^n \times \mathbb{R}^L \rightarrow \mathbb{R}^m$  are defined as

$$\begin{aligned}\hat{V}(x, \hat{W}_c) &\triangleq \hat{W}_c^T \sigma(x), \\ \hat{u}(x, \hat{W}_a) &\triangleq -\frac{1}{2} R^{-1} g^T(x) \nabla_x \sigma^T(x) \hat{W}_a,\end{aligned}\quad (4.6)$$

where  $\hat{W}_c \in \mathbb{R}^L$  and  $\hat{W}_a \in \mathbb{R}^L$  are the estimates of  $W$ . The use of two sets of weights to estimate the same set of ideal weights is motivated by the stability analysis and the fact that it enables a formulation of the Bellman error that is linear in the critic weight estimates  $\hat{W}_c$ , enabling a least-squares-based adaptive update law.

### 4.3.3 Simulation of Experience Via Bellman Error Extrapolation

In traditional reinforcement learning-based algorithms, the value function estimate and the policy estimate are updated based on observed data. The use of observed data to learn the value function naturally leads to a sufficient exploration condition which demands sufficient richness in the observed data. In stochastic systems, this is achieved using a randomized stationary policy (cf. [1, 35, 36]), whereas in deterministic systems, a probing noise is added to the derived control law (cf. [2, 6, 37–39]).

The technique developed in this result implements simulation of experience in a model-based reinforcement learning scheme by using  $\hat{Y}\hat{\theta}$  as an estimate of the uncertain drift dynamics  $f$  to extrapolate the approximate Bellman error to a pre-defined set of points  $\{x_i \in \mathbb{R}^n \mid i = 1, \dots, N\}$  in the state-space. In the following,  $\hat{\delta}_t : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  denotes the approximate Bellman error in (4.5) evaluated along the trajectories of (1.9), (4.2), (4.7), and (4.9) as  $\hat{\delta}_t(t) \triangleq \hat{\delta}(x(t), \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t))$  and  $\hat{\delta}_{ti} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  denotes the approximate Bellman error extrapolated to the points  $\{x_i \in \mathbb{R}^n \mid i = 1, \dots, N\}$  along the trajectories of (4.2), (4.7), and (4.9) as  $\hat{\delta}_{ti} \triangleq \hat{\delta}(x_i, \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t))$ .

A least-squares update law for the critic weights is designed based on the subsequent stability analysis as

$$\dot{\hat{W}}_c(t) = -\eta_{c1}\Gamma \frac{\omega(t)}{\rho(t)}\hat{\delta}_t(t) - \frac{\eta_{c2}}{N}\Gamma \sum_{i=1}^N \frac{\omega_i(t)}{\rho_i(t)}\hat{\delta}_{ti}(t), \quad (4.7)$$

$$\dot{\Gamma}(t) = \left( \beta\Gamma(t) - \eta_{c1} \frac{\Gamma(t)\omega(t)\omega(t)^T\Gamma(t)}{\rho^2(t)} \right) \mathbf{1}_{\{\|\Gamma\| \leq \bar{\Gamma}\}}, \quad (4.8)$$

where  $\Gamma : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L \times L}$  is a time-varying least-squares gain matrix,  $\|\Gamma(t_0)\| \leq \bar{\Gamma}$ ,  $\omega(t) \triangleq \nabla_x \sigma(x(t)) \left( Y(x(t))\hat{\theta}(t) + g(x(t))\hat{u}(x(t), \hat{W}_a(t)) \right)$ ,  $\omega_i(t) \triangleq \nabla_x \sigma(x_i) \left( Y(x_i)\hat{\theta}(t) + g(x_i)\hat{u}(x_i, \hat{W}_a(t)) \right)$ ,  $\rho(t) \triangleq 1 + v\omega^T(t)\Gamma(t)\omega(t)$ , and  $\rho_i(t) \triangleq 1 + v\omega_i^T(t)\Gamma(t)\omega_i(t)$ ,  $v \in \mathbb{R}$  is a constant positive normalization gain,  $\bar{\Gamma} > 0 \in \mathbb{R}$  is a saturation constant,  $\beta > 0 \in \mathbb{R}$  is a constant forgetting factor, and  $\eta_{c1}, \eta_{c2} > 0 \in \mathbb{R}$  are constant adaptation gains.

The actor weights are updated based on the subsequent stability analysis as

$$\begin{aligned} \dot{\hat{W}}_a(t) = & -\eta_{a1} \left( \hat{W}_a(t) - \hat{W}_c(t) \right) - \eta_{a2} \hat{W}_a(t) + \frac{\eta_{c1} G_\sigma^T(t) \hat{W}_a(t) \omega^T(t)}{4\rho(t)} \hat{W}_c(t) \\ & + \sum_{i=1}^N \frac{\eta_{c2} G_{\sigma i}^T \hat{W}_a(t) \omega_i^T(t)}{4N\rho_i(t)} \hat{W}_c(t), \end{aligned} \quad (4.9)$$

where  $\eta_{a1}, \eta_{a2} \in \mathbb{R}$  are positive constant adaptation gains,  $G_\sigma(t) \triangleq \nabla_x \sigma(x(t))g(x(t))R^{-1}g^T(x(t))\nabla_x \sigma^T(x(t))$ ,  $G_{\sigma i} \triangleq \sigma'_i g_i R^{-1} g_i^T \sigma'^T_i \in \mathbb{R}^{L \times L}$ , where  $g_i \triangleq g(x_i)$  and  $\sigma'_i \triangleq \nabla_x \sigma(x_i)$ .

The update law in (4.7) ensures that the adaptation gain matrix is bounded such that

$$\underline{\Gamma} \leq \|\Gamma(t)\| \leq \bar{\Gamma}, \quad \forall t \in \mathbb{R}_{\geq t_0}. \quad (4.10)$$

Using the weight estimates  $\hat{W}_a$ , the controller for the system in (1.9) is designed as

$$u(t) = \hat{u}(x(t), \hat{W}_a(t)). \quad (4.11)$$

The following rank condition facilitates the subsequent stability analysis.

**Assumption 4.2** There exists a finite set of fixed points  $\{x_i \in \mathbb{R}^n \mid i = 1, \dots, N\}$  such that  $\forall t \in \mathbb{R}_{\geq t_0}$

$$0 < \underline{c} \triangleq \frac{1}{N} \left( \inf_{t \in \mathbb{R}_{\geq t_0}} \left( \lambda_{\min} \left\{ \sum_{i=1}^N \frac{\omega_i(t)\omega_i^T(t)}{\rho_i(t)} \right\} \right) \right). \quad (4.12)$$

Since the rank condition in (4.12) depends on the estimates  $\hat{\theta}$  and  $\hat{W}_a$ , it is generally impossible to guarantee a priori. However, unlike the persistence of excitation condition in previous results such as [2, 6, 37–39], the condition in (4.12) can be verified online at each time  $t$ . Furthermore, the condition in (4.12) can be heuristically met by collecting redundant data (i.e., by selecting more points than the number of neurons by choosing  $N \gg L$ ).

The update law in (4.7) is fundamentally different from the concurrent learning adaptive update in results such as [30, 31] in the sense that the points  $\{x_i \in \mathbb{R}^n \mid i = 1, \dots, N\}$  are selected a priori based on information about the desired behavior of the system. Given the system dynamics, or an estimate of the system dynamics, the approximate Bellman error can be extrapolated to any desired point in the state-space, whereas the prediction error, which is used as a metric in adaptive control, can only be evaluated at observed data points along the state trajectory.

#### 4.3.4 Stability Analysis

To facilitate the subsequent stability analysis, the approximate Bellman error is expressed in terms of the weight estimation errors  $\tilde{W}_c \triangleq W - \hat{W}_c$  and  $\tilde{W}_a \triangleq W - \hat{W}_a$ . Subtracting (1.16) from (4.5), an unmeasurable form of the instantaneous Bellman error can be expressed as

$$\hat{\delta}_t = -\omega^T \tilde{W}_c - W^T \nabla_x \sigma Y \tilde{\theta} + \frac{1}{4} \tilde{W}_a^T G_\sigma \tilde{W}_a + \frac{1}{4} G_\epsilon - \nabla_x \epsilon f + \frac{1}{2} W^T \nabla_x \sigma G \nabla_x \epsilon^T, \quad (4.13)$$

where  $G \triangleq g R^{-1} g^T \in \mathbb{R}^{n \times n}$  and  $G_\epsilon \triangleq \nabla_x \epsilon G \nabla_x \epsilon^T \in \mathbb{R}$ . Similarly, the approximate Bellman error evaluated at the sampled states  $\{x_i \mid i = 1, \dots, N\}$  can be expressed as

$$\hat{\delta}_{ti} = -\omega_i^T \tilde{W}_c + \frac{1}{4} \tilde{W}_a^T G_{\sigma_i} \tilde{W}_a - W^T \sigma_i' Y_i \tilde{\theta} + \Delta_i, \quad (4.14)$$

where  $Y_i = Y(x_i)$ ,  $\epsilon'_i = \nabla_x \epsilon(x_i)$ ,  $f_i = f(x_i)$ ,  $G_i \triangleq g_i R^{-1} g_i^T \in \mathbb{R}^{n \times n}$ ,  $G_{\epsilon_i} \triangleq \epsilon'_i G_i \epsilon'^T \in \mathbb{R}$ , and  $\Delta_i \triangleq \frac{1}{2} W^T \sigma_i' G_i \epsilon'^T + \frac{1}{4} G_{\epsilon_i} - \epsilon'_i f_i \in \mathbb{R}$  is a constant.

On any compact set  $\chi \subset \mathbb{R}^n$  the function  $Y$  is Lipschitz continuous, and hence, there exists a positive constant  $L_Y \in \mathbb{R}$  such that

$$\|Y(x)\| \leq L_Y \|x\|, \forall x \in \chi. \quad (4.15)$$

In (4.15), the Lipschitz property is exploited for clarity of exposition. The bound in (4.15) can be easily generalized to  $\|Y(x)\| \leq L_Y(\|x\|) \|x\|$ , where  $L_Y : \mathbb{R} \rightarrow \mathbb{R}$  is a positive, non-decreasing, and radially unbounded function.

Using (4.10), the normalized regressor  $\frac{\omega}{\rho}$  can be bounded as

$$\sup_{t \in \mathbb{R}_{\geq t_0}} \left\| \frac{\omega}{\rho} \right\| \leq \frac{1}{2\sqrt{v\underline{\Gamma}}}. \quad (4.16)$$

For brevity of notation the following positive constants are defined:

$$\vartheta_1 \triangleq \frac{\eta_{c1} L_Y \|\theta\| \bar{\epsilon}}{4\sqrt{v\underline{\Gamma}}}, \quad \vartheta_2 \triangleq \sum_{i=1}^N \left( \frac{\eta_{c2} \|\sigma'_i Y_i\| \bar{W}}{4N\sqrt{v\underline{\Gamma}}} \right),$$

$$\vartheta_3 \triangleq \frac{L_Y \eta_{c1} \bar{W} \|\nabla_x \sigma\|}{4\sqrt{v\underline{\Gamma}}}, \quad \vartheta_4 \triangleq \left\| \frac{1}{4} G_\epsilon \right\|,$$

$$\vartheta_5 \triangleq \frac{\eta_{c1} \overline{\|2W^T \nabla_x \sigma G \nabla_x \sigma^T + G_\epsilon\|}}{8\sqrt{v\underline{\Gamma}}} + \left\| \sum_{i=1}^N \frac{\eta_{c2} \omega_i \Delta_i}{N\rho_i} \right\|,$$

$$\vartheta_6 \triangleq \left\| \frac{1}{2} W^T G_\sigma + \frac{1}{2} \nabla_x \epsilon G^T \nabla_x \sigma^T \right\| + \vartheta_7 \bar{W}^2 + \eta_{a2} \bar{W},$$

$$\vartheta_7 \triangleq \frac{\eta_{c1} \|G_\sigma\|}{8\sqrt{v\underline{\Gamma}}} + \sum_{i=1}^N \left( \frac{\eta_{c2} \|G_{\sigma i}\|}{8N\sqrt{v\underline{\Gamma}}} \right), \quad \underline{q} \triangleq \lambda_{\min}\{Q\},$$

$$\nu_l = \frac{1}{2} \min \left( \frac{\underline{q}}{2}, \frac{\eta_{c2} c}{3}, \frac{\eta_{a1} + 2\eta_{a2}}{6}, \frac{K}{4} \right),$$

$$\iota = \frac{3\vartheta_5^2}{4\eta_{c2} \underline{c}} + \frac{3\vartheta_6^2}{2(\eta_{a1} + 2\eta_{a2})} + \frac{D^2}{2K} + \vartheta_4, \quad (4.17)$$

where  $\overline{(\cdot)} \triangleq \sup_{x \in \chi} (\cdot) : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ .

Let  $Z : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{n+2L+p}$  be defined as  $Z(t) \triangleq [x^T(t), \tilde{W}_c^T(t), \tilde{W}_a^T(t), \tilde{\theta}^T(t)]^T$ , where  $x(\cdot)$ ,  $\tilde{W}_c(\cdot)$ ,  $\tilde{W}_a(\cdot)$ , and  $\tilde{\theta}(\cdot)$  denote the solutions of the differential equations in (1.9), (4.2), (4.7), and (4.9), respectively, with appropriate initial conditions. The sufficient conditions for ultimate boundedness of  $Z(\cdot)$  are derived based on the subsequent stability analysis as

$$\begin{aligned}
\frac{\eta_{a1} + 2\eta_{a2}}{6} &> \vartheta_7 \bar{W} \left( \frac{2\zeta_2 + 1}{2\zeta_2} \right), \\
\frac{K}{4} &> \frac{\vartheta_2 + \zeta_1 \zeta_3 \vartheta_3 \bar{Z}}{\zeta_1}, \\
\frac{\eta_{c2}}{3} &> \frac{\zeta_2 \vartheta_7 \bar{W} + \eta_{a1} + 2(\vartheta_1 + \zeta_1 \vartheta_2 + (\vartheta_3/\zeta_3) \bar{Z})}{2\underline{c}}, \\
\frac{q}{2} &> \vartheta_1,
\end{aligned} \tag{4.18}$$

where  $\bar{Z} \triangleq \underline{v}^{-1} \left( \bar{v} \left( \max \left( \|Z(t_0)\|, \sqrt{\frac{\iota}{v_l}} \right) \right) \right)$ ,  $\zeta_1, \zeta_2, \zeta_3 \in \mathbb{R}$  are known positive adjustable constants, and  $\underline{v}$  and  $\bar{v}$  are subsequently defined class  $\mathcal{K}$  functions. The Lipschitz constants in (4.15) and the neural network function approximation errors depend on the underlying compact set; hence, given a bound on the initial condition  $Z(t_0)$  for the concatenated state  $Z(\cdot)$ , a compact set that contains the concatenated state trajectory needs to be established before adaptation gains satisfying the conditions in (4.18) can be selected. Based on the subsequent stability analysis, an algorithm to compute the required compact set, denoted by  $\mathcal{Z} \subset \mathbb{R}^{2n+2L+p}$ , is developed in Appendix A.2.1. Since the constants  $\iota$  and  $v_l$  depend on  $L_Y$  only through the products  $L_Y \bar{\epsilon}$  and  $\frac{L_Y}{\zeta_3}$ , Algorithm A.2 ensures that

$$\sqrt{\frac{\iota}{v_l}} \leq \frac{1}{2} \text{diam}(\mathcal{Z}), \tag{4.19}$$

where  $\text{diam}(\mathcal{Z})$  denotes the diameter of the set  $\mathcal{Z}$  defined as  $\text{diam}(\mathcal{Z}) \triangleq \sup \{ \|x - y\| \mid x, y \in \mathcal{Z}\}$ . The main result of this section can now be stated as follows.

**Theorem 4.3** *Provided Assumptions 4.1 and 4.2 hold and gains  $q$ ,  $\eta_{c2}$ ,  $\eta_{a2}$ , and  $K$  are selected large enough using Algorithm A.2, the controller in (4.11) along with the adaptive update laws in (4.7) and (4.9) ensure that the  $x(\cdot)$ ,  $\tilde{W}_c(\cdot)$ , and  $\tilde{W}_a(\cdot)$  are uniformly ultimately bounded.*

*Proof* Let  $V_L : \mathbb{R}^{n+2L+p} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  be a continuously differentiable positive definite candidate Lyapunov function defined as

$$V_L(Z, t) \triangleq V^*(x) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1}(t) \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a + V_\theta(\tilde{\theta}, t), \tag{4.20}$$

where  $V^*$  is the optimal value function,  $V_\theta$  was introduced in Assumption 4.1. Using the fact that  $V^*$  is positive definite, (4.3), (4.10) and [40, Lemma 4.3] yield

$$\underline{v}(\|Z\|) \leq V_L(Z, t) \leq \bar{v}(\|Z\|), \tag{4.21}$$

for all  $t \in \mathbb{R}_{\geq t_0}$  and for all  $Z \in \mathbb{R}^{n+2L+p}$ , where  $\underline{v}, \bar{v} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions.

Provided the gains are selected according to Algorithm A.2, substituting for the approximate Bellman errors from (4.13) and (4.14), using the bounds in (4.15) and (4.16), and using Young's inequality, the time derivative of (4.20) evaluated along the trajectory  $Z(\cdot)$  can be upper-bounded as

$$\nabla_Z V_L(Z, t) h(Z, t) + \nabla_t V_L(Z, t) \leq -v_l \|Z\|^2, \quad (4.22)$$

for all  $\|Z\| \geq \sqrt{\frac{L}{v_l}} > 0$ ,  $Z \in \mathcal{Z}$  and  $t \in \mathbb{R}_{\geq t_0}$ , where  $h : \mathbb{R}^{n+2L+p} \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{n+2L+p}$  is a concatenation of the vector fields in (1.9), (4.2), (4.7), and (4.9). Since  $V_\theta$  is a common Lyapunov function for the switched subsystem in (4.2), and the terms introduced by the update law (4.8) do not contribute to the bound in (4.22),  $V_L$  is a common Lyapunov function for the complete error system.

Using (4.19), (4.21), and (4.22), [40, Theorem 4.18] can be invoked to conclude that  $Z(\cdot)$  is uniformly ultimately bounded in the sense that  $\limsup_{t \rightarrow \infty} \|Z(t)\| \leq \underline{v}^{-1} \left( \bar{v} \left( \sqrt{\frac{L}{v_l}} \right) \right)$ . Furthermore, the concatenated state trajectories are bounded such that  $\|Z(t)\| \leq \bar{Z}, \forall t \in \mathbb{R}_{\geq t_0}$ . Since the estimates  $\hat{W}_a(\cdot)$  approximate the ideal weights  $W$ , the policy  $\hat{u}$  approximates the optimal policy  $u^*$ .  $\square$

### 4.3.5 Simulation

This section presents two simulations to demonstrate the performance and the applicability of the developed technique. First, the performance of an approximate solution to an optimal control problem that has a known analytical solution. Based on the known solution, an exact polynomial basis is used for value function approximation. The second simulation demonstrates the applicability of the developed technique in the case where the analytical solution, and hence, an exact basis for value function approximation is not known. In this case, since the optimal solution is unknown, the optimal trajectories obtained using the developed technique are compared with optimal trajectories obtained through numerical optimal control techniques.

#### Problem With a Known Basis

The performance of the developed controller is demonstrated by simulating a non-linear control-affine system with a two dimensional state  $x = [x_1, x_2]^T$ . The system dynamics are described by (1.9), where

$$f = \begin{bmatrix} x_1 & x_2 & 0 & 0 \\ 0 & 0 & x_1 & x_2 (1 - (\cos(2x_1) + 2)^2) \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix},$$

$$g = [0 \ (\cos(2x_1) + 2)^T]^T. \quad (4.23)$$

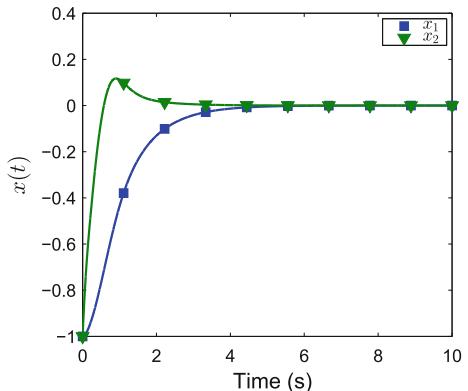
In (4.23),  $a, b, c, d \in \mathbb{R}$  are positive unknown parameters. The parameters are selected as  $a = -1$ ,  $b = 1$ ,  $c = -0.5$ , and  $d = -0.5$ . The control objective is to minimize the cost in (1.10) while regulating the system state to the origin. The origin is an unstable equilibrium point of the unforced system  $\dot{x} = f(x)$ . The weighting matrices in the cost function are selected as  $Q = I_2$  and  $R = 1$ . The optimal value function and optimal control for the system in (4.23) are given by  $V^*(x) = \frac{1}{2}x_1^2 + x_2^2$  and  $u^*(x) = -(\cos(2x_1) + 2)x_2$ , respectively (cf. [6]).

Thirty data points are recorded using a singular value maximizing algorithm (cf. [31]) for the concurrent learning-based adaptive update law in (A.18). The state derivative at each recorded data point is computed using a fifth order Savitzky–Golay smoothing filter (cf. [41]).

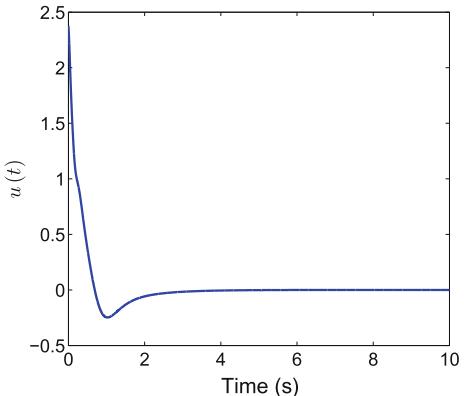
The basis function  $\sigma : \mathbb{R}^2 \rightarrow \mathbb{R}^3$  for value function approximation is selected as  $\sigma = [x_1^2, x_1x_2, x_2^2]$ . Based on the analytical solution, the ideal weights are  $W = [0.5, 0, 1]^T$ . The data points for the concurrent learning-based update law in (4.7) are selected to be on a  $5 \times 5$  grid around the origin. The learning gains are selected as  $\eta_{c1} = 1$ ,  $\eta_{c2} = 15$ ,  $\eta_{a1} = 100$ ,  $\eta_{a2} = 0.1$ , and  $v = 0.005$  and gains for the system identifier developed in Appendix A.2.3 are selected as  $k_x = 10I_2$ ,  $\Gamma_\theta = 20I_4$ , and  $k_\theta = 30$ . The actor and the critic weight estimates are initialized using a stabilizing set of initial weights as  $\hat{W}_c(0) = \hat{W}_a(0) = [1, 1, 1]^T$  and the least-squares gain is initialized as  $\Gamma(0) = 100I_3$ . The initial condition for the system state is selected as  $x(0) = [-1, -1]^T$ , the state estimates  $\hat{x}$  are initialized to be zero, the parameter estimates  $\hat{\theta}$  are initialized to be one, and the data stack for concurrent learning is recorded online.

Figures 4.2, 4.3, 4.4, 4.5 and 4.6 demonstrates that the system state is regulated to the origin, the unknown parameters in the drift dynamics are identified, and the value function and the actor weights converge to their true values. Furthermore, unlike previous results, a probing signal to ensure persistence of excitation is not required. Figures 4.7 and 4.8 demonstrate the satisfaction of Assumptions 4.2 and A.2, respectively.

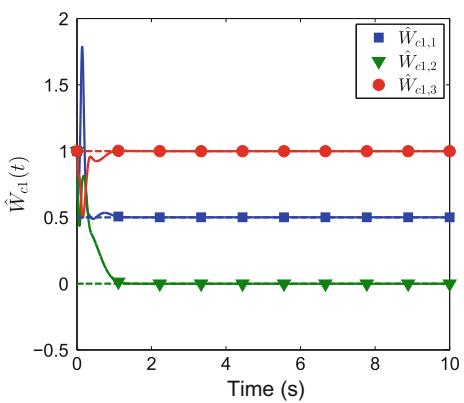
**Fig. 4.2** System state trajectories generated using the developed technique (reproduced with permission from [18], ©2016, Elsevier)



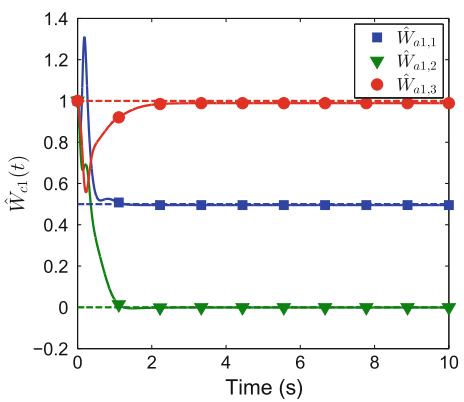
**Fig. 4.3** Control trajectory generated using the developed technique (reproduced with permission from [18], ©2016, Elsevier)



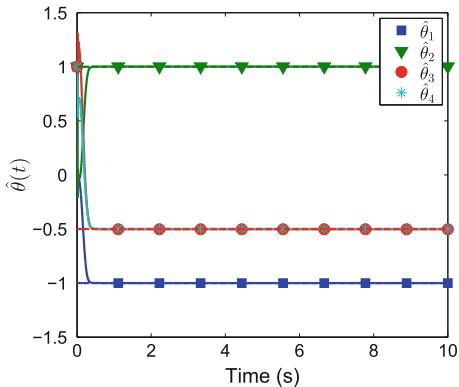
**Fig. 4.4** Critic weight estimates generated using the developed technique, and compared to the analytical solution (reproduced with permission from [18], ©2016, Elsevier)



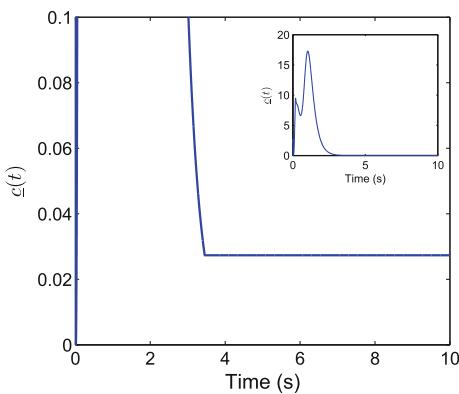
**Fig. 4.5** Actor weight estimates generated using the developed technique, and compared to the analytical solution (reproduced with permission from [18], ©2016, Elsevier)



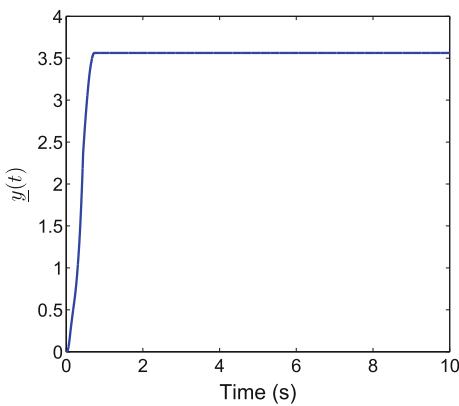
**Fig. 4.6** Estimates of the unknown plant parameters generated using the developed technique, and compared to the ideal values (reproduced with permission from [18], ©2016, Elsevier)



**Fig. 4.7** Satisfaction of Assumptions 4.2 and A.2 for the simulation with known basis (reproduced with permission from [18], ©2016, Elsevier)



**Fig. 4.8** Satisfaction of Assumptions 4.2 and A.2 for the simulation with known basis (reproduced with permission from [18], ©2016, Elsevier)



### Problem with an Unknown Basis

To demonstrate the applicability of the developed controller, a nonlinear control-affine system with a four dimensional state  $x = [x_1, x_2, x_3, x_4]^T$  is simulated. The system dynamics are described by

$$\begin{aligned} f &= \begin{bmatrix} x_3 \\ x_4 \\ -M^{-1}V_m \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} \end{bmatrix} + \begin{bmatrix} 0, 0, 0, 0 \\ 0, 0, 0, 0 \\ [M^{-1}, M^{-1}]D \end{bmatrix} \begin{bmatrix} f_{d1} \\ f_{d2} \\ f_{s1} \\ f_{s2} \end{bmatrix}, \\ g &= \left[ [0, 0]^T, [0, 0]^T, (M^{-1})^T \right]^T. \end{aligned} \quad (4.24)$$

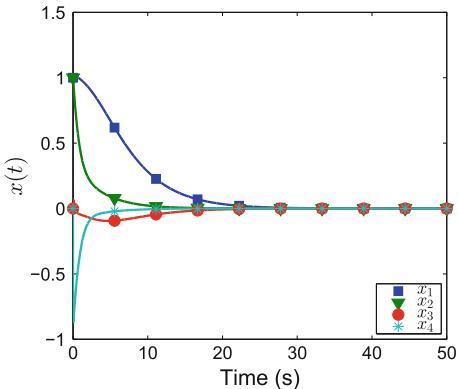
In (4.24),  $D \triangleq \text{diag}[x_3, x_4, \tanh(x_3), \tanh(x_4)]$  and the matrices  $M, V_m, F_d, F_s \in \mathbb{R}^{2 \times 2}$  are defined as  $M \triangleq \begin{bmatrix} p_1 + 2p_3c_2, p_2 + p_3c_2 \\ p_2 + p_3c_2, p_2 \end{bmatrix}$ ,  $F_d \triangleq \begin{bmatrix} f_{d1}, 0 \\ 0, f_{d2} \end{bmatrix}$ ,  $V_m \triangleq \begin{bmatrix} -p_3s_2x_4, -p_3s_2(x_3 + x_4) \\ p_3s_2x_3, 0 \end{bmatrix}$ , and  $F_s \triangleq \begin{bmatrix} f_{s1} \tanh(x_3), 0 \\ 0, f_{s2} \tanh(x_3) \end{bmatrix}$ , where  $c_2 = \cos(x_2)$ ,  $s_2 = \sin(x_2)$ ,  $p_1 = 3.473$ ,  $p_2 = 0.196$ , and  $p_3 = 0.242$ . The positive constants  $f_{d1}, f_{d2}, f_{s1}, f_{s2} \in \mathbb{R}$  are the unknown parameters. The parameters are selected as  $f_{d1} = 5.3$ ,  $f_{d2} = 1.1$ ,  $f_{s1} = 8.45$ , and  $f_{s2} = 2.35$ . The control objective is to minimize the cost in (1.10) with  $Q = \text{diag}([10, 10, 1, 1])$  and  $R = \text{diag}([1, 1])$  while regulating the system state to the origin. The origin is a marginally stable equilibrium point of the unforced system  $\dot{x} = f(x)$ .

The basis function  $\sigma : \mathbb{R}^4 \rightarrow \mathbb{R}^{10}$  for value function approximation is selected as  $\sigma(x) = [x_1x_3, x_2x_4, x_3x_2, x_4x_1, x_1x_2, x_4x_3, x_1^2, x_2^2, x_3^2, x_4^2]$ . The data points for the concurrent learning-based update law in (4.7) are selected to be on a  $3 \times 3 \times 3 \times 3$  grid around the origin, and the actor weights are updated using a projection-based update law. The learning gains are selected as  $\eta_{c1} = 1$ ,  $\eta_{c2} = 30$ ,  $\eta_{a1} = 0.1$ , and  $\nu = 0.0005$ . The gains for the system identifier developed in Appendix A.2.3 are selected as  $k_x = 10I_4$ ,  $\Gamma_\theta = \text{diag}([90, 50, 160, 50])$ , and  $k_\theta = 1.1$ . The least-squares gain is initialized as  $\Gamma(0) = 1000I_{10}$  and the policy and the critic weight estimates are initialized as  $\hat{W}_c(0) = \hat{W}_a(0) = [5, 5, 0, 0, 0, 0, 25, 0, 2, 2]^T$ . The initial condition for the system state is selected as  $x(0) = [1, 1, 0, 0]^T$ , the state estimates  $\hat{x}$  are initialized to be zero, the parameter estimates  $\hat{\theta}$  are initialized to be one, and the data stack for concurrent learning is recorded online.

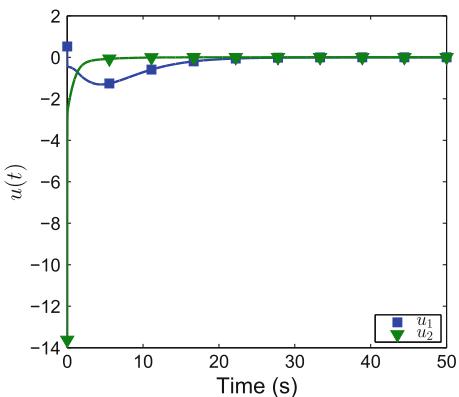
Figures 4.9, 4.10, 4.11, 4.12, 4.13 and 4.14 demonstrates that the system state is regulated to the origin, the unknown parameters in the drift dynamics are identified, and the value function and the actor weights converge. Figures 4.15 and 4.16 demonstrate the satisfaction of Assumptions 4.2 and A.2, respectively. The value function and the actor weights converge to the following values.

$$\hat{W}_c^* = \hat{W}_a^* = [24.7, 1.19, 2.25, 2.67, 1.18, 0.93, 44.34, 11.31, 3.81, 0.10]^T. \quad (4.25)$$

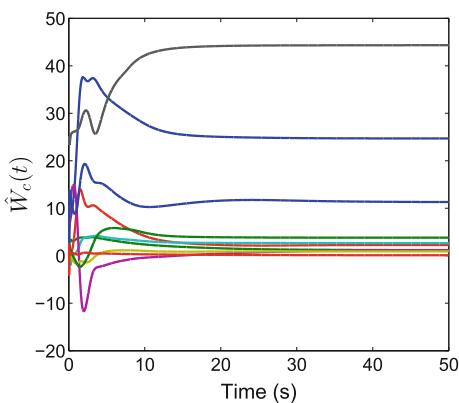
**Fig. 4.9** State trajectories generated using the developed technique (reproduced with permission from [18], ©2016, Elsevier)



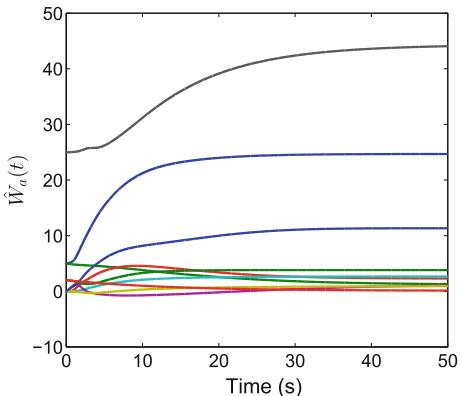
**Fig. 4.10** Control trajectories generated using the developed technique (reproduced with permission from [18], ©2016, Elsevier)



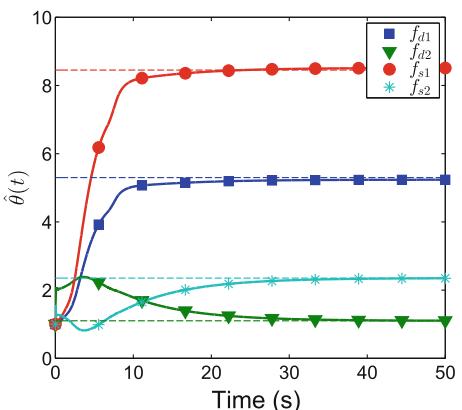
**Fig. 4.11** Critic weights generated using the developed technique (reproduced with permission from [18], ©2016, Elsevier)



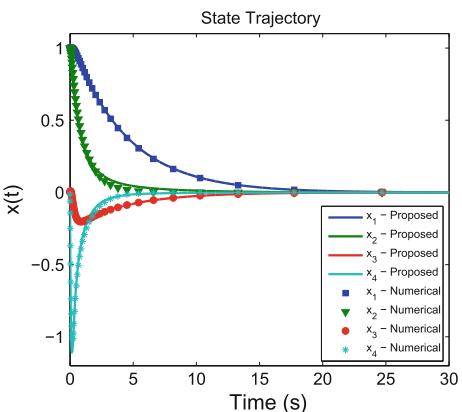
**Fig. 4.12** Actor weights generated using the developed technique (reproduced with permission from [18], ©2016, Elsevier)



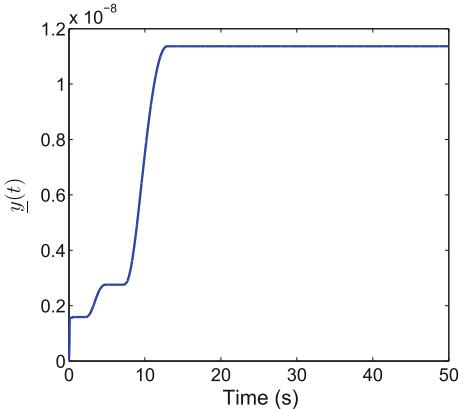
**Fig. 4.13** Drift parameter estimates generated using the developed technique compared to the actual drift parameters represented by dashed lines (reproduced with permission from [18], ©2016, Elsevier)



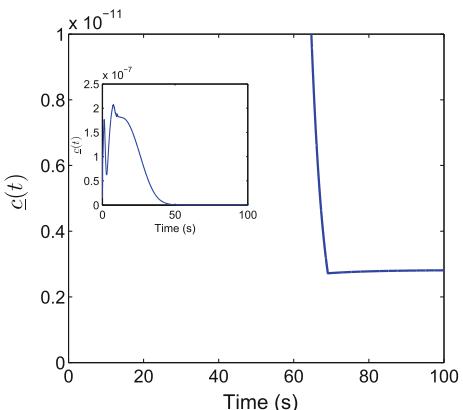
**Fig. 4.14** State trajectories generated using feedback policy  $u^*(x)$  compared to a numerical optimal solution (reproduced with permission from [18], ©2016, Elsevier)



**Fig. 4.15** Satisfaction of Assumption A.2 for the simulation with unknown basis (reproduced with permission from [18], ©2016, Elsevier)

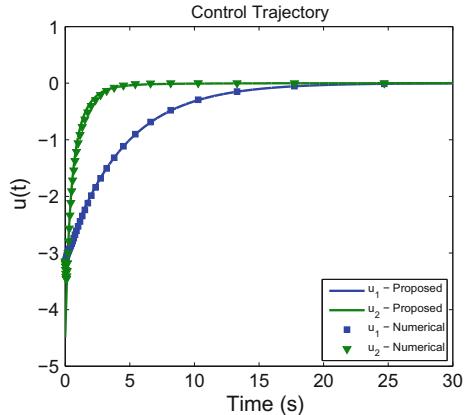


**Fig. 4.16** Satisfaction of Assumption 4.2 for the simulation with unknown basis (reproduced with permission from [18], ©2016, Elsevier)



Since the true values of the critic weights are unknown, the weights in (4.25) can not be compared to their true values. However, a measure of proximity of the weights in (4.25) to the ideal weights  $W$  can be obtained by comparing the system trajectories resulting from applying the feedback control policy  $\hat{u}^*(x) = -\frac{1}{2}R^{-1}g^T(x)\nabla_x\sigma^T(x)\hat{W}_a^*$  to the system, against numerically computed optimal system trajectories. Figures 4.14 and 4.17 indicate that the weights in (4.25) generate state and control trajectories that closely match the numerically computed optimal trajectories. The numerical optimal solution is obtained using an infinite-horizon Gauss-pseudospectral method (cf. [42]) using 45 collocation points.

**Fig. 4.17** Control trajectories generated using feedback policy  $\hat{u}^*(x)$  compared to a numerical optimal solution (reproduced with permission from [18], ©2016, Elsevier)



## 4.4 Extension to Trajectory Tracking<sup>2</sup>

This section provides an approximate online adaptive solution to the infinite-horizon optimal tracking problem for control-affine continuous-time nonlinear systems with unknown drift dynamics. To relax the persistence of excitation condition, model-based reinforcement learning is implemented using a concurrent learning-based system identifier to simulate experience by evaluating the Bellman error over unexplored areas of the state-space. Tracking of the desired trajectory and convergence of the developed policy to a neighborhood of the optimal policy are established via Lyapunov-based stability analysis. Simulation results demonstrate the effectiveness of the developed technique.

### 4.4.1 Problem Formulation and Exact Solution

The control objective in this section is to optimally track a time-varying desired trajectory  $x_d : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$ . Using the transformation in Sect. 3.3.1, the error system dynamics can be expressed in the autonomous form

$$\dot{\zeta}(t) = F(\zeta(t)) + G(\zeta(t))\mu(t).$$

The control objective is to simultaneously synthesize and utilize a control signal  $\mu(\cdot)$  to minimize the cost functional in (3.49) under the dynamic constraint in (3.47), while tracking the desired trajectory, where the local cost  $r_t : \mathbb{R}^{2n} \times \mathbb{R}^m \rightarrow \mathbb{R}_{\geq 0}$  is defined as

---

<sup>2</sup>Parts of the text in this section are reproduced, with permission, from [22], ©2017, IEEE.

$$r_t(\zeta, \mu) \triangleq Q(e) + \mu^T R \mu,$$

where  $R \in \mathbb{R}^{m \times m}$  is a positive definite symmetric matrix of constants, and  $Q : \mathbb{R}^n \rightarrow \mathbb{R}$  is a continuous positive definite function.

Assuming that an optimal policy exists, the optimal policy can be characterized in terms of the value function  $V^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}$  defined as

$$V^*(\zeta) \triangleq \min_{\mu(\tau) \in U | \tau \in \mathbb{R}_{\geq t}} \int_t^\infty r_t(\zeta(\tau; t, \zeta, \mu(\cdot)), \mu(\tau)) d\tau,$$

where  $U \subseteq \mathbb{R}^m$  is the action-space. Assuming that a minimizing policy exists and that  $V^*$  is continuously differentiable, a closed-form solution for the optimal policy can be obtained as [43]  $\mu^*(\zeta) = -\frac{1}{2} R^{-1} G^T(\zeta) (\nabla_\zeta V^*(\zeta))^T$ . The optimal policy and the optimal value function satisfy the Hamilton–Jacobi–Bellman equation [43]

$$\nabla_\zeta V^*(\zeta) (F(\zeta) + G(\zeta) \mu^*(\zeta)) + Q_t(\zeta) + \mu^{*T}(\zeta) R \mu^*(\zeta) = 0, \quad (4.26)$$

with the initial condition  $V^*(0) = 0$ , where the function  $Q_t : \mathbb{R}^{2n} \rightarrow \mathbb{R}$  is defined as  $Q_t([e^T, x_d^T]^T) = Q(e), \forall e, x_d \in \mathbb{R}^n$ .

#### 4.4.2 Bellman Error

The optimal value function  $V^*$  is replaced by a parametric estimate  $\hat{V}(\zeta, \hat{W}_c)$  and the optimal policy  $\mu^*$  by a parametric estimate  $\hat{\mu}(\zeta, \hat{W}_a)$ , where  $\hat{W}_c \in \mathbb{R}^L$  and  $\hat{W}_a \in \mathbb{R}^L$  denote vectors of estimates of the ideal parameters. Substituting the estimates  $\hat{V}$  and  $\hat{\mu}$  for  $V^*$  and  $\mu^*$  in the Hamilton–Jacobi–Bellman equation, respectively, yields the Bellman error

$$\begin{aligned} \delta(\zeta, \hat{W}_c, \hat{W}_a) &= Q_t(\zeta) + \hat{\mu}^T(\zeta, \hat{W}_a) R \hat{\mu}(\zeta, \hat{W}_a) \\ &\quad + \nabla_\zeta \hat{V}(\zeta, \hat{W}_c) (F(\zeta) + G(\zeta) \hat{\mu}(\zeta, \hat{W}_a)). \end{aligned} \quad (4.27)$$

Similar to the development in Sect. 4.3, the Bellman error is extrapolated to unexplored areas of the state-space using a system identifier. In this section, a neural network-based system identifier is employed.

### 4.4.3 System Identification

On a compact set  $\mathcal{C} \subset \mathbb{R}^n$  the function  $f$  is represented using a neural network as  $f(x) = \theta^T \sigma_f(Y^T x_1) + \epsilon_\theta(x)$ , where  $x_1 \triangleq [1, x^T]^T \in \mathbb{R}^{n+1}$ ,  $\theta \in \mathbb{R}^{p+1 \times n}$  and  $Y \in \mathbb{R}^{n+1 \times p}$  denote the constant unknown output-layer and hidden-layer neural network weights,  $\sigma_f : \mathbb{R}^p \rightarrow \mathbb{R}^{p+1}$  denotes a bounded neural network basis function,  $\epsilon_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^n$  denotes the function reconstruction error, and  $p \in \mathbb{N}$  denotes the number of neural network neurons. Let  $\bar{\theta}$  and  $\bar{\epsilon}_\theta$  be known constants such that  $\|\theta\|_F \leq \bar{\theta} < \infty$ ,  $\sup_{x \in \mathcal{C}} \|\epsilon_\theta(x)\| \leq \bar{\epsilon}_\theta$ , and  $\sup_{x \in \mathcal{C}} \|\nabla_x \epsilon_\theta(x)\| \leq \bar{\epsilon}_\theta$ . Using an estimate  $\hat{\theta} \in \mathbb{R}^{p+1 \times n}$  of the weight matrix  $\theta$ , the function  $f$  can be approximated by the function  $\hat{f} : \mathbb{R}^{2n} \times \mathbb{R}^{p+1 \times n} \rightarrow \mathbb{R}^n$  defined as  $\hat{f}(\zeta, \hat{\theta}) \triangleq \hat{\theta}^T \sigma_\theta(\zeta)$ , where  $\sigma_\theta : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{p+1}$  is defined as  $\sigma_\theta(\zeta) = \sigma_f(Y^T [1, e^T + x_d^T]^T)$ .

An estimator for online identification of the drift dynamics is developed as

$$\dot{\hat{x}}(t) = \hat{\theta}^T(t) \sigma_\theta(\zeta(t)) + g(x(t)) u(t) + k \tilde{x}(t), \quad (4.28)$$

where  $\tilde{x} \triangleq x - \hat{x}$ , and  $k \in \mathbb{R}$  is a positive constant learning gain.

**Assumption 4.4** ([30]) A history stack containing recorded state-action pairs  $\{x_j, u_j\}_{j=1}^M$  along with numerically computed state derivatives  $\{\dot{\tilde{x}}_j\}_{j=1}^M$  that satisfies  $\lambda_{\min}\left\{\sum_{j=1}^M \sigma_{fj} \sigma_{fj}^T\right\} = \underline{\sigma}_\theta > 0$ ,  $\|\dot{\tilde{x}}_j - \dot{x}_j\| < \bar{d}$ ,  $\forall j$  is available a priori, where  $\sigma_{fj} \triangleq \sigma_f(Y^T [1, x_j^T]^T)$ ,  $\bar{d} \in \mathbb{R}$  is a known positive constant, and  $\dot{x}_j = f(x_j) + g(x_j) u_j$ .

A priori availability of the history stack is used for ease of exposition, and is not necessary. Provided the system states are exciting over a finite time interval  $t \in [t_0, t_0 + \bar{t}]$  (versus  $t \in [t_0, \infty)$  as in traditional persistence of excitation-based approaches) the history stack can also be recorded online. The controller developed in [44] can be used over the time interval  $t \in [t_0, t_0 + \bar{t}]$  while the history stack is being recorded, and the controller developed in this result can be used thereafter. The use of two different controllers results in a switched system with one switching event. Since there is only one switching event, the stability of the switched system follows from the stability of the individual subsystems.

The weight estimates  $\hat{\theta}$  are updated using the concurrent learning-based update law

$$\dot{\hat{\theta}}(t) = \Gamma_\theta \sigma_f(Y^T x_1(t)) \tilde{x}^T(t) + k_\theta \Gamma_\theta \sum_{j=1}^M \sigma_{fj} \left( \dot{\tilde{x}}_j - g_j u_j - \hat{\theta}^T(t) \sigma_{fj} \right)^T, \quad (4.29)$$

where  $k_\theta \in \mathbb{R}$  is a constant positive concurrent learning gain and  $\Gamma_\theta \in \mathbb{R}^{p+1 \times p+1}$  is a constant, diagonal, and positive definite adaptation gain matrix. Using the identifier, the Bellman error in (4.27) can be approximated as

$$\begin{aligned}\hat{\delta}(\zeta, \hat{\theta}, \hat{W}_c, \hat{W}_a) &= Q_t(\zeta) + \hat{\mu}^T(\zeta, \hat{W}_a) R \hat{\mu}(\zeta, \hat{W}_a) \\ &\quad + \nabla_{\zeta} \hat{V}(\zeta, \hat{W}_a) (F_{\theta}(\zeta, \hat{\theta}) + F_1(\zeta) + G(\zeta) \hat{\mu}(\zeta, \hat{W}_a)),\end{aligned}\tag{4.30}$$

where

$$\begin{aligned}F_{\theta}(\zeta, \hat{\theta}) &\triangleq \begin{bmatrix} \hat{\theta}^T \sigma_{\theta}(\zeta) - g(x) g^+(x_d) \hat{\theta}^T \sigma_{\theta} \left( \begin{bmatrix} \mathbf{0}_{n \times 1} \\ x_d \end{bmatrix} \right) \\ \mathbf{0}_{n \times 1} \end{bmatrix}, \\ F_1(\zeta) &\triangleq \left[ (-h_d + g(e + x_d) g^+(x_d) h_d)^T, h_d^T \right]^T.\end{aligned}$$

#### 4.4.4 Value Function Approximation

Since  $V^*$  and  $\mu^*$  are functions of the augmented state  $\zeta$ , the minimization problem stated in Sect. 4.4.1 is intractable. To obtain a finite-dimensional minimization problem, the optimal value function is represented over any compact operating domain  $\mathcal{C} \subset \mathbb{R}^{2n}$  using a neural network as  $V^*(\zeta) = W^T \sigma(\zeta) + \epsilon(\zeta)$ , where  $W \in \mathbb{R}^L$  denotes a vector of unknown neural network weights,  $\sigma : \mathbb{R}^{2n} \rightarrow \mathbb{R}^L$  denotes a bounded neural network basis function,  $\epsilon : \mathbb{R}^{2n} \rightarrow \mathbb{R}$  denotes the function reconstruction error, and  $L \in \mathbb{N}$  denotes the number of neural network neurons. Using Property 2.3, for any compact set  $\mathcal{C} \subset \mathbb{R}^{2n}$ , there exist constant ideal weights  $W$  and known positive constants  $\bar{W}$ , and  $\bar{\epsilon}$  such that  $\|W\| \leq \bar{W} < \infty$ ,  $\sup_{\zeta \in \mathcal{C}} \|\epsilon(\zeta)\| \leq \bar{\epsilon}$ , and  $\sup_{\zeta \in \mathcal{C}} \|\nabla_{\zeta} \epsilon(\zeta)\| \leq \bar{\epsilon}$  [45].

A neural network representation of the optimal policy is obtained as  $\mu^*(\zeta) = -\frac{1}{2} R^{-1} G^T(\zeta) (\nabla_{\zeta} \sigma^T(\zeta) W + \nabla_{\zeta} \epsilon^T(\zeta))$ . Using estimates  $\hat{W}_c$  and  $\hat{W}_a$  for the ideal weights  $W$ , the optimal value function and the optimal policy are approximated as

$$\hat{V}(\zeta, \hat{W}_c) \triangleq \hat{W}_c^T \sigma(\zeta), \quad \hat{\mu}(\zeta, \hat{W}_a) \triangleq -\frac{1}{2} R^{-1} G^T(\zeta) \nabla_{\zeta} \sigma^T(\zeta) \hat{W}_a.\tag{4.31}$$

The optimal control problem is thus reformulated as the need to find a set of weights  $\hat{W}_c$  and  $\hat{W}_a$  online, to minimize the error  $\hat{E}_{\hat{\theta}}(\hat{W}_c, \hat{W}_a) \triangleq \sup_{\zeta \in \chi} |\hat{\delta}(\zeta, \hat{\theta}, \hat{W}_c, \hat{W}_a)|$ , for a given  $\hat{\theta}$ , while simultaneously improving  $\hat{\theta}$  using (4.29), and ensuring stability of the system using the control law

$$u(t) = \hat{\mu}(\zeta(t), \hat{W}_a(t)) + \hat{u}_d(\zeta(t), \hat{\theta}(t)),\tag{4.32}$$

where  $\hat{u}_d(\zeta, \hat{\theta}) \triangleq g_d^+(t) (h_d(t) - \hat{\theta}^T \sigma_{\theta d}(t))$  and  $\sigma_{\theta d}(t) \triangleq \sigma_{\theta} \left( [\mathbf{0}_{1 \times n} \ x_d^T(t)]^T \right)$ . The error between  $u_d$  and  $\hat{u}_d$  is included in the stability analysis based on the fact that

the error trajectories generated by the system  $\dot{e}(t) = f(x(t)) + g(x(t))u(t) - \dot{x}_d(t)$  under the controller in (4.32) are identical to the error trajectories generated by the system  $\dot{\zeta}(t) = F(\zeta(t)) + G(\zeta(t))\mu(t)$  under the control law  $\mu(t) = \hat{\mu}(\zeta(t), \hat{W}_a(t)) + g_d^+(t)\tilde{\theta}^T(t)\sigma_{\theta d}(t) + g_d^+(t)\epsilon_{\theta d}(t)$ , where  $\epsilon_{\theta d}(t) \triangleq \epsilon_\theta(x_d(t))$ .

#### 4.4.5 Simulation of Experience

Since computation of the supremum in  $\hat{E}_{\hat{\theta}}$  is intractable in general, simulation of experience is implemented by minimizing a squared sum of Bellman errors over finitely many points in the state-space. The following assumption facilitates the aforementioned approximation.

**Assumption 4.5** ([14]) There exists a finite set of points  $\{\zeta_i \in \mathcal{C} \mid i = 1, \dots, N\}$  and a constant  $c \in \mathbb{R}$  such that  $0 < c \triangleq \frac{1}{N} \left( \inf_{t \in \mathbb{R}_{\geq t_0}} \left( \lambda_{\min} \left\{ \sum_{i=1}^N \frac{\omega_i \omega_i^T}{\rho_i} \right\} \right) \right)$ , where  $\rho_i \triangleq 1 + v\omega_i^T \Gamma \omega_i \in \mathbb{R}$ , and  $\omega_i(t) \triangleq \nabla_{\zeta} \sigma(\zeta_i) \left( F_{\theta}(\zeta_i, \hat{\theta}(t)) + F_1(\zeta_i) + G(\zeta_i) \hat{\mu}(\zeta_i, \hat{W}_a(t)) \right)$ .

Using Assumption 4.5, simulation of experience is implemented by the weight update laws

$$\dot{\hat{W}}_c(t) = -k_{c1}\Gamma \frac{\omega(t)}{\rho(t)} \hat{\delta}_t(t) - \frac{k_{c2}}{N} \Gamma(t) \sum_{i=1}^N \frac{\omega_i(t)}{\rho_i(t)} \hat{\delta}_{ti}(t), \quad (4.33)$$

$$\dot{\Gamma}(t) = \left( \beta \Gamma(t) - k_{c1} \Gamma(t) \frac{\omega(t) \omega^T(t)}{\rho^2(t)} \Gamma(t) \right) \mathbf{1}_{\{\|\Gamma\| \leq \bar{\Gamma}\}}, \quad \|\Gamma(t_0)\| \leq \bar{\Gamma}, \quad (4.34)$$

$$\begin{aligned} \dot{\hat{W}}_a(t) &= -k_{a1} \left( \hat{W}_a(t) - \hat{W}_c(t) \right) - k_{a2} \hat{W}_a(t) \\ &+ \left( \frac{k_{c1} G_{\sigma}^T(t) \hat{W}_a(t) \omega^T(t)}{4\rho(t)} + \sum_{i=1}^N \frac{k_{c2} G_{\sigma i}^T(t) \hat{W}_a(t) \omega_i^T(t)}{4N\rho_i(t)} \right) \hat{W}_c(t), \end{aligned} \quad (4.35)$$

where  $\omega(t) \triangleq \nabla_{\zeta} \sigma(\zeta(t)) \left( F_{\theta}(\zeta(t), \hat{\theta}(t)) + F_1(\zeta(t)) + G(\zeta(t)) \hat{\mu}(\zeta(t), \hat{W}_a(t)) \right)$ ,  $\Gamma \in \mathbb{R}^{L \times L}$  is the least-squares gain matrix,  $\bar{\Gamma} \in \mathbb{R}$  denotes a positive saturation constant,  $\beta \in \mathbb{R}$  denotes a constant forgetting factor,  $k_{c1}, k_{c2}, k_{a1}, k_{a2} \in \mathbb{R}$  denote constant positive adaptation gains,  $G_{\sigma}(t) \triangleq \nabla_{\zeta} \sigma(\zeta(t)) G(\zeta(t)) R^{-1} G^T(\zeta(t)) \nabla_{\zeta} \sigma^T(\zeta(t))$ , and  $\rho(t) \triangleq 1 + v\omega^T(t) \Gamma(t) \omega(t)$ , where  $v \in \mathbb{R}$  is a positive normalization constant. In (4.33)–(4.35) and in the subsequent development, the notation  $\xi_i$ , is defined as  $\xi_i \triangleq \xi(\zeta_i, \cdot)$  for any function  $\xi(\zeta, \cdot)$  and the instantaneous Bellman errors  $\hat{\delta}_t$  and  $\hat{\delta}_{ti}$  are given by  $\hat{\delta}_t(t) = \hat{\delta}(\zeta(t), \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t))$  and  $\hat{\delta}_{ti}(t) = \hat{\delta}(\zeta_i, \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t))$ .

#### 4.4.6 Stability Analysis

If the state penalty function  $Q_t$  is positive definite, then the optimal value function  $V^*$  is positive definite (cf. [2, 6, 46]), and serves as a Lyapunov function for the concatenated system under the optimal control policy  $\mu^*$ . As a result,  $V^*$  is used as a candidate Lyapunov function for the closed-loop system under the policy  $\hat{\mu}$ . In this case, the function  $Q_t$ , and hence, the function  $V^*$  are positive semidefinite. Therefore, the function  $V^*$  is not a valid candidate Lyapunov function. However, the results in [44] can be used to show that a nonautonomous form of the optimal value function denoted by  $V_t^* : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ , defined as  $V_t^*(e, t) = V^*\left([e^T, x_d^T(t)]^T\right)$ ,  $\forall e \in \mathbb{R}^n$ ,  $t \in \mathbb{R}$ , is positive definite and decrescent. Hence,  $V_t^*(0, t) = 0$ ,  $\forall t \in \mathbb{R}$  and there exist class  $\mathcal{K}$  functions  $\underline{v} : \mathbb{R} \rightarrow \mathbb{R}$  and  $\bar{v} : \mathbb{R} \rightarrow \mathbb{R}$  such that  $\underline{v}(\|e\|) \leq V_t^*(e, t) \leq \bar{v}(\|e\|)$ ,  $\forall e \in \mathbb{R}^n$  and  $\forall t \in \mathbb{R}$ .

To facilitate the stability analysis, a concatenated state  $Z \in \mathbb{R}^{2n+2L+n(p+1)}$  is defined as

$$Z \triangleq \left[ e^T \tilde{W}_c^T \tilde{W}_a^T \tilde{x}^T \left( \text{vec}(\tilde{\theta}) \right)^T \right]^T,$$

and a candidate Lyapunov function is defined as

$$V_L(Z, t) \triangleq V_t^*(e, t) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a + \frac{1}{2} \tilde{x}^T \tilde{x} + \frac{1}{2} \text{tr}(\tilde{\theta}^T \Gamma_\theta^{-1} \tilde{\theta}). \quad (4.36)$$

The saturated least-squares update law in (4.34) ensures that there exist positive constants  $\underline{\gamma}, \bar{\gamma} \in \mathbb{R}$  such that  $\underline{\gamma} \leq \|\Gamma^{-1}(t)\| \leq \bar{\gamma}$ ,  $\forall t \in \mathbb{R}$ . Using the bounds on  $\Gamma$  and  $V_t^*$  and the fact that  $\text{tr}(\tilde{\theta}^T \Gamma_\theta^{-1} \tilde{\theta}) = (\text{vec}(\tilde{\theta}))^T (\Gamma_\theta^{-1} \otimes I_{p+1}) (\text{vec}(\tilde{\theta}))$ , the candidate Lyapunov function in (4.36) can be bounded as

$$\underline{v}_l(\|Z\|) \leq V_L(Z, t) \leq \bar{v}_l(\|Z\|), \quad (4.37)$$

$\forall Z \in \mathbb{R}^{2n+2L+n(p+1)}$  and  $\forall t \in \mathbb{R}$ , where  $\underline{v}_l : \mathbb{R} \rightarrow \mathbb{R}$  and  $\bar{v}_l : \mathbb{R} \rightarrow \mathbb{R}$  are class  $\mathcal{K}$  functions.

Given any compact set  $\chi \subset \mathbb{R}^{2n+2L+n(p+1)}$  containing an open ball of radius  $\rho \in \mathbb{R}$  centered at the origin, a positive constant  $\iota \in \mathbb{R}$  is defined as

$$\begin{aligned} \iota \triangleq & \frac{3 \left( \frac{(k_{a1} + k_{a2}) \bar{W}^2 \|G_\sigma\|}{16\sqrt{v}\underline{\Gamma}} + \frac{\|(W^T G_\sigma + \epsilon' G_\sigma \sigma'^T)\|}{4} + \frac{k_a \bar{W}}{2} \right)^2}{(k_{a1} + k_{a2})} \\ & + \frac{3 \left( \left( \|\bar{W}^T \sigma' G g_d^+\| + \|\epsilon' G g_d^+\| \right) \bar{\sigma}_g + k_\theta \bar{d}_\theta \right)^2}{4k_\theta \underline{\sigma}_\theta} \end{aligned}$$

$$\begin{aligned}
& + \frac{(k_{c1} + k_{c2})^2 \|\Delta\|^2}{4v\underline{\Gamma}k_{c2}\underline{c}} + \frac{\overline{\epsilon_\theta}^2}{2k} + \overline{\|\epsilon' G g_d^+ \epsilon_{\theta d}\|} \\
& + \overline{\left\| \frac{1}{2} G_\epsilon \right\|} + \overline{\left\| \frac{1}{2} W^T \sigma' G_r \epsilon'^T \right\|} + \overline{\|W^T \sigma' G g_d^+ \epsilon_{\theta d}\|}, \tag{4.38}
\end{aligned}$$

where  $G_r \triangleq GR^{-1}G^T$  and  $G_\epsilon \triangleq \epsilon' G_r (\epsilon')^T$ . Let  $v_l : \mathbb{R} \rightarrow \mathbb{R}$  be a class  $\mathcal{K}$  function such that

$$v_l(\|Z\|) \leq \frac{q(\|e\|)}{2} + \frac{k_{c2}\underline{c}}{8} \|\tilde{W}_c\|^2 + \frac{(k_{a1} + k_{a2})}{6} \|\tilde{W}_a\|^2 + \frac{k}{4} \|\tilde{x}\|^2 + \frac{k_\theta \sigma_\theta}{6} \|\text{vec}(\tilde{\theta})\|^2. \tag{4.39}$$

The sufficient gain conditions used in the subsequent Theorem 4.6 are

$$v_l^{-1}(\iota) < \overline{v_l}^{-1}(v_l(\rho)), \tag{4.40}$$

$$k_{c2}\underline{c} > \frac{3(k_{c2} + k_{c1})^2 \overline{W}^2 \overline{\|\sigma'\|}^2 \overline{\sigma_g}^2}{4k_\theta \sigma_\theta \underline{\Gamma}}, \tag{4.41}$$

$$(k_{a1} + k_{a2}) > \frac{3(k_{c1} + k_{c2}) \overline{W} \|G_\sigma\|}{8\sqrt{\nu \underline{\Gamma}}} + \frac{3}{\underline{c} k_{c2}} \left( \frac{(k_{c1} + k_{c2}) \overline{W} \|G_\sigma\|}{8\sqrt{\nu \underline{\Gamma}}} + k_{a1} \right)^2, \tag{4.42}$$

where  $\overline{\sigma_g} \triangleq \overline{\|\sigma_\theta\|} + \overline{\|g g_d^+\|} \overline{\|\sigma_{\theta d}\|}$ . In (4.38)–(4.42), the notation  $\|\varpi\|$  denotes  $\sup_{y \in \chi_l} \|\varpi(y)\|$  for any function  $\varpi : \mathbb{R}^l \rightarrow \mathbb{R}$ , where  $l \in \mathbb{N}$  and  $\chi_l$  denotes the projection of  $\chi$  onto  $\mathbb{R}^l$ .

The sufficient condition in (4.40) requires the set  $\chi$  to be large enough based on the constant  $\iota$ . Since the neural network approximation errors depend on the compact set  $\chi$ , in general, the constant  $\iota$  increases with the size of the set  $\chi$  for a fixed number of neural network neurons. However, for a fixed set  $\chi$ , the constant  $\iota$  can be reduced by reducing function reconstruction errors (i.e., by increasing number of neural network neurons) and by increasing the learning gains provided  $\sigma_\theta$  is large enough. Hence a sufficient number of neural network neurons and extrapolation points are required to satisfy the condition in (4.40).

**Theorem 4.6** *Provided Assumptions 4.4 and 4.5 hold and  $L$ ,  $\underline{c}$ , and  $\sigma_\theta$  are large enough to satisfy the sufficient gain conditions in (4.40)–(4.42), the controller in (4.32) with the weight update laws (4.33)–(4.35), and the identifier in (4.28) with the weight update law (4.29), ensure that the system states remain bounded, the tracking error is ultimately bounded, and that the control policy  $\hat{\mu}$  converges to a neighborhood around the optimal control policy  $\mu^*$ .*

*Proof* Using (3.47) and the fact that  $\dot{V}_t^*(e(t), t) = \dot{V}^*(\zeta(t))$ ,  $\forall t \in \mathbb{R}$ , the time-derivative of the candidate Lyapunov function in (4.36) is

$$\begin{aligned}\dot{V}_L &= \nabla_{\zeta} V^* (F + G\mu^*) - \tilde{W}_c^T \Gamma^{-1} \dot{\tilde{W}}_c - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \dot{\Gamma} \Gamma^{-1} \tilde{W}_c \\ &\quad - \tilde{W}_a^T \dot{\tilde{W}}_a + \dot{V}_0 + \nabla_{\zeta} V^* G\mu - \nabla_{\zeta} V^* G\mu^*. \end{aligned}\quad (4.43)$$

Under sufficient gain conditions in (4.40)–(4.42), using (4.26), (4.28)–(4.31), and the update laws in (4.33)–(4.35) the expression in (4.43) can be bounded as

$$\dot{V}_L \leq -v_l (\|Z\|), \quad \forall \|Z\| \geq v_l^{-1}(\iota), \quad \forall Z \in \chi, \quad (4.44)$$

where  $\iota$  is a positive constant and  $\chi \subset \mathbb{R}^{2n+2L+n(p+1)}$  is a compact set. Using (4.37) and (4.44), [40, Theorem 4.18] can be invoked to conclude that every trajectory  $Z(\cdot)$  satisfying  $\|Z(t_0)\| \leq \bar{v}_l^{-1}(\underline{v}_l(\rho))$ , where  $\rho$  is a positive constant, is bounded for all  $t \in \mathbb{R}$  and satisfies  $\limsup_{t \rightarrow \infty} \|Z(t)\| \leq \underline{v}_l^{-1}(\bar{v}_l(\underline{v}_l^{-1}(\iota)))$ . The ultimate bound can be decreased by increasing learning gains and the number of neurons in the neural networks, provided the points in the history stack and Bellman error extrapolation can be selected to increase  $\sigma_\theta$  and  $\underline{v}_l$ .  $\square$

#### 4.4.7 Simulation

##### Linear System

In the following, the developed technique is applied to solve a linear quadratic tracking problem. A linear system is selected because the optimal solution to the linear quadratic tracking problem can be computed analytically and compared against the solution generated by the developed technique. To demonstrate convergence to the ideal weights, the following linear system is simulated:  $\dot{x} = \begin{bmatrix} -1 & 1 \\ -0.5 & 0.5 \end{bmatrix}x + \begin{bmatrix} 0 \\ 1 \end{bmatrix}u$ . The control objective is to follow a desired trajectory, which is the solution to the initial value problem  $\dot{x}_d = \begin{bmatrix} -1 & 1 \\ -2 & 1 \end{bmatrix}x_d, \quad x_d(0) = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$ , while ensuring convergence of the estimated policy  $\hat{\mu}$  to a neighborhood of the policy  $\mu^*$ , such that the control law  $\mu(t) = \mu^*(\zeta(t))$  minimizes the cost  $\int_0^\infty (e^T(t) \text{diag}([10, 10]) e(t) + \mu^2(t)) dt$ .

Since the system is linear, the optimal value function is known to be quadratic. Hence, the value function is approximated using the quadratic basis  $\sigma(\zeta) = [e_1^2, e_2^2, e_1 e_2, e_1 x_{d1}, e_2 x_{d2}, e_1 x_{d2}, e_2 x_{d1}]^T$ , and the unknown drift dynamics is approximated using the linear basis  $\sigma_\theta(x) = [x_1, x_2]^T$ .

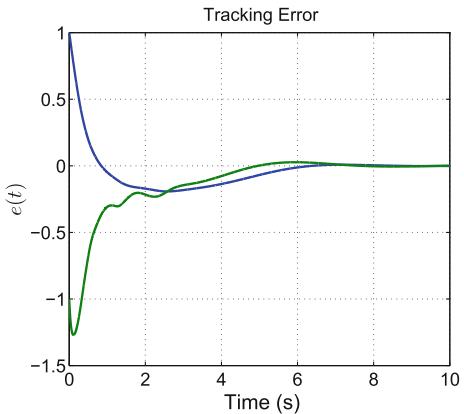
The linear system and the linear desired dynamics result in a linear time-invariant concatenated system. Since the system is linear, the optimal tracking problem reduces to an optimal regulation problem, which can be solved using the resulting Algebraic Riccati Equation. The optimal value function is given by  $V^*(\zeta) = \zeta^T P_\zeta \zeta$ , where the matrix  $P_\zeta$  is given by

$$P_\zeta = \begin{bmatrix} 4.43 & 0.67 & \mathbf{0}_{2 \times 2} \\ 0.67 & 2.91 & \mathbf{0}_{2 \times 2} \\ \mathbf{0}_{2 \times 2} & \mathbf{0}_{2 \times 2} \end{bmatrix}.$$

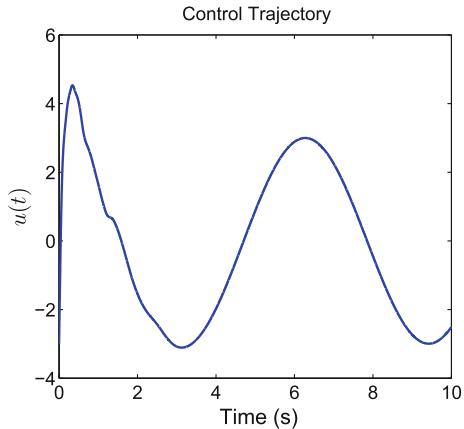
Using the matrix  $P_\zeta$ , the ideal weights corresponding to the selected basis can be computed as  $W = [4.43, 1.35, 0, 0, 2.91, 0, 0]^T$ .

Figures 4.18, 4.19, 4.20 and 4.21 demonstrate that the controller remains bounded, the tracking error goes to zero, and the weight estimates  $\hat{W}_c$ ,  $\hat{W}_a$  and  $\hat{\theta}$  go to their true values, establishing convergence of the approximate policy to the optimal policy. Figures 4.22 and 4.23 demonstrate satisfaction of the rank conditions in Assumptions 4.4 and 4.5, respectively.

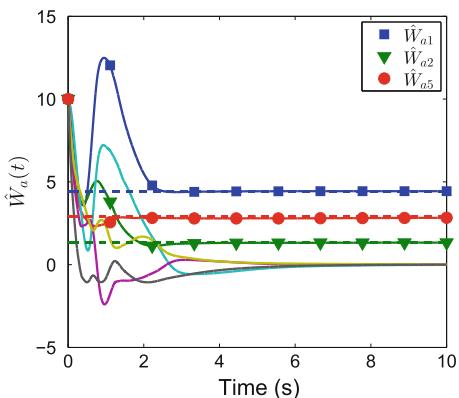
**Fig. 4.18** State trajectories generated using the developed method for the linear system (reproduced with permission from [22], ©2017, IEEE)



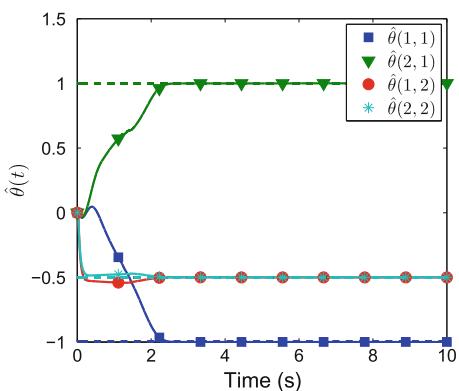
**Fig. 4.19** Control trajectory generated using the developed method for the linear system (reproduced with permission from [22], ©2017, IEEE).eps



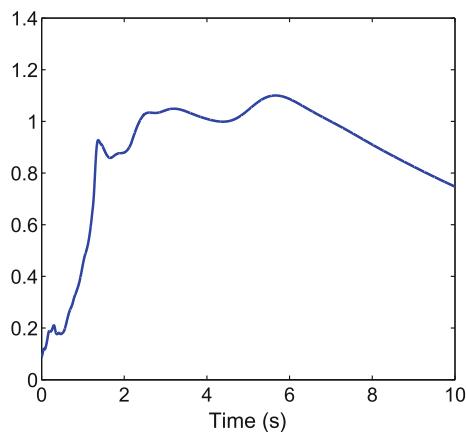
**Fig. 4.20** Actor weight estimates generated using the developed method for the linear system. Dashed lines denote the ideal values (reproduced with permission from [22], ©2017, IEEE)



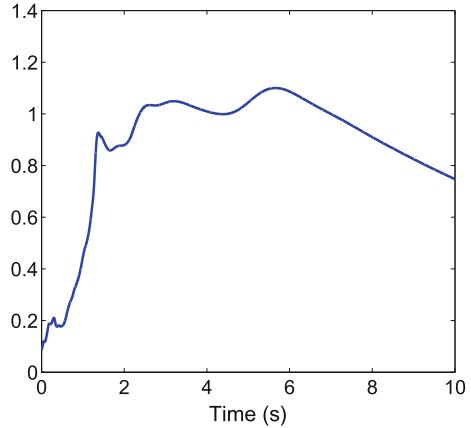
**Fig. 4.21** Drift parameter estimates generated using the developed method for the linear system. Dashed lines denote the ideal values (reproduced with permission from [22], ©2017, IEEE)



**Fig. 4.22** Evolution of minimum singular value of the concurrent learning history stack for the linear system (reproduced with permission from [22], ©2017, IEEE)



**Fig. 4.23** Satisfaction of Assumptions 4.4 and 4.5 for the linear system (reproduced with permission from [22], ©2017, IEEE)



### Nonlinear System

Effectiveness of the developed technique is demonstrated via numerical simulation on the nonlinear system  $\dot{x} = f(x) + (\cos(2x) + 2)^2 u$ ,  $x \in \mathbb{R}$ , where  $f(x) = x^2$  is assumed to be unknown. The control objective is to track the desired trajectory  $x_d(t) = 2 \sin(2t)$ , while ensuring convergence of the estimated policy  $\hat{\mu}$  to a neighborhood of the policy  $\mu^*$ , such that  $\mu^*$  minimizes the cost  $\int_0^\infty (10e^2(t) + \frac{1}{10}\mu^2(t)) dt$ .

Since the system is linear, the optimal value function is known to be quadratic. Hence, the value function is approximated using the quadratic basis  $\sigma(\zeta) = [e_1^2, e_2^2, e_1e_2, e_1x_{d1}, e_2x_{d2}, e_1x_{d2}, e_2x_{d1}]^T$ , and the unknown drift dynamics is approximated using the linear basis  $\sigma_\theta(x) = [x_1, x_2]^T$ .

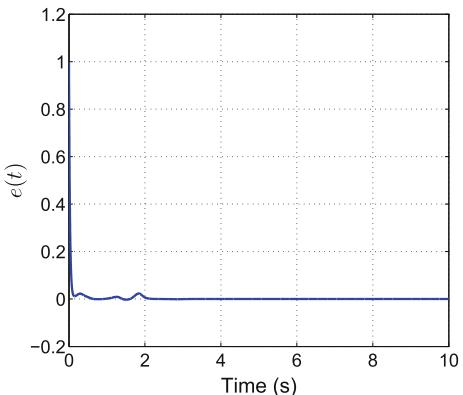
The value function is approximated using the polynomial basis  $\sigma(\zeta) = [e^2, e^4, e^6, e^2x_d^2, e^4x_d^2, e^6x_d^2]$ , and  $f(x)$  is approximated using the polynomial basis  $\sigma_\theta(x) = [x, x^2, x^3]$ . The higher order terms in  $\sigma(\zeta)$  are used to compensate for the higher order terms in  $\sigma_\theta$ .

The initial values for the state and the state estimate are selected to be  $x(0) = 1$  and  $\hat{x}(0) = 0$ , respectively, and the initial values for the neural network weights for the value function, the policy, and the drift dynamics are selected to be zero. Since the selected system exhibits a finite escape time for any initial condition other than zero, the initial policy  $\hat{\mu}(\zeta, \mathbf{0}_{6 \times 1})$  is not stabilizing. The stabilization demonstrated in Fig. 4.24 is achieved via fast simultaneous learning of the system dynamics and the value function.

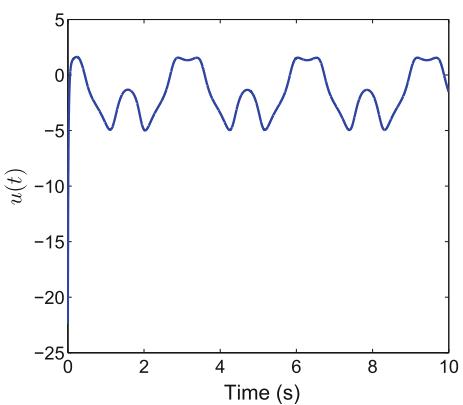
Figures 4.24, 4.25, 4.26 and 4.27 demonstrate that the controller remains bounded, the tracking error is regulated to the origin, and the neural network weights converge.

Figures 4.28 and 4.29 demonstrate satisfaction of the rank conditions in Assumptions 4.4 and 4.5, respectively. The rank condition on the history stack in Assumption 4.4 is ensured by selecting points using a singular value maximization algorithm [30], and the condition in Assumption 4.5 is met via oversampling (i.e., by selecting fifty-six points to identify six unknown parameters). Unlike previous results that rely on

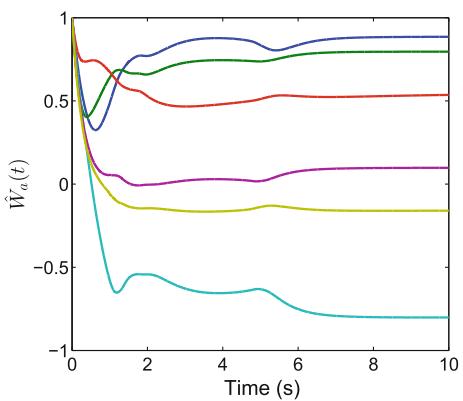
**Fig. 4.24** State trajectory for the nonlinear system generated using the developed method



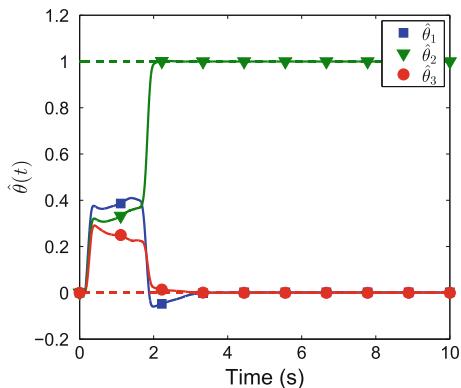
**Fig. 4.25** Control trajectory for the nonlinear system generated using the developed method



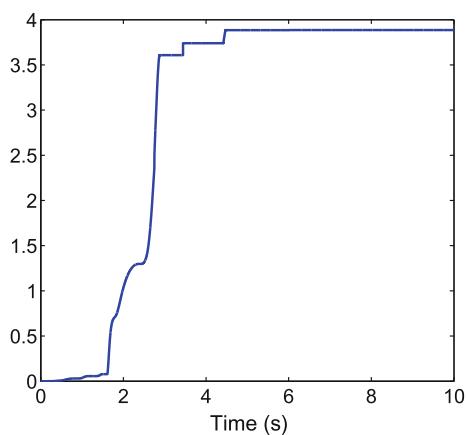
**Fig. 4.26** Critic weight estimates for the nonlinear system generated using the developed method



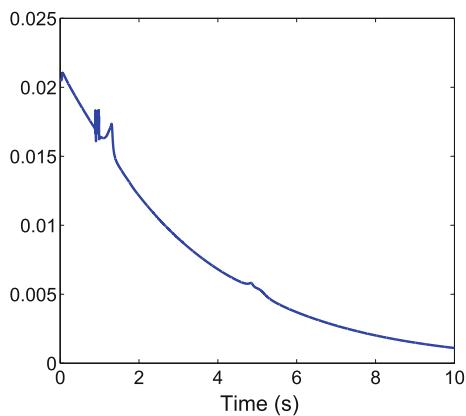
**Fig. 4.27** Drift parameter estimates for the nonlinear system generated using the developed method. Dashed lines represent true values of the drift parameters



**Fig. 4.28** Evolution of minimum singular value of the concurrent learning history stack for the nonlinear system



**Fig. 4.29** Evolution of minimum singular value of the Bellman error extrapolation matrix for the nonlinear system



the addition of an ad-hoc probing signal to satisfy the persistence of excitation condition, this result ensures sufficient exploration via Bellman error extrapolation. Since an analytical solution to the nonlinear optimal tracking problem is not available, the value function and the actor weights can not be compared against the ideal values. However, a comparison between the learned weights and the optimal weights is possible for linear systems provided the dynamics  $h_d$  of the desired trajectory are also linear.

The learning gains, the basis functions for the neural networks, and the points for Bellman error extrapolation are selected using a trial and error approach. Alternatively, global optimization methods such as a genetic algorithm, or simulation-based methods such as a Monte-Carlo simulation can be used to tune the gains.

## 4.5 *N*-Player Nonzero-Sum Differential Games<sup>3</sup>

This section presents a concurrent learning-based actor-critic-identifier architecture to obtain an approximate feedback-Nash equilibrium solution to an infinite-horizon *N*-player nonzero-sum differential game. The solution is obtained online for a nonlinear control-affine system with uncertain linearly parameterized drift dynamics. It is shown that under a condition milder than persistence of excitation, uniformly ultimately bounded convergence of the developed control policies to the feedback-Nash equilibrium policies can be established. Simulation results are presented to demonstrate the performance of the developed technique without an added excitation signal.

Consider the class of control-affine multi-input systems introduced in (3.82). In this section, the unknown function  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is assumed to be linearly parameterizable, the functions  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m_i}$  are assumed to be known and uniformly bounded, the functions  $f$  and  $g_i$  are assumed to be locally Lipschitz, and  $f(0) = 0$ .

Recall the Bellman errors introduced in (3.88):

$$\begin{aligned} \delta_i(x, \hat{W}_{ci}, \hat{W}_{a1}, \dots, \hat{W}_{aN}) &= \nabla_x \hat{V}_i(x, \hat{W}_{ci}) \left( f(x) + \sum_{j=1}^N g_j(x) \hat{u}_j(x, \hat{W}_{aj}) \right) \\ &\quad + r_i(x, \hat{u}_1(x, \hat{W}_{a1}), \dots, \hat{u}_N(x, \hat{W}_{aN})). \end{aligned} \quad (4.45)$$

To obtain a feedback-Nash equilibrium solution to the *N*-player differential game, the estimates  $\hat{W}_{ci}$  and  $\hat{W}_{ai}$  are recursively improved to drive the Bellman errors to zero. The computation of the Bellman errors in (4.45) requires knowledge of the drift dynamics  $f$ . To eliminate this requirement and to facilitate simulation of experience, a concurrent learning-based system identifier that satisfies Assumption 4.1 is developed in the following section.

---

<sup>3</sup>Parts of the text in this section are reproduced, with permission, from [16], ©2014, IEEE.

### 4.5.1 System Identification

Let  $f(x) = Y(x)\theta$  be the linear parametrization of the drift dynamics, where  $Y : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times p_\theta}$  denotes the locally Lipschitz regression matrix, and  $\theta \in \mathbb{R}^{p_\theta}$  denotes the vector of constant unknown drift parameters. The system identifier is designed as

$$\dot{\tilde{x}}(t) = Y(x(t))\hat{\theta}(t) + \sum_{i=1}^N g_i(x(t))u_i(t) + k_x \tilde{x}(t), \quad (4.46)$$

where the measurable state estimation error  $\tilde{x}$  is defined as  $\tilde{x}(t) \triangleq x(t) - \hat{x}(t)$ ,  $k_x \in \mathbb{R}^{n \times n}$  is a constant positive definite diagonal observer gain matrix, and  $\hat{\theta} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{p_\theta}$  denotes the vector of estimates of the unknown drift parameters. In traditional adaptive systems, the estimates are updated to minimize the instantaneous state estimation error, and convergence of parameter estimates to their true values can be established under a restrictive persistence of excitation condition. In this result, a concurrent learning-based data-driven approach is developed to relax the persistence of excitation condition to a weaker, verifiable rank condition.

**Assumption 4.7** ([30, 31]) A history stack  $\mathcal{H}_{id}$  containing state-action tuples  $\{(x_j, \hat{u}_j) \mid i = 1, \dots, N, j = 1, \dots, M_\theta\}$  recorded along the trajectories of (3.82) that satisfies

$$\text{rank} \left( \sum_{j=1}^{M_\theta} Y_j^T Y_j \right) = p_\theta, \quad (4.47)$$

is available a priori, where  $Y_j = Y(x_j)$ , and  $p_\theta$  denotes the number of unknown parameters in the drift dynamics.

To facilitate the concurrent learning-based parameter update, numerical methods are used to compute the state derivative  $\dot{x}_j$  corresponding to  $(x_j, \hat{u}_j)$ . The update law for the drift parameter estimates is designed as

$$\dot{\hat{\theta}}(t) = \Gamma_\theta Y^T(x(t))\tilde{x}(t) + \Gamma_\theta k_\theta \sum_{j=1}^{M_\theta} Y_j^T \left( \dot{x}_j - \sum_{i=1}^N g_{ij} u_{ij} - Y_j \hat{\theta}(t) \right), \quad (4.48)$$

where  $g_{ij} \triangleq g_i(x_j)$ ,  $\Gamma_\theta \in \mathbb{R}^{p \times p}$  is a constant positive definite adaptation gain matrix, and  $k_\theta \in \mathbb{R}$  is a constant positive concurrent learning gain. The update law in (4.48) requires the unmeasurable state derivative  $\dot{x}_j$ . Since the state derivative at a past recorded point on the state trajectory is required, past and future recorded values of the state can be used along with accurate noncausal smoothing techniques to obtain good estimates of  $\dot{x}_j$ . In the presence of derivative estimation errors, the parameter estimation errors can be shown to be uniformly ultimately bounded, where the size of the ultimate bound depends on the error in the derivative estimate [31].

To incorporate new information, the history stack is updated with new data. Thus, the resulting closed-loop system is a switched system. To ensure the stability of the switched system, the history stack is updated using a singular value maximizing algorithm (cf. [31]). Using (3.82), the state derivative can be expressed as

$$\dot{x}_j - \sum_{i=1}^N g_{ij} u_{ij} = Y_j \theta,$$

and hence, the update law in (4.48) yields the parameter estimation error dynamics

$$\dot{\tilde{\theta}}(t) = -\Gamma_\theta Y^T(x(t)) \tilde{x}(t) - \Gamma_\theta k_\theta \left( \sum_{j=1}^{M_\theta} Y_j^T Y_j \right) \tilde{\theta}(t), \quad (4.49)$$

where  $\tilde{\theta}(t) \triangleq \theta - \hat{\theta}(t)$  denotes the drift parameter estimation error. The closed-loop dynamics of the state estimation error are given by

$$\dot{\tilde{x}}(t) = Y(x(t)) \tilde{\theta}(t) - k_x \tilde{x}(t). \quad (4.50)$$

### 4.5.2 Model-Based Reinforcement Learning

Based on (4.46), measurable approximations to the Bellman errors in (4.45) are developed as

$$\begin{aligned} \hat{\delta}_i \left( x, \hat{W}_{ci}, \hat{W}_{a1}, \dots, \hat{W}_{aN}, \hat{\theta} \right) &= \nabla_x \hat{V}_i \left( x, \hat{W}_{ci} \right) \left( Y(x) \hat{\theta} + \sum_{j=1}^N g_j(x) \hat{u}_j \left( x, \hat{W}_{aj} \right) \right) \\ &\quad + r_i \left( x, \hat{u}_1 \left( x, \hat{W}_{a1} \right), \dots, \hat{u}_N \left( x, \hat{W}_{aN} \right) \right). \end{aligned} \quad (4.51)$$

The following assumption, which in general is weaker than the persistence of excitation assumption, is required for convergence of the concurrent learning-based critic weight estimates.

**Assumption 4.8** For each  $i \in \{1, \dots, N\}$ , there exists a finite set of  $M_{xi}$  points  $\{x_{ij} \in \mathbb{R}^n \mid j = 1, \dots, M_{xi}\}$  such that for all  $t \in \mathbb{R}_{\geq 0}$ ,

$$\underline{c}_{xi} \triangleq \frac{\left( \inf_{t \in \mathbb{R}_{\geq 0}} \left( \lambda_{\min} \left\{ \sum_{k=1}^{M_{xi}} \frac{\omega_i^k(t)(\omega_i^k(t))^T}{\rho_i^k(t)} \right\} \right) \right)}{M_{xi}} > 0, \quad (4.52)$$

where  $\underline{c}_{xi} \in \mathbb{R}$  is a positive constant. In (4.52),  $\omega_i^k(t) \triangleq \sigma_i'^{ik} Y^{ik} \hat{\theta}(t) - \frac{1}{2} \sum_{j=1}^N \sigma_i'^{ik} G_j^{ik} \left( \sigma_j'^{ik} \right)^T \hat{W}_{aj}(t)$ , where  $\sigma_j'^{ik} \triangleq \nabla_x \sigma_j(x_{ik})$ , the superscript  $(\cdot)^{ik}$  indicates that the

term is evaluated at  $x = x_{ik}$ , and  $\rho_i^k \triangleq 1 + v_i (\omega_i^k)^T \Gamma_i \omega_i^k$ , where  $v_i \in \mathbb{R}_{>0}$  is the normalization gain, and  $\Gamma_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{L_i \times L_i}$  is the adaptation gain matrix.

The concurrent learning-based least-squares update law for the critic weights is designed as

$$\begin{aligned}\dot{\hat{W}}_{ci}(t) &= -k_{c1i} \Gamma_i(t) \frac{\omega_i(t)}{\rho_i(t)} \hat{\delta}_{ti}(t) - \frac{k_{c2i} \Gamma_i(t)}{M_{xi}} \sum_{k=1}^{M_{xi}} \frac{\omega_i^k(t)}{\rho_i^k(t)} \hat{\delta}_{ti}^k(t), \\ \dot{\Gamma}_i(t) &= \left( \beta_i \Gamma_i(t) - k_{c1i} \Gamma_i(t) \frac{\omega_i(t) \omega_i^T(t)}{\rho_i^2(t)} \Gamma_i(t) \right) \mathbf{1}_{\{\|\Gamma_i\| \leq \bar{\Gamma}_i\}}, \quad \|\Gamma_i(t_0)\| \leq \bar{\Gamma}_i,\end{aligned}\quad (4.53)$$

where

$$\begin{aligned}\hat{\delta}_{ti}(t) &= \hat{\delta}_i \left( x(t), \hat{W}_{ci}(t), \hat{W}_{a1}(t), \dots, \hat{W}_{aN}(t), \hat{\theta}(t) \right), \\ \hat{\delta}_{ti}^k(t) &= \hat{\delta}_i \left( x_{ik}, \hat{W}_{ci}(t), \hat{W}_{a1}(t), \dots, \hat{W}_{aN}(t), \hat{\theta}(t) \right),\end{aligned}$$

$\omega_i(t) \triangleq \nabla_x \sigma_i(x(t)) Y(x(t)) \hat{\theta}(t) - \frac{1}{2} \sum_{j=1}^N \nabla_x \sigma_i(x(t)) G_j(x(t)) \nabla_x \sigma_j^T(x(t))$   
 $\hat{W}_{aj}(t)$ ,  $\rho_i(t) \triangleq 1 + v_i \omega_i^T(t) \Gamma_i(t) \omega_i(t)$ ,  $\bar{\Gamma}_i > 0 \in \mathbb{R}$  is the saturation constant,  
 $\beta_i \in \mathbb{R}$  is the constant positive forgetting factor, and  $k_{c1i}, k_{c2i} \in \mathbb{R}$  are constant positive adaptation gains.

The actor weight update laws are designed based on the subsequent stability analysis as

$$\begin{aligned}\dot{\hat{W}}_{ai}(t) &= -k_{a1i} \left( \hat{W}_{ai}(t) - \hat{W}_{ci}(t) \right) - k_{a2i} \hat{W}_{ai}(t) \\ &+ \frac{1}{4} \sum_{j=1}^N k_{c1i} \nabla_x \sigma_j(x(t)) G_{ij}(x(t)) \nabla_x \sigma_j^T(x(t)) \hat{W}_{aj}^T(t) \frac{\omega_i^T(t)}{\rho_i(t)} \hat{W}_{ci}^T(t) \\ &+ \frac{1}{4} \sum_{k=1}^{M_{xi}} \sum_{j=1}^N \frac{k_{c2i}}{M_{xi}} \sigma_j'^{ik} G_{ij}^k (\sigma_j'^k)^T \hat{W}_{aj}^T(t) \frac{(\omega_i^k(t))^T}{\rho_i^k(t)} \hat{W}_{ci}^T(t),\end{aligned}\quad (4.54)$$

where  $k_{a1i}, k_{a2i} \in \mathbb{R}$  are positive constant adaptation gains. The forgetting factor  $\beta_i$  along with the saturation in the update law for the least-squares gain matrix in (4.53) ensure (cf. [47]) that the least-squares gain matrix  $\Gamma_i$  and its inverse is positive definite and bounded for all  $i \in \{1, \dots, N\}$  as

$$\underline{\Gamma}_i \leq \|\Gamma_i(t)\| \leq \bar{\Gamma}_i, \quad \forall t \in \mathbb{R}_{\geq 0}, \quad (4.55)$$

where  $\underline{\Gamma}_i \in \mathbb{R}$  is a positive constant, and the normalized regressor is bounded as

$$\left\| \frac{\omega_i}{\rho_i} \right\|_{\infty} \leq \frac{1}{2\sqrt{v_i \underline{\Gamma}_i}}.$$

### 4.5.3 Stability Analysis

Subtracting (3.85) from (4.51), the approximate Bellman error can be expressed in an unmeasurable form as

$$\begin{aligned}\hat{\delta}_{ti} = & \omega_i^T \hat{W}_{ci} + x^T Q_i x + \sum_{j=1}^N \frac{1}{4} \hat{W}_{aj}^T \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T \hat{W}_{aj} \\ & - \left( x^T Q_i x + \sum_{j=1}^N u_j^{*T} R_{ij} u_j^* + \nabla_x V_i^* f + \nabla_x V_i^* \sum_{j=1}^N g_j u_j^* \right).\end{aligned}$$

Substituting for  $V^*$  and  $u^*$  from (3.113) and using  $f = Y\theta$ , the approximate Bellman error can be expressed as

$$\begin{aligned}\hat{\delta}_{ti} = & \omega_i^T \hat{W}_{ci} + \sum_{j=1}^N \frac{1}{4} \hat{W}_{aj}^T \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T \hat{W}_{aj} - W_i^T \nabla_x \sigma_i Y\theta - \nabla_x \epsilon_i Y\theta \\ & - \sum_{j=1}^N \frac{1}{4} (W_j^T \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T W_j + 2\nabla_x \epsilon_j G_{ij} \nabla_x \sigma_j^T W_j + \nabla_x \epsilon_j G_{ij} \nabla_x \epsilon_j^T) \\ & + \frac{1}{2} \sum_{j=1}^N (W_i^T \nabla_x \sigma_i G_j \nabla_x \sigma_j^T W_j + \nabla_x \epsilon_i G_j \nabla_x \sigma_j^T W_j + W_i^T \nabla_x \sigma_i G_j \nabla_x \epsilon_j^T) \\ & + \frac{1}{2} \sum_{j=1}^N \nabla_x \epsilon_i G_j \nabla_x \epsilon_j^T.\end{aligned}$$

Adding and subtracting  $\frac{1}{4} \hat{W}_{aj}^T \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T W_j + \omega_i^T W_i$  yields

$$\begin{aligned}\hat{\delta}_{ti} = & -\omega_i^T \tilde{W}_{ci} + \frac{1}{4} \sum_{j=1}^N \tilde{W}_{aj}^T \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T \tilde{W}_{aj} - W_i^T \nabla_x \sigma_i Y\tilde{\theta} \\ & - \frac{1}{2} \sum_{j=1}^N (W_i^T \nabla_x \sigma_i G_j - W_j^T \nabla_x \sigma_j G_{ij}) \nabla_x \sigma_j^T \tilde{W}_{aj} - \nabla_x \epsilon_i Y\theta + \Delta_i,\end{aligned}\quad (4.56)$$

where  $\Delta_i \triangleq \frac{1}{2} \sum_{j=1}^N (W_i^T \nabla_x \sigma_i G_j - W_j^T \nabla_x \sigma_j G_{ij}) \nabla_x \epsilon_j^T + \frac{1}{2} \sum_{j=1}^N W_j^T \nabla_x \sigma_j G_j \nabla_x \epsilon_i^T + \frac{1}{2} \sum_{j=1}^N \nabla_x \epsilon_i G_j \nabla_x \epsilon_j^T - \sum_{j=1}^N \frac{1}{4} \nabla_x \epsilon_j G_{ij} \nabla_x \epsilon_j^T$ . Similarly, the approximate Bellman error evaluated at the selected points can be expressed in an unmeasurable form as

$$\begin{aligned}\hat{\delta}_{ti}^k &= -\omega_i^{kT} \tilde{W}_{ci} + \frac{1}{4} \sum_{j=1}^N \tilde{W}_{aj}^T \sigma_j'^{ik} G_{ij}^{ik} (\sigma_j'^{ik})^T \tilde{W}_{aj} + \Delta_i^k \\ &\quad - \frac{1}{2} \sum_{j=1}^N (W_i^T \sigma_i'^{ik} G_j^{ik} - W_j^T \sigma_j'^{ik} G_{ij}^{ik}) (\sigma_j'^{ik})^T \tilde{W}_{aj} - W_i^T \sigma_i'^{ik} Y^{ik} \tilde{\theta},\end{aligned}\quad (4.57)$$

where the constant  $\Delta_i^k \in \mathbb{R}$  is defined as  $\Delta_i^k \triangleq -\epsilon_i'^{ik} Y^{ik} \theta + \Delta_i^{ik}$ . To facilitate the stability analysis, a candidate Lyapunov function is defined as

$$V_L \triangleq \sum_{i=1}^N V_i^* + \frac{1}{2} \sum_{i=1}^N \tilde{W}_{ci}^T \Gamma_i^{-1} \tilde{W}_{ci} + \frac{1}{2} \sum_{i=1}^N \tilde{W}_{ai}^T \tilde{W}_{ai} + \frac{1}{2} \tilde{x}^T \tilde{x} + \frac{1}{2} \tilde{\theta}^T \Gamma_\theta^{-1} \tilde{\theta}. \quad (4.58)$$

Since  $V_i^*$  are positive definite, the bound in (4.55) and [40, Lemma 4.3] can be used to bound the candidate Lyapunov function as

$$\underline{v}(\|Z\|) \leq V_L(Z, t) \leq \bar{v}(\|Z\|), \quad (4.59)$$

where  $Z = \left[ x^T, \tilde{W}_{c1}^T, \dots, \tilde{W}_{cN}^T, \tilde{W}_{a1}^T, \dots, \tilde{W}_{aN}^T, \tilde{x}, \tilde{\theta} \right]^T \in \mathbb{R}^{2n+2N \sum_i L_i + p_\theta}$  and  $\underline{v}, \bar{v} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions. For any compact set  $\mathcal{Z} \subset \mathbb{R}^{2n+2N \sum_i L_i + p_\theta}$ , define

$$\iota_1 \triangleq \max_{i,j} \left( \sup_{Z \in \mathcal{Z}} \left\| \frac{1}{2} W_i^T \nabla_x \sigma_i G_j \nabla_x \sigma_j^T + \frac{1}{2} \nabla_x \epsilon_i G_j \nabla_x \sigma_j^T \right\| \right),$$

$$\begin{aligned}\iota_2 &\triangleq \max_{i,j} \left( \sup_{Z \in \mathcal{Z}} \left\| \frac{k_{cli} \omega_i}{4\rho_i} (3W_j \nabla_x \sigma_j G_{ij} - 2W_i^T \nabla_x \sigma_i G_j) \nabla_x \sigma_j^T \right. \right. \\ &\quad \left. \left. + \sum_{k=1}^{M_{xi}} \frac{k_{c2i} \omega_i^k}{4M_{xi} \rho_i^k} (3W_j^T \sigma_j'^{ik} G_{ij}^{ik} - 2W_i^T \sigma_i'^{ik} G_j^{ik}) (\sigma_j'^{ik})^T \right\| \right),\end{aligned}$$

$$\begin{aligned}\iota_3 &\triangleq \max_{i,j} \left( \sup_{Z \in \mathcal{Z}} \left\| \frac{1}{2} \sum_{i,j=1}^N (W_i^T \nabla_x \sigma_i + \nabla_x \epsilon_i) G_j \nabla_x \epsilon_j^T \right. \right. \\ &\quad \left. \left. - \frac{1}{4} \sum_{i,j=1}^N (2W_j^T \nabla_x \sigma_j + \nabla_x \epsilon_j) G_{ij} \nabla_x \epsilon_j^T \right\| \right),\end{aligned}$$

$$\iota_4 \triangleq \max_{i,j} \left( \sup_{Z \in \mathcal{Z}} \left\| \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T \right\| \right), \quad \iota_{5i} \triangleq \frac{k_{c1i} L_Y \bar{\epsilon}_i \bar{\theta}}{4\sqrt{v_i \underline{\Gamma}_i}},$$

$$\iota_{6i} \triangleq \frac{k_{c1i} L_Y \bar{W}_i \bar{\sigma}_i}{4\sqrt{v_i \underline{\Gamma}_i}}, \quad \iota_{7i} \triangleq \frac{k_{c2i} \max_k \|\sigma_i^{ik} Y^{ik}\| \bar{W}_i}{4\sqrt{v_i \underline{\Gamma}_i}},$$

$$\iota_8 \triangleq \sum_{i=1}^N \frac{(k_{c1i} + k_{c2i}) \bar{W}_i \iota_4}{8\sqrt{v_i \underline{\Gamma}_i}}, \quad \iota_{9i} \triangleq (\iota_1 N + (k_{a2i} + \iota_8) \bar{W}_i),$$

$$\iota_{10i} \triangleq \frac{k_{c1i} \sup_{Z \in \mathcal{Z}} \|\Delta_i\| + k_{c2i} \max_k \|\Delta_i^k\|}{2\sqrt{v_i \underline{\Gamma}_i}},$$

$$\nu_l \triangleq \min \left( \frac{q_i}{2}, \frac{k_{c2i} \underline{c}_{xi}}{4}, \underline{k}_x, \frac{2k_{a1i} + k_{a2i}}{8}, \frac{k_\theta \underline{y}}{2} \right),$$

$$\iota \triangleq \sum_{i=1}^N \left( \frac{2\iota_{9i}^2}{2k_{a1i} + k_{a2i}} + \frac{\iota_{10i}^2}{k_{c2i} \underline{c}_{xi}} \right) + \iota_3, \quad (4.60)$$

where  $q_i$  denotes the minimum eigenvalue of  $Q_i$ ,  $\underline{y}$  denotes the minimum eigenvalue of  $\sum_{j=1}^{M_o} Y_j^T Y_j$ ,  $\underline{k}_x$  denotes the minimum eigenvalue of  $k_x$ , and the suprema exist since  $\frac{\omega_i}{\rho_i}$  is uniformly bounded for all  $Z$ , and the functions  $G_i$ ,  $G_{ij}$ ,  $\sigma_i$ , and  $\nabla_x \epsilon_i$  are continuous. In (4.60),  $L_Y \in \mathbb{R}_{\geq 0}$  denotes the Lipschitz constant such that  $\|Y(\varpi)\| \leq L_Y \|\varpi\|$ ,  $\forall \varpi \in \mathcal{Z}_n$ , where  $\mathcal{Z}_n$  denotes the projection of  $\mathcal{Z}$  on  $\mathbb{R}^n$ . The sufficient conditions for uniformly ultimately bounded convergence are derived based on the subsequent stability analysis as

$$\begin{aligned} q_i &> 2\iota_{5i}, \\ k_{c2i} \underline{c}_{xi} &> 2\iota_{5i} + 2\zeta_1 \iota_{7i} + \iota_2 \zeta_2 N + k_{a1i} + 2\zeta_3 \iota_{6i} \bar{Z}, \\ 2k_{a1i} + k_{a2i} &> 4\iota_8 + \frac{2\iota_2 N}{\zeta_2}, \\ k_\theta \underline{y} &> \frac{2\iota_{7i}}{\zeta_1} + 2\frac{\iota_{6i}}{\zeta_3} \bar{Z}, \end{aligned} \quad (4.61)$$

where  $\bar{Z} \triangleq \underline{y}^{-1} \left( \bar{v} \left( \max \left( \|Z(t_0)\|, \frac{\iota}{v_l} \right) \right) \right)$  and  $\zeta_1, \zeta_2, \zeta_3 \in \mathbb{R}$  are known positive adjustable constants.

Since the neural network function approximation error and the Lipschitz constant  $L_Y$  depend on the compact set that contains the state trajectories, the compact set needs to be established before the gains can be selected using (4.61). Based on the subsequent stability analysis, an algorithm is developed in Appendix A.2.2 to compute the required compact set (denoted by  $\mathcal{Z}$ ) based on the initial conditions. Since the constants  $\iota$  and  $v_l$  depend on  $L_Y$  only through the products  $L_Y \bar{\epsilon}_i$  and  $L_Y \zeta_3$ , Algorithm A.3 ensures that

$$\frac{\iota}{v_l} \leq \frac{1}{2} \text{diam}(\mathcal{Z}), \quad (4.62)$$

where  $\text{diam}(\mathcal{Z})$  denotes the diameter of the set  $\mathcal{Z}$ .

**Theorem 4.9** *Provided Assumptions 4.7–4.8 hold and the control gains satisfy the sufficient conditions in (4.61), where the constants in (4.60) are computed based on the compact set  $\mathcal{Z}$  selected using Algorithm A.3, the system identifier in (4.46) along with the adaptive update law in (4.48) and the controllers in (3.114) along with the adaptive update laws in (4.53) and (4.54) ensure that the state  $x(\cdot)$ , the state estimation error  $\tilde{x}(\cdot)$ , the critic weight estimation errors  $\tilde{W}_{ci}(\cdot)$ , and the actor weight estimation errors  $\tilde{W}_{ai}(\cdot)$  are uniformly ultimately bounded, resulting in uniformly ultimately bounded convergence of the policies  $\hat{u}_i$  to the feedback-Nash equilibrium policies  $u_i^*$ .*

*Proof* The derivative of the candidate Lyapunov function in (4.58) along the trajectories of (3.82), (4.49), (4.50), (4.53), and (4.54) is given by

$$\begin{aligned} \dot{V}_L = & \sum_{i=1}^N \left( \nabla_x V_i^* \left( f + \sum_{j=1}^N g_j u_j \right) \right) + \tilde{x}^T \left( Y\tilde{\theta} - k_x \tilde{x} \right) \\ & + \sum_{i=1}^N \tilde{W}_{ci}^T \left( \frac{k_{cli}\omega_i}{\rho_i} \hat{\delta}_{ti} + \frac{k_{c2i}}{M_{xi}} \sum_{i=1}^{M_{xi}} \frac{\omega_i^k}{\rho_i^k} \hat{\delta}_{ti}^k \right) - \frac{1}{2} \sum_{i=1}^N \tilde{W}_{ci}^T \left( \beta_i \Gamma_i^{-1} - k_{cli} \frac{\omega_i \omega_i^T}{\rho_i^2} \right) \tilde{W}_{ci} \\ & + \tilde{\theta}^T \left( -Y^T \tilde{x} - k_\theta \left( \sum_{j=1}^{M_\theta} Y_j^T Y_j \right) \tilde{\theta} \right) - \sum_{i=1}^N \tilde{W}_{ai}^T \left( -k_{a1i} \left( \hat{W}_{ai}^T - \hat{W}_{ci}^T \right) - k_{a2i} \hat{W}_{ai}^T \right. \\ & \left. + \frac{1}{4} \sum_{j=1}^N k_{cli} \hat{W}_{ci}^T \frac{\omega_i}{\rho_i} \hat{W}_{aj}^T \nabla_x \sigma_j G_{ij} \nabla_x \sigma_j^T + \frac{1}{4} \sum_{k=1}^{M_{xi}} \sum_{j=1}^N \frac{k_{c2i}}{M_{xi}} \hat{W}_{ci}^T \frac{\omega_i^k}{\rho_i^k} \hat{W}_{aj}^T \sigma_j^{ik} G_{ij}^k \left( \sigma_j^{ik} \right)^T \right). \end{aligned} \quad (4.63)$$

Substituting the unmeasurable forms of the Bellman errors from (4.56) and (4.57) into (4.63) and using the triangle inequality, the Cauchy–Schwarz inequality, and Young’s inequality, the Lyapunov derivative in (4.63) can be bounded as

$$\begin{aligned}
\dot{V} \leq & -\sum_{i=1}^N \frac{q_i}{2} \|x\|^2 - \sum_{i=1}^N \frac{k_{c2i}\underline{c}_{xi}}{2} \|\tilde{W}_{ci}\|^2 - \underline{k}_x \|\tilde{x}\|^2 - \frac{k_\theta \underline{y}}{2} \|\tilde{\theta}\|^2 - \sum_{i=1}^N \left( \frac{2k_{a1i} + k_{a2i}}{4} \right) \|\tilde{W}_{ai}\|^2 \\
& - \sum_{i=1}^N \left( \frac{k_{c2i}\underline{c}_{xi}}{2} - \iota_{5i} - \zeta_1 \iota_{7i} - \frac{1}{2} \iota_2 \zeta_2 N - \frac{1}{2} k_{a1i} - \zeta_3 \iota_{6i} \|x\| \right) \|\tilde{W}_{ci}\|^2 \\
& + \sum_{i=1}^N \left( \frac{k_\theta \underline{y}}{2} - \frac{\iota_{7i}}{\zeta_1} - \frac{\iota_{6i}}{\zeta_3} \|x\| \right) \|\tilde{\theta}\|^2 + \sum_{i=1}^N \iota_{9i} \|\tilde{W}_{ai}\| + \sum_{i=1}^N \iota_{10i} \|\tilde{W}_{ci}\| \\
& + \sum_{i=1}^N \left( \frac{2k_{a1i} + k_{a2i}}{4} - \iota_8 - \frac{\iota_{2N}}{2\zeta_2} \right) \|\tilde{W}_{ai}\|^2 + \iota_3 - \sum_{i=1}^N \left( \frac{q_i}{2} - \iota_{5i} \right) \|x\|^2. \tag{4.64}
\end{aligned}$$

Provided the sufficient conditions in (4.61) hold and the conditions

$$\begin{aligned}
\frac{k_{c2i}\underline{c}_{xi}}{2} &> \iota_{5i} + \zeta_1 \iota_{7i} + \frac{1}{2} \iota_2 \zeta_2 N + \frac{1}{2} k_{a1i} + \zeta_3 \iota_{6i} \|x\|, \\
\frac{k_\theta \underline{y}}{2} &> \frac{\iota_{7i}}{\zeta_1} + \frac{\iota_{6i}}{\zeta_3} \|x\|, \tag{4.65}
\end{aligned}$$

hold for all  $Z \in \mathcal{Z}$ , completing the squares in (4.64), the bound on the Lyapunov derivative can be expressed as

$$\begin{aligned}
\dot{V} \leq & -\sum_{i=1}^N \frac{q_i}{2} \|x\|^2 - \sum_{i=1}^N \frac{k_{c2i}\underline{c}_{xi}}{4} \|\tilde{W}_{ci}\|^2 - \underline{k}_x \|\tilde{x}\|^2 - \sum_{i=1}^N \left( \frac{2k_{a1i} + k_{a2i}}{8} \right) \|\tilde{W}_{ai}\|^2 \\
& - \frac{k_\theta \underline{y}}{2} \|\tilde{\theta}\|^2 + \iota \\
\leq & -v_l \|Z\|, \quad \forall \|Z\| > \frac{\iota}{v_l}, \quad Z \in \mathcal{Z}. \tag{4.66}
\end{aligned}$$

Using (4.59), (4.62), and (4.66), [40, Theorem 4.18] can be invoked to conclude that  $\limsup_{t \rightarrow \infty} \|Z(t)\| \leq v^{-1}(\bar{v}(\frac{\iota}{v}))$ . Furthermore, the system trajectories are bounded as  $\|Z(t)\| \leq \bar{Z}$  for all  $t \in \mathbb{R}_{\geq 0}$ . Hence, the conditions in (4.61) are sufficient for the conditions in (4.65) to hold for all  $t \in \mathbb{R}_{\geq 0}$ .

The error between the feedback-Nash equilibrium policy and the approximate policy can be bounded above as

$$\|u_i^* - \hat{u}_i\| \leq \frac{1}{2} \|R_{ii}\| \overline{g_i \sigma_i} \left( \|\tilde{W}_{ai}\| + \bar{\epsilon}'_i \right),$$

for all  $i = 1, \dots, N$ , where  $\overline{g_i} \triangleq \sup_x \|g_i(x)\|$ . Since the weights  $\tilde{W}_{ai}$  are uniformly ultimately bounded, uniformly ultimately bounded convergence of the approximate policies to the feedback-Nash equilibrium policies is obtained.  $\square$

*Remark 4.10* The closed-loop system analyzed using the candidate Lyapunov function in (4.58) is a switched system. The switching happens when the history stack

is updated and when the least-squares regression matrices  $I_i$  reach their saturation bound. Similar to least-squares-based adaptive control (cf. [47]), (4.58) can be shown to be a common Lyapunov function for the regression matrix saturation, and the use of a singular value maximizing algorithm to update the history stack ensures that (4.58) is a common Lyapunov function for the history stack updates (cf. [31]). Since (4.58) is a common Lyapunov function, (4.59), (4.62), and (4.66) establish uniformly ultimately bounded convergence of the switched system.

#### 4.5.4 Simulation

To portray the performance of the developed approach, the concurrent learning-based adaptive technique is applied to the nonlinear control-affine system

$$\dot{x} = f(x) + g_1(x)u_1 + g_2(x)u_2, \quad (4.67)$$

where  $x \in \mathbb{R}^2$ ,  $u_1, u_2 \in \mathbb{R}$ , and

$$f = \begin{bmatrix} x_2 - 2x_1 \\ \left( -\frac{1}{2}x_1 - x_2 + \frac{1}{4}x_2(\cos(2x_1) + 2)^2 \right. \\ \left. + \frac{1}{4}x_2(\sin(4x_1^2) + 2)^2 \right) \end{bmatrix},$$

$$g_1 = \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix}, \quad g_2 = \begin{bmatrix} 0 \\ \sin(4x_1^2) + 2 \end{bmatrix}.$$

The value function has the structure shown in (3.83) with the weights  $Q_1 = 2Q_2 = 2I_2$  and  $R_{11} = R_{12} = 2R_{21} = 2R_{22} = 2$ . The system identification protocol given in Sect. 4.5.1 and the concurrent learning-based scheme given in Sect. 4.5.2 are implemented simultaneously to provide an approximate online feedback-Nash equilibrium solution to the given nonzero-sum two-player game.

The control-affine system in (4.67) is selected for this simulation because it is constructed using the converse Hamilton–Jacobi approach [48] where the analytical feedback-Nash equilibrium solution to the nonzero-sum game is

$$V_1^* = \begin{bmatrix} 0.5 \\ 0 \\ 1 \end{bmatrix}^T \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{bmatrix}, \quad V_2^* = \begin{bmatrix} 0.25 \\ 0 \\ 0.5 \end{bmatrix}^T \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{bmatrix},$$

and the feedback-Nash equilibrium control policies for Player 1 and Player 2 are

$$u_1^* = -\frac{1}{2}R_{11}^{-1}g_1^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.5 \\ 0 \\ 1 \end{bmatrix}, \quad u_2^* = -\frac{1}{2}R_{22}^{-1}g_2^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.25 \\ 0 \\ 0.5 \end{bmatrix}.$$

Since the analytical solution is available, the performance of the developed method can be evaluated by comparing the obtained approximate solution against the analytical solution.

The dynamics are linearly parameterized as  $f = Y(x)\theta$ , where

$$Y(x) = \begin{bmatrix} x_2 & 0 \\ x_1 & 0 \\ 0 & x_1 \\ 0 & x_2 \\ 0 & x_2(\cos(2x_1) + 2)^2 \\ 0 & x_2(\sin(4x_1^2) + 2)^2 \end{bmatrix}^T$$

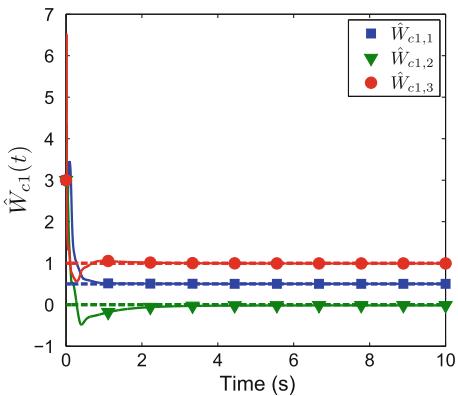
is known and the constant vector of parameters  $\theta = [1, -2, -\frac{1}{2}, -1, \frac{1}{4}, -\frac{1}{4}]^T$  is assumed to be unknown. The initial guess for  $\theta$  is selected as  $\hat{\theta}(t_0) = 0.5 * [1, 1, 1, 1, 1, 1]^T$ . The system identification gains are selected as  $k_x = 5$ ,  $\Gamma_\theta = \text{diag}(20, 20, 100, 100, 60, 60)$ , and  $k_\theta = 1.5$ . A history stack of thirty points is selected using a singular value maximizing algorithm (cf. [31]) for the concurrent learning-based update law in (4.48), and the state derivatives are estimated using a fifth order Savitzky–Golay filter (cf. [41]). Based on the structure of the feedback-Nash equilibrium value functions, the basis function for value function approximation is selected as  $\sigma = [x_1^2, x_1x_2, x_2^2]^T$ , and the adaptive learning parameters and initial conditions are shown for both players in Table 4.1. Twenty-five points lying on a  $5 \times 5$  grid around the origin are selected for the concurrent learning-based update laws in (4.53) and (4.54).

Figures 4.30, 4.31, 4.32 and 4.33 show the rapid convergence of the actor and critic weights to the approximate feedback-Nash equilibrium values for both players, resulting in the value functions and control policies

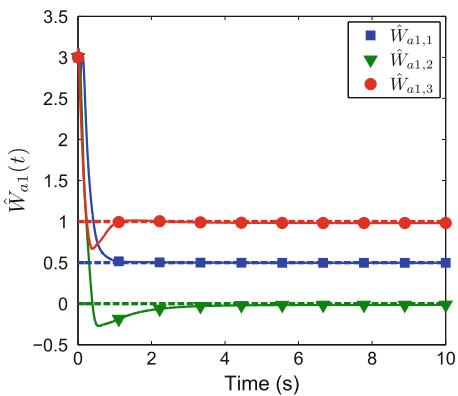
**Table 4.1** Approximate dynamic programming learning gains and initial conditions

Learning Gains			Initial Conditions		
	Player 1	Player 2		Player 1	Player 2
$v$	0.005	0.005	$\widehat{W}_c(t_0)$	$[3, 3, 3]^T$	$[3, 3, 3]^T$
$k_{c1}$	1	1	$\widehat{W}_a(t_0)$	$[3, 3, 3]^T$	$[3, 3, 3]^T$
$k_{c2}$	1.5	1	$\Gamma(t_0)$	$100I_3$	$100I_3$
$k_{a1}$	10	10	$x(t_0)$	$[1, 1]^T$	$[1, 1]^T$
$k_{a2}$	0.1	0.1	$\hat{x}(t_0)$	$[0, 0]^T$	$[0, 0]^T$
$\beta$	3	3			
$\bar{\Gamma}$	10,000	10,000			

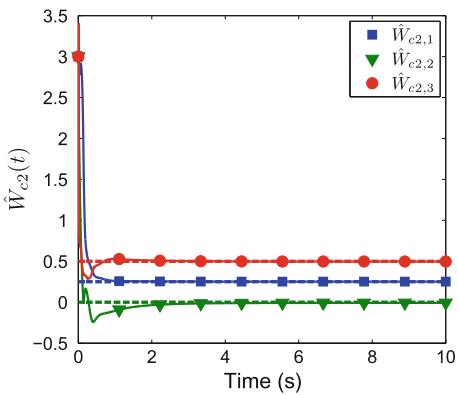
**Fig. 4.30** Player 1 critic weight estimates. Dashed lines indicate the ideal values (reproduced with permission from [16], ©2014, IEEE)



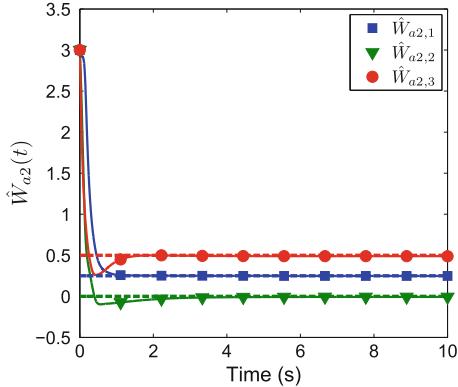
**Fig. 4.31** Player 1 actor weight estimates. Dashed lines indicate the ideal values (reproduced with permission from [16], ©2014, IEEE)



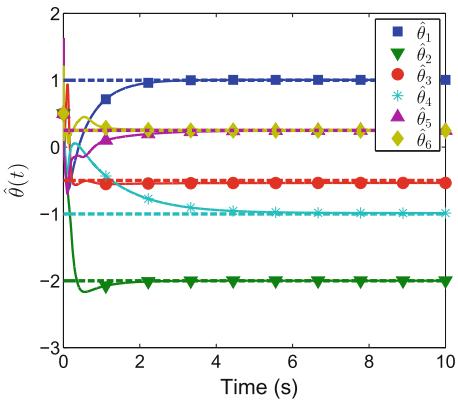
**Fig. 4.32** Player 2 critic weight estimates. Dashed lines indicate the ideal values (reproduced with permission from [16], ©2014, IEEE)



**Fig. 4.33** Player 2 actor weight estimates. Dashed lines indicate the ideal values (reproduced with permission from [16], ©2014, IEEE)



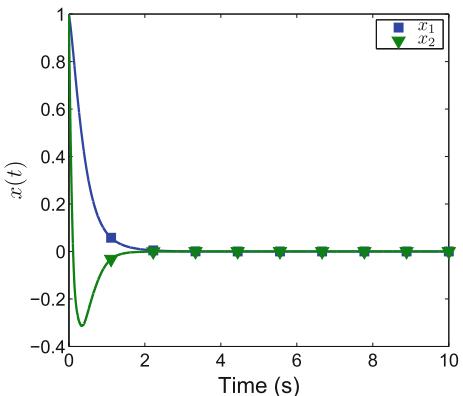
**Fig. 4.34** Drift parameter estimates. Dashed lines indicate the ideal values (reproduced with permission from [16], ©2014, IEEE)



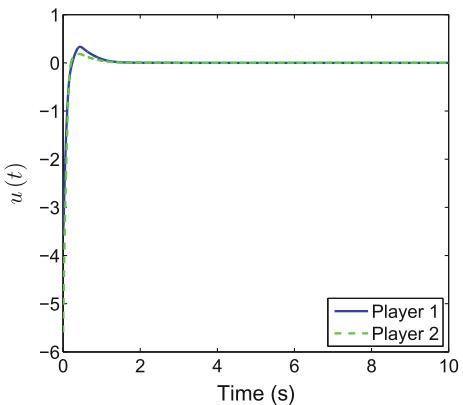
$$\begin{aligned}\hat{V}_1 &= \begin{bmatrix} 0.5021 \\ -0.0159 \\ 0.9942 \end{bmatrix}^T \sigma, \quad \hat{V}_2 = \begin{bmatrix} 0.2510 \\ -0.0074 \\ 0.4968 \end{bmatrix}^T \sigma, \\ \hat{u}_1 &= -\frac{1}{2} R_{11}^{-1} g_1^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.4970 \\ -0.0137 \\ 0.9810 \end{bmatrix}, \\ \hat{u}_2 &= -\frac{1}{2} R_{22}^{-1} g_2^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.2485 \\ -0.0055 \\ 0.4872 \end{bmatrix}.\end{aligned}$$

Figure 4.34 demonstrates that (without the injection of a persistently exciting signal) the system identification parameters also approximately converged to the correct values. The state and control signal trajectories are displayed in Figs. 4.35 and 4.36.

**Fig. 4.35** State trajectory convergence to the origin (reproduced with permission from [16], ©2014, IEEE)



**Fig. 4.36** Control trajectories of Player 1 and Player 2 (reproduced with permission from [16], ©2014, IEEE)



## 4.6 Background and Further Reading

Online implementation of reinforcement learning is comparable to adaptive control (cf., [2, 6, 49–52] and the references therein). In adaptive control, the estimates for the uncertain parameters in the plant model are updated using the tracking error as a performance metric; whereas, in online reinforcement learning-based techniques, estimates for the uncertain parameters in the value function are updated using the Bellman error as a performance metric. Typically, to establish regulation or tracking, adaptive control methods do not require the adaptive estimates to converge to the true values. However, convergence of the reinforcement learning-based controller to a neighborhood of the optimal controller requires convergence of the parameter estimates to a neighborhood of their ideal values.

Results such as [7, 10, 37, 44, 53–56] solve optimal tracking and differential game problems for linear and nonlinear systems online, where persistence of excitation of the error states is used to establish convergence. In general, it is impossible to guarantee persistence of excitation a priori. As a result, a probing signal designed using trial and error is added to the controller to ensure persistence of excitation. However, the probing signal is typically not considered in the stability analysis.

Contemporary results on data-driven approximate dynamic programming methods include methods to solve set-point and output regulation [11, 57–61], trajectory tracking [56, 62, 63], and differential game [64–69] problems.

## References

1. Mehta P, Meyn S (2009) Q-learning and pontryagin's minimum principle. In: Proceedings of IEEE conference on decision control, pp 3598–3605
2. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92
3. Vrabie D (2010) Online adaptive optimal control for continuous-time systems. PhD thesis, University of Texas at Arlington
4. Al-Tamimi A, Lewis FL, Abu-Khalaf M (2007) Model-free q-learning designs for linear discrete-time zero-sum games with application to  $H_\infty$  control. *Automatica* 43:473–481
5. Lewis FL, Vrabie D (2009) Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits Syst Mag* 9(3):32–50
6. Vamvoudakis KG, Lewis FL (2010) Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica* 46(5):878–888
7. Vamvoudakis KG, Lewis FL (2011) Multi-player non-zero-sum games: Online adaptive learning solution of coupled hamilton-jacobi equations. *Automatica* 47:1556–1569
8. Vamvoudakis KG, Lewis FL, Hudas GR (2012) Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality. *Automatica* 48(8):1598–1611
9. Modares H, Lewis FL, Naghibi-Sistani MB (2013) Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks. *IEEE Trans Neural Netw Learn Syst* 24(10):1513–1525
10. Kiumarsi B, Lewis FL, Modares H, Karimpour A, Naghibi-Sistani MB (2014) Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. *Automatica* 50(4):1167–1175
11. Modares H, Lewis FL, Naghibi-Sistani MB (2014) Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica* 50(1):193–202
12. Modares H, Lewis FL (2014) Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning. *Automatica* 50(7):1780–1792
13. Kamalapurkar R, Walters P, Dixon WE (2013) Concurrent learning-based approximate optimal regulation. In: Proceedings of IEEE conference on decision control, Florence, IT, pp 6256–6261
14. Kamalapurkar R, Andrews L, Walters P, Dixon WE (2014) Model-based reinforcement learning for infinite-horizon approximate optimal tracking. In: Proceedings of IEEE conference on decision control, Los Angeles, CA, pp 5083–5088
15. Kamalapurkar R, Klotz J, Dixon WE (2014) Model-based reinforcement learning for on-line feedback-Nash equilibrium solution of N-player nonzero-sum differential games. In: Proceedings of the American control conference, pp 3000–3005

16. Kamalapurkar R, Klotz J, Dixon WE (2014) Concurrent learning-based online approximate feedback Nash equilibrium solution of N-player nonzero-sum differential games. *IEEE/CAA J Autom Sin* 1(3):239–247
17. Kamalapurkar R, Rosenfeld JA, Dixon WE (2015) State following (StaF) kernel functions for function approximation Part II: Adaptive dynamic programming. In: Proceedings of the American control conference, pp 521–526
18. Kamalapurkar R, Walters P, Dixon WE (2016) Model-based reinforcement learning for approximate optimal regulation. *Automatica* 64:94–104
19. Kamalapurkar R (2014) Model-based reinforcement learning for online approximate optimal control. PhD thesis, University of Florida
20. Kamalapurkar R, Rosenfeld J, Dixon WE (2016) Efficient model-based reinforcement learning for approximate online optimal control. *Automatica* 74:247–258
21. Kamalapurkar R, Klotz JR, Walters P, Dixon WE (2018) Model-based reinforcement learning for differential graphical games. *IEEE Trans Control Netw Syst* 5:423–433
22. Kamalapurkar R, Andrews L, Walters P, Dixon WE (2017) Model-based reinforcement learning for infinite-horizon approximate optimal tracking. *IEEE Trans Neural Netw Learn Syst* 28(3):753–758
23. Singh SP (1992) Reinforcement learning with a hierarchy of abstract models. *AAAI Natl. Conf. Artif. Intell.* 92:202–207
24. Atkeson CG, Schaal S (1997) Robot learning from demonstration. In: International conference on machine learning 97:12–20
25. Abbeel P, Quigley M, Ng AY (2006) Using inaccurate models in reinforcement learning. In: International conference on machine learning. ACM, New York, NY, USA, pp 1–8
26. Deisenroth MP (2010) Efficient reinforcement learning using Gaussian processes. KIT Scientific Publishing, Karlsruhe
27. Mitrovic D, Klanke S, Vijayakumar S (2010) Adaptive optimal feedback control with learned internal dynamics models. In: Sigaud O, Peters J (eds) From motor learning to interaction learning in robots, vol 264. Studies in computational intelligence. Springer, Berlin, pp 65–84
28. Deisenroth MP, Rasmussen CE (2011) Pilco: A model-based and data-efficient approach to policy search. In: International Conference on Machine Learning, pp 465–472
29. Chowdhary G (2010) Concurrent learning for convergence in adaptive control without persistency of excitation. PhD thesis, Georgia Institute of Technology
30. Chowdhary GV, Johnson EN (2011) Theory and flight-test validation of a concurrent-learning adaptive controller. *J Guid Control Dyn* 34(2):592–607
31. Chowdhary G, Yucelen T, Mühllegg M, Johnson EN (2013) Concurrent learning adaptive control of linear systems with exponentially convergent bounds. *Int J Adapt Control Signal Process* 27(4):280–301
32. Parikh A, Kamalapurkar R, Chen HY, Dixon WE (2015) Homography based visual servo control with scene reconstruction. In: Proceedings of IEEE Conference on Decision Control, pp 6972–6977
33. Kamalapurkar R, Reish B, Chowdhary G, Dixon WE (2017) Concurrent learning for parameter estimation using dynamic state-derivative estimators. *IEEE Trans Autom Control* 62:3594–3601
34. Parikh A, Kamalapurkar R, Dixon WE (2015) Integral concurrent learning: adaptive control with parameter convergence without PE or state derivatives. [arXiv:1512.03464](https://arxiv.org/abs/1512.03464)
35. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge
36. Konda V, Tsitsiklis J (2004) On actor-critic algorithms. *SIAM J Control Optim* 42(4):1143–1166
37. Dierks T, Jagannathan S (2009) Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics. In: Proceedings of IEEE Conference on Decision Control, Shanghai, CN, pp 6750–6755
38. Vamvoudakis KG, Lewis FL (2009) Online synchronous policy iteration method for optimal control. In: Yu W (ed) Recent advances in intelligent control systems. Springer, Berlin, pp 357–374

39. Dierks T, Jagannathan S (2010) Optimal control of affine nonlinear continuous-time systems. In: Proceedings of the American control conference, pp 1568–1573
40. Khalil HK (2002) Nonlinear systems, 3rd edn. Prentice Hall, Upper Saddle River
41. Savitzky A, Golay MJE (1964) Smoothing and differentiation of data by simplified least squares procedures. *Anal Chem* 36(8):1627–1639
42. Garg D, Hager WW, Rao AV (2011) Pseudospectral methods for solving infinite-horizon optimal control problems. *Automatica* 47(4):829–837
43. Kirk D (2004) Optimal control theory: an introduction. Dover, Mineola
44. Kamalapurkar R, Dinh H, Bhasin S, Dixon WE (2015) Approximate optimal trajectory tracking for continuous-time nonlinear systems. *Automatica* 51:40–48
45. Lewis FL, Jagannathan S, Yesildirak A (1998) Neural network control of robot manipulators and nonlinear systems. CRC Press, Philadelphia
46. Lewis FL, Vrabie D, Syrmos VL (2012) Optimal Control, 3rd edn. Wiley, Hoboken
47. Ioannou P, Sun J (1996) Robust adaptive control. Prentice Hall, New Jersey
48. Nevistic V, Prims JA (1996) Constrained nonlinear optimal control: a converse HJB approach. Technical Report CIT-CDS 96-021, California Institute of Technology, Pasadena, CA 91125
49. Padhi R, Unnikrishnan N, Wang X, Balakrishnan S (2006) A single network adaptive critic (SNAC) architecture for optimal control synthesis for a class of nonlinear systems. *Neural Netw* 19(10):1648–1660
50. He P, Jagannathan S (2007) Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. *IEEE Trans Syst Man Cybern Part B Cybern* 37(2):425–436
51. Zhang H, Wei Q, Luo Y (2008) A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy hdp iteration algorithm. *IEEE Trans Syst Man Cybern Part B Cybern* 38(4):937–942
52. Zhang H, Liu D, Luo Y, Wang D (2013) Adaptive dynamic programming for control algorithms and stability. Communications and control engineering. Springer, London
53. Johnson M, Bhasin S, Dixon WE (2011) Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm. In: Proceedings of conference on decision and control, pp 142–147
54. Modares H, Lewis FL (2013) Online solution to the linear quadratic tracking problem of continuous-time systems using reinforcement learning. In: Proceedings of conference on decision and control, Florence, IT, pp 3851–3856
55. Qin C, Zhang H, Luo Y (2014) Online optimal tracking control of continuous-time linear systems with unknown dynamics by using adaptive dynamic programming. *Int J Control* 87(5):1000–1009
56. Modares H, Lewis FL, Jiang ZP (2015)  $H\infty$  tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. *IEEE Trans Neural Netw Learn Syst* 26(10):2550–2562
57. Luo B, Wu HN, Huang T, Liu D (2014) Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design. *Automatica* 50:3281–3290
58. Yang X, Liu D, Wei Q (2014) Online approximate optimal control for affine non-linear systems with unknown internal dynamics using adaptive dynamic programming. *IET Control Theory Appl* 8(16):1676–1688
59. Jiang Y, Jiang ZP (2015) Global adaptive dynamic programming for continuous-time nonlinear systems. *IEEE Trans Autom Control* 60(11):2917–2929
60. Bian T, Jiang ZP (2016) Value iteration and adaptive dynamic programming for data-driven adaptive optimal control design. *Automatica* 71:348–360
61. Gao W, Jiang ZP (2016) Adaptive dynamic programming and adaptive optimal output regulation of linear systems. *IEEE Trans Autom Control* 61(12):4164–4169
62. Xiao G, Luo Y, Zhang H, Jiang H (2016) Data-driven optimal tracking control for a class of affine non-linear continuous-time systems with completely unknown dynamics. *IET Control Theory Appl* 10(6):700–710

63. Gao W, Jiang ZP (to appear) Learning-based adaptive optimal tracking control of strict-feedback nonlinear systems. *IEEE Trans Neural Netw Learn Syst*
64. Li H, Liu D, Wang D (2014) Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics. *IEEE Trans Autom Sci Eng* 11(3):706–714
65. Wei Q, Song R, Yan P (2016) Data-driven zero-sum neuro-optimal control for a class of continuous-time unknown nonlinear systems with disturbance using adp. *IEEE Trans Neural Netw Learn Syst* 27(2):444–458
66. Song R, Lewis FL, Wei Q (2017) Off-policy integral reinforcement learning method to solve nonlinear continuous-time multiplayer nonzero-sum games. *IEEE Trans Neural Netw Learn Syst* 28(3):704–713
67. Song R, Wei Q, Song B (2017) Neural-network-based synchronous iteration learning method for multi-player zero-sum games. *Neurocomputing* 242:73–82
68. Vamvoudakis KG, Modares H, Kiumarsi B, Lewis FL (2017) Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online. *IEEE Control Syst* 37(1):33–52
69. Wei Q, Liu D, Lin Q, Song R (2017) Adaptive dynamic programming for discrete-time zero-sum games. *IEEE Trans Neural Netw Learn Syst* 29(4):957–969

# Chapter 5

## Differential Graphical Games



### 5.1 Introduction

Reinforcement learning techniques are valuable not only for optimization but also for control synthesis in complex systems such as a distributed network of cognitive agents. Combined efforts from multiple autonomous agents can yield tactical advantages including improved munitions effects, distributed sensing, detection, and threat response, and distributed communication pipelines. While coordinating behaviors among autonomous agents is a challenging problem that has received mainstream focus, unique challenges arise when seeking optimal autonomous collaborative behaviors. For example, most collaborative control literature focuses on centralized approaches that require all nodes to continuously communicate with a central agent, yielding a heavy communication demand that is subject to failure due to delays, and missing information. Furthermore, the central agent is required to carry enough on-board computational resources to process the data and to generate command signals. These challenges motivate the need to minimize communication for guidance, navigation and control tasks, and to distribute the computational burden among the agents. Since all the agents in a network have independent collaborative or competitive objectives, the resulting optimization problem is a multi-objective optimization problem.

In this chapter (see also, [1]), the objective is to obtain an online forward-in-time feedback-Nash equilibrium solution (cf. [2–7]) to an infinite-horizon formation tracking problem, where each agent desires to follow a mobile leader while the group maintains a desired formation. The agents try to minimize cost functions that penalize their own formation tracking errors and their own control efforts.

For multi-agent problems with decentralized objectives, the desired action by an individual agent depends on the actions and the resulting trajectories of its neighbors; hence, the error system for each agent is a complex nonautonomous dynamical system. Nonautonomous systems, in general, have non-stationary value functions. Since non-stationary functions are difficult to approximate using parameterized function

approximation schemes such as neural networks, designing approximate optimal policies for nonautonomous systems is challenging.

Since the external influence from neighbors renders the dynamics of each agent nonautonomous, optimization in a network of agents presents challenges similar to optimal tracking problems. Using insights gained from the development in Chap. 4, this chapter develops a model-based reinforcement learning technique to generate feedback-Nash equilibrium policies online for agents in a network with cooperative or competitive objectives. In particular, the network of agents is separated into autonomous subgraphs, and the differential game is solved separately on each subgraph.

In addition to control, this chapter also explores applications of differential graphical games in monitoring and intent detection (see also, [8]). Implementing a network of cooperating agents (e.g., flocks of unmanned air vehicles, teams of ground vehicles) helps to ensure mission completion and provides more advanced tactical capabilities. Networks containing agents enacting decentralized control policies, wherein only information from neighboring agents is used to internally make decisions, benefit from autonomy: each agent is encoded with a (possibly disaggregated) tactical mission objective and has no need to maintain contact with a mission coordinator.

However, networked systems must be cognizant of the reliability of neighbors' influence. Network neighbors may be unreliable due to input disturbances, faulty dynamics, or network subterfuge, such as cyber-attacks. Existing monitoring results use an agent's state trajectory to judge its performance. An issue with only using a neighbor's state to judge performance is that, owing to nonlinearities in the dynamics, a small deviation in the control may cause a large deviation away from the expected state of the system at the following time step.. Thus, if judging only by the trajectory of a dynamical system, minimally deviant behavior may be exaggerated during monitoring or significantly deviant behavior may not be noticed. Thus, motivation exists to examine more information than just the state when judging an agent's behavior. The intuition behind considering both state errors and control effort is clear upon recalling that both state errors and control effort are used in common cost functions, such as that in linear quadratic regulators.

One of the contribution of this chapter is the development of a novel metric, based on the Bellman error, which provides a condition for determining if a network neighbor with uncertain nonlinear dynamics is behaving near optimally; furthermore, this monitoring procedure only requires neighbor communication and may be implemented online. The contribution is facilitated by the use of approximate dynamic programming and concurrent learning to approximately determine how close optimality conditions are to being satisfied.

## 5.2 Cooperative Formation Tracking Control of Heterogeneous Agents<sup>1</sup>

### 5.2.1 Graph Theory Preliminaries

Consider a set of  $N$  autonomous agents moving in the state space  $\mathbb{R}^n$ . The control objective is for the agents to maintain a desired formation with respect to a leader. The state of the leader is denoted by  $x_0 \in \mathbb{R}^n$ . The agents are assumed to be on a network with a fixed communication topology modeled as a static directed graph (i.e., digraph).

Each agent forms a node in the digraph. The set of all nodes excluding the leader is denoted by  $\mathcal{N} = \{1, \dots, N\}$  and the leader is denoted by node 0. If node  $i$  can receive information from node  $j$  then there exists a directed edge from the  $j$ th to the  $i$ th node of the digraph, denoted by the ordered pair  $(j, i)$ . Let  $E$  denote the set of all edges. Let there be a positive weight  $a_{ij} \in \mathbb{R}$  associated with each edge  $(j, i)$ . Note that  $a_{ij} \neq 0$  if and only if  $(j, i) \in E$ . The digraph is assumed to have no repeated edges (i.e.,  $(i, i) \notin E, \forall i$ ), which implies  $a_{ii} = 0, \forall i$ . The neighborhood sets of node  $i$  are denoted by  $\mathcal{N}_{-i}$  and  $\mathcal{N}_i$ , defined as  $\mathcal{N}_{-i} \triangleq \{j \in \mathcal{N} \mid (j, i) \in E\}$  and  $\mathcal{N}_i \triangleq \mathcal{N}_{-i} \cup \{i\}$ .

To streamline the analysis, an adjacency matrix  $\mathcal{A} \in \mathbb{R}^{N \times N}$  is defined as  $\mathcal{A} \triangleq [a_{ij} \mid i, j \in \mathcal{N}]$ , a diagonal pinning gain matrix  $\mathcal{A}_0 \in \mathbb{R}^{N \times N}$  is defined as  $\mathcal{A}_0 \triangleq \text{diag}([a_{10}, \dots, a_{N0}])$ , an in-degree matrix  $\mathcal{D} \in \mathbb{R}^{N \times N}$  is defined as  $\mathcal{D} \triangleq \text{diag}(d_i)$ , where  $d_i \triangleq \sum_{j \in \mathcal{N}_i} a_{ij}$ , and a graph Laplacian matrix  $\mathcal{L} \in \mathbb{R}^{N \times N}$  is defined as  $\mathcal{L} \triangleq \mathcal{D} - \mathcal{A}$ . The graph is assumed to have a spanning tree (i.e., given any node  $i$ , there exists a directed path from the leader 0 to node  $i$ ). A node  $j$  is said to be an extended neighbor of node  $i$  if there exists a directed path from node  $j$  to node  $i$ . The extended neighborhood set of node  $i$ , denoted by  $\mathcal{S}_{-i}$ , is defined as the set of all extended neighbors of node  $i$ . Formally,  $\mathcal{S}_{-i} \triangleq \{j \in \mathcal{N} \mid j \neq i \wedge \exists k \leq N, \{j_1, \dots, j_k\} \subset \mathcal{N} \mid \{(j, j_1), (j_1, j_2), \dots, (j_k, i)\} \subset 2^E\}$ . Let  $\mathcal{S}_i \triangleq \mathcal{S}_{-i} \cup \{i\}$ , and let the edge weights be normalized such that  $\sum_j a_{ij} = 1, \forall i \in \mathcal{N}$ . Note that the sub-graphs are nested in the sense that  $\mathcal{S}_j \subseteq \mathcal{S}_i$  for all  $j \in \mathcal{S}_i$ .

### 5.2.2 Problem Formulation

The state  $x_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  of each agent evolves according to the control-affine dynamics

$$\dot{x}_i(t) = f_i(x_i(t)) + g_i(x_i(t)) u_i(t), \quad (5.1)$$

where  $u_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{m_i}$  denotes the control input, and  $f_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$  and  $g_i : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m_i}$  are locally Lipschitz continuous functions. The dynamics of the leader are

---

<sup>1</sup>Parts of the text in this section are reproduced, with permission, from [1], ©2016, IEEE.

assumed to be autonomous (i.e.,  $\dot{x}_0 = f_0(x_0)$ , where  $f_0 : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is a locally Lipschitz continuous function). The function  $f_0$  and the initial condition  $x_0(t_0)$  are selected such that the trajectory  $x_0(\cdot)$  is uniformly bounded.

The control objective is for the agents to maintain a predetermined formation (with respect to an inertial reference frame) around the leader while minimizing their own cost functions. For all  $i \in \mathcal{N}$ , the  $i$ th agent is aware of its constant desired relative position  $x_{dij} \in \mathbb{R}^n$  with respect to all its neighbors  $j \in \mathcal{N}_{-i}$ , such that the desired formation is realized when  $x_i(t) - x_j(t) \rightarrow x_{dij}$ ,  $\forall i, j \in \mathcal{N}$ . The vectors  $x_{dij}$  are assumed to be fixed in an inertial reference frame (i.e., the final desired formation is rigid and its motion in an inertial reference frame can be described as pure translation).

To facilitate the control design, the formation is expressed in terms of a set of constant vectors  $\{x_{di0} \in \mathbb{R}^n\}_{i \in \mathcal{N}}$  where each  $x_{di0}$  denotes the constant final desired position of agent  $i$  with respect to the leader. The vectors  $\{x_{di0}\}_{i \in \mathcal{N}}$  are unknown to the agents not connected to the leader, and the known desired inter agent relative position can be expressed in terms of  $\{x_{di0}\}_{i \in \mathcal{N}}$  as  $x_{dij} = x_{di0} - x_{dj0}$ . The control objective is thus satisfied when  $x_i(t) \rightarrow x_{di0} + x_0(t)$ ,  $\forall i \in \mathcal{N}$ . To quantify the objective, a local neighborhood tracking error signal is defined as

$$e_i(t) = \sum_{j \in \{0\} \cup \mathcal{N}_{-i}} a_{ij} ((x_i(t) - x_j(t)) - x_{dij}). \quad (5.2)$$

To facilitate the analysis, the error signal in (5.2) is expressed in terms of the unknown leader-relative desired positions as

$$e_i(t) = \sum_{j \in \{0\} \cup \mathcal{N}_{-i}} a_{ij} ((x_i(t) - x_{di0}) - (x_j(t) - x_{dj0})). \quad (5.3)$$

Stacking the error signals in a vector  $\mathcal{E}(t) \triangleq [e_1^T(t), e_2^T(t), \dots, e_N^T(t)]^T \in \mathbb{R}^{nN}$  the equation in (5.3) can be expressed in a matrix form

$$\mathcal{E}(t) = ((\mathcal{L} + \mathcal{A}_0) \otimes \mathbf{I}_n) (\mathcal{X}(t) - \mathcal{X}_d - \mathcal{X}_0(t)), \quad (5.4)$$

where  $\mathcal{X}(t) = [x_1^T(t), x_2^T(t), \dots, x_N^T(t)]^T \in \mathbb{R}^{nN}$ ,  $\mathcal{X}_d = [x_{d10}^T, x_{d20}^T, \dots, x_{dN0}^T]^T \in \mathbb{R}^{nN}$ , and  $\mathcal{X}_0(t) = [x_0^T(t), x_0^T(t), \dots, x_0^T(t)]^T \in \mathbb{R}^{nN}$ . Using (5.4), it can be concluded that provided the matrix  $((\mathcal{L} + \mathcal{A}_0) \otimes \mathbf{I}_n) \in \mathbb{R}^{nN \times nN}$  is nonsingular,  $\|\mathcal{E}(t)\| \rightarrow 0$  implies  $x_i(t) \rightarrow x_{di0} + x_0(t)$ ,  $\forall i$ , and hence, the satisfaction of control objective. The matrix  $((\mathcal{L} + \mathcal{A}_0) \otimes \mathbf{I}_n)$  is nonsingular provided the graph has a spanning tree with the leader at the root [9]. To facilitate the formulation of an optimization problem, the following section explores the functional dependence of the state value functions for the network of agents.

### 5.2.3 Elements of the Value Function

The dynamics for the open-loop neighborhood tracking error are

$$\dot{e}_i(t) = \sum_{j \in \{0\} \cup \mathcal{N}_{-i}} a_{ij} \left( f_i(x_i(t)) + g_i(x_i(t)) u_i(t) - f_j(x_j(t)) - g_j(x_j(t)) u_j(t) \right).$$

Under the temporary assumption that each controller  $u_i(\cdot)$  is an error-feedback controller (i.e.,  $u_i(t) = \hat{u}_i(e_i(t), t)$ ), the error dynamics are expressed as

$$\dot{e}_i(t) = \sum_{j \in \{0\} \cup \mathcal{N}_{-i}} a_{ij} \left( f_i(x_i(t)) + g_i(x_i(t)) \hat{u}_i(e_i(t), t) - f_j(x_j(t)) - g_j(x_j(t)) \hat{u}_j(e_j(t), t) \right).$$

Thus, the error trajectory  $\{e_i(t)\}_{t=t_0}^{\infty}$ , where  $t_0$  denotes the initial time, depends on  $\hat{u}_j(e_j(t), t)$ ,  $\forall j \in \mathcal{N}_i$ . Similarly, the error trajectory  $\{e_j(t)\}_{t=t_0}^{\infty}$  depends on  $\hat{u}_k(e_k(t), t)$ ,  $\forall k \in \mathcal{N}_j$ . Recursively, the trajectory  $\{e_i(t)\}_{t=t_0}^{\infty}$  depends on  $\hat{u}_j(e_j(t), t)$ , and hence, on  $e_j(t)$ ,  $\forall j \in \mathcal{S}_i$ . Thus, even if the controller for each agent is restricted to use local error feedback, the resulting error trajectories are interdependent. In particular, a change in the initial condition of one agent in the extended neighborhood causes a change in the error trajectories corresponding to all the extended neighbors. Consequently, the value function corresponding to an infinite-horizon optimal control problem where each agent tries to minimize  $\int_{t_0}^{\infty} (Q(e_i(\tau)) + R(u_i(\tau))) d\tau$ , where  $Q : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $R : \mathbb{R}^{m_i} \rightarrow \mathbb{R}$  are positive definite functions, is dependent on the error states of all the extended neighbors.

Since the steady-state controllers required for formation tracking are generally nonzero, quadratic total-cost optimal control problems result in infinite costs, and hence, are infeasible. In the following section, relative steady-state controllers are derived to facilitate the formulation of a feasible optimal control problem.

### 5.2.4 Optimal Formation Tracking Problem

When the agents are perfectly tracking the desired trajectory in the desired formation, even though the states of all the agents are different, the time-derivatives of the states of all the agents are identical. Hence, in steady state, the control signal applied by each agent must be such that the time derivatives of the states corresponding to the set of extended neighbors are identical. In particular, the relative control signal  $u_{ij} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{m_i}$  that will keep node  $i$  in its desired relative position with respect to node  $j \in \mathcal{S}_{-i}$  (i.e.,  $x_i(t) = x_j(t) + x_{dij}$ ), must be such that the time derivative of  $x_i(\cdot)$  is the same as the time derivative of  $x_j(\cdot)$ . Using the dynamics of the agent from (5.1) and substituting the desired relative position  $x_j(\cdot) + x_{dij}$  for the state  $x_i(\cdot)$ , the relative control signal  $u_{ij}(\cdot)$  must satisfy

$$f_i(x_j(t) + x_{dij}) + g_i(x_j(t) + x_{dij}) u_{ij}(t) = \dot{x}_j(t). \quad (5.5)$$

The relative steady-state control signal can be expressed in an explicit form provided the following assumption is satisfied.

**Assumption 5.1** The matrix  $g_i(x)$  is full rank for all  $i \in \mathcal{N}$  and for all  $x \in \mathbb{R}^n$ ; furthermore, the relative steady-state control signal expressed as  $u_{ij}(t) = f_{ij}(x_j(t)) + g_{ij}(x_j(t)) u_j(t)$ , satisfies (5.5) along the desired trajectory, where  $f_{ij}(x_j) \triangleq g_i^+(x_j + x_{dij})(f_j(x_j) - f_i(x_j + x_{dij})) \in \mathbb{R}^{m_i}$ ,  $g_{ij}(x_j) \triangleq g_i^+(x_j + x_{dij}) g_j(x_j) \in \mathbb{R}^{m_i \times m_j}$ , where the control effectiveness and the control input relative to the leader are understood to be  $g_0(x) = 0$ ,  $\forall x \in \mathbb{R}^n$  and  $u_{i0} \equiv 0$ ,  $\forall i \in \mathcal{N}$ , respectively, and  $g_i^+(x)$  denotes a pseudoinverse of the matrix  $g_i(x)$ ,  $\forall x \in \mathbb{R}^n$ .

To facilitate the formulation of an optimal formation tracking problem, define the control error  $\mu_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{m_i}$  as

$$\mu_i(t) \triangleq \sum_{j \in \mathcal{N}_{-i} \cup \{0\}} a_{ij} (u_i(t) - u_{ij}(t)). \quad (5.6)$$

In the remainder of this section, the control errors  $\{\mu_i(\cdot)\}$  will be treated as the design variables. To implement the controllers using the designed control errors, it is essential to invert the relationship in (5.6). To facilitate the inversion, let  $\mathcal{S}_i^o \triangleq \{1, \dots, s_i\}$ , where  $s_i \triangleq |\mathcal{S}_i|$ . Let  $\lambda_i : \mathcal{S}_i^o \rightarrow \mathcal{S}_i$  be a bijective map such that  $\lambda_i(1) = i$ . For notational brevity, let  $(\cdot)_{\mathcal{S}_i}$  denote the concatenated vector  $\left[ (\cdot)_{\lambda_i^1}^T, (\cdot)_{\lambda_i^2}^T, \dots, (\cdot)_{\lambda_i^{s_i}}^T \right]^T$ ,  $(\cdot)_{\mathcal{S}_{-i}}$  denote the concatenated vector  $\left[ (\cdot)_{\lambda_i^2}^T, \dots, (\cdot)_{\lambda_i^{s_i}}^T \right]^T$ ,  $\sum^i$  denote  $\sum_{j \in \mathcal{N}_{-i} \cup \{0\}}$ ,  $\lambda_i^j$  denote  $\lambda_i(j)$ ,  $\mathcal{E}_i(t) \triangleq \left[ e_{\mathcal{S}_i}^T(t), x_{\lambda_i^1}^T(t) \right]^T \in \mathbb{R}^{n(s_i+1)}$ , and  $\mathcal{E}_{-i}(t) \triangleq \left[ e_{\mathcal{S}_{-i}}^T(t), x_{\lambda_i^1}^T(t) \right]^T \in \mathbb{R}^{ns_i}$ . Then, the control error vector  $\mu_{\mathcal{S}_i}(t) \in \mathbb{R}^{\sum_{k \in \mathcal{S}_i} m_k}$  can be expressed as

$$\mu_{\mathcal{S}_i}(t) = \mathcal{L}_{g_i}(\mathcal{E}_i(t)) u_{\mathcal{S}_i}(t) - F_i(\mathcal{E}_i(t)), \quad (5.7)$$

where the matrix  $\mathcal{L}_{g_i} : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}^{\sum_{k \in \mathcal{S}_i} m_k \times \sum_{k \in \mathcal{S}_i} m_k}$  is defined by

$$[\mathcal{L}_{g_i}(\mathcal{E}_i)]_{kl} = \begin{cases} -a_{\lambda_i^k \lambda_i^l} g_{\lambda_i^k \lambda_i^l}(x_{\lambda_i^l}), & \forall l \neq k, \\ \sum_{j=1}^{s_i} a_{\lambda_i^k j} I_{m_{\lambda_i^k}}, & \forall l = k, \end{cases}$$

where  $k, l = 1, 2, \dots, s_i$ , and  $F_i : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}^{\sum_{k \in \mathcal{S}_i} m_k}$  is defined as

$$F_i(\mathcal{E}_i) \triangleq \left[ \sum_{j=1}^{s_i} a_{\lambda_i^1 j} f_{\lambda_i^1 j}^T(x_j), \dots, \sum_{j=1}^{s_i} a_{\lambda_i^{s_i} j} f_{\lambda_i^{s_i} j}^T(x_j) \right]^T.$$

**Assumption 5.2** The matrix  $\mathcal{L}_{gi}(\mathcal{E}_i(t))$  is invertible for all  $t \in \mathbb{R}$ .

Assumption 5.2 is a controllability-like condition. Intuitively, Assumption 5.2 requires the control effectiveness matrices to be compatible to ensure the existence of relative control inputs that allow the agents to follow the desired trajectory in the desired formation.

Using Assumption 5.2, the control vector can be expressed as

$$u_{\mathcal{S}_i}(t) = \mathcal{L}_{gi}^{-1}(\mathcal{E}_i(t)) \mu_{\mathcal{S}_i}(t) + \mathcal{L}_{gi}^{-1}(\mathcal{E}_i(t)) F_i(\mathcal{E}_i(t)). \quad (5.8)$$

Let  $\mathcal{L}_{gi}^k$  denote the  $(\lambda_i^{-1}(k))$ th block row of  $\mathcal{L}_{gi}^{-1}$ . Then, the controller  $u_i(\cdot)$  can be implemented as

$$u_i(t) = \mathcal{L}_{gi}^i(\mathcal{E}_i(t)) \mu_{\mathcal{S}_i}(t) + \mathcal{L}_{gi}^i(\mathcal{E}_i(t)) F_i(\mathcal{E}_i(t)), \quad (5.9)$$

and for any  $j \in \mathcal{N}_{-i}$ ,

$$u_j(t) = \mathcal{L}_{gi}^j(\mathcal{E}_i(t)) \mu_{\mathcal{S}_i}(t) + \mathcal{L}_{gi}^j(\mathcal{E}_i(t)) F_i(\mathcal{E}_i(t)). \quad (5.10)$$

Using (5.9) and (5.10), the error and the state dynamics for the agents can be represented as

$$\dot{e}_i(t) = \mathcal{F}_i(\mathcal{E}_i(t)) + \mathcal{G}_i(\mathcal{E}_i(t)) \mu_{\mathcal{S}_i}(t), \quad (5.11)$$

and

$$\dot{x}_i(t) = \mathcal{F}_i(\mathcal{E}_i(t)) + \mathcal{G}_i(\mathcal{E}_i(t)) \mu_{\mathcal{S}_i}(t), \quad (5.12)$$

where

$$\begin{aligned} \mathcal{F}_i(\mathcal{E}_i) &\triangleq \sum^i a_{ij} g_i(x_i) \mathcal{L}_{gi}^i(\mathcal{E}_i) F_i(\mathcal{E}_i) - \sum^i a_{ij} g_j(x_j) \mathcal{L}_{gi}^j(\mathcal{E}_i) F_i(\mathcal{E}_i) \\ &+ \sum^i a_{ij} f_i(x_i) - \sum^i a_{ij} f_j(x_j), \\ \mathcal{G}_i(\mathcal{E}_i) &\triangleq \sum^i a_{ij} \left( g_i(x_i) \mathcal{L}_{gi}^i(\mathcal{E}_i) - g_j(x_j) \mathcal{L}_{gi}^j(\mathcal{E}_i) \right), \\ \mathcal{F}_i(\mathcal{E}_i) &\triangleq f_i(x_i) + g_i(x_i) \mathcal{L}_{gi}^i(\mathcal{E}_i) F_i(\mathcal{E}_i), \\ \mathcal{G}_i(\mathcal{E}_i) &\triangleq g_i(x_i) \mathcal{L}_{gi}^i(\mathcal{E}_i). \end{aligned}$$

Let  $h_{ei}(t; t_0, \mathcal{E}_{i0}, \bar{\mu}_i, \bar{\mu}_{\mathcal{S}_{-i}})$  and  $h_{xi}(t; t_0, \mathcal{E}_{i0}, \bar{\mu}_i, \bar{\mu}_{\mathcal{S}_{-i}})$  denote the trajectories of (5.11) and (5.12), respectively, with the initial time  $t_0$ , initial condition  $\mathcal{E}_i(t_0) = \mathcal{E}_{i0}$ , and policies  $\bar{\mu}_j : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}^{m_i}$ ,  $j \in \mathcal{S}_i$ , and let  $\mathcal{H}_i \triangleq \left[ (h_e)_{\mathcal{S}_i}^T, h_{x\lambda_i}^T \right]^T$ . Define a cost functional

$$J_i(e_i(\cdot), \mu_i(\cdot)) \triangleq \int_0^\infty r_i(e_i(\sigma), \mu_i(\sigma)) d\sigma \quad (5.13)$$

where  $r_i : \mathbb{R}^n \times \mathbb{R}^{m_i} \rightarrow \mathbb{R}_{\geq 0}$  denotes the local cost defined as  $r_i(e_i, \mu_i) \triangleq Q_i(e_i) + \mu_i^T R_i \mu_i$ , where  $Q_i : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  is a positive definite function, and  $R_i \in \mathbb{R}^{m_i \times m_i}$  is a constant positive definite matrix. The objective of each agent is to minimize the cost functional in (5.13). To facilitate the definition of a feedback-Nash equilibrium solution, define value functions  $V_i : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}_{\geq 0}$  as

$$V_i(\mathcal{E}_i; \bar{\mu}_i, \bar{\mu}_{\mathcal{S}_{-i}}) \triangleq \int_t^\infty r_i(h_{ei}(\sigma; t, \mathcal{E}_i, \bar{\mu}_i, \bar{\mu}_{\mathcal{S}_{-i}}), \bar{\mu}_i(\mathcal{H}_i(\sigma; t, \mathcal{E}_i, \bar{\mu}_i, \bar{\mu}_{\mathcal{S}_{-i}}))) d\sigma, \quad (5.14)$$

where  $V_i(\mathcal{E}_i; \bar{\mu}_i, \bar{\mu}_{\mathcal{S}_{-i}})$  denotes the total cost-to-go under the policies  $\bar{\mu}_{\mathcal{S}_i}$ , starting from the state  $\mathcal{E}_i$ . Note that the value functions in (5.14) are time-invariant because the dynamical systems  $\{\dot{e}_j(t) = \mathcal{F}_j(\mathcal{E}_i(t)) + \mathcal{G}_j(\mathcal{E}_i(t)) \mu_{\mathcal{S}_j}(t)\}_{j \in \mathcal{S}_i}$  and  $\dot{x}_i(t) = \mathcal{F}_i(\mathcal{E}_i(t)) + \mathcal{G}_i(\mathcal{E}_i(t)) \mu_{\mathcal{S}_i}(t)$  together form an autonomous dynamical system.

A graphical feedback-Nash equilibrium solution within the subgraph  $\mathcal{S}_i$  is defined as the tuple of policies  $\{\mu_j^* : \mathbb{R}^{n(s_j+1)} \rightarrow \mathbb{R}^{m_j}\}_{j \in \mathcal{S}_i}$  such that the value functions in (5.14) satisfy

$$V_j^*(\mathcal{E}_j) \triangleq V_j(\mathcal{E}_j; \mu_j^*, \mu_{\mathcal{S}_{-j}}^*) \leq V_j(\mathcal{E}_j; \bar{\mu}_j, \mu_{\mathcal{S}_{-j}}^*),$$

$\forall j \in \mathcal{S}_i, \forall \mathcal{E}_i \in \mathbb{R}^{n(s_i+1)}$ , and for all admissible policies  $\bar{\mu}_j$ . Provided a feedback-Nash equilibrium solution exists and the value functions (5.14) are continuously differentiable, the feedback-Nash equilibrium value functions can be characterized in terms of the following system of Hamilton–Jacobi equations:

$$\begin{aligned} \sum_{j \in \mathcal{S}_i} \nabla_{e_j} V_i^*(\mathcal{E}_i) (\mathcal{F}_j(\mathcal{E}_i) + \mathcal{G}_j(\mathcal{E}_i) \mu_{\mathcal{S}_j}^*(\mathcal{E}_i)) + \bar{Q}_i(\mathcal{E}_i) + \mu_i^{*T}(\mathcal{E}_i) R_i \mu_i^*(\mathcal{E}_i) \\ + \nabla_{x_i} V_i^*(\mathcal{E}_i) (\mathcal{F}_i(\mathcal{E}_i) + \mathcal{G}_i(\mathcal{E}_i) \mu_{\mathcal{S}_i}^*(\mathcal{E}_i)) = 0, \quad \forall \mathcal{E}_i \in \mathbb{R}^{n(s_i+1)}, \end{aligned} \quad (5.15)$$

where  $\bar{Q}_i : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}$  is defined as  $\bar{Q}_i(\mathcal{E}_i) \triangleq Q_i(e_i)$ .

**Theorem 5.3** *Provided a feedback-Nash equilibrium solution exists and that the value functions in (5.14) are continuously differentiable, the system of Hamilton–Jacobi equations in (5.15) constitutes a necessary and sufficient condition for feedback-Nash equilibrium.*

*Proof* Consider the cost functional in (5.13), and assume that all the extended neighbors of the  $i$ th agent follow their feedback-Nash equilibrium policies. The value function corresponding to any admissible policy  $\bar{\mu}_i$  can be expressed as

$$V_i \left( \left[ e_i^T, \mathcal{E}_{-i}^T \right]^T ; \bar{\mu}_i, \mu_{\mathcal{S}_{-i}}^* \right) = \int_t^\infty r_i(h_{ei}(\sigma; t, \mathcal{E}_i, \bar{\mu}_i, \mu_{\mathcal{S}_{-i}}^*), \bar{\mu}_i(\mathcal{H}_i(\sigma; t, \mathcal{E}_i, \bar{\mu}_i, \mu_{\mathcal{S}_{-i}}^*))) d\sigma.$$

Treating the dependence on  $\mathcal{E}_{-i}$  as explicit time dependence define

$$\bar{V}_i(e_i, t; \bar{\mu}_i, \mu_{\mathcal{S}_{-i}}^*) \triangleq V_i \left( [e_i^T, \mathcal{E}_{-i}^T(t)]^T ; \bar{\mu}_i, \mu_{\mathcal{S}_{-i}}^* \right), \quad (5.16)$$

$\forall e_i \in \mathbb{R}^n$  and  $\forall t \in \mathbb{R}_{\geq 0}$ . Assuming that the optimal controller that minimizes (5.13) when all the extended neighbors follow their feedback-Nash equilibrium policies exists, and that the optimal value function  $\bar{V}_i^* \triangleq \bar{V}_i(\cdot; \mu_i^*, \mu_{\mathcal{S}_{-i}}^*)$  exists and is continuously differentiable, optimal control theory for single objective optimization problems (cf. [10]) can be used to derive the following necessary and sufficient condition

$$\frac{\partial \bar{V}_i^*(e_i, t)}{\partial e_i} (\mathcal{F}_i(\mathcal{E}_i) + \mathcal{G}_i(\mathcal{E}_i) \mu_{\mathcal{S}_i}^*(\mathcal{E}_i)) + \frac{\partial \bar{V}_i^*(e_i, t)}{\partial t} + Q_i(e_i) + \mu_i^{*T}(\mathcal{E}_i) R_i \mu_i^*(\mathcal{E}_i) = 0. \quad (5.17)$$

Using (5.16), the partial derivative with respect to the state can be expressed as

$$\frac{\partial \bar{V}_i^*(e_i, t)}{\partial e_i} = \frac{\partial V_i^*(\mathcal{E}_i)}{\partial e_i}, \quad (5.18)$$

$\forall e_i \in \mathbb{R}^n$  and  $\forall t \in \mathbb{R}_{\geq 0}$ . The partial derivative with respect to time can be expressed as

$$\begin{aligned} \frac{\partial \bar{V}_i^*(e_i, t)}{\partial t} &= \sum_{j \in \mathcal{S}_{-i}} \frac{\partial V_i^*(\mathcal{E}_i)}{\partial e_j} (\mathcal{F}_j(\mathcal{E}_i) + \mathcal{G}_j(\mathcal{E}_i) \mu_{\mathcal{S}_j}^*(\mathcal{E}_i)) + \frac{\partial V_i^*(\mathcal{E}_i)}{\partial x_i} \mathcal{F}_i(\mathcal{E}_i), \\ &\quad + \frac{\partial V_i^*(\mathcal{E}_i)}{\partial x_i} \mathcal{G}_i(\mathcal{E}_i) \mu_{\mathcal{S}_i}^*(\mathcal{E}_i), \end{aligned} \quad (5.19)$$

$\forall e_i \in \mathbb{R}^n$  and  $\forall t \in \mathbb{R}_{\geq 0}$ . Substituting (5.18) and (5.19) into (5.17) and repeating the process for each  $i$ , the system of Hamilton–Jacobi equations in (5.15) is obtained.  $\square$

Minimizing the Hamilton–Jacobi equations using the stationary condition, the feedback-Nash equilibrium solution is expressed in the explicit form

$$\mu_i^*(\mathcal{E}_i) = -\frac{1}{2} R_i^{-1} \sum_{j \in \mathcal{S}_i} \left( \mathcal{G}_j^i(\mathcal{E}_i) \right)^T \left( \nabla_{e_j} V_i^*(\mathcal{E}_i) \right)^T - \frac{1}{2} R_i^{-1} \left( \mathcal{G}_i^i(\mathcal{E}_i) \right)^T \left( \nabla_{x_i} V_i^*(\mathcal{E}_i) \right)^T, \quad (5.20)$$

$\forall \mathcal{E}_i \in \mathbb{R}^{n(s_i+1)}$ , where  $\mathcal{G}_j^i \triangleq \mathcal{G}_j \frac{\partial \mu_{\mathcal{S}_j}^*}{\partial \mu_i^*}$  and  $\mathcal{G}_i^i \triangleq \mathcal{G}_i \frac{\partial \mu_{\mathcal{S}_i}^*}{\partial \mu_i^*}$ . As it is generally infeasible to obtain an analytical solution to the system of the Hamilton–Jacobi equations in (5.15), the feedback-Nash value functions and the feedback-Nash policies are approximated using parametric approximation schemes as  $\hat{V}_i(\mathcal{E}_i, \hat{W}_{ci})$  and  $\hat{\mu}_i(\mathcal{E}_i, \hat{W}_{ai})$ , respectively, where  $\hat{W}_{ci} \in \mathbb{R}^{L_i}$  and  $\hat{W}_{ai} \in \mathbb{R}^{L_i}$  are parameter estimates. Substitution of the approximations  $\hat{V}_i$  and  $\hat{\mu}_i$  in (5.15) leads to a set of Bellman errors  $\delta_i$  defined as

$$\begin{aligned}
\delta_i \left( \mathcal{E}_i, \hat{W}_{ci}, \left( \hat{W}_a \right)_{\mathcal{S}_i} \right) &\triangleq \hat{\mu}_i^T \left( \mathcal{E}_i, \hat{W}_{ai} \right) R \hat{\mu}_i \left( \mathcal{E}_i, \hat{W}_{ai} \right) + \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \hat{V}_i \left( \mathcal{E}_i, \hat{W}_{ci} \right) \mathcal{F}_j \left( \mathcal{E}_j \right) \\
&+ \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \hat{V}_i \left( \mathcal{E}_i, \hat{W}_{ci} \right) \mathcal{G}_j \left( \mathcal{E}_j \right) \hat{\mu}_{\mathcal{S}_j} \left( \mathcal{E}_j, \left( \hat{W}_a \right)_{\mathcal{S}_j} \right) \\
&+ \nabla_{x_i} \hat{V}_i \left( \mathcal{E}_i, \hat{W}_{ci} \right) \left( \mathcal{F}_i \left( \mathcal{E}_i \right) + \mathcal{G}_i \left( \mathcal{E}_i \right) \hat{\mu}_{\mathcal{S}_i} \left( \mathcal{E}_i, \left( \hat{W}_a \right)_{\mathcal{S}_i} \right) \right) + Q_i \left( e_i \right).
\end{aligned} \tag{5.21}$$

Approximate feedback-Nash equilibrium control is realized by tuning the estimates  $\hat{V}_i$  and  $\hat{\mu}_i$  so as to minimize the Bellman errors  $\delta_i$ . However, computation of  $\delta_i$  and that of  $u_{ij}$  in (5.6) requires exact model knowledge. In the following, a concurrent learning-based system identifier is developed to relax the exact model knowledge requirement and to facilitate the implementation of model-based reinforcement learning via Bellman error extrapolation (cf. [11]). In particular, the developed controllers do not require knowledge of the system drift functions  $f_i$ .

### 5.2.5 System Identification

On any compact set  $\chi \subset \mathbb{R}^n$  the function  $f_i$  can be represented using a neural network as

$$f_i(x) = \theta_i^T \sigma_{\theta i}(x) + \epsilon_{\theta i}(x), \tag{5.22}$$

$\forall x \in \mathbb{R}^n$ , where  $\theta_i \in \mathbb{R}^{P_i+1 \times n}$  denote the unknown output-layer neural network weights,  $\sigma_{\theta i} : \mathbb{R}^n \rightarrow \mathbb{R}^{P_i+1}$  denotes a bounded neural network basis function,  $\epsilon_{\theta i} : \mathbb{R}^n \rightarrow \mathbb{R}^n$  denotes the function reconstruction error, and  $P_i \in \mathbb{N}$  denotes the number of neural network neurons. Using the universal function approximation property of single layer neural networks, provided the rows of  $\sigma_{\theta i}(x)$  form a proper basis, there exist constant ideal weights  $\theta_i$  and positive constants  $\bar{\theta}_i \in \mathbb{R}$  and  $\bar{\epsilon}_{\theta i} \in \mathbb{R}$  such that  $\|\theta_i\|_F \leq \bar{\theta}_i < \infty$  and  $\sup_{x \in \chi} \|\epsilon_{\theta i}(x)\| \leq \bar{\epsilon}_{\theta i}$ , where  $\|\cdot\|_F$  denotes the Frobenius norm.

**Assumption 5.4** The bounds  $\bar{\theta}_i$  and  $\bar{\epsilon}_{\theta i}$  are known for all  $i \in \mathcal{N}$ .

Using an estimate  $\hat{\theta}_i : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{P_i+1 \times n}$  of the weight matrix  $\theta_i$ , the function  $f_i$  can be approximated by the function  $\hat{f}_i : \mathbb{R}^n \times \mathbb{R}^{P_i+1 \times n} \rightarrow \mathbb{R}^n$  defined by  $\hat{f}_i(x, \hat{\theta}) \triangleq \hat{\theta}^T \sigma_{\theta i}(x)$ . Based on (5.22), an estimator for online identification of the drift dynamics is developed as

$$\dot{\tilde{x}}_i(t) = \hat{\theta}_i^T(x_i(t)) \sigma_{\theta i}(x_i(t)) + g_i(x_i(t)) u_i(t) + k_i \tilde{x}_i(t), \tag{5.23}$$

where  $\tilde{x}_i(t) \triangleq x_i(t) - \hat{x}_i(t)$  and  $k_i \in \mathbb{R}$  is a positive constant learning gain. The following assumption facilitates concurrent learning-based system identification.

**Assumption 5.5** [12, 13] A history stack containing recorded state-action pairs  $\{x_i^k, u_i^k\}_{k=1}^{M_{\theta i}}$  along with numerically computed state derivatives  $\{\dot{x}_i^k\}_{k=1}^{M_{\theta i}}$  that satisfies

$$\lambda_{\min} \left\{ \sum_{k=1}^{M_{\theta i}} \sigma_{\theta i}^k (\sigma_{\theta i}^k)^T \right\} = \underline{\sigma_{\theta i}} > 0, \quad \|\dot{x}_i^k - \dot{x}_i^k\| < \bar{d}_i, \quad \forall k \quad (5.24)$$

is available a priori. In (5.24),  $\sigma_{\theta i}^k \triangleq \sigma_{\theta i}(x_i^k)$ , and  $\bar{d}_i, \underline{\sigma_{\theta i}} \in \mathbb{R}$  are known positive constants.

The weight estimates  $\hat{\theta}_i(\cdot)$  are updated using the following concurrent learning-based update law:

$$\dot{\hat{\theta}}_i(t) = k_{\theta i} \Gamma_{\theta i} \sum_{k=1}^{M_{\theta i}} \sigma_{\theta i}^k \left( \dot{x}_i^k - g_i^k u_i^k - \hat{\theta}_i^T(t) \sigma_{\theta i}^k \right)^T + \Gamma_{\theta i} \sigma_{\theta i}(x_i(t)) \tilde{x}_i^T(t), \quad (5.25)$$

where  $g_i^k \triangleq g_i(x_i^k)$ ,  $k_{\theta i} \in \mathbb{R}$  is a constant positive concurrent learning gain, and  $\Gamma_{\theta i} \in \mathbb{R}^{P_i+1 \times P_i+1}$  is a constant, diagonal, and positive definite adaptation gain matrix.

To facilitate the subsequent stability analysis, a candidate Lyapunov function  $V_{0i} : \mathbb{R}^n \times \mathbb{R}^{P_i+1 \times n} \rightarrow \mathbb{R}$  is selected as

$$V_{0i}(\tilde{x}_i, \tilde{\theta}_i) \triangleq \frac{1}{2} \tilde{x}_i^T \tilde{x}_i + \frac{1}{2} \text{tr} \left( \tilde{\theta}_i^T \Gamma_{\theta i}^{-1} \tilde{\theta}_i \right), \quad (5.26)$$

where  $\tilde{\theta}_i(t) \triangleq \theta_i(t) - \hat{\theta}_i(t)$ . Using (5.23)–(5.25), a bound on the time derivative of  $V_{0i}$  is established as

$$\dot{V}_{0i}(\tilde{x}_i, \tilde{\theta}_i) \leq -k_i \|\tilde{x}_i\|^2 - k_{\theta i} \underline{\sigma_{\theta i}} \left\| \tilde{\theta}_i \right\|_F^2 + \bar{\epsilon}_{\theta i} \|\tilde{x}_i\| + k_{\theta i} \bar{d}_{\theta i} \left\| \tilde{\theta}_i \right\|_F, \quad (5.27)$$

where  $\bar{d}_{\theta i} \triangleq \bar{d}_i \sum_{k=1}^{M_{\theta i}} \|\sigma_{\theta i}^k\| + \sum_{k=1}^{M_{\theta i}} (\|\epsilon_{\theta i}^k\| \|\sigma_{\theta i}^k\|)$ . Using (5.26) and (5.27), a Lyapunov-based stability analysis can be used to show that  $\hat{\theta}_i(\cdot)$  converges exponentially to a neighborhood around  $\theta_i$ .

*Remark 5.6* Using an integral formulation, the system identifier can also be implemented without using state-derivative measurements (see, e.g., [14]).

### 5.2.6 Approximation of the Bellman Error and the Relative Steady-State Controller

Using the approximations  $\hat{f}_i$  for the functions  $f_i$ , the Bellman errors in (5.21) can be approximated as

$$\begin{aligned} \hat{\delta}_i \left( \mathcal{E}_i, \hat{W}_{ci}, \left( \hat{W}_a \right)_{\mathcal{S}_i}, \hat{\theta}_{\mathcal{S}_i} \right) &\triangleq \hat{\mu}_i^T \left( \mathcal{E}_i, \hat{W}_{ai} \right) R_i \hat{\mu}_i \left( \mathcal{E}_i, \hat{W}_{ai} \right) + \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \hat{V}_i \left( \mathcal{E}_i, \hat{W}_{ci} \right) \hat{\mathcal{F}}_j \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_j} \right) \\ &+ \nabla_{x_i} \hat{V}_i \left( \mathcal{E}_i, \hat{W}_{ci} \right) \left( \hat{\mathcal{F}}_i \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_i} \right) + \mathcal{G}_i \left( \mathcal{E}_i \right) \hat{\mu}_{\mathcal{S}_i} \left( \mathcal{E}_i, \left( \hat{W}_a \right)_{\mathcal{S}_i} \right) \right) + Q_i \left( e_i \right) \\ &+ \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \hat{V}_i \left( \mathcal{E}_i, \hat{W}_{ci} \right) \mathcal{G}_j \left( \mathcal{E}_j \right) \hat{\mu}_{\mathcal{S}_j} \left( \mathcal{E}_j, \left( \hat{W}_a \right)_{\mathcal{S}_j} \right). \end{aligned} \quad (5.28)$$

In (5.28),

$$\begin{aligned} \hat{\mathcal{F}}_i \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_i} \right) &\triangleq \sum^i a_{ij} \left( \hat{f}_i \left( x_i, \hat{\theta}_i \right) - \hat{f}_j \left( x_j, \hat{\theta}_j \right) \right) + \sum^i a_{ij} \left( g_i \left( x_i \right) \mathcal{L}_{gi}^i - g_j \left( x_j \right) \mathcal{L}_{gi}^j \right) \hat{f}_i \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_i} \right), \\ \hat{f}_i \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_i} \right) &\triangleq \hat{\theta}_i^T \sigma_{\theta i} \left( x_i \right) + g_i \left( x_i \right) \mathcal{L}_{gi}^i \hat{f}_i \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_i} \right), \\ \hat{F}_i \left( \mathcal{E}_i, \hat{\theta}_{\mathcal{S}_i} \right) &\triangleq \begin{bmatrix} \left( \sum^{\lambda_i^1} a_{\lambda_i^1 j} \hat{f}_{\lambda_i^1 j} \left( x_{\lambda_i^1}, \hat{\theta}_{\lambda_i^1}, x_j, \hat{\theta}_j \right) \right) \\ \vdots \\ \left( \sum^{\lambda_i^{s_i}} a_{\lambda_i^{s_i} j} \hat{f}_{\lambda_i^{s_i} j} \left( x_{\lambda_i^{s_i}}, \hat{\theta}_{\lambda_i^{s_i}}, x_j, \hat{\theta}_j \right) \right) \end{bmatrix}, \\ \hat{f}_{ij} \left( x_i, \hat{\theta}_i, x_j, \hat{\theta}_j \right) &\triangleq g_i^+ \left( x_j + x_{dij} \right) \hat{f}_j \left( x_j, \hat{\theta}_j \right) - g_i^+ \left( x_j + x_{dij} \right) \hat{f}_i \left( x_j + x_{dij}, \hat{\theta}_i \right). \end{aligned}$$

The approximations  $\hat{F}_i$ ,  $\hat{\mathcal{F}}_i$ , and  $\hat{\mathcal{F}}_i$  are related to the original unknown function as  $\hat{F}_i \left( \mathcal{E}_i, \theta_{\mathcal{S}_i} \right) + B_i \left( \mathcal{E}_i \right) = F_i \left( \mathcal{E}_i \right)$ ,  $\hat{\mathcal{F}}_i \left( \mathcal{E}_i, \theta_{\mathcal{S}_i} \right) + \mathcal{B}_i \left( \mathcal{E}_i \right) = \mathcal{F}_i \left( \mathcal{E}_i \right)$ , and  $\hat{\mathcal{F}}_i \left( \mathcal{E}_i, \theta_{\mathcal{S}_i} \right) + \mathcal{B}_i \left( \mathcal{E}_i \right) = \mathcal{F}_i \left( \mathcal{E}_i \right)$ , where  $B_i$ ,  $\mathcal{B}_i$ , and  $\mathcal{B}_i$  are  $O \left( (\bar{\epsilon}_\theta)_{\mathcal{S}_i} \right)$  terms that denote bounded function approximation errors.

Using the approximations  $\hat{f}_i$ , an implementable form of the controllers in (5.8) is expressed as

$$u_{\mathcal{S}_i} \left( t \right) = \mathcal{L}_{gi}^{-1} \left( \mathcal{E}_i \left( t \right) \right) \hat{\mu}_{\mathcal{S}_i} \left( \mathcal{E}_i \left( t \right), \left( \hat{W}_a \left( t \right) \right)_{\mathcal{S}_i} \right) + \mathcal{L}_{gi}^{-1} \left( \mathcal{E}_i \left( t \right) \right) \hat{F}_i \left( \mathcal{E}_i \left( t \right), \hat{\theta}_{\mathcal{S}_i} \left( t \right) \right). \quad (5.29)$$

Using (5.7) and (5.29), an unmeasurable form of the virtual controllers for the systems (5.11) and (5.12) is given by

$$\mu_{\mathcal{S}_i} \left( t \right) = \hat{\mu}_{\mathcal{S}_i} \left( \mathcal{E}_i \left( t \right), \left( \hat{W}_a \left( t \right) \right)_{\mathcal{S}_i} \right) - \hat{F}_i \left( \mathcal{E}_i \left( t \right), \tilde{\theta}_{\mathcal{S}_i} \left( t \right) \right) - B_i \left( \mathcal{E}_i \left( t \right) \right). \quad (5.30)$$

### 5.2.7 Value Function Approximation

On any compact set  $\chi \in \mathbb{R}^{n(s_i+1)}$ , the value functions can be represented as

$$V_i^* \left( \mathcal{E}_i \right) = W_i^T \sigma_i \left( \mathcal{E}_i \right) + \epsilon_i \left( \mathcal{E}_i \right), \quad \forall \mathcal{E}_i \in \mathbb{R}^{n(s_i+1)}, \quad (5.31)$$

where  $W_i \in \mathbb{R}^{L_i}$  are ideal neural network weights,  $\sigma_i : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}^{L_i}$  are neural network basis functions, and  $\epsilon_i : \mathbb{R}^{n(s_i+1)} \rightarrow \mathbb{R}$  are function approximation errors. Using the universal function approximation property of single layer neural networks,

provided  $\sigma_i(\mathcal{E}_i)$  forms a proper basis, there exist constant ideal weights  $W_i$  and positive constants  $\overline{W}_i, \overline{\epsilon}_i, \overline{\nabla\epsilon}_i \in \mathbb{R}$  such that  $\|W_i\| \leq \overline{W}_i < \infty$ ,  $\sup_{\mathcal{E}_i \in \chi} \|\epsilon_i(\mathcal{E}_i)\| \leq \overline{\epsilon}_i$ , and  $\sup_{\mathcal{E}_i \in \chi} \|\nabla\epsilon_i(\mathcal{E}_i)\| \leq \overline{\nabla\epsilon}_i$ .

**Assumption 5.7** The constants  $\overline{\epsilon}_i$ ,  $\overline{\nabla\epsilon}_i$ , and  $\overline{W}_i$  are known for all  $i \in \mathcal{N}$ .

Using (5.20) and (5.31), the feedback-Nash equilibrium policies are

$$\mu_i^*(\mathcal{E}_i) = -\frac{1}{2} R_i^{-1} G_{\sigma i}(\mathcal{E}_i) W_i - \frac{1}{2} R_i^{-1} G_{\epsilon i}(\mathcal{E}_i),$$

$\forall \mathcal{E}_i \in \mathbb{R}^{n(s_i+1)}$ , where

$$\begin{aligned} G_{\sigma i}(\mathcal{E}_i) &\triangleq \sum_{j \in \mathcal{S}_i} (\mathcal{G}_j^i(\mathcal{E}_i))^T (\nabla_{e_j} \sigma_i(\mathcal{E}_i))^T + (\mathcal{G}_i^i(\mathcal{E}_i))^T (\nabla_{x_i} \sigma_i(\mathcal{E}_i))^T \\ G_{\epsilon i}(\mathcal{E}_i) &\triangleq \sum_{j \in \mathcal{S}_i} (\mathcal{G}_j^i(\mathcal{E}_i))^T (\nabla_{e_j} \epsilon_i(\mathcal{E}_i))^T + (\mathcal{G}_i^i(\mathcal{E}_i))^T (\nabla_{x_i} \epsilon_i(\mathcal{E}_i))^T. \end{aligned}$$

The value functions and the policies are approximated using neural networks as

$$\begin{aligned} \hat{V}_i(\mathcal{E}_i, \hat{W}_{ci}) &\triangleq \hat{W}_{ci}^T \sigma_i(\mathcal{E}_i), \\ \hat{\mu}_i(\mathcal{E}_i, \hat{W}_{ai}) &\triangleq -\frac{1}{2} R_i^{-1} G_{\sigma i}(\mathcal{E}_i) \hat{W}_{ai}, \end{aligned} \quad (5.32)$$

where  $\hat{W}_{ci}$  and  $\hat{W}_{ai}$  are estimates of the ideal weights  $W_i$  introduced in (5.21).

### 5.2.8 Simulation of Experience via Bellman Error Extrapolation

A consequence of Theorem 5.3 is that the Bellman error provides an indirect measure of how close the estimates  $\hat{W}_{ci}$  and  $\hat{W}_{ai}$  are to the ideal weights  $W_i$ . From a reinforcement learning perspective, each evaluation of the Bellman error along the system trajectory can be interpreted as experience gained by the critic, and each evaluation of the Bellman error at points not yet visited can be interpreted as simulated experience. In previous results such as [15–19], the critic is restricted to the experience gained (in other words Bellman errors evaluated) along the system state trajectory. The development in [15–18] can be extended to employ simulated experience; however, the extension requires exact model knowledge. The formulation in (5.28) employs the system identifier developed in Sect. 5.2.5 to facilitate approximate evaluation of the Bellman error at off-trajectory points.

To simulate experience, a set of points  $\{\mathcal{E}_i^k\}_{k=1}^{M_i}$  is selected corresponding to each agent  $i$ , and the instantaneous Bellman error in (5.21) is approximated at the current state and the selected points using (5.36). The approximation at the current state is denoted by  $\hat{\delta}_{ti}$  and the approximation at the selected points is denoted by  $\hat{\delta}_{ti}^k$ , where  $\hat{\delta}_{ti}$  and  $\hat{\delta}_{ti}^k$  are defined as

$$\begin{aligned}\hat{\delta}_{ti}(t) &\triangleq \hat{\delta}_i\left(\mathcal{E}_i(t), \hat{W}_{ci}(t), \left(\hat{W}_a(t)\right)_{\mathcal{S}_i}, \left(\hat{\theta}(t)\right)_{\mathcal{S}_i}\right), \\ \hat{\delta}_{ti}^k(t) &\triangleq \hat{\delta}_i\left(\mathcal{E}_i^k, \hat{W}_{ci}(t), \left(\hat{W}_a(t)\right)_{\mathcal{S}_i}, \left(\hat{\theta}(t)\right)_{\mathcal{S}_i}\right).\end{aligned}$$

Note that once  $\{e_j\}_{j \in \mathcal{S}_i}$  and  $x_i$  are selected, the  $i$ th agent can compute the states of all the remaining agents in the sub-graph.

The critic uses simulated experience to update the critic weights using the least-squares-based update law

$$\begin{aligned}\dot{\hat{W}}_{ci}(t) &= -\eta_{c1i}\Gamma_i(t) \frac{\omega_i(t)}{\rho_i(t)} \hat{\delta}_{ti}(t) - \frac{\eta_{c2i}\Gamma_i(t)}{M_i} \sum_{k=1}^{M_i} \frac{\omega_i^k(t)}{\rho_i^k(t)} \hat{\delta}_{ti}^k(t), \\ \dot{\Gamma}_i(t) &= \left( \beta_i \Gamma_i(t) - \eta_{c1i}\Gamma_i(t) \frac{\omega_i(t) \omega_i^T(t)}{\rho_i^2(t)} \Gamma_i \right) \mathbf{1}_{\{\|\Gamma_i\| \leq \bar{\Gamma}_i\}}, \quad \|\Gamma_i(t_0)\| \leq \bar{\Gamma}_i,\end{aligned}\tag{5.33}$$

where  $\rho_i(t) \triangleq 1 + v_i \omega_i^T(t) \Gamma_i \omega_i(t)$ ,  $\Gamma_i \in \mathbb{R}^{L_i \times L_i}$  denotes the time-varying least-squares learning gain,  $\bar{\Gamma}_i \in \mathbb{R}$  denotes the saturation constant, and  $\eta_{c1i}, \eta_{c2i}, \beta_i, v_i \in \mathbb{R}$  are constant positive learning gains. In (5.33),

$$\begin{aligned}\omega_i(t) &\triangleq \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \sigma_i(\mathcal{E}_i(t)) \left( \hat{\mathcal{F}}_j\left(\mathcal{E}_j(t), \hat{\theta}_{\mathcal{S}_j}(t)\right) + \mathcal{G}_j(\mathcal{E}_j(t)) \hat{\mu}_{\mathcal{S}_j}\left(\mathcal{E}_j(t), \left(\hat{W}_a(t)\right)_{\mathcal{S}_j}\right) \right) \\ &\quad + \nabla_{x_i} \sigma_i(\mathcal{E}_i(t)) \left( \hat{\mathcal{F}}_i\left(\mathcal{E}_i(t), \hat{\theta}_{\mathcal{S}_i}(t)\right) + \mathcal{G}_i(\mathcal{E}_i(t)) \hat{\mu}_{\mathcal{S}_i}\left(\mathcal{E}_i(t), \left(\hat{W}_a(t)\right)_{\mathcal{S}_i}\right) \right), \\ \omega_i^k(t) &\triangleq \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \sigma_i^k \left( \hat{\mathcal{F}}_j^k\left(\hat{\theta}_{\mathcal{S}_j}(t)\right) + \mathcal{G}_j^k \hat{\mu}_{\mathcal{S}_j}^k\left(\left(\hat{W}_a(t)\right)_{\mathcal{S}_j}\right) \right) \\ &\quad + \nabla_{x_i} \sigma_i^k \left( \hat{\mathcal{F}}_i^k\left(\hat{\theta}_{\mathcal{S}_i}(t)\right) + \mathcal{G}_i^k \hat{\mu}_{\mathcal{S}_i}^k\left(\left(\hat{W}_a(t)\right)_{\mathcal{S}_i}\right) \right),\end{aligned}$$

where the notation  $\phi_i^k$  indicates evaluation at  $\mathcal{E}_i = \mathcal{E}_i^k$  for a function  $\phi_i(\mathcal{E}_i, (\cdot))$  (i.e.,  $\phi_i^k(\cdot) \triangleq \phi_i(\mathcal{E}_i^k, (\cdot))$ ). The actor updates the actor weights using the following update law derived based on a Lyapunov-based stability analysis:

$$\begin{aligned}\dot{\hat{W}}_{ai}(t) = & \frac{1}{4} \eta_{c1i} G_{\sigma i}^T(\mathcal{E}_i(t)) R_i^{-1} G_{\sigma i}(\mathcal{E}_i(t)) \hat{W}_{ai}(t) \frac{\omega_i^T(t)}{\rho_i(t)} \hat{W}_{ci}(t) - \eta_{a2i} \hat{W}_{ai}(t) \\ & + \frac{1}{4} \sum_{k=1}^{M_i} \frac{\eta_{c2i}}{M_i} \left( G_{\sigma i}^k \right)^T R_i^{-1} G_{\sigma i}^k \hat{W}_{ai}(t) \frac{(\omega_i^k(t))^T}{\rho_i^k(t)} \hat{W}_{ci}(t) - \eta_{a1i} (\hat{W}_{ai}(t) - \hat{W}_{ci}(t)),\end{aligned}\quad (5.34)$$

where  $\eta_{a1i}, \eta_{a2i} \in \mathbb{R}$  are constant positive learning gains. The following assumption facilitates simulation of experience.

**Assumption 5.8** [13] For each  $i \in \mathcal{N}$ , there exists a finite set of  $M_i$  points  $\{\mathcal{E}_i^k\}_{k=1}^{M_i}$  such that

$$\underline{\rho}_i \triangleq \frac{\left( \inf_{t \in \mathbb{R}_{\geq 0}} \left( \lambda_{\min} \left\{ \sum_{k=1}^{M_i} \frac{\omega_i^k(t)(\omega_i^k(t))^T}{\rho_i^k(t)} \right\} \right) \right)}{M_i} > 0,\quad (5.35)$$

where  $\underline{\rho}_i \in \mathbb{R}$  is a positive constant.

### 5.2.9 Stability Analysis

To facilitate the stability analysis, the left hand side of (5.15) is subtracted from (5.28) to express the Bellman error in terms of the weight estimation errors as

$$\begin{aligned}\hat{\delta}_{ti} = & -\tilde{W}_{ci}^T \omega_i - W_i^T \nabla_{x_i} \sigma_i \hat{\mathcal{F}}_i(\mathcal{E}_i, \tilde{\theta}_{\mathcal{S}_i}) + \frac{1}{4} \tilde{W}_{ai}^T G_{\sigma i}^T R_i^{-1} G_{\sigma i} \tilde{W}_{ai} \\ & - W_i^T \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \sigma_i \hat{\mathcal{F}}_j(\mathcal{E}_j, \tilde{\theta}_{\mathcal{S}_j}) + \frac{1}{2} W_i^T \sum_{j \in \mathcal{S}_i} \nabla_{e_j} \sigma_i \mathcal{G}_j \mathcal{R}_{\mathcal{S}_j}(\tilde{W}_a)_{\mathcal{S}_j} \\ & - \frac{1}{2} W_i^T G_{\sigma i}^T R_i^{-1} G_{\sigma i} \tilde{W}_{ai} + \frac{1}{2} W_i^T \nabla_{x_i} \sigma_i \mathcal{G}_i \mathcal{R}_{\mathcal{S}_i}(\tilde{W}_a)_{\mathcal{S}_i} + \Delta_i,\end{aligned}\quad (5.36)$$

where  $(\tilde{\cdot}) \triangleq (\cdot) - (\hat{\cdot})$ ,  $\Delta_i = O\left((\bar{\epsilon})_{\mathcal{S}_i}, (\bar{\nabla \epsilon})_{\mathcal{S}_i}, (\bar{\epsilon_\theta})_{\mathcal{S}_i}\right)$ , and  $\mathcal{R}_{\mathcal{S}_j} \triangleq \text{diag}\left(\left[R_{\lambda_j^1}^{-1} G_{\sigma \lambda_j^1}^T, \dots, R_{\lambda_j^{s_j}}^{-1} G_{\sigma \lambda_j^{s_j}}^T\right]\right)$  is a block diagonal matrix. Consider a set of extended neighbors  $\mathcal{S}_p$  corresponding to the  $p$ th agent. To analyze asymptotic properties of the agents in  $\mathcal{S}_p$ , consider the following candidate Lyapunov function

$$V_{Lp}(Z_p, t) \triangleq \sum_{i \in \mathcal{S}_p} V_{ti}(e_{\mathcal{S}_i}, t) + \sum_{i \in \mathcal{S}_p} \frac{1}{2} \tilde{W}_{ci}^T \Gamma_i^{-1} \tilde{W}_{ci} + \sum_{i \in \mathcal{S}_p} \frac{1}{2} \tilde{W}_{ai}^T \tilde{W}_{ai} + \sum_{i \in \mathcal{S}_p} V_{0i}(\tilde{x}_i, \tilde{\theta}_i),\quad (5.37)$$

where  $Z_p \in \mathbb{R}^{(2ns_i + 2L_i s_i + n(P_i + 1)s_i)}$  is defined as

$$Z_p \triangleq \left[ e_{\mathcal{S}_p}^T, \left( \tilde{W}_c \right)_{\mathcal{S}_p}^T, \left( \tilde{W}_a \right)_{\mathcal{S}_p}^T, \tilde{x}_{\mathcal{S}_p}^T, \text{vec} \left( \tilde{\theta}_{\mathcal{S}_p} \right)^T \right]^T,$$

$\text{vec}(\cdot)$  denotes the vectorization operator, and  $V_{ti} : \mathbb{R}^{ns_i} \times \mathbb{R} \rightarrow \mathbb{R}$  is defined as

$$V_{ti}(e_{\mathcal{S}_i}, t) \triangleq V_i^* \left( [e_{\mathcal{S}_i}^T, x_i^T(t)]^T \right), \quad (5.38)$$

$\forall e_{\mathcal{S}_i} \in \mathbb{R}^{ns_i}$  and  $\forall t \in \mathbb{R}_{\geq t_0}$ .

Since  $V_{ti}^*$  depends on  $t$  only through uniformly bounded leader trajectories, Lemma 1 from [20] can be used to show that  $V_{ti}$  is a positive definite and decreasing function. Since the graph has a spanning tree, the mapping between the errors and the states is invertible. Hence, the state of an agent can be expressed as  $x_i = h_i(e_{\mathcal{S}_i}, x_0)$  for some function  $h_i$ . Thus, the value function can be expressed as  $V_i^*(e_{\mathcal{S}_i}, x_0) = V_i^*(e_{\mathcal{S}_i}, h(e_{\mathcal{S}_i}, x_0))$ . Then,  $V_{ti}^*$  can be alternatively defined as  $V_{ti}(e_{\mathcal{S}_i}, t) \triangleq V_i^* \left( \begin{bmatrix} e_{\mathcal{S}_i} \\ x_0(t) \end{bmatrix} \right)$ . Since  $x_0$  is a uniformly bounded function of  $t$  by assumption, [20, Lemma 1] can be used to conclude that  $V_{ti}$  is a positive definite and decreasing function.

Thus, using [21, Lemma 4.3], the following bounds on the candidate Lyapunov function in (5.37) are established

$$\underline{v}_{lp}(\|Z_p\|) \leq V_{Lp}(Z_p, t) \leq \overline{v}_{lp}(\|Z_p\|), \quad (5.39)$$

$\forall Z_p \in \mathbb{R}^{(2ns_i+2L_is_i+n(P_i+1)s_i)}$  and  $\forall t \in \mathbb{R}_{\geq t_0}$ , where  $\underline{v}_{lp}, \overline{v}_{lp} : \mathbb{R} \rightarrow \mathbb{R}$  are class  $\mathcal{K}$  functions.

To facilitate the stability analysis, given any compact ball  $\chi_p \subset \mathbb{R}^{2ns_i+2L_is_i+n(P_i+1)s_i}$  of radius  $r_p \in \mathbb{R}$  centered at the origin, a positive constant  $\iota_p \in \mathbb{R}$  is defined as

$$\begin{aligned} \iota_p &\triangleq \sum_{i \in \mathcal{S}_p} \left( \frac{\overline{\epsilon_{\theta i}}^2}{2k_i} + \frac{3 \left( k_{\theta i} \overline{d_{\theta i}} + \|A_i^\theta\| \|B_i^\theta\| \right)^2}{4\sigma_{\theta i}} \right) + \sum_{i \in \mathcal{S}_p} \frac{5 (\eta_{c1i} + \eta_{c2i})^2 \left\| \frac{\omega_i}{\rho_i} \Delta_i \right\|^2}{4\eta_{c2i} \rho_i} \\ &+ \sum_{i \in \mathcal{S}_p} \frac{1}{2} \overline{\left\| \nabla_{x_i} V_i^*(\mathcal{E}_i) \mathcal{G}_i \mathcal{R}_{\mathcal{S}_i} \epsilon_{\mathcal{S}_i} + \sum_{j \in \mathcal{S}_i} \nabla_{e_j} V_i^*(\mathcal{E}_i) \mathcal{G}_j \mathcal{R}_{\mathcal{S}_j} \epsilon_{\mathcal{S}_j} \right\|} \\ &+ \sum_{i \in \mathcal{S}_p} \overline{\left\| \sum_{j \in \mathcal{S}_i} \nabla_{e_j} V_i^*(\mathcal{E}_i) \mathcal{G}_j B_j + \nabla_{x_i} V_i^*(\mathcal{E}_i) \mathcal{G}_i B_i \right\|} \\ &+ \frac{3 \sum_{i \in \mathcal{S}_p} \left( \frac{\|A_i^{a1}\|}{2} + \eta_{a2i} \overline{W_i} + \frac{(\eta_{c1i} + \eta_{c2i})}{4} \overline{\left\| W_i^T \frac{\omega_i}{\rho_i} W_i^T G_{\sigma i}^T R_i^{-1} G_{\sigma i} \right\|} \right)^2}{4(\eta_{a1i} + \eta_{a2i})}, \end{aligned}$$

where for any function  $\varpi : \mathbb{R}^l \rightarrow \mathbb{R}$ ,  $l \in \mathbb{N}$ , the notation  $\|\varpi\|$  denotes  $\sup_{y \in \chi_{pl}} \|\varpi(y)\|$ ,  $\chi_{pl}$  denotes the projection of  $\chi_p$  onto  $\mathbb{R}^l$ , and  $A_i^\theta$ ,  $B_i^\theta$ , and  $A_i^{a1}$  are uniformly bounded state-dependent terms. Based on the subsequent stability analysis, the following sufficient gain conditions can be developed to facilitate Theorem 5.9:

$$\frac{\eta_{c2i}\underline{\rho}_i}{5} > \sum_{j \in \mathcal{S}_p} \frac{3s_p \mathbf{1}_{j \in \mathcal{S}_i} (\eta_{c1i} + \eta_{c2i})^2 \overline{\|A_{ij}^{1a\theta}\|^2 \|B_{ij}^{1a\theta}\|^2}}{4k_{\theta j} \underline{\sigma}_{\theta j}}, \quad (5.40)$$

$$\begin{aligned} \frac{(\eta_{a1i} + \eta_{a2i})}{3} &> \sum_{j \in \mathcal{S}_p} \frac{5s_p \mathbf{1}_{i \in \mathcal{S}_j} (\eta_{c1j} + \eta_{c2j})^2 \overline{\|A_{ji}^{1ac}\|^2}}{16\eta_{c2j}\underline{\rho}_j} + \frac{5\underline{\eta}_{a1i}^2}{4\eta_{c2i}\underline{\rho}_i} \\ &+ \frac{(\eta_{c1i} + \eta_{c2i}) \overline{W_i} \left\| \frac{\omega_i}{\rho_i} \right\| \overline{\|G_{\sigma i}^T R_i^{-1} G_{\sigma i}\|}}{4}, \end{aligned} \quad (5.41)$$

$$v_{lp}^{-1}(\iota_p) < \overline{v_{lp}}^{-1} \left( v_{lp}(r_p) \right), \quad (5.42)$$

where  $A_{ij}^{1a\theta}$ ,  $B_{ij}^{1a\theta}$ , and  $A_{ji}^{1ac}$  are uniformly bounded state-dependent terms.

**Theorem 5.9** *Provided Assumptions 5.2–5.8 hold and the sufficient gain conditions in (5.40)–(5.42) are satisfied, the controller in (5.32) along with the actor and critic update laws in (5.33) and (5.34), and the system identifier in (5.23) along with the weight update laws in (5.25) ensure that the local neighborhood tracking errors  $e_i$  are ultimately bounded and that the policies  $\hat{\mu}_i$  converge to a neighborhood around the feedback-Nash policies  $\mu_i^*$ ,  $\forall i \in \mathcal{N}$ .*

*Proof* The time derivative of the candidate Lyapunov function in (5.37) is given by

$$\begin{aligned} \dot{V}_{lp} &= \sum_{i \in \mathcal{S}_p} \dot{V}_{ti}(e_{\mathcal{S}_i}, t) - \frac{1}{2} \sum_{i \in \mathcal{S}_p} \tilde{W}_{ci}^T \Gamma_i^{-1} \dot{\Gamma}_i \Gamma_i^{-1} \tilde{W}_{ci} - \sum_{i \in \mathcal{S}_p} \tilde{W}_{ci}^T \Gamma_i^{-1} \dot{\hat{W}}_{ci} - \sum_{i \in \mathcal{S}_p} \tilde{W}_{ai}^T \dot{\hat{W}}_{ai} \\ &+ \sum_{i \in \mathcal{S}_p} \dot{V}_{0i} \left( \tilde{x}_i, \tilde{\theta}_i \right). \end{aligned} \quad (5.43)$$

Using (5.15), (5.27), (5.30), and (5.36), the update laws in (5.33) and (5.34), and the definition of  $V_{ti}$  in (5.38), the derivative in (5.43) can be bounded as

$$\begin{aligned} \dot{V}_{lp} &\leq \sum_{i \in \mathcal{S}_p} \left( -\frac{\eta_{c2i}\underline{\rho}_i}{5} \left\| \tilde{W}_{ci} \right\|^2 - \frac{(\eta_{a1i} + \eta_{a2i})}{3} \left\| \tilde{W}_{ai} \right\|^2 \right) \\ &+ \sum_{i \in \mathcal{S}_p} \left( -\underline{q}_i(\|e_i\|) - \frac{k_i}{2} \left\| \tilde{x}_i \right\|^2 - \frac{k_{\theta i}\underline{\sigma}_{\theta i}}{3} \left\| \tilde{\theta}_i \right\|_F^2 \right) + \iota_p. \end{aligned}$$

Let  $v_{lp} : \mathbb{R} \rightarrow \mathbb{R}$  be a class  $\mathcal{K}$  function such that

$$\begin{aligned} v_{lp}(\|Z_p\|) &\leq \frac{1}{2} \sum_{i \in \mathcal{S}_p} \underline{q}_i(\|e_i\|) + \frac{1}{2} \sum_{i \in \mathcal{S}_p} \frac{\eta_{c2i}\rho_i}{5} \|\tilde{W}_{ci}\|^2 + \frac{1}{2} \sum_{i \in \mathcal{S}_p} \frac{(\eta_{a1i} + \eta_{a2i})}{3} \|\tilde{W}_{ai}\|^2 \\ &\quad + \frac{1}{2} \sum_{i \in \mathcal{S}_p} \frac{k_i}{2} \|\tilde{x}_i\|^2 + \frac{1}{2} \sum_{i \in \mathcal{S}_p} \frac{k_{\theta i}\sigma_{\theta i}}{3} \|\tilde{\theta}_i\|_F^2, \end{aligned} \quad (5.44)$$

where  $\underline{q}_i : \mathbb{R} \rightarrow \mathbb{R}$  are class  $\mathcal{K}$  functions such that  $\underline{q}_i(\|e\|) \leq Q_i(e)$ ,  $\forall e \in \mathbb{R}^n$ ,  $\forall i \in \mathcal{N}$ . Then, the Lyapunov derivative can be bounded as

$$\dot{V}_{lp} \leq -v_{lp}(\|Z_p\|) \quad (5.45)$$

$\forall Z_p$  such that  $Z_p \in \chi_p$  and  $\|Z_p\| \geq \underline{v}_{lp}^{-1}(\iota_p)$ . Using the bounds in (5.39), the sufficient conditions in (5.40)–(5.42), and the inequality in (5.45), [21, Theorem 4.18] can be invoked to conclude that every trajectory  $Z_p(\cdot)$  satisfying  $\|Z_p(t_0)\| \leq \underline{v}_{lp}^{-1}(\underline{v}_{lp}(r_p))$ , is bounded for all  $t \in \mathbb{R}_{\geq t_0}$  and satisfies  $\limsup_{t \rightarrow \infty} \|Z_p(t)\| \leq \underline{v}_{lp}^{-1}(\overline{v}_{lp}(\overline{v}_{lp}^{-1}(\iota_p)))$ .

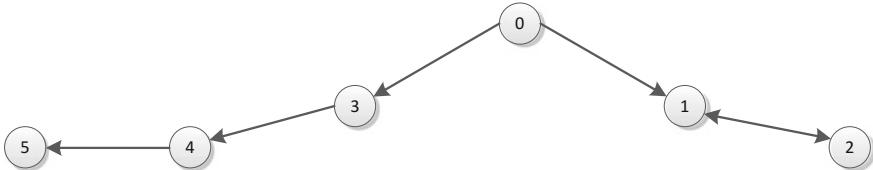
Since the choice of the subgraph  $\mathcal{S}_p$  was arbitrary, the neighborhood tracking errors  $e_i(\cdot)$  are ultimately bounded for all  $i \in \mathcal{N}$ . Furthermore, the weight estimates  $\hat{W}_{ai}(\cdot)$  converge to a neighborhood of the ideal weights  $W_i$ ; hence, invoking Theorem 5.3, the policies  $\hat{\mu}_i$  converge to a neighborhood of the feedback-Nash equilibrium policies  $\mu_i^*$ ,  $\forall i \in \mathcal{N}$ .

### 5.2.10 Simulations

#### One-Dimensional Example

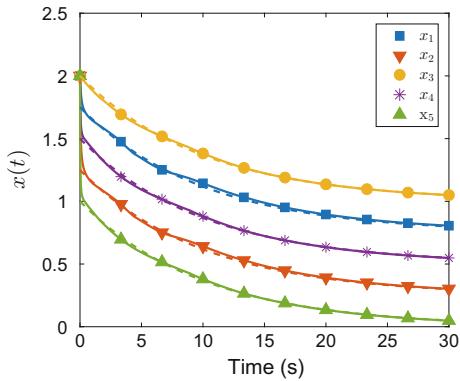
This section provides a simulation example to demonstrate the applicability of the developed technique. The agents are assumed to have the communication topology as shown in Fig. 5.1 with unit pinning gains and edge weights. Agent motion is described by identical nonlinear one-dimensional dynamics of the form (5.1) where  $f_i(x_i) = \theta_{i1}x_i + \theta_{i2}x_i^2$  and  $g_i(x_i) = (\cos(2x_{i1}) + 2)$ ,  $\forall i = 1, \dots, 5$ . The ideal values of the unknown parameters are selected to be  $\theta_{i1} = 0, 0, 0.1, 0.5$ , and  $0.2$ , and  $\theta_{i2} = 1, 0.5, 1, 1$ , and  $1$ , for  $i = 1, \dots, 5$ , respectively. The agents start at  $x_i = 2$ ,  $\forall i$ , and their final desired locations with respect to each other are given by  $x_{d12} = 0.5$ ,  $x_{d21} = -0.5$ ,  $x_{d43} = -0.5$ , and  $x_{d53} = -0.5$ . The leader traverses an exponentially decaying trajectory  $x_0(t) = e^{-0.1t}$ . The desired positions of agents 1 and 3 with respect to the leader are  $x_{d10} = 0.75$  and  $x_{d30} = 1$ , respectively.

For each agent  $i$ , five values of  $e_i$ , three values of  $x_i$ , and three values of errors corresponding to all the extended neighbors are selected for Bellman error extrapolation, resulting in  $5 \times 3^{s_i}$  total values of  $\mathcal{E}_i$ . All agents estimate the unknown drift parameters using history stacks containing thirty points recorded online using a singular value maximizing algorithm (cf. [22]), and compute the required state derivatives

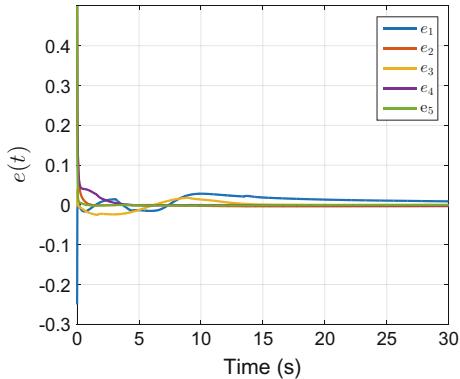


**Fig. 5.1** Communication topology: A network containing five agents (reproduced with permission from [1], ©2016, IEEE)

**Fig. 5.2** State trajectories for the five agents for the one-dimensional example. The dotted lines show the desired state trajectories (reproduced with permission from [1], ©2016, IEEE)

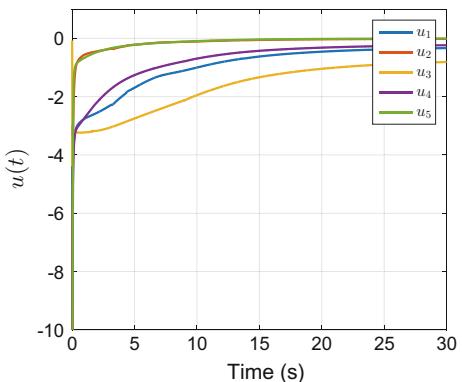


**Fig. 5.3** Tracking error trajectories for the agents for the one-dimensional example (reproduced with permission from [1], ©2016, IEEE)

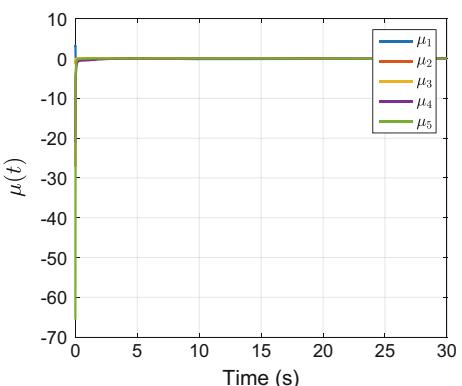


using a fifth order Savitzky–Golay smoothing filter (cf. [23]). Figures 5.2, 5.3, 5.4 and 5.5 show the tracking error, the state trajectories compared with the desired trajectories, and the control inputs for all the agents demonstrating convergence to the desired formation and the desired trajectory. Note that Agents 2, 4, and 5 do not have a communication link to the leader, nor do they know their desired relative position from the leader. The convergence to the desired formation is achieved via cooperative control based on decentralized objectives. Figures 5.6 and 5.7 show the evolution and convergence of the critic weights and the parameters estimates for the drift dynamics for Agent 1. See Table 5.1 for the optimal control problem parameters,

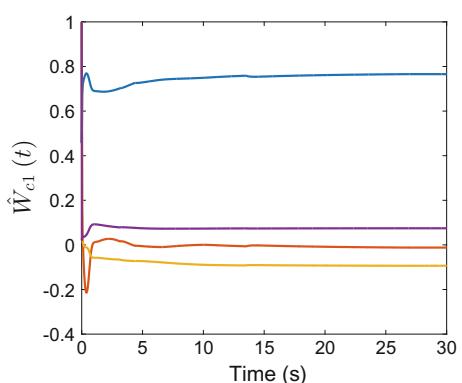
**Fig. 5.4** Trajectories of the control input for all agents for the one-dimensional example (reproduced with permission from [1], ©2016, IEEE)



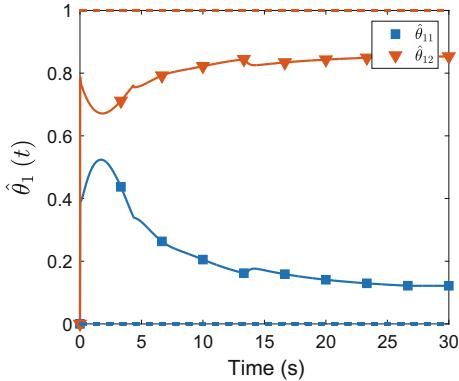
**Fig. 5.5** Trajectories of the relative control error for all agents for the one-dimensional example (reproduced with permission from [1], ©2016, IEEE)



**Fig. 5.6** Critic weight estimates for Agent 1 for the one-dimensional example (reproduced with permission from [1], ©2016, IEEE)



**Fig. 5.7** Drift dynamics parameters estimates for Agent 1 for the one-dimensional example. The dotted lines are the ideal values of the drift parameters (reproduced with permission from [1], ©2016, IEEE)



basis functions, and adaptation gains for all the agents. The errors between the ideal drift parameters and their respective estimates are large; however, as demonstrated by Fig. 5.3, the resulting dynamics are sufficiently close to the actual dynamics for the developed technique to generate stabilizing policies. It is unclear whether the value function and the actor weights converge to their ideal values.

### Two-Dimensional Example

In this simulation, the dynamics of all the agents are assumed to be exactly known, and are selected to be of the form (5.1) where for all  $i = 1, \dots, 5$ ,

$$f_i(x_i) = \begin{bmatrix} -x_{i1} + x_{i2} \\ -0.5x_{i1} - 0.5x_{i2}(1 - (\cos(2x_{i1}) + 2)^2) \end{bmatrix},$$

$$g_i(x_i) = \begin{bmatrix} \sin(2x_{i1}) + 2 & 0 \\ 0 & \cos(2x_{i1}) + 2 \end{bmatrix}.$$

The agents start at the origin, and their final desired relative positions are given by  $xd_{12} = [-0.5, 1]^T$ ,  $xd_{21} = [0.5, -1]^T$ ,  $xd_{43} = [0.5, 1]^T$ , and  $xd_{53} = [-1, 1]^T$ . The relative positions are designed such that the final desired formation is a pentagon with the leader node at the center. The leader traverses a sinusoidal trajectory  $x_0(t) = [2 \sin(t), 2 \sin(t) + 2 \cos(t)]^T$ . The desired positions of agents 1 and 3 with respect to the leader are  $x_{d10} = [-1, 0]^T$  and  $x_{d30} = [0.5, -1]^T$ , respectively.

The optimal control problem parameters, basis functions, and adaptation gains for the agents are available in Table 5.2. Nine values of  $e_i$ ,  $x_i$ , and the errors corresponding to all the extended neighbors are selected for Bellman error extrapolation for each agent  $i$  on a uniform  $3 \times 3$  grid in a  $1 \times 1$  square around the origin, resulting in  $9^{(s_i+1)}$  total values of  $\mathcal{E}_i$ . Figures 5.8, 5.9, 5.10 and 5.11 show the tracking error, the state trajectories, and the control inputs for all the agents demonstrating convergence to the desired formation and the desired trajectory. Note that Agents 2, 4, and 5 do not have a communication link to the leader, nor do they know their desired relative position from the leader. The convergence to the desired formation is achieved via

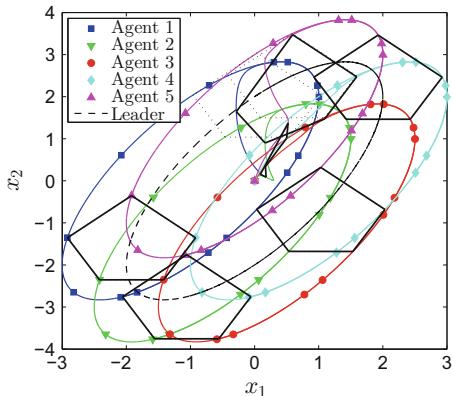
**Table 5.1** Simulation parameters for the two-dimensional example

	Agent 1	Agent 2	Agent 3	Agent 4	Agent 5
$Q_i$	10	10	10	10	10
$R_i$	0.1	0.1	0.1	0.1	0.1
$\sigma_i(\mathcal{E}_i)$	$\frac{1}{2}[e_1^2, \frac{1}{2}e_1^4, e_1^2x_1^2, e_2^2]^T$	$\frac{1}{2}[e_2^2, \frac{1}{2}e_2^4, e_2^2x_2^2, e_1^2]^T$	$\frac{1}{2}[e_3^2, \frac{1}{2}e_3^4, e_3^2x_3^2, \frac{1}{2}e_3^2x_3^2]^T$	$\frac{1}{2}[e_4^2, \frac{1}{2}e_4^4, e_4^2x_4^2, e_4^2e_4^2, e_3^4x_3^2, e_3^2e_4^2, e_3^2e_4^2, e_4^2e_4^2]^T$	$\frac{1}{2}[e_5^2; \frac{1}{2}e_5^4, e_5^2e_5^2, e_5^2e_5^2, e_5^2e_5^2, e_5^2e_5^2, e_5^2e_5^2, e_5^2e_5^2, e_5^2e_5^2]^T$
$x_i(0)$	2	2	2	2	2
$\hat{x}_i(0)$	0	0	0	0	0
$\hat{W}_{ci}(0)$	$\mathbf{1}_{4 \times 1}$	$\mathbf{1}_{4 \times 1}$	$\mathbf{1}_{4 \times 1}$	$\mathbf{1}_{5 \times 1}$	$\mathbf{1}_{8 \times 1}$
$\hat{W}_{ai}(0)$	$\mathbf{1}_{4 \times 1}$	$\mathbf{1}_{4 \times 1}$	$\mathbf{1}_{4 \times 1}$	$\mathbf{1}_{5 \times 1}$	$\mathbf{1}_{8 \times 1}$
$\hat{\theta}_i(0)$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$
$\Gamma_i(0)$	$500\mathbf{I}_4$	$500\mathbf{I}_4$	$500\mathbf{I}_4$	$500\mathbf{I}_5$	$500\mathbf{I}_8$
$\eta_{cli}$	0.1	0.1	0.1	0.1	0.1
$\eta_{c2i}$	10	10	10	10	10
$\eta_{ali}$	5	5	5	5	5
$\eta_{a2i}$	0.1	0.1	0.1	0.1	0.1
$v_i$	0.005	0.005	0.005	0.005	0.005
$\Gamma_{\theta i}$	$\mathbf{I}_2$	$0.8\mathbf{I}_2$	$\mathbf{I}_2$	$\mathbf{I}_2$	$\mathbf{I}_2$
$k_i$	500	500	500	500	500
$k_{\theta i}$	30	30	25	20	30

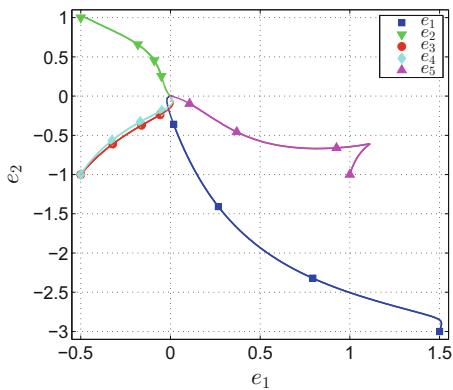
**Table 5.2** Simulation parameters for the two-dimensional example

	Agent 1	Agent 2	Agent 3	Agent 4	Agent 5
$Q_i$	$10I_2$	$10I_2$	$I_2$	$I_2$	$I_2$
$R_i$	$I_2$	$I_2$	$I_2$	$I_2$	$I_2$
$\sigma_i (\mathcal{E}_i)$	$\frac{1}{2}[2e_{11}^2, 2e_{11}e_{12}, 2e_{12}^2, e_{21}^2, 2e_{21}e_{22}, e_{22}^2, e_{11}^2e_{11}^2, e_{12}^2e_{12}^2, e_{21}^2e_{21}^2, e_{12}^2e_{11}^2, e_{12}^2e_{21}^2, e_{21}^2e_{22}^2, e_{12}^2e_{22}^2]^T$	$\frac{1}{2}[2e_{21}^2, 2e_{21}e_{22}, 2e_{22}^2, e_{11}^2, 2e_{11}e_{12}, e_{12}^2, e_{21}^2e_{21}^2, e_{22}^2e_{21}^2, e_{21}^2e_{22}^2, e_{22}^2e_{22}^2]^T$	$\frac{1}{2}[2e_{31}^2, 2e_{31}e_{32}, 2e_{32}^2, e_{31}^2e_{31}^2, e_{32}^2e_{31}^2, e_{31}^2e_{32}^2, e_{32}^2e_{32}^2, e_{31}^2e_{32}^2, e_{32}^2e_{31}^2]^T$	$\frac{1}{2}[2e_{41}^2, 2e_{41}e_{42}, 2e_{42}^2, e_{41}^2, 2e_{41}e_{42}, e_{42}^2, e_{41}^2e_{41}^2, e_{42}^2e_{41}^2, e_{41}^2e_{42}^2, e_{42}^2e_{41}^2, e_{41}^2e_{42}^2, e_{42}^2e_{41}^2]^T$	$\frac{1}{2}[2e_{51}^2, 2e_{51}e_{52}, 2e_{52}^2, e_{51}^2, 2e_{51}e_{52}, e_{52}^2, e_{51}^2e_{51}^2, e_{52}^2e_{51}^2, e_{51}^2e_{52}^2, e_{52}^2e_{51}^2]^T$
$x_i (0)$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$	$\mathbf{0}_{2 \times 1}$
$\hat{\mathbf{W}}_{ci} (0)$	$\mathbf{1}_{10 \times 1}$	$\mathbf{1}_{10 \times 1}$	$\mathbf{1}_{7 \times 1}$	$\mathbf{1}_{10 \times 1}$	$\mathbf{1}_{13 \times 1}$
$\hat{\mathbf{W}}_{ai} (0)$	$\mathbf{1}_{10 \times 1}$	$\mathbf{1}_{10 \times 1}$	$\mathbf{1}_{7 \times 1}$	$\mathbf{1}_{10 \times 1}$	$\mathbf{1}_{13 \times 1}$
$\Gamma_i (0)$	$500\mathbf{I}_{10}$	$500\mathbf{I}_{10}$	$500\mathbf{I}_4$	$500\mathbf{I}_5$	$500\mathbf{I}_8$
$\eta_{cli}$	0.1	0.1	0.1	0.1	0.1
$\eta_{cli}$	2.5	5	2.5	2.5	2.5
$\eta_{cli}$	2.5	0.5	2.5	2.5	2.5
$\eta_{a2i}$	0.01	0.01	0.01	0.01	0.01
$v_i$	0.005	0.005	0.005	0.005	0.005

**Fig. 5.8** Phase portrait in the state-space for the two-dimensional example. The actual pentagonal formation is represented by a solid black pentagon, and the desired pentagonal formation around the leader is represented by a dotted black pentagon



**Fig. 5.9** Phase portrait of all agents in the error space for the two-dimensional example



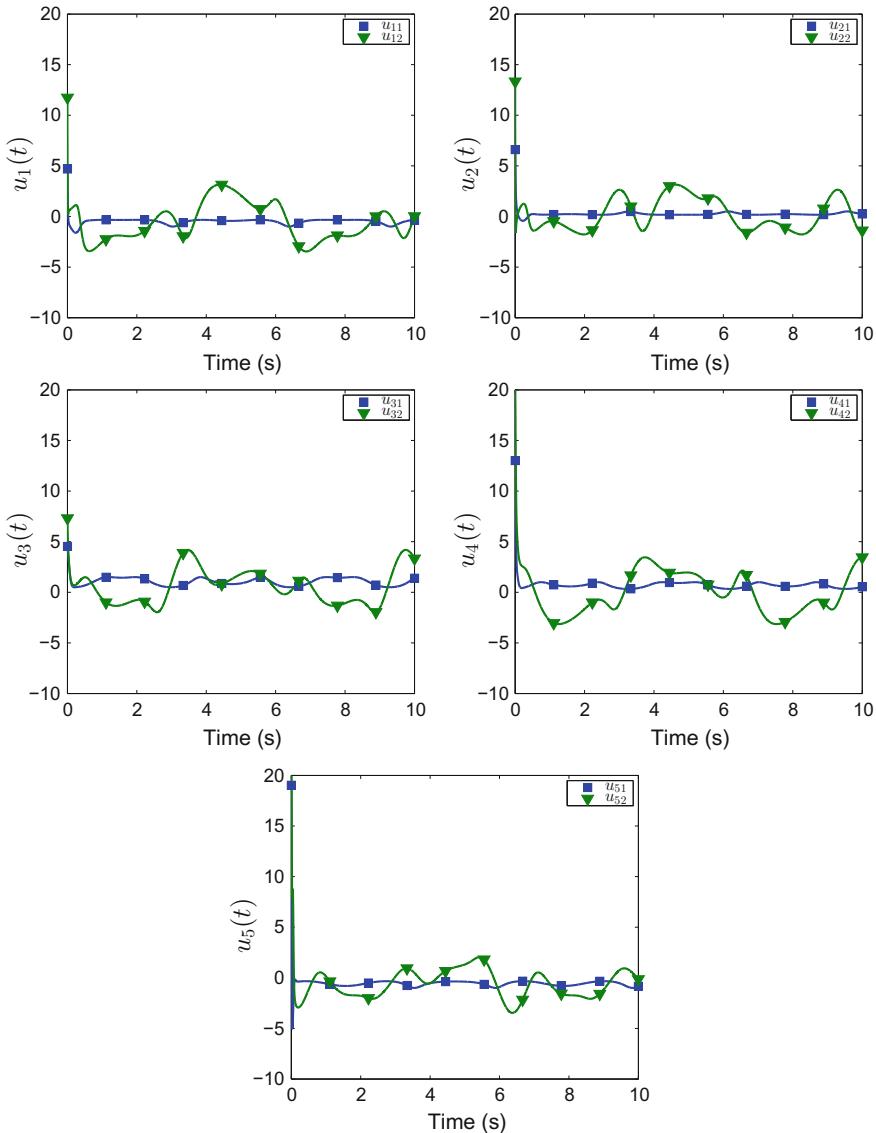
cooperative control based on decentralized objectives. Figure 5.12 show the evolution and convergence of the actor weights for all the agents.

### Three Dimensional Example

To demonstrate the applicability of the developed method to nonholonomic systems, simulations are performed on a five agent network of wheeled mobile robots. The dynamics of the wheeled mobile robots are given by

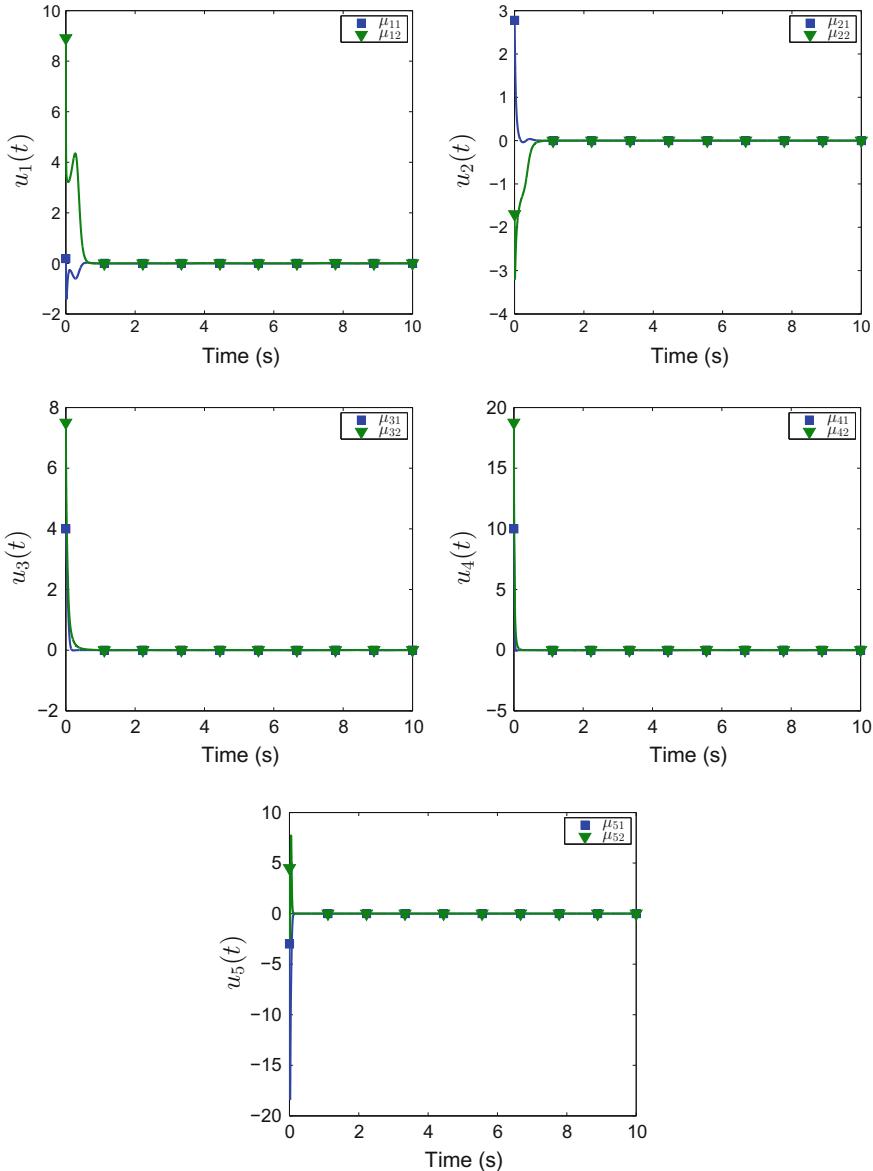
$$\dot{x}_i = g(x_i) u_i, \quad g(x_i) = \begin{bmatrix} \cos(x_{i3}) & 0 \\ \sin(x_{i3}) & 0 \\ 0 & 1 \end{bmatrix},$$

where  $x_{ij}(t)$  denotes the  $j$ th element of the vector  $x_i(t) \in \mathbb{R}^3$ . The desired trajectory is selected to be a circular trajectory that slowly comes to a halt after three rotations, generated using the following dynamical system.



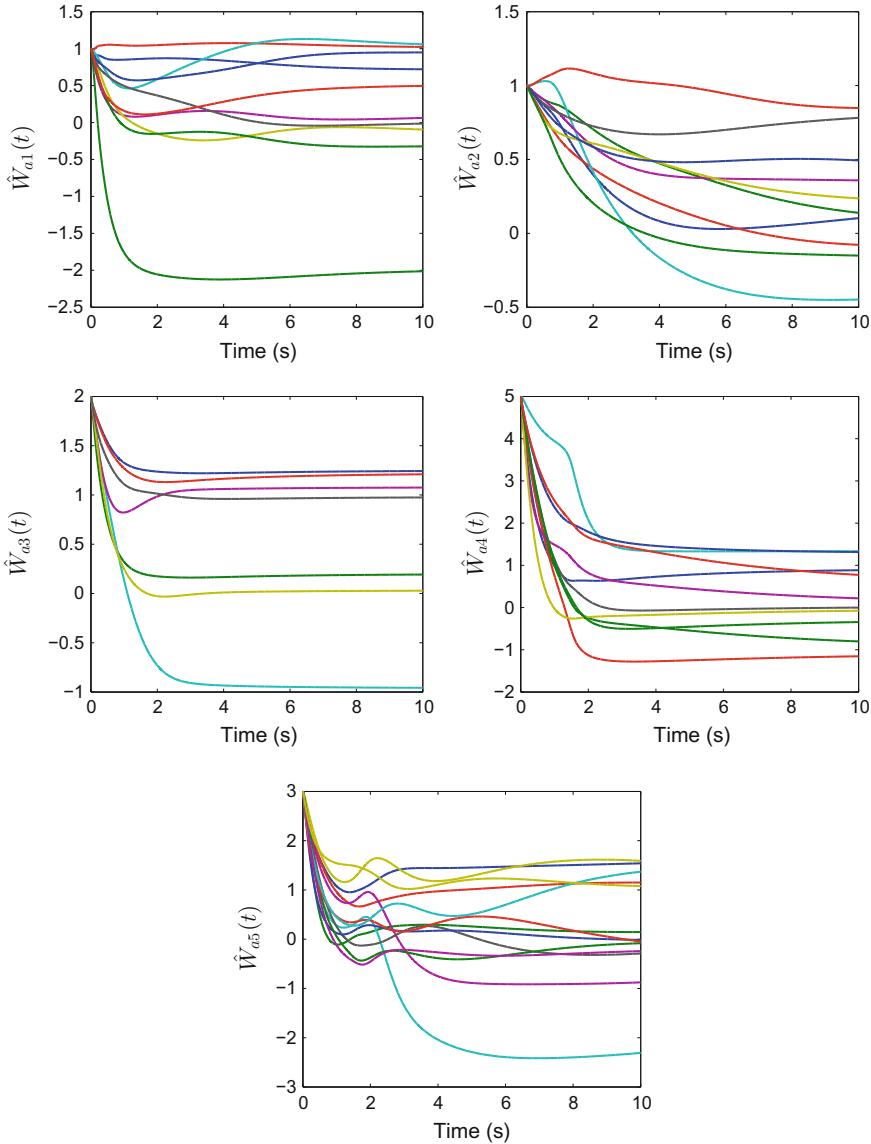
**Fig. 5.10** Trajectories of the control input for Agents 1–5 for the two-dimensional example

$$\dot{x}_0 = \begin{bmatrix} \frac{F_r}{T_p} (T_p^2 - x_{03}^2) \cos(x_{03}) \\ \frac{F_r}{T_p} (T_p^2 - x_{03}^2) \sin(x_{03}) \\ \frac{F_r}{T_p} (T_p^2 - x_{03}^2) \end{bmatrix},$$



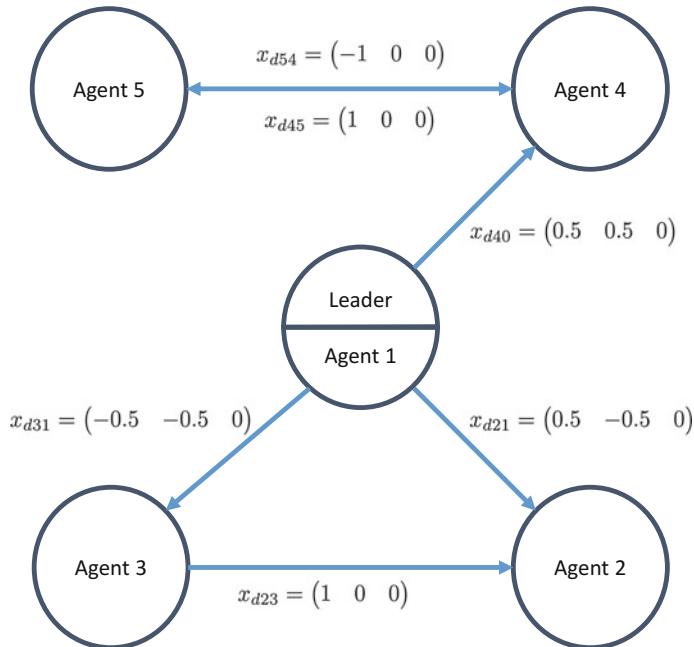
**Fig. 5.11** Trajectories of the relative control error for Agents 1–5 for the two-dimensional example

where  $x_{03}$  denotes the third element of  $x_0$ , and the parameters  $F_r$  and  $T_p$  are selected to be  $F_r = 0.1$  and  $T_p = 6\pi$ . The desired formation and the communication topology are shown in Fig. 5.13. For each agent, a random point is selected in the state space each control iteration for Bellman error extrapolation. Figures 5.14, 5.15, 5.16 and

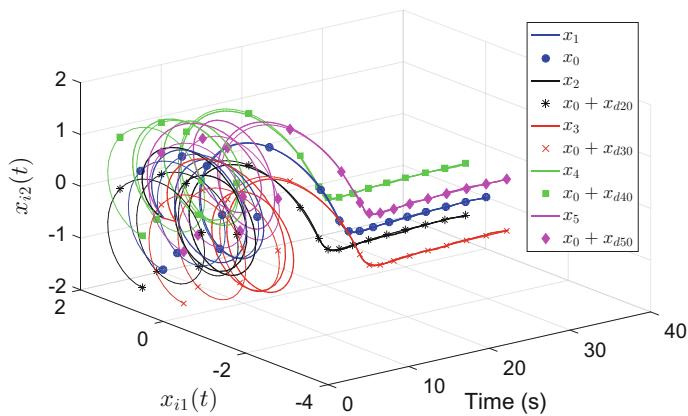


**Fig. 5.12** actor weights for Agents 1–5 for the two-dimensional example

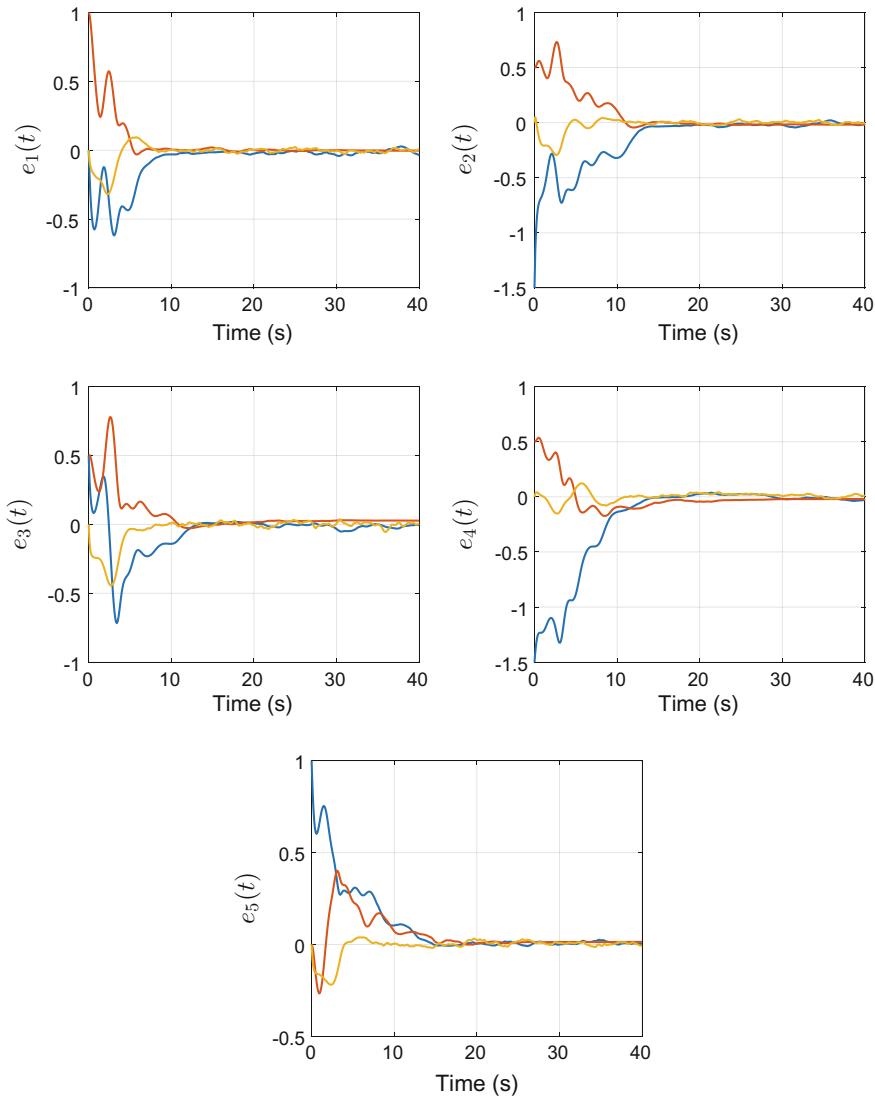
5.17 show the tracking error, the control inputs, and the actor weight estimates for all the agents demonstrating convergence to the desired formation and the desired trajectory. Note that Agents 2, 3 and 5 do not have a direct communication link to the leader, nor do they know their desired relative position with respect to the leader.



**Fig. 5.13** Network topology

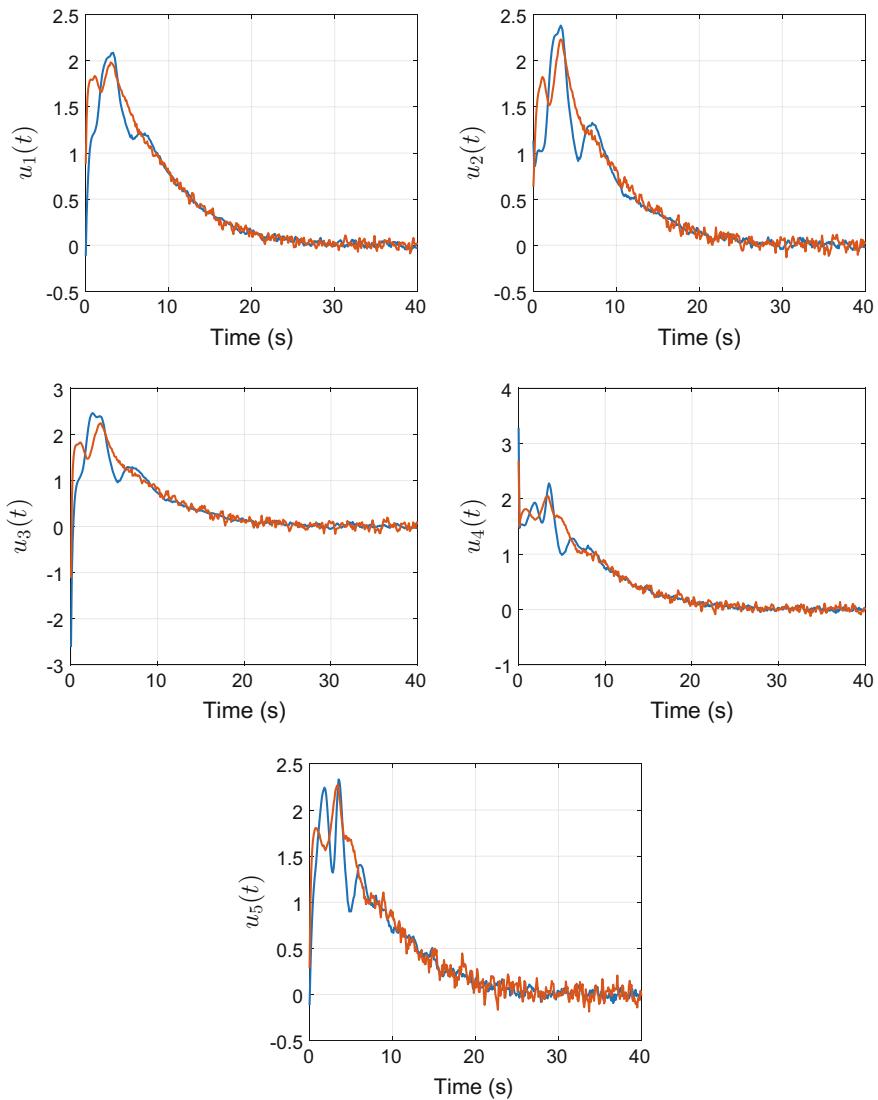


**Fig. 5.14** State trajectories and desired state trajectories as a function of time

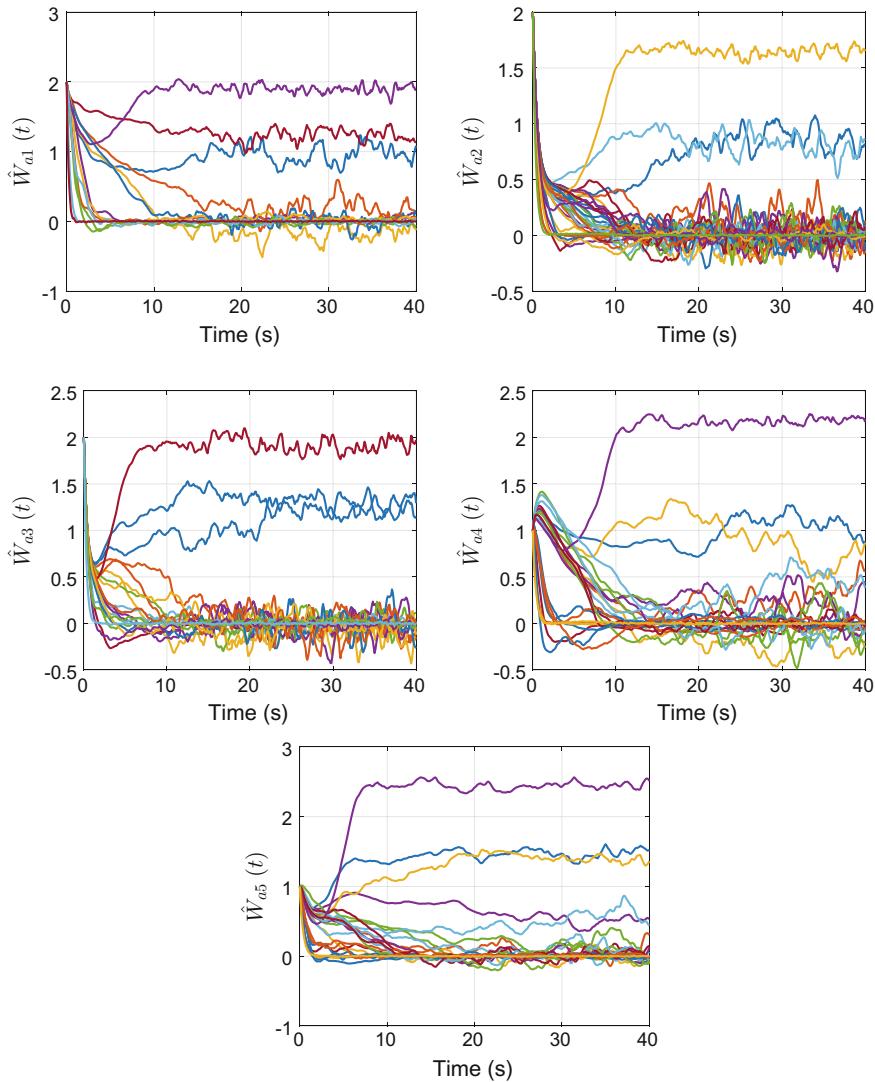


**Fig. 5.15** Tracking error trajectories for the five agents

The convergence to the desired formation is achieved via cooperative control based on decentralized objectives.



**Fig. 5.16** Control trajectories for the five agents



**Fig. 5.17** Actor weight estimates for the five agents

## 5.3 Reinforcement Learning-Based Network Monitoring<sup>2</sup>

### 5.3.1 Problem Description

Consider a network of  $N$  agents with a communication topology described by the directed graph  $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ , where  $\mathcal{V} = \{1, 2, \dots, N\}$  is the set of agents and  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  are the corresponding communication links. The set  $\mathcal{E}$  contains an ordered pair  $(j, i)$  such that  $(j, i) \in \mathcal{E}$  if agent  $j$  communicates information to agent  $i$ . The neighborhood of agent  $i$  is defined as  $\mathcal{N} \triangleq \{j \in \mathcal{V} \mid (j, i) \in \mathcal{E}\}$ , the set of all agents which communicate to  $i$ . It is assumed that the graph is simple, i.e., there are no self loops:  $(i, i) \notin \mathcal{E}$ . Each communication link is weighted by a constant  $a_{ij} \in \mathbb{R}$ , where  $a_{ij} > 0$  if  $(j, i) \in \mathcal{E}$  and  $a_{ij} = 0$  otherwise. The graph adjacency matrix  $A \in \mathbb{R}^{N \times N}$  is constructed from these weights as  $A \triangleq [a_{ij} \mid i, j \in \mathcal{V}]$ .

The interaction of the network leader with the other agents is described by the graph  $\bar{\mathcal{G}} = \{\bar{\mathcal{V}}, \bar{\mathcal{E}}\}$ , which is a supergraph of  $\mathcal{G}$  that includes the leader, denoted by 0, such that  $\bar{\mathcal{V}} = \mathcal{V} \cup \{0\}$ . The communication link set  $\bar{\mathcal{E}}$  is constructed such that  $\bar{\mathcal{E}} \supset \mathcal{E}$  and  $(0, i) \in \bar{\mathcal{E}}$  if the leader communicates to  $i$ . The leader-included neighborhood is accordingly defined as  $\bar{\mathcal{N}}_i \triangleq \{j \in \bar{\mathcal{V}} \mid (j, i) \in \bar{\mathcal{E}}\}$ . Leader connections are weighted by the pinning matrix  $A_0 \in \mathbb{R}^{N \times N}$ , where  $A_0 \triangleq \text{diag}(a_{i0}) \mid i \in \mathcal{V}$  and  $a_{i0} > 0$  if  $(0, i) \in \bar{\mathcal{E}}$  and  $a_{i0} = 0$  otherwise.

The objective of the follower agents is to synchronize in state towards the leader's (possibly time-varying) state. The dynamics for agent  $i \in \mathcal{V}$  are

$$\dot{x}_i(t) = f_i(x_i(t)) + g_i(t)u_i(t), \quad (5.46)$$

where  $x_i : \mathbb{R}_{\geq 0} \rightarrow S$  is the state, the set  $S \subset \mathbb{R}^n$  is the agents' state space,  $f_i : S \rightarrow \mathbb{R}^n$  is a locally Lipschitz function,  $g_i \in \mathbb{R}^{n \times m}$  is a known constant matrix,  $u_i : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^m$  is a pre-established stabilizing and synchronizing control input, and the derivative of the leader state,  $\dot{x}_0$ , is continuous and bounded.

The monitoring objective applied at each agent is to cooperatively monitor the network for satisfaction of its control objective, wherein the network may be affected by input disturbances that cause suboptimal performance. Moreover, the monitoring protocol should be decentralized and passive, i.e., only information from one-hop neighbors should be used and the protocol should not interfere with the monitored systems.

For typical synchronization techniques, such as model predictive control, inverse-optimal, or approximate dynamic programming, a control law is developed based on a cost functional of the form

$$J_i(e_i(\cdot), u_i(\cdot)) \triangleq \int_0^{t_f} (Q_i(e_i(\tau)) + u_i^T(\tau) R_i u_i(\tau)) d\tau, \quad (5.47)$$

---

<sup>2</sup>Parts of the text in this section are reproduced, with permission, from [8], ©2015, IEEE.

where  $t_f > 0$  is the final time of the optimal control problem,  $Q_i : \mathbb{R}^n \rightarrow \mathbb{R}$  is a tracking error weighting function,  $R_i$  is a constant positive definite symmetric weighting matrix, and  $e_i : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$  is the neighborhood tracking error defined as

$$e_i(t) \triangleq \sum_{j \in \mathcal{N}_i} a_{ij} (x_i(t) - x_j(t)) + a_{i0} (x_i(t) - x_0(t)).$$

For notational brevity, let  $\mathcal{E}$  denote the collection  $e_1, \dots, e_N$ .

Even if a controller is not developed based on a cost function, such as in robust and adaptive control, techniques exist which can be used to develop an expression for a meaningful cost functional in the form of (5.47) for a given control policy (cf. [24–26]). The following monitoring approach uses the cost functional in (5.47) to observe how well the networked dynamical systems are satisfying optimality conditions; specifically, satisfaction of the Hamilton–Jacobi–Bellman equation will be monitored to determine how “closely” to optimal the networked systems are operating. Assuming that an optimal controller exists, an equivalent representation of the optimal control problem in (5.47) can be developed in terms of the value function  $V_i : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ , defined as

$$V_i(\mathcal{E}, t) \triangleq \int_t^{t_f} (Q_i(e_i(\tau; t, e_i, u_i(\cdot))) + u_i^T(\tau) R_i u_i(\tau)) d\tau,$$

which is minimized as

$$V_i^*(\mathcal{E}, t) = \min_{u_i \in \mathbb{U}_i} \int_t^{t_f} (Q_i(e_i(\tau; t, e_i, u_i(\cdot))) + u_i^T(\tau) R_i u_i(\tau)) d\tau, \quad (5.48)$$

where  $\mathbb{U}_i$  is the set of admissible controllers for agent  $i$ . Because the minimization of the value function  $V_i$  is inherently coupled with the minimization of other value functions in the network, the value function  $V_i$  can naturally be a function of the error signal  $e_j$ ,  $j \in \mathcal{V}$ , if there exists a directed path from the leader to agent  $i$  that includes agent  $j$ . Using techniques similar to Theorem 5.3, it can be shown that provided a feedback-Nash equilibrium solution exists and that the value functions in (5.48) are continuously differentiable, a necessary and sufficient condition for feedback-Nash equilibrium is given by the system of Hamilton–Jacobi equations

$$\begin{aligned} \sum_{j \in \mathcal{V}} \frac{\partial V_i^*(\mathcal{E}, t)}{\partial e_j} \sum_{k \in \mathcal{N}_j} a_{jk} (f_j(x_j) + g_j u_j^*(\mathcal{E}, t) - f_k(x_k) - g_k u_k^*(\mathcal{E}, t)) \\ + Q_i(e_i) + u_i^{*T}(\mathcal{E}, t) R_i u_i^*(\mathcal{E}, t) + \frac{\partial V_i^*(\mathcal{E}, t)}{\partial t} \equiv 0. \end{aligned} \quad (5.49)$$

Thus, one method for monitoring the network’s operating conditions is to monitor the expression in (5.49), which equals zero for the implementation of optimal control efforts.

Because the optimal value function  $V_i^*$  is often infeasible to solve analytically, an approximate dynamic programming-based approximation scheme is subsequently developed so that the approximate value of  $\frac{\partial V_i^*}{\partial t} + \mathcal{H}_i$  may be monitored. However, the Hamilton–Jacobi equation for agent  $i$  in (5.49) is inherently coupled with the state and control of every agent  $j \in \mathcal{V}$  such that there exists a directed path from the leader to agent  $i$ . Consequently, checking for satisfaction of the Hamilton–Jacobi equations is seemingly unavoidably centralized in information communication. To overcome this restriction, the developed approximate dynamic programming-based approximation scheme is constructed such that only information concerning one-hop neighbors' states, one-hop neighbors' control policies and time is necessary for value function approximation.

To make the current problem tenable, it is assumed that authentic information is exchanged between the agents, i.e., communication is not maliciously compromised; rather, the agents are cooperatively monitoring each other's performance. If necessary, communication authentication algorithms such as in [27] or [28] can be used to verify digitally communicated information.

To evaluate the expression in (5.49), knowledge of the drift dynamics  $f_i$  is required. The following section provides a method to estimate the function  $f_i$  using a data-based approach.

### 5.3.2 System Identification

**Assumption 5.10** The uncertain, locally Lipschitz drift dynamics,  $f_i$ , are linear-in-the-parameters, such that  $f_i(x_i) = Y_i(x_i)\theta_i^*$ , where  $Y_i : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times p_i}$  is a known regression matrix and  $\theta_i^* \in \mathbb{R}^{p_i}$  is a vector of constant unknown parameters.

The function  $\hat{f}_i : \mathbb{R}^n \times \mathbb{R}^{p_i} \rightarrow \mathbb{R}^n$  is an estimate of the uncertain drift dynamics  $f_i$  and is defined as  $\hat{f}_i(x_i, \hat{\theta}_i) \triangleq Y_i(x_i)\hat{\theta}_i$ , where  $\hat{\theta}_i \in \mathbb{R}^{p_i}$  is an estimate of the unknown vector  $\theta_i^*$ . Estimation of  $\theta_i^*$  is facilitated by construction of the identifier

$$\dot{\tilde{x}}_i(t) = \hat{f}_i(x_i(t), \hat{\theta}_i(t)) + g_i(x_i(t))u_i(t) + k_{xi}\tilde{x}_i(t), \quad (5.50)$$

where  $\tilde{x}_i(t) \triangleq x_i(t) - \hat{x}_i(t)$  is the state estimation error, and  $k_{xi} \in \mathbb{R}^{n \times n}$  is a constant positive definite diagonal gain matrix. The state identification error dynamics are expressed using (5.46) and (5.50) as

$$\dot{\tilde{x}}_i(t) = Y_i(x_i(t))\tilde{\theta}_i(t) - k_{xi}\tilde{x}_i(t), \quad (5.51)$$

where  $\tilde{\theta}_i(t) \triangleq \theta^* - \hat{\theta}_i(t)$ . The state estimator in (5.50) is used to develop a data-driven concurrent learning-based update law for  $\hat{\theta}_i(\cdot)$  as

$$\dot{\hat{\theta}}_i(t) = \Gamma_{\theta i} (Y_i(x_i(t)))^T \tilde{x}_i(t) + \Gamma_{\theta i} k_{\theta i} \sum_{\xi=1}^K \left( Y_i^\xi \right)^T \left( \dot{x}_i^\xi - g_i^\xi u_i^\xi - Y_i^\xi \hat{\theta}_i(t) \right), \quad (5.52)$$

where  $\Gamma_{\theta i} \in \mathbb{R}^{p_i \times p_i}$  is a constant positive definite symmetric gain matrix,  $k_{\theta i} \in \mathbb{R}_{>0}$  is a constant concurrent learning gain, and the superscript  $(\cdot)^\xi$  denotes evaluation at one of the unique recorded values in the state data stack  $\{x_i^\xi \mid \xi = 1, \dots, K\}$  or corresponding control value data stack  $\{u_i^\xi \mid \xi = 1, \dots, K\}$ . It is assumed that these data stacks are recorded prior to use of the drift dynamics estimator. The following assumption specifies the necessary data richness of the recorded data.

**Assumption 5.11** [22] There exists a finite set of collected data  $\{x_i^\xi \mid \xi = 1, \dots, K\}$  such that

$$\text{rank} \left( \sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi \right) = p_i. \quad (5.53)$$

Note that persistence of excitation is not mentioned as a necessity for this identification algorithm; instead of guaranteeing data richness by assuming that the dynamics are persistently exciting for all time, it is only assumed that there exists a *finite* set of data points that provide the necessary data richness. This also eliminates the common requirement for injection of a persistent dither signal to attempt to ensure persistence of excitation, which would interfere with the monitored systems. Furthermore, contrary to persistence of excitation-based approaches, the condition in (5.53) can be verified. Note that, because (5.52) depends on the state derivative  $\dot{x}_i^\xi$  at a past value, numerical techniques can be used to approximate  $\dot{x}_i^\xi$  using preceding and proceeding recorded state information.

To facilitate an analysis of the performance of the identifier in (5.52), the identifier dynamics are expressed in terms of estimation errors as

$$\dot{\tilde{\theta}}_i(t) = -\Gamma_{\theta i} (Y_i(x_i(t)))^T \tilde{x}_i(t) - \Gamma_{\theta i} k_{\theta i} \sum_{\xi=1}^K \left( Y_i^\xi \right)^T \left( \dot{x}_i^\xi - g_i^\xi u_i^\xi - Y_i^\xi \hat{\theta}_i(t) \right). \quad (5.54)$$

To describe the performance of the identification of  $\theta_i^*$ , consider the positive definite continuously differentiable Lyapunov function  $V_{\theta i} : \mathbb{R}^{n+p_i} \rightarrow [0, \infty)$  defined as

$$V_{\theta i}(z_i) \triangleq \frac{1}{2} \tilde{x}_i^T \tilde{x}_i + \frac{1}{2} \tilde{\theta}_i^T \Gamma_{\theta i}^{-1} \tilde{\theta}_i, \quad (5.55)$$

where  $z_i \triangleq [\tilde{x}_i^T, \tilde{\theta}_i^T]^T$ . The expression in (5.55) satisfies the inequalities

$$\underline{V}_{\theta i} \|z_i\|^2 \leq V_{\theta i}(z_i) \leq \bar{V}_{\theta i} \|z_i\|^2, \quad (5.56)$$

where  $\underline{V}_{\theta i} \triangleq \frac{1}{2} \min(1, \lambda_{\min}\{\Gamma_{\theta i}^{-1}\})$  and  $\bar{V}_{\theta i} \triangleq \frac{1}{2} \max(1, \lambda_{\max}\{\Gamma_{\theta i}^{-1}\})$  are positive known constants. Using the dynamics in (5.51) and (5.54), the time derivative of (5.55) is

$$\dot{V}_{\theta i}(z_i) = -\tilde{x}_i^T k_{xi} \tilde{x}_i - \tilde{\theta}_i^T k_{\theta i} \left( \sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi \right) \tilde{\theta}_i. \quad (5.57)$$

Note that because the matrix  $\sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi$  is symmetric and positive semi-definite, its eigenvalues are real and greater than or equal to zero. Furthermore, by Assumption 5.11, none of the eigenvalues of  $\sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi$  are equal to zero. Thus, all of the eigenvalues of the symmetric matrix  $\sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi$  are positive, and the matrix  $\sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi$  is positive definite. Using this property and the inequalities in (5.56), (5.57) is upper bounded as

$$\dot{V}_{\theta i}(z_i) \leq -c_i \|z_i\|^2 \leq -\frac{c_i}{\bar{V}_{\theta i}} V_{\theta i}(z_i), \quad (5.58)$$

where  $c_i \triangleq \min\left(\lambda_{\min}\{k_{xi}\}, k_{\theta i} \lambda_{\min}\left\{\sum_{\xi=1}^K \left( Y_i^\xi \right)^T Y_i^\xi\right\}\right)$ . The inequalities in (5.56) and (5.58) can then be used to conclude that  $\|\tilde{x}_i(t)\|, \|\tilde{\theta}_i(t)\| \rightarrow 0$  exponentially fast. Thus, the drift dynamics  $f_i(x_i) = Y_i(x_i) \theta_i^*$  are exponentially identified.

Note that even with state derivative estimation errors, the parameter estimation error  $\tilde{\theta}_i(\cdot)$  can be shown to be uniformly ultimately bounded, where the magnitude of the ultimate bound depends on the derivative estimation error [22].

*Remark 5.12* Using an integral formulation, the system identifier can also be implemented without using state-derivative measurements (see, e.g., [14]).

### 5.3.3 Value Function Approximation

For the approximate value of (5.49) to be evaluated for monitoring purposes, the unknown optimal value function  $V_i^*$  needs to be approximated for each agent  $i$ . Because the coupled Hamilton–Jacobi equations are typically difficult to solve analytically, this section provides an approach to approximate  $V_i^*$  using neural networks.

Assuming the networked agents' states remain bounded, the universal function approximation property of neural networks can be used with  $w_i$  neurons to equivalently represent  $V_i^*$  as

$$V_i^*(e_i, t') = W_i^T \sigma_i(e_i, t') + \epsilon_i(e_i, t'), \quad (5.59)$$

where  $t' \triangleq \frac{t}{t_f}$  is the normalized time,  $W_i \in \mathbb{R}^{w_i}$  is an unknown ideal neural network weight vector bounded above by a known constant  $\bar{W}_i \in \mathbb{R}_{>0}$  as  $\|W_i\| \leq \bar{W}_i$ ,  $\sigma_i : S \times [0, 1] \rightarrow \mathbb{R}^{w_i}$  is a bounded nonlinear continuously differentiable activation function with the property  $\sigma(0, 0) = 0$ , where  $\epsilon_i : S \times [0, 1] \rightarrow \mathbb{R}$  is the unknown function reconstruction error. From the universal function approximation property, the reconstruction error  $\epsilon_i$  satisfies the properties  $\sup_{\rho \in S, \varphi \in [0, 1]} |\epsilon_i(\rho, \varphi)| \leq \bar{\epsilon}_i$ ,  $\sup_{\rho \in S, \varphi \in [0, 1]} \frac{\partial |\epsilon_i(\rho, \varphi)|}{\partial \rho} \leq \bar{\epsilon}_{ei}$ , and  $\sup_{\rho \in S, \varphi \in [0, 1]} \frac{\partial |\epsilon_i(\rho, \varphi)|}{\partial \varphi} \leq \bar{\epsilon}_{ti}$ , where  $\bar{\epsilon}_i, \bar{\epsilon}_{ei}, \bar{\epsilon}_{ti} \in \mathbb{R}_{>0}$  are constant upper bounds.

Note that only the state of agent  $i$ , states of neighbors of agent  $i$  ( $j \in \bar{\mathcal{N}}_i$ ), and time are used as arguments in the neural network representation of  $V_i^*$  in (5.59), instead of the states of all agents in the network. This is justified by treating the error states of other agents simply as functions of time, the effect of which is accommodated by including time in the basis function  $\sigma_i$  and function reconstruction error  $\epsilon_i$ . Inclusion of time in the basis function is feasible due to the finite horizon of the optimization problem in (5.47). Using state information from additional agents (e.g., two-hop communication) in the network may increase the practical fidelity of function reconstruction and may be done in an approach similar to that developed in this section.

Using this neural network representation,  $V_i^*$  is approximated for use in computing the Hamiltonian as

$$\hat{V}_i(e_i, \hat{W}_{ci}, t') \triangleq \hat{W}_{ci}^T \sigma_i(e_i, t'),$$

where  $\hat{W}_{ci}$  is an estimate of the ideal neural network weight vector  $W_i$ .

To facilitate the development of a feedback-based update policy to drive  $\hat{W}_{ci}(\cdot)$  towards  $W_i$ , the Bellman error for agent  $i$  is defined as

$$\begin{aligned} \delta_i(t') &\triangleq \hat{W}_{ci}^T \sigma_{ti}(e_i, t') + \hat{\mathcal{H}}_i(E_i, X_i, \hat{W}_{ci}, \hat{\omega}_{ai}, t') \\ &\quad - \left( \frac{\partial V_i^*(\mathcal{E}, t')}{\partial t} + \mathcal{H}_i(\mathcal{E}, \mathcal{X}, \mathcal{U}^*, V_i^*, t') \right), \end{aligned} \quad (5.60)$$

where  $\sigma_{ti}(e_i, t') \triangleq \frac{\partial \sigma_i(e_i, t')}{\partial t}$ ,  $E_i \triangleq \{e_j \mid j \in \{i\} \cup \bar{\mathcal{N}}_i\}$  is the set of error states of agent  $i$  and the neighbors of agent  $i$ ,  $X_i \triangleq \{x_j \mid j \in \{i\} \cup \bar{\mathcal{N}}_i\}$  is the set of states of agent  $i$  and the neighbors of agent  $i$ ,  $\hat{\omega}_{ai} \triangleq \{\hat{W}_{ai} \mid j \in \{i\} \cup \bar{\mathcal{N}}_i\}$  is the set of actor weights of agent  $i$  and the neighbors of agent  $i$ ,  $\mathcal{H}_i$  is the Hamiltonian defined as

$$\begin{aligned} \mathcal{H}_i(\mathcal{E}, \mathcal{X}, \mathcal{U}^*, V_i^*, t) &\triangleq Q_i(e_i) + u_i^{*T}(\mathcal{E}, t) R_i u_i^*(\mathcal{E}, t) \\ &\quad + \sum_{j \in \mathcal{V}} \frac{\partial V_i^*(\mathcal{E}, t)}{\partial e_j} \sum_{k \in \bar{\mathcal{N}}_j} a_{jk} (f_j(x_j) + g_j u_j^*(\mathcal{E}, t) - f_k(x_k) - g_k u_k^*(\mathcal{E}, t)) \end{aligned} \quad (5.61)$$

where the sets  $X$  and  $U$  are defined as  $\mathcal{X} \triangleq \{x_i \mid i \in \mathcal{V}\}$  and  $\mathcal{U} \triangleq \{u_i \mid i \in \mathcal{V}\}$  and  $\hat{\mathcal{H}}_i$  is the approximate Hamiltonian defined as

$$\begin{aligned} \hat{\mathcal{H}}_i(E_i, X_i, \hat{W}_{ci}, \hat{\omega}_{ai}, t') &\triangleq Q_i(e_i) + \hat{u}_i^T(e_i, \hat{W}_{ai}, t') R_i \hat{u}_i(e_i, \hat{W}_{ai}, t') \\ &+ \hat{W}_{ci}^T \sigma_{ei}(e_i, t') \left( \sum_{j \in \mathcal{N}_i} a_{ij} (\hat{f}_i(x_i) + g_i \hat{u}_i(e_i, \hat{W}_{ai}, t') - \hat{f}_j(x_j) - g_j \hat{u}_j(e_j, \hat{W}_{aj}, t')) \right. \\ &\left. + a_{i0} (\hat{f}_i(x_i) + g_i \hat{u}_i(e_i, \hat{W}_{ai}, t') - \dot{x}_0(t')) \right), \end{aligned} \quad (5.62)$$

where  $\sigma_{ei}(e_i, t') \triangleq \frac{\partial \sigma_i(e_i, t')}{\partial e_i}$ ,  $\hat{u}_i(e_i, \hat{W}_{ai}, t') \triangleq -\frac{1}{2} \left( \sum_{j \in \mathcal{N}_i} a_{ij} \right) R_i^{-1} g_i^T \sigma_{ei}^T(e_i, t')$ .  $\hat{W}_{ai}$  is the approximated optimal control for agent  $i$ , and  $\hat{W}_{ai}$  is another estimate of the ideal neural network weight vector  $W_i$ . Noting that the expression in (5.62) is measurable (assuming that the leader state derivative is available to those communicating with the leader), the Bellman error in (5.60) may be put into measurable form, after recalling that  $\frac{\partial V_i^*}{\partial t} + \mathcal{H}_i(\mathcal{E}, \mathcal{X}, \mathcal{U}^*, V_i^*, t) \equiv 0$ , as

$$\delta_i(t') \triangleq \hat{W}_{ci}^T \sigma_{ti}(e_i, t') + \hat{\mathcal{H}}_i(E_i, X_i, \hat{W}_{ci}, \hat{\omega}_{ai}, t'), \quad (5.63)$$

which is the feedback to be used to train the neural network estimate  $\hat{W}_{ci}$ . The use of the two neural network estimates  $\hat{W}_{ci}$  and  $\hat{W}_{ai}$  allows for least-squares based adaptation for the feedback in (5.63), since only the use of  $\hat{W}_{ci}$  would result in nonlinearity of  $\hat{W}_{ci}$  in (5.63).

The difficulty in making a non-interfering monitoring scheme with approximate dynamic programming lies in obtaining sufficient data richness for learning. Contrary to typical approximate dynamic programming-based control methods, the developed data-driven adaptive learning policy uses concepts from concurrent learning (cf. [22, 29]) to provide data richness. Let  $s_i \triangleq \{\rho_l \in S \mid l = 1, \dots, (|\mathcal{N}_i| + 1)\} \cup \{\varphi \mid \varphi \in [0, t_f]\}$  be a pre-selected sample point in the state space of agent  $i$ , its neighbors, and time. Additionally, let  $S_i \triangleq \{s_i^{c_l} \mid c_l = 1, \dots, L\}$  be a collection of these unique sample points. The Bellman error will be evaluated over the set  $S_i$  in the neural network update policies to guarantee data richness. As opposed to the common practice of injecting an exciting signal into a system's control input to provide sufficient data richness for adaptive learning, this strategy evaluates the Bellman error at preselected points in the state space and time to mimic exploration of the state space. The following assumption specifies a sufficient condition on the set  $S_i$  for convergence of the subsequently defined update policies.

**Assumption 5.13** For each agent  $i \in \mathcal{V}$ , the set of sample points  $S_i$  satisfies

$$\mu_i \triangleq \frac{1}{L} \inf_{t \in [0, t_f]} \lambda_{\min} \left\{ \sum_{c_l=1}^L \frac{\chi_i^{c_l} (\chi_i^{c_l})^T}{\gamma_i^{c_l}} \right\} > 0, \quad (5.64)$$

where  $(\cdot)^{c_l}$  denotes evaluation at the  $c_l^{\text{th}}$  sample point for the indicated agent,  $\chi_i \triangleq \sigma_{ti} + \sigma_{ei} \left( \sum_{j \in \mathcal{N}_i} a_{ij} (\hat{f}_i(x_i) + g_i \hat{u}_i - \hat{f}_j(x_j) - g_j \hat{u}_j) + a_{i0} (\hat{f}_i(x_i) + g_i \hat{u}_i - \dot{x}_0) \right)$

is a regressor vector in the developed neural network update law,  $\gamma_i \triangleq 1 + \lambda_i (\chi_i)^T \Gamma_i \chi_i$  provides normalization to the developed neural network update law,  $\lambda_i \in \mathbb{R}_{>0}$  is a constant normalization gain, and  $\Gamma_i \in \mathbb{R}^{w_i \times w_i}$  is a subsequently defined least-squares positive definite gain matrix.

Note that when evaluating expressions at the pre-selected concurrent learning sample points, the current values of parameter estimates are used since they are approximating constant values. In general, similar to the selection of a dither signal in persistence of excitation-based approaches (as in [30, 31]), the satisfaction of (5.64) cannot be guaranteed a priori. However, this strategy benefits from the ability to accommodate this condition by selecting more information than theoretically necessary (i.e., selecting sample points such that  $L \gg w_i$ ). Additionally, satisfaction of the condition in (5.64) up until the current time can be verified online.

Using the measurable form of the Bellman error in (5.63) as feedback, a concurrent learning-based least-squares update policy is developed to approximate  $W_i$  as [29]

$$\dot{\hat{W}}_{ci} = -\phi_{c1i} \Gamma_i \frac{\chi_i}{\gamma_i} \delta_i - \frac{\phi_{c2i}}{L} \Gamma_i \sum_{c_l=1}^L \frac{\chi_i^{c_l}}{\gamma_i^{c_l}} \delta_i^{c_l}, \quad (5.65)$$

$$\dot{\Gamma}_i = \left( \beta_i \Gamma_i - \phi_{c1i} \Gamma_i \frac{\chi_i \chi_i^T}{\gamma_i^2} \Gamma_i \right) \mathbf{1}_{\{\|\Gamma_i \leq \bar{\Gamma}_i\|\}}, \quad (5.66)$$

where  $\phi_{c1i}, \phi_{c2i} \in \mathbb{R}_{>0}$  are constant adaptation gains,  $\beta_i \in \mathbb{R}_{>0}$  is a constant forgetting factor,  $\bar{\Gamma}_i \in \mathbb{R}_{>0}$  is a saturation constant, and  $\Gamma_i(0)$  is positive definite, symmetric, and bounded such that  $\|\Gamma_i(0)\| \leq \bar{\Gamma}_i$ . The form of the least-squares gain matrix update law in (5.66) is constructed such that  $\Gamma_i$  remains positive definite and

$$\underline{\Gamma}_i \leq \|\Gamma_i(t)\| \leq \bar{\Gamma}_i, \quad \forall t \in \mathbb{R}_{\geq 0}, \quad (5.67)$$

where  $\underline{\Gamma}_i \in \mathbb{R}_{>0}$  is constant [32]. The neural network estimate  $\hat{W}_{ai}$  is updated towards the estimate  $\hat{W}_{ci}$  as

$$\dot{\hat{W}}_{ai} = -\phi_{a1i} \left( \hat{W}_{ai} - \hat{W}_{ci} \right) - \phi_{a2i} \hat{W}_{ai} + \left( \frac{\phi_{c1i} G_{\sigma i} \hat{W}_{ai} \chi_i^T}{4\gamma_i} + \sum_{c_l=1}^L \frac{\phi_{c2i} G_{\sigma i}^{c_l} \hat{W}_{ai} (\chi_i^{c_l})^T}{4L\gamma_i^{c_l}} \right) \hat{W}_{ci}, \quad (5.68)$$

where  $\phi_{a1i}, \phi_{a2i} \in \mathbb{R}_{>0}$  are constant adaptation gains,  $G_{\sigma i} \triangleq \left( \sum_{j \in \bar{\mathcal{N}}_i} a_{ij} \right)^2 \sigma_{ei} G_i \sigma_{ei}^T \in \mathbb{R}^{w_i}$ , and  $G_i \triangleq g_i R_i^{-1} g_i^T$ .

### 5.3.4 Stability Analysis

The following theorem summarizes the stability properties of the leader-follower network.

**Theorem 5.14** *For every agent  $i \in \mathcal{V}$ , the identifier in (5.52) along with the adaptive update laws in (5.65)–(5.68) guarantee that the estimation errors  $\tilde{W}_{ci} \triangleq W_i - \hat{W}_{ci}$  and  $\tilde{W}_{ai} \triangleq W_i - \hat{W}_{ai}$  uniformly converge in a finite time  $T_W$  to  $B_r$ , provided Assumptions (5.10)–(5.13) hold, the adaptation gains are selected sufficiently large, and the concurrent learning sample points are selected to produce a sufficiently large  $\mu_i$ , where  $T_W$  and  $r$  can be made smaller by selecting more concurrent learning sample points to increase the value of  $\mu_i$  and increasing the gains  $k_{xi}$ ,  $k_{\theta i}$ ,  $\phi_{a1i}$ ,  $\phi_{a2i}$ , and  $\phi_{c2i}$ .*

*Proof* (Sketch) Consider the Lyapunov function

$$V_L \triangleq \sum_{i \in \mathcal{V}} \left( \frac{1}{2} \tilde{W}_{ci}^T \Gamma_i^{-1} \tilde{W}_{ci} + \frac{1}{2} \tilde{W}_{ai}^T \tilde{W}_{ai} + V_{\theta i}(z_i) \right). \quad (5.69)$$

An upper bound of  $\dot{V}_L$  along the trajectories of (5.51), (5.54), (5.65), and (5.68) can be obtained after expressing  $\delta_i$  in terms of estimation errors, using the property  $\frac{d}{dt}(\Gamma_i^{-1}) = -\Gamma_i^{-1} \dot{\Gamma}_i \Gamma_i^{-1}$ , using the inequality  $\left\| \frac{\lambda_i}{\gamma_i} \right\| \leq \frac{1}{2\sqrt{\lambda_i \Gamma_i}}$ , applying the Cauchy–Schwarz and triangle inequalities, and performing nonlinear damping, such that  $\dot{V}_L$  is bounded from above by an expression that is negative definite in terms of the state of  $V_L$  plus a constant upper-bounding term. Using [21, Theorem 4.18], it can be shown that the estimation errors are uniformly ultimately bounded.

### 5.3.5 Monitoring Protocol

With the estimation of the unknown neural network weight  $W_i$  by  $\hat{W}_{ci}$  from the previous section, the performance of an agent in satisfying the Hamilton–Jacobi–Bellman optimality constraint can be monitored through use of  $\hat{V}_i = \hat{W}_{ci}^T \sigma_i$ , the approximation of  $V_i^*$ . From (5.49),  $\frac{\partial V_i^*}{\partial t} + \mathcal{H}_i(\mathcal{E}, \mathcal{X}, \mathcal{U}^*, V_i^*, t) \equiv 0$  (where  $\mathcal{U}^*$  denotes the optimal control efforts). Let  $M_i \in \mathbb{R}$  denote the signal to be monitored by agent  $i \in \mathcal{V}$ , defined as

$$\begin{aligned} M_i(e_i, X_i, \hat{W}_{ci}, U_i, t') = & \left| \hat{W}_{ci}^T \sigma_{ti} + \hat{W}_{ci}^T \sigma_{ei} \left( \sum_{j \in \mathcal{N}_i} a_{ij} (\hat{f}_i(x_i) + g_i u_i - \hat{f}_j(x_j) - g_j u_j) \right. \right. \\ & \left. \left. + Q_i(e_i) + u_i^T R_i u_i + a_{i0} (\hat{f}_i(x_i) + g_i u_i - \dot{x}_0) \right) \right|, \end{aligned}$$

which differs from (5.63) in that the measured values of control efforts are being used, where  $U_i \triangleq \{u_j \mid j \in \{i\} \cup \mathcal{N}_i\}$ . Because the identification of  $f_i$  is performed exponentially fast and the uniform convergence of  $\hat{W}_{ci}$  to a ball around the origin occurs in the finite learning time  $T_W$ , the monitored signal  $M_i$  satisfies the relationship  $\left| \frac{\partial V_i^*}{\partial t} + \mathcal{H}_i(\mathcal{E}, \mathcal{X}, \mathcal{U}^*, V_i^*, t) - M_i(e_i, X_i, \hat{W}_{ci}, U_i^*, t') \right| < \varsigma \forall t \geq T_W$ , where  $\varsigma \in \mathbb{R}_{>0}$  is some bounded constant that can be made smaller by selecting larger gains and more concurrent learning sample points. In other words, when using the neural network approximation of  $V_i^*$  and appropriate gains and sample points,  $M_i(e_i, X_i, \hat{W}_{ci}, U_i^*, t') \approx 0$ , where  $U_i^* \triangleq \{u_j^* \mid j \in \{i\} \cup \mathcal{N}_i\}$ . In this manner, due to continuity of the Hamiltonian, observing large values for  $M_i(e_i, X_i, \hat{W}_{ci}, U_i, t')$  indicates significant deviation away from optimal operating conditions. Let  $\bar{M} \in \mathbb{R}_{>0}$  be a constant threshold used for the monitoring process which satisfies  $\bar{M} > \varsigma$ , where the value for  $\bar{M}$  can be increased if a greater tolerance for sub-optimal performance is acceptable. The monitoring protocol, which is separated into a learning phase and a monitoring phase, can be summarized as follows.

---

**Algorithm 5.1** Monitoring protocol.

---

Learning phase:

For each agent  $i \in \mathcal{V}$ , use the update policies in (5.50), (5.52), (5.65), (5.66), (5.68) to train  $\hat{\theta}_i$  and  $\hat{W}_{ci}$ , the estimates for  $\theta_i^*$  and  $W_i$ .

Monitoring phase:

At  $t = T_W$ , terminate updates to  $\hat{\theta}_i$  and  $\hat{W}_{ci}$ .

For each agent  $i \in \mathcal{V}$ , monitor the value of  $M_i(e_i, X_i, \hat{W}_{ci}, U_i, t')$  using  $\hat{\theta}_i$ ,  $\hat{W}_{ci}$ , neighbor communication  $\{x_j, u_j, \hat{f}_j, g_j\}$  and  $\dot{x}_0$  if  $(i, 0) \in \bar{E}$ .

If  $M_i > \bar{M}$ , then undesirable performance has been observed.

---

## 5.4 Background and Further Reading

Online real-time solutions to differential games are presented in results such as [15, 16, 33–35]; however, since these results solve problems with centralized objectives (i.e., each agent minimizes or maximizes a cost function that penalizes the states of all the agents in the network), they are not applicable for a network of agents with independent decentralized objectives (i.e., each agent minimizes or maximizes a cost function that penalizes only the error states corresponding to itself).

Various methods have been developed to solve formation tracking problems for linear systems (cf. [36–40] and the references therein). An optimal control approach is used in [41] to achieve consensus while avoiding obstacles. In [42], an optimal controller is developed for agents with known dynamics to cooperatively track a desired trajectory. In [43], an inverse optimal controller is developed for unmanned

aerial vehicles to cooperatively track a desired trajectory while maintaining a desired formation. In [44], a differential game-based approach is developed for unmanned aerial vehicles to achieve distributed Nash strategies. In [45], an optimal consensus algorithm is developed for a cooperative team of agents with linear dynamics using only partial information. A value function approximation based approach is presented in [17] for cooperative synchronization in a strongly connected network of agents with known linear dynamics.

For nonlinear systems, model-predictive control-based approaches ([46, 47]) and approximate dynamic programming-based approaches ([17, 48]) have been proposed. A model-predictive control-based approach is presented in [46]; however, no stability or convergence analysis is presented. A stable distributed model-predictive control-based approach is presented in [47] for nonlinear discrete-time systems with known nominal dynamics. Asymptotic stability is proved without any interaction between the nodes; however, a nonlinear optimal control problem needs to be solved at every iteration to implement the controller. An optimal tracking approach for formation control is presented in [48] using single network adaptive critics where the value function is learned offline. Online feedback-Nash equilibrium solution of differential graphical games in a network of agents with continuous-time uncertain nonlinear dynamics has remained an open problem. Recently, a leader-based consensus algorithm is developed in [49] where exact model of the system dynamics is utilized, and convergence to optimality is obtained under a persistence of excitation condition. The model-predictive control-based controllers require extensive numerical computations and lack stability and optimality guarantees. The approximate dynamic programming-based approaches either require offline computations or are suboptimal because not all the inter-agent interactions are considered in the value function.

Efforts have been made to structure the layout of a network such that the impact of network corruption can be abated [50, 51]. Researchers have also investigated the ability to allay types of network subterfuge by creating control algorithms which are resilient to attacks on sensors and actuators [52]. Other efforts seek to have agents detect undesired performance in their network neighbors. The results in [53] and [54] provide methods to detect “sudden” faulty behavior which is modeled as a step function multiplied by fault dynamics. Other works develop procedures to detect generally undesired behavior in networks of linear systems [27, 55] and unpredictable state trajectories of nonlinear systems using neural networks [56, 57]. Adaptive thresholds used for determining if the state of a neighboring agent is within an acceptable tolerance are developed in [58] and [59].

Contemporary results on online near-optimal control of multi-agent systems include disturbance rejection methods [60], off-policy methods [61], and Q-learning methods [62]. For a complete description of recent developments on online methods to solve multiplayer games, see [63].

## References

1. Kamalapurkar R, Klotz JR, Walters P, Dixon WE (2018) Model-based reinforcement learning for differential graphical games. *IEEE Trans Control Netw Syst* 5:423–433
2. Case J (1969) Toward a theory of many player differential games. *SIAM J Control* 7:179–197
3. Starr A, Ho CY (1969) Nonzero-sum differential games. *J Optim Theory Appl* 3(3):184–206
4. Starr A, Ho CY (1969) Further properties of nonzero-sum differential games. *J Optim Theory Appl* 4:207–219
5. Friedman A (1971) Differential games. Wiley
6. Bressan A, Priuli FS (2006) Infinite horizon noncooperative differential games. *J Differ Equ* 227(1):230–257
7. Bressan A (2011) Noncooperative differential games. *Milan J Math* 79(2):357–427
8. Klotz J, Andrews L, Kamalapurkar R, Dixon WE (2015) Decentralized monitoring of leader-follower networks of uncertain nonlinear systems. In: Proceedings of the American control conference, pp 1393–1398
9. Khoo S, Xie L (2009) Robust finite-time consensus tracking algorithm for multirobot systems. *IEEE/ASME Trans Mechatron* 14(2):219–228
10. Liberzon D (2012) Calculus of variations and optimal control theory: a concise introduction. Princeton University Press
11. Kamalapurkar R, Andrews L, Walters P, Dixon WE (2017) Model-based reinforcement learning for infinite-horizon approximate optimal tracking. *IEEE Trans Neural Netw Learn Syst* 28(3):753–758
12. Chowdhary GV, Johnson EN (2011) Theory and flight-test validation of a concurrent-learning adaptive controller. *J Guid Control Dyn* 34(2):592–607
13. Kamalapurkar R, Walters P, Dixon WE (2016) Model-based reinforcement learning for approximate optimal regulation. *Automatica* 64:94–104
14. Bell Z, Parikh A, Nezadovitz J, Dixon WE (2016) Adaptive control of a surface marine craft with parameter identification using integral concurrent learning. In: Proceedings of the IEEE conference on decision and control, pp 389–394
15. Vamvoudakis KG, Lewis FL (2011) Multi-player non-zero-sum games: online adaptive learning solution of coupled hamilton-jacobi equations. *Automatica* 47:1556–1569
16. Johnson M, Bhasin S, Dixon WE (2011) Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm. In: Proceedings of the IEEE conference on decision and control, pp 142–147
17. Vamvoudakis KG, Lewis FL, Hudas GR (2012) Multi-agent differential graphical games: online adaptive learning solution for synchronization with optimality. *Automatica* 48(8):1598–1611
18. Kamalapurkar R, Dinh HT, Walters P, Dixon WE (2013) Approximate optimal cooperative decentralized control for consensus in a topological network of agents with uncertain nonlinear dynamics. In: Proceedings of the American control conference, Washington, DC, pp 1322–1327
19. Modares H, Lewis FL, Naghibi-Sistani MB (2014) Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica* 50(1):193–202
20. Kamalapurkar R, Dinh H, Bhasin S, Dixon WE (2015) Approximate optimal trajectory tracking for continuous-time nonlinear systems. *Automatica* 51:40–48
21. Khalil HK (2002) Nonlinear Systems, 3rd edn. Prentice Hall, Upper Saddle River, NJ
22. Chowdhary G, Yucelen T, Mühlegg M, Johnson EN (2013) Concurrent learning adaptive control of linear systems with exponentially convergent bounds. *Int J Adapt Control Signal Process* 27(4):280–301
23. Savitzky A, Golay MJE (1964) Smoothing and differentiation of data by simplified least squares procedures. *Anal Chem* 36(8):1627–1639
24. Krstic M, Li ZH (1998) Inverse optimal design of input-to-state stabilizing nonlinear controllers. *IEEE Trans Autom Control* 43(3):336–350
25. Mombaur K, Truong A, Laumond JP (2010) From human to humanoid locomotion - an inverse optimal control approach. *Auton Robots* 28(3):369–383

26. Ratliff ND, Bagnell JA, Zinkevich MA (2006) Maximum margin planning. In: Proceedings of the international conference on learning
27. Pang Z, Liu G (2012) Design and implementation of secure networked predictive control systems under deception attacks. *IEEE Trans Control Syst Technol* 20(5):1334–1342
28. Clark A, Zhu Q, Poovendran R, Başar T (2013) An impact-aware defense against stuxnet. In: Proceedings of the American control conference, pp 4146–4153
29. Kamalapurkar R, Walters P, Dixon WE (2013) Concurrent learning-based approximate optimal regulation. In: Proceedings of the IEEE conference on decision and control, Florence, IT, pp 6256–6261
30. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92
31. Vamvoudakis KG, Lewis FL (2009) Online synchronous policy iteration method for optimal control. In: Yu W (ed) Recent advances in intelligent control systems, Springer, pp 357–374
32. Ioannou P, Sun J (1996) Robust adaptive control. Prentice Hall
33. Johnson M, Hiramatsu T, Fitz-Coy N, Dixon WE (2010) Asymptotic stackelberg optimal control design for an uncertain Euler-Lagrange system. In: Proceedings of the IEEE conference on decision and control, Atlanta, GA, pp 6686–6691
34. Vamvoudakis KG, Lewis FL (2010) Online neural network solution of nonlinear two-player zero-sum games using synchronous policy iteration. In: Proceedings of the IEEE conference on decision and control
35. Vrabie D, Lewis FL (2010) Integral reinforcement learning for online computation of feedback nash strategies of nonzero-sum differential games. In: Proceedings of the IEEE conference on decision and control, pp 3066–3071
36. Lewis M, Tan K (1997) High precision formation control of mobile robots using virtual structures. *Auton Robots* 4(4):387–403
37. Balch T, Arkin R (1998) Behavior-based formation control for multirobot teams. *IEEE Trans Robot Autom* 14(6):926–939
38. Das A, Fierro R, Kumar V, Ostrowski J, Spletzer J, Taylor C (2002) A vision-based formation control framework. *IEEE Trans Robot Autom* 18(5):813–825
39. Fax J, Murray R (2004) Information flow and cooperative control of vehicle formations. *IEEE Trans Autom Control* 49(9):1465–1476
40. Murray R (2007) Recent research in cooperative control of multivehicle systems. *J Dyn Syst Meas Control* 129:571–583
41. Wang J, Xin M (2010) Multi-agent consensus algorithm with obstacle avoidance via optimal control approach. *Int J Control* 83(12):2606–2621
42. Wang J, Xin M (2012) Distributed optimal cooperative tracking control of multiple autonomous robots. *Robot Auton Syst* 60(4):572–583
43. Wang J, Xin M (2013) Integrated optimal formation control of multiple unmanned aerial vehicles. *IEEE Trans Control Syst Technol* 21(5):1731–1744
44. Lin W (2014) Distributed uav formation control using differential game approach. *Aerosp Sci Technol* 35:54–62
45. Semsar-Kazerooni E, Khorasani K (2008) Optimal consensus algorithms for cooperative team of agents subject to partial information. *Automatica* 44(11):2766–2777
46. Shim DH, Kim HJ, Sastry S (2003) Decentralized nonlinear model predictive control of multiple flying robots. *Proceedings of the IEEE conference on decision and control* 4:3621–3626
47. Magni L, Scattolini R (2006) Stabilizing decentralized model predictive control of nonlinear systems. *Automatica* 42(7):1231–1236
48. Heydari A, Balakrishnan SN (2012) An optimal tracking approach to formation control of nonlinear multi-agent systems. In: Proceedings of AIAA guidance, navigation and control conference
49. Zhang H, Zhang J, Yang GH, Luo Y (2015) Leader-based optimal coordination control for the consensus problem of multiagent differential games via fuzzy adaptive dynamic programming. *IEEE Trans Fuzzy Syst* 23(1):152–163

50. Sundaram S, Revzen S, Pappas G (2012) A control-theoretic approach to disseminating values and overcoming malicious links in wireless networks. *Automatica* 48(11):2894–2901
51. Abbas W, Egerstedt M (2012) Securing multiagent systems against a sequence of intruder attacks. In: Proceedings of the American control conference, pp 4161–4166
52. Fawzi H, Tabuada P, Diggavi S (2012) Security for control systems under sensor and actuator attacks. In: Proceedings of the IEEE conference on decision and control, pp 3412–3417
53. Jung D, Selmic RR (2008) Power leader fault detection in nonlinear leader-follower networks. In: Proceedings of the IEEE conference on decision and control, pp 404–409
54. Zhang X (2010) Decentralized fault detection for a class of large-scale nonlinear uncertain systems. In: Proceedings of the American control conference, pp 5650–5655
55. Li X, Zhou K (2009) A time domain approach to robust fault detection of linear time-varying systems. *Automatica* 45(1):94–102
56. Potula K, Selmic RR, Polycarpou MM (2010) Dynamic leader-followers network model of human emotions and their fault detection. In: Proceedings of the IEEE conference on decision and control, pp 744–749
57. Ferdowsi H, Raja DL, Jagannathan S (2012) A decentralized fault prognosis scheme for nonlinear interconnected discrete-time systems. In: Proceedings of the American control conference, pp 5900–5905
58. Luo X, Dong M, Huang Y (2006) On distributed fault-tolerant detection in wireless sensor networks. *IEEE Trans Comput* 55(1):58–70
59. Fernández-Bes J, Cid-Sueiro J (2012) Decentralized detection with energy-aware greedy selective sensors. In: International workshop on cognitive information process, pp 1–6
60. Jiao Q, Modares H, Xu S, Lewis FL, Vamvoudakis KG (2016) Multi-agent zero-sum differential graphical games for disturbance rejection in distributed control. *Automatica* 69:24–34
61. Li J, Modares H, Chai T, Lewis FL, Xie L (2017) Off-policy reinforcement learning for synchronization in multiagent graphical games. *IEEE Trans Neural Netw Learn Syst* 28(10):2434–2445
62. Vamvoudakis KG (2017) Q-learning for continuous-time graphical games on large networks with completely unknown linear system dynamics. *Int J Robust Nonlinear Control* 27(16):2900–2920
63. Vamvoudakis KG, Modares H, Kiumarsi B, Lewis FL (2017) Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online. *IEEE Control Syst* 37(1):33–52

# Chapter 6

## Applications



### 6.1 Introduction

This chapter is dedicated to applications of model-based reinforcement learning to closed-loop control of autonomous ground and marine vehicles. Marine craft, which include ships, floating platforms, autonomous underwater vehicles, etc, play a vital role in commercial, military, and recreational objectives. Marine craft are often required to remain on a station for an extended period of time, e.g., floating oil platforms, support vessels, and autonomous underwater vehicles acting as a communication link for multiple vehicles or persistent environmental monitors. The success of the vehicle often relies on the vehicle's ability to hold a precise station (e.g., station keeping near structures or underwater features). The cost of holding that station is correlated to the energy expended for propulsion through consumption of fuel and wear on mechanical systems, especially when station keeping in environments with a persistent current. Therefore, by reducing the energy expended for station keeping objectives, the cost of holding a station can be reduced.

In this chapter, an optimal station keeping policy that captures the desire to balance the need to accurately hold a station and the cost of holding that station through a quadratic performance criterion is generated for a fully actuated marine craft (see also, [1]). The developed controller differs from results such as [2, 3] in that it tackles the challenges associated with the introduction of a time-varying irrotational current. Since the hydrodynamic parameters of a marine craft are often difficult to determine, a concurrent learning system identifier is developed. As outlined in [4], concurrent learning uses additional information from recorded data to remove the persistence of excitation requirement associated with traditional system identifiers. Due to a unique structure, the proposed model-based approximate dynamic programming method generates the optimal station keeping policy using a combination of on-policy and off-policy data, eliminating the need for physical exploration of the state-space. A Lyapunov-based stability analysis is presented which guarantees uniformly ultimately bounded convergence of the marine craft to its station

and uniformly ultimately bounded convergence of the approximated policy to the optimal policy. The developed strategy is validated for planar motion of an autonomous underwater vehicle. The experiments are conducted in a second-magnitude spring located in central Florida.

Advances in sensing and computational capabilities have enabled autonomous ground vehicles to become vital assets across multiple disciplines. This surge of interest over the last few decades has drawn considerable attention to motion control of mobile robots. As the technology matures, there is a desire to improve the performance (e.g., minimum control effort, time, distance) of mobile robots to better achieve their objectives.

Motivated by the desire for optimal path-following, an approximate dynamic programming-based controller is developed for a unicycle-type mobile robot where the optimal policy is parameterized by a neural network (see also, [5]). By simultaneously identifying and utilizing the feedback policy, the approximate dynamic programming-based controller does not need offline training for new desired paths or performance criteria. Path-following is achieved by tracking a virtual target placed on the desired path. The motion of the virtual target is described by a predefined state-dependent ordinary differential equation (cf. [6–8]). The state associated with the virtual target’s location along the path is unbounded due to the infinite time horizon of the guidance law, which presents several challenges related to the use of a neural network. In addition, the vehicle requires a constant control effort to remain on the path; therefore, any policy that results in path-following also results in infinite cost, rendering the associated control problem ill-defined.

In this chapter, the motion of the virtual target is redefined to facilitate the use of the neural network, and a modified control input is developed to render feasible optimal policies. The cost function is formulated in terms of the modified control and redefined virtual target motion, a unique challenge not addressed in previous approximate dynamic programming literature. A Lyapunov-based stability analysis is presented to establish uniformly ultimately bounded convergence of the approximate policy to the optimal policy and the vehicle state to the path while maintaining a desired speed profile. Simulation results compare the policy obtained using the developed technique to an offline numerical optimal solution. These results demonstrate that the controller approximates the optimal solution with similar accuracy as an offline numerical approach. Experimental results on a two-wheel differential drive mobile robot demonstrate the ability of the controller to learn the approximate optimal policy in real-time.

## 6.2 Station-Keeping of a Marine Craft

### 6.2.1 Vehicle Model

The nonlinear equations of motion for a marine craft, including the effects of irrotational ocean current, are given by [9]

$$\dot{\eta}(t) = J_E(\eta(t)) v(t), \quad (6.1)$$

$$\begin{aligned} \tau_b(t) &= M_{RB}\dot{v}(t) + C_{RB}(v(t))v(t) + M_A\dot{v}_r(t) + C_A(v_r(t))v_r(t) \\ &\quad + D_A(v_r(t))v_r(t) + G(\eta(t)), \end{aligned} \quad (6.2)$$

where  $v : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the body-fixed translational and angular velocity vector,  $v_c : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the body-fixed irrotational current velocity vector,  $v_r = v - v_c$  is the relative body-fixed translational and angular fluid velocity vector,  $\eta : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the earth-fixed position and orientation vector,  $J_E : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$  is the coordinate transformation between the body-fixed and earth-fixed coordinates, <sup>1</sup> $M_{RB} \in \mathbb{R}^{n \times n}$  is the constant rigid body inertia matrix,  $C_{RB} : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$  is the rigid body centripetal and Coriolis matrix,  $M_A \in \mathbb{R}^{n \times n}$  is the constant hydrodynamic added mass matrix,  $C_A : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$  is the unknown hydrodynamic centripetal and Coriolis matrix,  $D_A : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$  is the unknown hydrodynamic damping and friction matrix,  $G : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is the gravitational and buoyancy force and moment vector, and  $\tau_b : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the body-fixed force and moment control input.

For a three degree-of-freedom planar model with orientation represented as Euler angles, the state vectors in (6.1) and (6.2) are defined as

$$\eta \triangleq [x \ y \ \psi]^T,$$

$$v \triangleq [u \ v \ r]^T,$$

where  $x, y : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$ , are the earth-fixed position vector components of the center of mass,  $\psi : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  represents the yaw angle,  $u, v : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  are the body-fixed translational velocities, and  $r : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the body-fixed angular velocity. The irrotational current vector is defined as

$$v_c \triangleq [u_c \ v_c \ 0]^T,$$

where  $u_c, v_c : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  are the body-fixed current translational velocities. The coordinate transformation  $J_E$  is given as

$$J_E(\eta) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

**Assumption 6.1** The marine craft is neutrally buoyant if submerged and the center of gravity is located vertically below the center of buoyancy on the  $z$  axis if the vehicle model includes roll and pitch.

---

<sup>1</sup>The orientation of the vehicle may be represented as Euler angles, quaternions, or angular rates. In this development, the use of Euler angles is assumed, see Sect. 7.5 in [9] for details regarding other representations.

Assumption 6.1 simplifies the subsequent analysis and can often be met by trimming the vehicle. For marine craft where this assumption cannot be met, an additional term may be added to the controller, similar to how terms dependent on the irrotational current are handled.

### 6.2.2 System Identifier

Since the hydrodynamic effects pertaining to a specific marine craft may be unknown, an online system identifier is developed for the vehicle drift dynamics. Consider the control-affine form of the vehicle model,

$$\dot{\zeta}(t) = Y(\zeta(t), v_c(t))\theta + f_0(\zeta(t), \dot{v}_c(t)) + g\tau_b(t), \quad (6.3)$$

where  $\zeta \triangleq [\eta \ v]^T : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^{2n}$  is the state vector. The unknown hydrodynamics are linear-in-the-parameters with  $p$  unknown parameters where  $Y : \mathbb{R}^{2n} \times \mathbb{R}^n \rightarrow \mathbb{R}^{2n \times p}$  is the regression matrix and  $\theta \in \mathbb{R}^p$  is the vector of unknown parameters. The unknown hydrodynamic effects are modeled as

$$Y(\zeta, v_c)\theta = \begin{bmatrix} 0 \\ -M^{-1}C_A(v_r)v_r - M^{-1}D_A(v_r)v_r \end{bmatrix},$$

and known rigid body drift dynamics  $f_0 : \mathbb{R}^{2n} \times \mathbb{R}^n \rightarrow \mathbb{R}^{2n}$  are modeled as

$$f_0(\zeta, \dot{v}_c) = \begin{bmatrix} J_E(\eta)v \\ M^{-1}M_A\dot{v}_c - M^{-1}C_{RB}(v)v - M^{-1}G(\eta) \end{bmatrix},$$

where  $M \triangleq M_{RB} + M_A$  and the body-fixed current velocity  $v_c$  and acceleration  $\dot{v}_c$  are assumed to be measurable. The body-fixed current velocity  $v_c$  may be trivially measured using sensors commonly found on marine craft, such as a Doppler velocity log, while the current acceleration  $\dot{v}_c$  may be determined using numerical differentiation and smoothing. The known constant control effectiveness matrix  $g \in \mathbb{R}^{2n \times n}$  is defined as

$$g \triangleq \begin{bmatrix} 0 \\ M^{-1} \end{bmatrix}.$$

An identifier is designed as

$$\dot{\hat{\zeta}}(t) = Y(\zeta(t), v_c(t))\hat{\theta}(t) + f_0(\zeta(t), \dot{v}_c(t)) + g\tau_b(t) + k_\zeta\tilde{\zeta}(t), \quad (6.4)$$

where  $\tilde{\zeta} \triangleq \zeta - \hat{\zeta}$  is the measurable state estimation error, and  $k_\zeta \in \mathbb{R}^{2n \times 2n}$  is a constant positive definite, diagonal gain matrix. Subtracting (6.4) from (6.3) yields

$$\dot{\tilde{\zeta}}(t) = Y(\zeta(t), v_c(t)) \tilde{\theta}(t) - k_\zeta \tilde{\zeta}(t),$$

where  $\tilde{\theta} \triangleq \theta - \hat{\theta}$  is the parameter identification error.

Traditional adaptive control techniques require persistence of excitation to ensure the parameter estimates  $\hat{\theta}$  converge to their true values  $\theta$  (cf. [10, 11]). Persistence of excitation often requires an excitation signal to be applied to the vehicle's input resulting in unwanted deviations in the vehicle state. These deviations are often in opposition to the vehicle's control objectives. Alternatively, a concurrent learning-based system identifier can be developed (cf. [12, 13]). The concurrent learning-based system identifier relaxes the persistence of excitation requirement through the use of a prerecorded history stack of state-action pairs.

**Assumption 6.2** There exists a prerecorded data set of sampled data points  $\{\zeta_j, v_{cj}, \dot{v}_{cj}, \tau_{bj} \in \chi \mid j = 1, 2, \dots, M\}$  with numerically calculated state derivatives  $\dot{\zeta}_j$  at each recorded state-action pair such that  $\forall t \in [0, \infty)$ ,

$$\begin{aligned} \text{rank} \left( \sum_{j=1}^M Y_j^T Y_j \right) &= p, \\ \left\| \dot{\tilde{\zeta}}_j - \dot{\zeta}_j \right\| &< \bar{d}, \forall j, \end{aligned} \quad (6.5)$$

where  $Y_j \triangleq Y(\zeta_j, v_{cj})$ ,  $f_{0j} \triangleq f_0(\zeta_j)$ ,  $\dot{\zeta}_j = Y_j \theta + f_{0j} + g \tau_{bj}$ , and  $\bar{d} \in [0, \infty)$  is a constant.

In this development, it is assumed that a data set of state-action pairs is available a priori. Experiments to collect state-action pairs do not necessarily need to be conducted in the presence of a current (e.g., the data may be collected in a pool). Since the current affects the dynamics only through the  $v_r$  terms, data that is sufficiently rich and satisfies Assumption 6.2 may be collected by merely exploring the  $\zeta$  state-space. Note, this is the reason the body-fixed current  $v_c$  and acceleration  $\dot{v}_c$  are not considered a part of the state. If state-action data is not available for the given system then it is possible to build the history stack in real-time (see Appendix A.2.3).

The parameter estimate update law is

$$\dot{\hat{\theta}}(t) = \Gamma_\theta Y(\zeta(t), v_c(t))^T \tilde{\zeta}(t) + \Gamma_\theta k_\theta \sum_{j=1}^M Y_j^T (\dot{\tilde{\zeta}}_j - f_{0j} - g \tau_{bj} - Y_j \hat{\theta}(t)), \quad (6.6)$$

where  $\Gamma_\theta$  is a positive definite diagonal gain matrix, and  $k_\theta$  is a positive scalar gain matrix. To facilitate the stability analysis, the parameter estimate update law is expressed in the advantageous form

$$\dot{\hat{\theta}}(t) = \Gamma_\theta Y(\zeta(t), v_c(t))^T \tilde{\zeta}(t) + \Gamma_\theta k_\theta \sum_{j=1}^M Y_j^T (Y_j \tilde{\theta} + d_j),$$

where  $d_j = \dot{\tilde{\zeta}}_j - \dot{\tilde{\zeta}}_j$ .

To analyze the developed identifier, consider the candidate Lyapunov function  $V_P : \mathbb{R}^{2n+p} \rightarrow [0, \infty)$  defined as

$$V_P(Z_P) \triangleq \frac{1}{2} \tilde{\zeta}^T \tilde{\zeta} + \frac{1}{2} \tilde{\theta}^T \Gamma_\theta^{-1} \tilde{\theta}, \quad (6.7)$$

where  $Z_P \triangleq [\tilde{\zeta}^T \ \tilde{\theta}^T]$ . The candidate Lyapunov function can be bounded as

$$\frac{1}{2} \min \left\{ 1, \underline{\gamma}_\theta \right\} \|Z_P\|^2 \leq V_P(Z_P) \leq \frac{1}{2} \max \{ 1, \overline{\gamma}_\theta \} \|Z_P\|^2 \quad (6.8)$$

where  $\underline{\gamma}_\theta, \overline{\gamma}_\theta$  are the minimum and maximum eigenvalues of  $\Gamma_\theta$ , respectively.

The time derivative of the candidate Lyapunov function in (6.7) is

$$\dot{V}_P(t) = -\tilde{\zeta}^T(t) k_\zeta \tilde{\zeta}(t) - k_\theta \tilde{\theta}^T(t) \sum_{j=1}^M Y_j^T Y_j \tilde{\theta}(t) - k_\theta \tilde{\theta}^T(t) \sum_{j=1}^M Y_j^T d_j.$$

The time derivative can be upper bounded as

$$\dot{V}_P(t) \leq -\underline{k}_\zeta \|\tilde{\zeta}(t)\|^2 - k_\theta \underline{y} \|\tilde{\theta}(t)\|^2 + k_\theta d_\theta \|\tilde{\theta}(t)\|, \quad (6.9)$$

where  $\underline{k}_\zeta, \underline{y}$  are the minimum eigenvalues of  $k_\zeta$  and  $\sum_{j=1}^M Y_j^T Y_j$ , respectively, and  $d_\theta = \bar{d} \sum_{j=1}^M \|Y_j\|$ . After completing the squares, (6.9) can be upper bounded as

$$\dot{V}_P(t) \leq -\underline{k}_\zeta \|\tilde{\zeta}(t)\|^2 - \frac{k_\theta \underline{y}}{2} \|\tilde{\theta}(t)\|^2 + \frac{k_\theta d_\theta^2}{2\underline{y}},$$

which may be further upper bounded as

$$\dot{V}_P(t) \leq -\alpha_P \|Z_P(t)\|^2, \quad \forall \|Z_P(t)\| \geq K_P > 0, \quad (6.10)$$

where  $\alpha_P \triangleq \frac{1}{2} \min \left\{ 2\underline{k}_\zeta, k_\theta \underline{y} \right\}$  and  $K_P \triangleq \sqrt{\frac{k_\theta d_\theta^2}{2\alpha_P \underline{y}}}$ . Using (6.8) and (6.10),  $\tilde{\zeta}$  and  $\tilde{\theta}$  can be shown to exponentially decay to a ultimate bound as  $t \rightarrow \infty$ .

### 6.2.3 Problem Formulation

The presence of a time-varying irrotational current yields unique challenges in the formulation of the optimal regulation problem. Since the current renders the system non-autonomous, a residual model that does not include the effects of the irrotational

current is introduced. The residual model is used in the development of the optimal control problem in place of the original model. A disadvantage of this approach is that the optimal policy is developed for the current-free model. In the case where the earth-fixed current is constant, the effects of the current may be included in the development of the optimal control problem as detailed in Appendix A.3.2.

The dynamic constraints can be written in a control-affine form as

$$\dot{\zeta}(t) = Y_{res}(\zeta(t))\theta + f_{0_{res}}(\zeta(t)) + gu(t), \quad (6.11)$$

where the unknown hydrodynamics are linear-in-the-parameters with  $p$  unknown parameters where  $Y_{res} : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n \times p}$  is a regression matrix, the function  $f_{0_{res}} : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n}$  is the known portion of the dynamics, and  $u : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n$  is the control vector. The drift dynamics, defined as  $f_{res}(\zeta) \triangleq Y_{res}(\zeta)\theta + f_{0_{res}}(\zeta)$ , satisfies  $f_{res}(0) = 0$  when Assumption 6.1 is satisfied.

The drift dynamics in (6.11) are modeled as

$$\begin{aligned} Y_{res}(\zeta)\theta &= \begin{bmatrix} 0 \\ -M^{-1}C_A(v)v - M^{-1}D(v)v \end{bmatrix}, \\ f_{0_{res}}(\zeta) &= \begin{bmatrix} J_E v \\ -M^{-1}C_{RB}(v)v - M^{-1}G(\eta) \end{bmatrix}, \end{aligned} \quad (6.12)$$

and the virtual control vector  $u$  is defined as

$$u = \tau_b - \tau_c(\zeta, v_c, \dot{v}_c), \quad (6.13)$$

where  $\tau_c : \mathbb{R}^{2n} \times \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$  is a feedforward term to compensate for the effect of the variable current, which includes cross-terms generated by the introduction of the residual dynamics and is given as

$$\tau_c(\zeta, v_c, \dot{v}_c) = C_A(v_r)v_r + D(v_r)v_r - M_A\dot{v}_c - C_A(v)v - D(v)v.$$

The current feedforward term is represented in the advantageous form

$$\tau_c(\zeta, v_c, \dot{v}_c) = -M_A\dot{v}_c + Y_c(\zeta, v_c)\theta,$$

where  $Y_c : \mathbb{R}^{2n} \times \mathbb{R}^n \rightarrow \mathbb{R}^{2n \times p}$  is the regression matrix and

$$Y_c\theta(\zeta, v_c) = C_A(v_r)v_r + D(v_r)v_r - C_A(v)v - D(v)v.$$

Since the parameters are unknown, an approximation of the compensation term  $\tau_c$  given by

$$\hat{\tau}_c \left( \zeta, v_c, \dot{v}_c, \hat{\theta} \right) = -M_A \dot{v}_c + Y_c \hat{\theta} \quad (6.14)$$

is implemented, and the approximation error is defined by

$$\tilde{\tau}_c \triangleq \tau_c - \hat{\tau}_c.$$

The performance index for the optimal regulation problem is selected as

$$J(t_0, \zeta_0, u(\cdot)) = \int_{t_0}^{\infty} r(\zeta(\tau; t_0, \zeta_0, u(\cdot)), u(\tau)) d\tau, \quad (6.15)$$

where  $\zeta(\tau; t_0, \zeta_0, u(\cdot))$  denotes a solution to (6.11), evaluated at  $t = \tau$ , under the controller  $u(\cdot)$ , with the initial condition  $\zeta_0 \triangleq \zeta(t_0)$ , and  $r : \mathbb{R}^{2n} \rightarrow [0, \infty)$  is the local cost defined as

$$r(\zeta, u) \triangleq \zeta^T Q \zeta + u^T R u. \quad (6.16)$$

In (6.16),  $Q \in \mathbb{R}^{2n \times 2n}$  and  $R \in \mathbb{R}^{n \times n}$  are symmetric positive definite weighting matrices. The matrix  $Q$  has the property  $\underline{q} \|\xi_q\|^2 \leq \xi_q^T Q \xi_q \leq \bar{q} \|\xi_q\|^2$ ,  $\forall \xi_q \in \mathbb{R}^{2n}$  where  $\underline{q}$  and  $\bar{q}$  are positive constants. Assuming existence of the optimal controller, the infinite-time scalar value function  $V : \mathbb{R}^{2n} \rightarrow [0, \infty)$  for the optimal solution is written as

$$V^*(\zeta) = \min_{u_{[t, \infty]}} \int_t^{\infty} r(\zeta(\tau; t, \zeta, u(\cdot)), u(\tau)) d\tau, \quad (6.17)$$

where the minimization is performed over the set of admissible controllers.

The objective of the optimal control problem is to find the optimal policy  $u^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}^n$  that minimizes the performance index (6.15) subject to the dynamic constraints in (6.11). Assuming that a minimizing policy exists and the value function is continuously differentiable, the Hamilton–Jacobi–Bellman equation is given as [14]

$$0 = \nabla_t V^*(\zeta) + r(\zeta, u^*(\zeta)) + \nabla_{\zeta} V^*(\zeta) (Y_{res}(\zeta) \theta + f_{0res}(\zeta) + g u^*(\zeta)), \quad (6.18)$$

where  $\frac{\partial V(\zeta)}{\partial t} = 0$  since the value function is not an explicit function of time. After substituting (6.16) into (6.18), the optimal policy is given by [14]

$$u^*(\zeta) = -\frac{1}{2} R^{-1} g^T (\nabla_{\zeta} V^*(\zeta))^T. \quad (6.19)$$

### 6.2.4 Approximate Policy

The subsequent development is based on a neural network approximation of the value function and optimal policy. Over any compact domain  $\chi \subset \mathbb{R}^{2n}$ , the optimal value function  $V^* : \mathbb{R}^{2n} \rightarrow [0, \infty)$  can be represented by a single-layer neural network with  $l$  neurons as

$$V^*(\zeta) = W^T \sigma(\zeta) + \epsilon(\zeta), \quad (6.20)$$

where  $W \in \mathbb{R}^l$  is the ideal weight vector bounded above by a known positive constant,  $\sigma : \mathbb{R}^{2n} \rightarrow \mathbb{R}^l$  is a bounded continuously differentiable activation function, and  $\epsilon : \mathbb{R}^{2n} \rightarrow \mathbb{R}$  is the bounded continuously differentiable function reconstruction error.

Using (6.19) and (6.20), the optimal policy can be expressed as

$$u^*(\zeta) = -\frac{1}{2} R^{-1} g^T (\nabla_\zeta \sigma^T(\zeta) W + \nabla_\zeta \epsilon^T(\zeta)). \quad (6.21)$$

Based on (6.20) and (6.21), neural network approximations of the value function and the optimal policy are defined as

$$\hat{V}(\zeta, \hat{W}_c) = \hat{W}_c^T \sigma(\zeta), \quad (6.22)$$

$$\hat{u}(\zeta, \hat{W}_a) = -\frac{1}{2} R^{-1} g^T \nabla_\zeta \sigma^T(\zeta) \hat{W}_a, \quad (6.23)$$

where  $\hat{W}_c, \hat{W}_a : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^l$  are estimates of the constant ideal weight vector  $W$ . The weight estimation errors are defined as  $\tilde{W}_c \triangleq W - \hat{W}_c$  and  $\tilde{W}_a \triangleq W - \hat{W}_a$ .

Substituting (6.11), (6.22), and (6.23) into (6.18), the Bellman error,  $\hat{\delta} : \mathbb{R}^{2n} \times \mathbb{R}^p \times \mathbb{R}^l \times \mathbb{R}^l \rightarrow \mathbb{R}$ , given as

$$\hat{\delta}(\zeta, \hat{\theta}, \hat{W}_c, \hat{W}_a) = r(\zeta, \hat{u}(\zeta, \hat{W}_a)) + \nabla_\zeta \hat{V}(\zeta, \hat{W}_c) (Y_{res}(\zeta) \hat{\theta} + f_{0_{res}}(\zeta) + g \hat{u}(\zeta, \hat{W}_a)) \quad (6.24)$$

The Bellman error, evaluated along the system trajectories, can be expressed as

$$\hat{\delta}_t(t) \triangleq \delta(\zeta(t), \hat{\theta}(t), \hat{W}_c(t), \hat{W}_a(t)) = r(\zeta(t), \hat{u}(\zeta(t), \hat{W}_a(t))) + \hat{W}_c^T(t) \omega(t),$$

where  $\omega : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^l$  is given by

$$\omega(t) = \nabla_\zeta \sigma(\zeta(t)) (Y_{res}(\zeta(t)) \hat{\theta}(t) + f_{0_{res}}(\zeta(t)) + g \hat{u}(\zeta(t), \hat{W}_a(t))).$$

The Bellman error may be extrapolated to unexplored regions of the state-space since it depends solely on the approximated system model and the neural network weight estimates. In Sect. 6.2.5, Bellman error extrapolation is employed to establish uniformly ultimately bounded convergence of the approximate policy to the optimal policy without requiring persistence of excitation provided the following assumption is satisfied.

**Assumption 6.3** There exists a positive constant  $\underline{c}$  and set of states  $\{\zeta_k \in \chi | k = 1, 2, \dots, N\}$  such that

$$\inf_{t \in [0, \infty)} \left[ \lambda_{\min} \left\{ \sum_{k=1}^N \frac{\omega_k \omega_k^T}{\rho_k} \right\} \right] = \underline{c}, \quad (6.25)$$

where  $\omega_k(t) \triangleq \nabla_\zeta \sigma(\zeta_k) \left( Y_{res}(\zeta_k) \hat{\theta}(t) + f_{0_{res}}(\zeta_k) + g \hat{u}(\zeta_k, \hat{W}_a(t)) \right)$  and  $\rho_k \triangleq 1 + k_\rho \omega_k^T \Gamma \omega_k$ .

In general, the condition in (6.25) cannot be guaranteed to hold a priori; however, heuristically, the condition can be met by sampling redundant data (i.e.,  $N \gg l$ ).

The value function least-squares update law based on minimization of the Bellman error is given by

$$\dot{\hat{W}}_c(t) = -\Gamma(t) \left( k_{c1} \frac{\omega(t)}{\rho(t)} \hat{\delta}_t(t) + \frac{k_{c2}}{N} \sum_{k=1}^N \frac{\omega_k(t)}{\rho_k(t)} \hat{\delta}_{tk}(t) \right), \quad (6.26)$$

$$\dot{\Gamma}(t) = \begin{cases} \beta \Gamma(t) - k_{c1} \Gamma(t) \frac{\omega(t) \omega(t)^T}{\rho(t)} \Gamma, & \|\Gamma(t)\| \leq \bar{\Gamma}, \\ 0 & \text{otherwise} \end{cases}, \quad (6.27)$$

where  $k_{c1}, k_{c2} \in \mathbb{R}$  are positive adaptation gains,  $\hat{\delta}_{tk}(t) \triangleq \hat{\delta}(\zeta_k, \hat{\theta}(t), \hat{W}_c(t), \hat{W}_a(t))$  is the extrapolated approximate Bellman error,  $\|\Gamma(t_0)\| = \|\Gamma_0\| \leq \bar{\Gamma}$  is the initial adaptation gain,  $\bar{\Gamma} \in \mathbb{R}$  is a positive saturation gain,  $\beta \in \mathbb{R}$  is a positive forgetting factor, and  $\rho \triangleq 1 + k_\rho \omega^T \Gamma \omega$  is a normalization constant, where  $k_\rho \in \mathbb{R}$  is a positive gain. The update law in (6.26) and (6.27) ensures that

$$\underline{\Gamma} \leq \|\Gamma(t)\| \leq \bar{\Gamma}, \quad \forall t \in [0, \infty).$$

The actor neural network update law is given by

$$\dot{\hat{W}}_a(t) = \text{proj} \left\{ -k_a \left( \hat{W}_a(t) - \hat{W}_c(t) \right) \right\}, \quad (6.28)$$

where  $k_a \in \mathbb{R}$  is a positive gain, and  $\text{proj}\{\cdot\}$  is a smooth projection operator used to bound the weight estimates. Using properties of the projection operator, the actor

neural network weight estimation error can be bounded above by positive constant. See Sect. 4.4 in [11] or Remark 3.6 in [15] for details of smooth projection operators.

Using the definition in (6.13), the force and moment applied to the vehicle, described in (6.3), is given in terms of the approximated optimal virtual control (6.23) and the approximate compensation term in (6.14) as

$$\hat{\tau}_b(t) = \hat{u}\left(\zeta(t), \hat{W}_a(t)\right) + \hat{\tau}_c\left(\zeta(t), \hat{\theta}(t), v_c(t), \dot{v}_c(t)\right). \quad (6.29)$$

### 6.2.5 Stability Analysis

An unmeasurable form of the Bellman error can be written using (6.18) and (6.24) as

$$\begin{aligned} \hat{\delta}_t = & -\tilde{W}_c^T \omega - W^T \nabla_{\zeta} \sigma Y_{res} \tilde{\theta} - \nabla_{\zeta} \epsilon (Y_{res} \theta + f_{0_{res}}) + \frac{1}{4} \tilde{W}_a^T G_{\sigma} \tilde{W}_a + \frac{1}{2} \nabla_{\zeta} \epsilon G \nabla_{\zeta} \sigma^T W \\ & + \frac{1}{4} \nabla_{\zeta} \epsilon G \nabla_{\zeta} \epsilon, \end{aligned} \quad (6.30)$$

where  $G \triangleq g R^{-1} g^T \in \mathbb{R}^{2n \times 2n}$  and  $G_{\sigma} \triangleq \nabla_{\zeta} \sigma G \nabla_{\zeta} \sigma^T \in \mathbb{R}^{l \times l}$  are symmetric, positive semi-definite matrices. Similarly, the Bellman error at the sampled data points can be written as

$$\hat{\delta}_{tk} = -\tilde{W}_c^T \omega_k - W^T \sigma'_k (Y_{res_k} \tilde{\theta}) + \frac{1}{4} \tilde{W}_a^T G_{\sigma k} \tilde{W}_a + E_k, \quad (6.31)$$

where

$$E_k \triangleq \frac{1}{2} \epsilon'_k G \sigma'^T W + \frac{1}{4} \epsilon'_k G \epsilon'^T - \epsilon'_k (Y_{res_k} \theta + f_{0_{res_k}}) \in \mathbb{R}$$

is a constant at each data point, and the notation  $F_k$  denotes the function  $F(\zeta, \cdot)$  evaluated at the sampled state (i.e.,  $F_k(\cdot) = F(\zeta_k, \cdot)$ ).

The functions  $Y_{res}$  and  $f_{0_{res}}$  are Lipschitz continuous on the compact set  $\chi$  and can be bounded by

$$\|Y_{res}(\zeta)\| \leq L_{Y_{res}} \|\zeta\|, \forall \zeta \in \chi,$$

$$\|f_{0_{res}}(\zeta)\| \leq L_{f_{0_{res}}} \|\zeta\|, \forall \zeta \in \chi,$$

respectively, where  $L_{Y_{res}}$  and  $L_{f_{0_{res}}}$  are positive constants.

To facilitate the subsequent stability analysis, consider the candidate Lyapunov function  $V_L : \mathbb{R}^{2n} \times \mathbb{R}^l \times \mathbb{R}^l \times \mathbb{R}^p \rightarrow [0, \infty)$  given by

$$V_L(Z) = V(\zeta) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a + V_P(Z_P),$$

where  $Z \triangleq [\zeta^T \tilde{W}_c^T \tilde{W}_a^T Z_P^T]^T \in \chi \cup \mathbb{R}^l \times \mathbb{R}^l \times \mathbb{R}^p$ . Since the value function  $V$  in (6.17) is positive definite,  $V_L$  can be bounded by

$$\underline{v}_L(\|Z\|) \leq V_L(Z) \leq \overline{v}_L(\|Z\|) \quad (6.32)$$

using [16, Lemma 4.3] and (6.8), where  $\underline{v}_L, \overline{v}_L : [0, \infty) \rightarrow [0, \infty)$  are class  $\mathcal{K}$  functions. Let  $\beta \subset \chi \cup \mathbb{R}^l \times \mathbb{R}^l \times \mathbb{R}^p$  be a compact set, and let  $\varphi_\zeta, \varphi_c, \varphi_a, \varphi_\theta, \kappa_c, \kappa_a, \kappa_\theta$ , and  $\kappa$  be constants as defined in Appendix A.3.1. When Assumptions 6.2 and 6.3, and the sufficient gain conditions in Appendix A.3.1 are satisfied, the constant  $K \in \mathbb{R}$  defined as

$$K \triangleq \sqrt{\frac{\kappa_c^2}{2\alpha\varphi_c} + \frac{\kappa_a^2}{2\alpha\varphi_a} + \frac{\kappa_\theta^2}{2\alpha\varphi_\theta} + \frac{\kappa}{\alpha}}$$

is positive, where  $\alpha \triangleq \frac{1}{2} \min \left\{ \varphi_\zeta, \varphi_c, \varphi_a, \varphi_\theta, 2\underline{k}_\zeta \right\}$ .

**Theorem 6.4** *If Assumptions 6.1–6.3, the sufficient conditions (A.33)–(A.35), and*

$$K < \underline{v}_L^{-1}(\overline{v}_L(r)) \quad (6.33)$$

*are satisfied, where  $r \in \mathbb{R}$  is the radius of the compact set  $\beta$ , then the policy in (6.23) with the neural network update laws in (6.26)–(6.28) guarantee uniformly ultimately bounded regulation of the state  $\zeta$  and uniformly ultimately bounded convergence of the approximated policies  $\hat{u}$  to the optimal policy  $u^*$ .*

*Proof* The time derivative of the candidate Lyapunov function is

$$\dot{V}_L = \frac{\partial V}{\partial \zeta} (Y\theta + f_0) + \frac{\partial V}{\partial \zeta} g(\hat{u} + \hat{\tau}_c) - \tilde{W}_c^T \Gamma^{-1} \dot{\hat{W}}_c - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \dot{\Gamma} \Gamma^{-1} \tilde{W}_c - \tilde{W}_a^T \dot{\hat{W}}_a + \dot{V}_P. \quad (6.34)$$

Using (6.18),  $\frac{\partial V}{\partial \zeta} (Y\theta + f_0) = -\frac{\partial V}{\partial \zeta} g(u^* + \tau_c) - r(\zeta, u^*)$ . Then,

$$\begin{aligned} \dot{V}_L = & \frac{\partial V}{\partial \zeta} g(\hat{u} + \hat{\tau}_c) - \frac{\partial V}{\partial \zeta} g(u^* + \tau_c) - r(\zeta, u^*) - \tilde{W}_c^T \Gamma^{-1} \dot{\hat{W}}_c - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \dot{\Gamma} \Gamma^{-1} \tilde{W}_c \\ & - \tilde{W}_a^T \dot{\hat{W}}_a + \dot{V}_P. \end{aligned}$$

Substituting (6.26) and (6.28) for  $\dot{\hat{W}}_c$  and  $\dot{\hat{W}}_a$ , respectively, yields

$$\begin{aligned}\dot{V}_L &= -\zeta^T Q \zeta - u^{*T} R u^* + \frac{\partial V}{\partial \zeta} g \tilde{\tau}_c + \frac{\partial V}{\partial \zeta} g \hat{u} - \frac{\partial V}{\partial \zeta} g u^* + \tilde{W}_c^T \left[ k_{c1} \frac{\omega}{\rho} \delta + \frac{k_{c2}}{N} \sum_{j=1}^N \frac{\omega_k}{\rho_k} \delta_k \right] \\ &\quad + \tilde{W}_a^T k_a \left( \hat{W}_a - \hat{W}_c \right) - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \left[ \left( \beta \Gamma - k_{c1} \Gamma \frac{\omega \omega^T}{\rho} \Gamma \right) \mathbf{1}_{\|\Gamma\| \leq \bar{\Gamma}} \right] \Gamma^{-1} \tilde{W}_c + \dot{V}_P.\end{aligned}$$

Using Young's inequality, (6.20), (6.21), (6.23), (6.30), and (6.31) the Lyapunov derivative can be upper bounded as

$$\begin{aligned}\dot{V}_L &\leq -\varphi_\zeta \|\zeta\|^2 - \varphi_c \|\tilde{W}_c\|^2 - \varphi_a \|\tilde{W}_a\|^2 - \varphi_\theta \|\tilde{\theta}\|^2 - \underline{k}_\zeta \|\tilde{\zeta}\|^2 + \kappa_a \|\tilde{W}_a\| + \kappa_c \|\tilde{W}_c\| \\ &\quad + \kappa_\theta \|\tilde{\theta}\| + \kappa.\end{aligned}$$

Completing the squares, the upper bound on the Lyapunov derivative may be written as

$$\begin{aligned}\dot{V}_L &\leq -\frac{\varphi_\zeta}{2} \|\zeta\|^2 - \frac{\varphi_c}{2} \|\tilde{W}_c\|^2 - \frac{\varphi_a}{2} \|\tilde{W}_a\|^2 - \frac{\varphi_\theta}{2} \|\tilde{\theta}\|^2 - \underline{k}_\zeta \|\tilde{\zeta}\|^2 + \frac{\kappa_c^2}{2\varphi_c} + \frac{\kappa_a^2}{2\varphi_a} \\ &\quad + \frac{\kappa_\theta^2}{2\varphi_\theta} + \kappa,\end{aligned}$$

which can be further upper bounded as

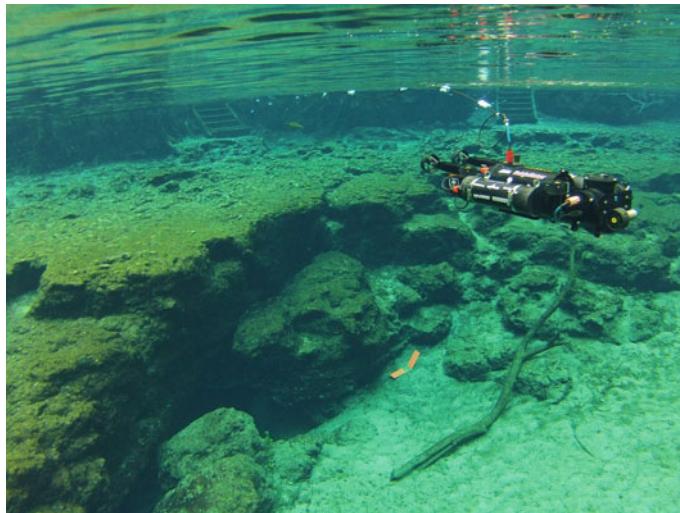
$$\dot{V}_L \leq -\alpha \|Z\|, \quad \forall \|Z\| \geq K > 0. \quad (6.35)$$

Using (6.32), (6.33), and (6.35), [16, Theorem 4.18] is invoked to conclude that  $Z$  is uniformly ultimately bounded, in the sense that  $\limsup_{t \rightarrow \infty} \|Z(t)\| \leq \underline{v}_L^{-1}(\bar{v}_L(K))$ .

Based on the definition of  $Z$  and the inequalities in (6.32) and (6.35),  $\zeta(\cdot)$ ,  $\tilde{W}_c(\cdot)$ ,  $\tilde{W}_a(\cdot) \in \mathcal{L}_\infty$ . From the definition of  $W$  and the neural network weight estimation errors,  $\hat{W}_c(\cdot)$ ,  $\hat{W}_a(\cdot) \in \mathcal{L}_\infty$ . Using the actor update laws,  $\dot{\hat{W}}_a(\cdot) \in \mathcal{L}_\infty$ . It follows that  $t \mapsto \hat{V}(x(t)) \in \mathcal{L}_\infty$  and  $t \mapsto \hat{u}(x(t), \hat{W}_a(t)) \in \mathcal{L}_\infty$ . From the dynamics in (6.12),  $\dot{\zeta}(\cdot) \in \mathcal{L}_\infty$ . By the definition in (6.24),  $\hat{\delta}_t(\cdot) \in \mathcal{L}_\infty$ . By the definition of the normalized critic update law,  $\dot{\hat{W}}_c(\cdot) \in \mathcal{L}_\infty$ .  $\square$

### 6.2.6 Experimental Validation

The performance of the controller is demonstrated with experiments conducted at Ginnie Springs in High Springs, FL. Ginnie Springs is a second-magnitude spring discharging 142 million liters of freshwater daily with a spring pool measuring 27.4 m in diameter and 3.7 m deep [17]. Ginnie Springs was selected because of its relatively high flow rate and clear waters for vehicle observation. For clarity of exposition and to remain within the vehicle's depth limitations, the proposed method is implemented



**Fig. 6.1** SubjuGator 7 autonomous underwater vehicle operating at Ginnie Springs, FL

on three degrees of freedom of an autonomous underwater vehicle (i.e., surge, sway, yaw). The number of basis functions and weights required to support a six degree-of-freedom model greatly increases from the set required for the three degree-of-freedom model. The vehicle's Doppler velocity log has a minimum height over bottom of approximately 3 m that is required to measure water velocity. A minimum depth of approximately 0.5 m is required to remove the vehicle from surface effects. With the depth of the spring nominally 3.7 m, a narrow window of about 20 cm is left operate the vehicle in heave.

Experiments were conducted on an autonomous underwater vehicle, SubjuGator 7, developed at the University of Florida. The autonomous underwater vehicle, shown in Fig. 6.1, is a small two man portable autonomous underwater vehicle with a mass of 40.8 kg. The vehicle is over-actuated with eight bidirectional thrusters.

Designed to be modular, the vehicle has multiple specialized pressure vessels that house computational capabilities, sensors, batteries, and mission specific payloads. The central pressure vessel houses the vehicle's motor controllers, network infrastructure, and core computing capability. The core computing capability services the vehicle's environmental sensors (e.g., visible light cameras, scanning sonar, etc.), the vehicle's high-level mission planning, and low-level command and control software. A standard small form factor computer makes up the computing capability and utilizes a 2.13 GHz server grade quad-core processor. Located near the front of the vehicle, the navigation vessel houses the vehicle's basic navigation sensors. The suite of navigation sensors include an inertial measurement unit, a Doppler velocity log, a depth sensor, and a digital compass. The navigation vessel also includes an embedded 720 MHz processor for preprocessing and packaging navigation data. Along the

sides of the central pressure vessel, two vessels house 44 Ah of batteries used for propulsion and electronics.

The vehicle's software runs within the Robot Operating System framework in the central pressure vessel. For the experiment, three main software nodes were used: navigation, control, and thruster mapping nodes. The navigation node receives packaged navigation data from the navigation pressure vessel where an unscented Kalman filter estimates the vehicle's full state at 50 Hz. The desired force and moment produced by the controller are mapped to the eight thrusters using a least-squares minimization algorithm. The controller node contains the proposed controller and system identifier.

The implementation of the proposed method may have three parts: system identification, value function iteration, and control iteration. Implementing the system identifier requires (6.4), (6.6), and the data set alluded to in Assumption 6.2. The data set in Assumption 6.2 was collected in a swimming pool. The vehicle was commanded to track a data-rich trajectory with a RISE controller [18] while the state-action pairs were recorded. The recorded data was trimmed to a subset of 40 sampled points that were selected to maximize the minimum singular value of  $[Y_1 \ Y_2 \ \dots \ Y_j]$  as in Appendix A.2.3. The system identifier is updated at 50 Hz.

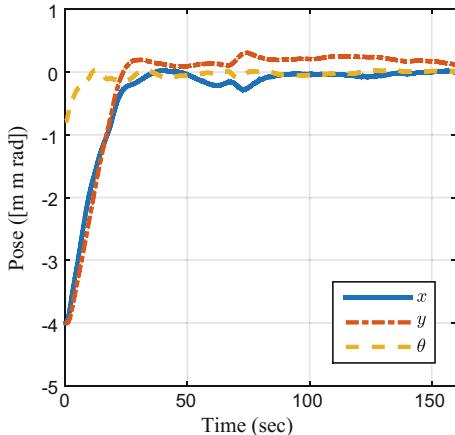
Equations (6.24) and (6.26) form the value function iteration. Evaluating the extrapolated Bellman error (6.24) with each control iteration is computational expensive. Due to the limited computational resources available on-board the autonomous underwater vehicle, the update of the critic weights was selected to be calculated at a different rate (5 Hz) than the main control loop.

For the experiments, the controller in (6.4), (6.6), (6.23), (6.24), (6.26), (6.28), and (6.29) was restricted to three degrees of freedom (i.e., surge, sway, and yaw). The RISE controller is used to regulate the remaining degrees-of-freedom (i.e., heave, roll, and pitch), to maintain the implied assumption that roll and pitch remain at zero and the depth remains constant. The RISE controller in conjunction with the proposed controller is executed at 50 Hz.

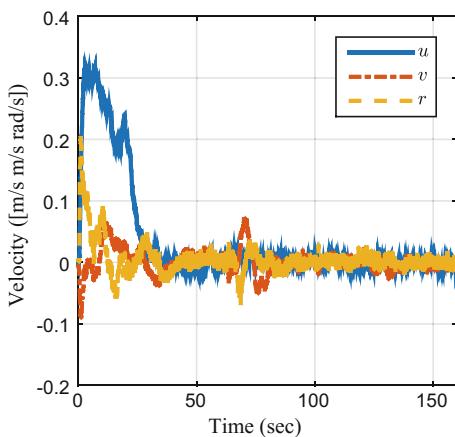
The vehicle uses water profiling data from the Doppler velocity log to measure the relative water velocity near the vehicle in addition to bottom tracking data for the state estimator. Between the state estimator, water profiling data, and recorded data, the equations used to implement the developed controller only contain known or measurable quantities.

The vehicle was commanded to hold a station near the vent of Ginnie Spring. An initial condition of  $\zeta(t_0) = [4 \text{ m } 4 \text{ m } [\frac{\pi}{4}] \text{ rad } 0 \text{ m/s } 0 \text{ ms } 0 \text{ rad/s}]^T$  was given to demonstrate the method's ability to regulate the state. The optimal control weighting matrices were selected to be  $Q = \text{diag}([20, 50, 20, 10, 10, 10])$  and  $R = I_3$ . The system identifier adaptation gains were selected to be  $k_x = 25 \times I_6$ ,  $k_\theta = 12.5$ , and  $\Gamma_\theta = \text{diag}([187.5, 937.5, 37.5, 37.5, 37.5, 37.5, 37.5, 37.5])$ . The parameter estimate was initialized with  $\hat{\theta}(t_0) = 0_{8 \times 1}$ . The neural network weights were initialized to match the ideal values for the linearized optimal control problem, selected by solving the algebraic Riccati equation with the dynamics linearized about the station. The actor adaptation gains were selected to be  $k_{c1} = 0.25 \times I_{21}$ ,  $k_{c2} = 0.5 \times I_{21}$ ,  $k_a = I_{21}$ ,

**Fig. 6.2** Inertial position error  $\eta$  of the autonomous underwater vehicle



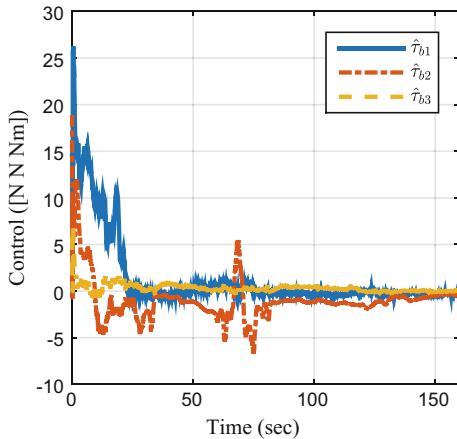
**Fig. 6.3** Body-fixed velocity error  $v$  (bottom) of the autonomous underwater vehicle



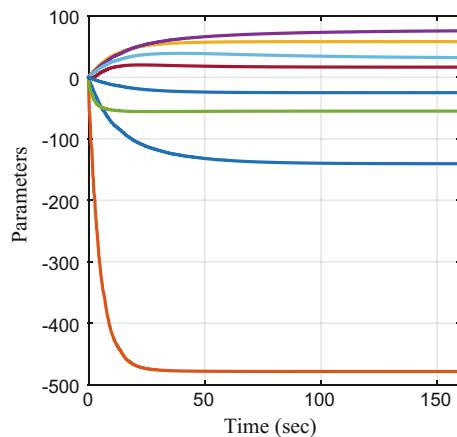
$k_p = 0.25$ , and  $\beta = 0.025$ . The adaptation matrix was initialized to  $\Gamma_0 = 400 \times I_{21}$ . The Bellman error was extrapolated to 2025 points in a grid about the station.

Figures 6.2 and 6.3 illustrate the ability of the generated policy to regulate the state in the presence of the spring's current. Figure 6.4 illustrates the total control effort applied to the body of the vehicle, which includes the estimate of the current compensation term and approximate optimal control. Figure 6.5 illustrates the output of the approximate optimal policy for the residual system. Figure 6.6 illustrates the convergence of the parameters of the system identifier and Figs. 6.7 and 6.8 illustrate convergence of the neural network weights representing the value function.

**Fig. 6.4** Body-fixed total control effort  $\hat{\tau}_b$  commanded about the center of mass of the vehicle

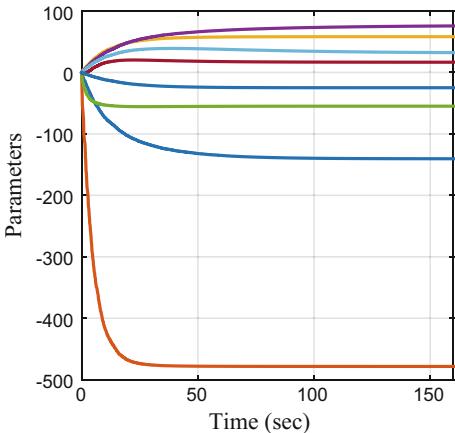


**Fig. 6.5** Body-fixed optimal control effort  $\hat{u}$  commanded about the center of mass of the vehicle

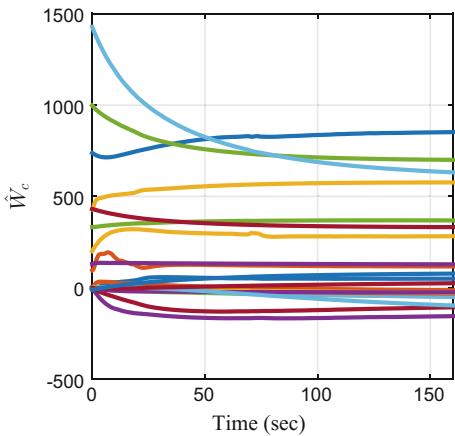


The anomaly seen at  $\sim 70$  s in the total control effort (Fig. 6.4) is attributed to a series of incorrect current velocity measurements. The corruption of the current velocity measurements is possibly due in part to the extremely low turbidity in the spring and/or relatively shallow operating depth. Despite presence of unreliable current velocity measurements the vehicle was able to regulate the vehicle to its station. The results demonstrate the developed method's ability to concurrently identify the unknown hydrodynamic parameters and generate an approximate optimal policy using the identified model. The vehicle follows the generated policy to achieve its station keeping objective using industry standard navigation and environmental sensors (i.e., inertial measurement unit, Doppler velocity log).

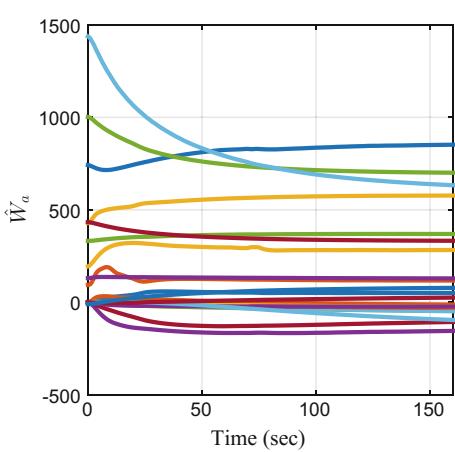
**Fig. 6.6** Identified system parameters determined for the vehicle online. The parameter definitions may be found in Example 6.2 and Eq. (6.100) of [9]



**Fig. 6.7** Value function  $(\hat{W}_c)$  neural network weight estimates online convergence



**Fig. 6.8** Policy  $(\hat{W}_a)$  neural network weight estimates online convergence



## 6.3 Online Optimal Control for Path-Following<sup>2</sup>

### 6.3.1 Problem Description

Path-following refers to a class of problems where the control objective is to converge to and remain on a desired geometric path. The desired path is not necessarily parameterized by time, but by some convenient parameter (e.g., path length). The path-following method in this section utilizes a virtual target that moves along the desired path. The error dynamics are defined kinematically between the virtual target and vehicle as [6]

$$\begin{aligned}\dot{x}(t) &= v(t) \cos \theta(t) + \dot{s}_p(t) (\kappa(t) y(t) - 1), \\ \dot{y}(t) &= v(t) \sin \theta(t) - x(t) \kappa(t) \dot{s}_p(t), \\ \dot{\theta}(t) &= w(t) - \kappa(t) \dot{s}_p(t),\end{aligned}\tag{6.36}$$

where  $x(t)$ ,  $y(t) \in \mathbb{R}$  denote the planar position error between the vehicle and the virtual target,  $\theta(t) \in \mathbb{R}$  denotes the rotational error between the vehicle heading and the heading of the virtual target,  $v(t) \in \mathbb{R}$  denotes the linear velocity of the vehicle,  $w(t) \in \mathbb{R}$  denotes the angular velocity of the vehicle,  $\kappa(t) \in \mathbb{R}$  denotes the path curvature evaluated at the virtual target, and  $s_p(t) \in \mathbb{R}$  denotes velocity of the virtual target along the path. For a detailed derivation of the dynamics in (6.36) see Appendix A.3.3.

**Assumption 6.5** The desired path is regular and  $C^2$  continuous; hence, the path curvature  $\kappa$  is bounded and continuous.

As described in [6], the location of the virtual target is determined by

$$\dot{s}_p(t) \triangleq v_{des}(t) \cos \theta(t) + k_1 x(t),\tag{6.37}$$

where  $v_{des} : \mathbb{R} \rightarrow \mathbb{R}$  is a desired positive, bounded and time-invariant speed profile, and  $k_1 \in \mathbb{R}_{>0}$  is an adjustable gain.

To facilitate the subsequent control development, an auxiliary function  $\phi : \mathbb{R} \rightarrow (-1, 1)$  is defined as

$$\phi(s_p) \triangleq \tanh(k_2 s_p),\tag{6.38}$$

where  $k_2 \in \mathbb{R}_{>0}$  is an adjustable gain. From (6.37) and (6.38), the time derivative of  $\phi$  is

$$\dot{\phi}(t) = k_2 (1 - \phi^2(t)) (v_{des}(t) \cos \theta(t) + k_1 x(t)).\tag{6.39}$$

---

<sup>2</sup>Parts of the text in this section are reproduced, with permission, from [5], ©2014, IEEE.)

Note that the path curvature and desired speed profile can be written as functions of  $\phi$ .

Based on (6.36) and (6.37), auxiliary control inputs  $v_e, w_e \in \mathbb{R}$  are designed as

$$\begin{aligned} v_e(t) &\triangleq v(t) - v_{ss}(\phi(t)), \\ w_e(t) &\triangleq w(t) - w_{ss}(\phi(t)), \end{aligned} \quad (6.40)$$

where  $w_{ss} \triangleq \kappa v_{des}$  and  $v_{ss} \triangleq v_{des}$  are computed based on the control input required to remain on the path.

Substituting (6.37) and (6.40) into (6.36), and augmenting the system state with (6.39), the closed-loop system is

$$\begin{aligned} \dot{x}(t) &= \kappa(\phi(t)) y(t) v_{des}(\phi(t)) \cos \theta(t) + k_1 \kappa x(t) y(t) - k_1 x(t) + v_e(t) \cos \theta(t), \\ \dot{y} &= v_{des}(\phi(t)) \sin \theta(t) - \kappa(\phi(t)) x(t) v_{des}(\phi(t)) \cos \theta(t) - k_1 \kappa(\phi(t)) x^2(t) \\ &\quad + v_e(t) \sin \theta(t), \\ \dot{\theta} &= \kappa(\phi(t)) v_{des}(\phi(t)) - \kappa(\phi(t))(v_{des}(\phi(t)) \cos \theta(t) + k_1 x(t)) + w_e(t), \\ \dot{\phi} &= k_2(1 - \phi^2(t))(v_{des}(\phi(t)) \cos \theta(t) + k_1 x(t)). \end{aligned} \quad (6.41)$$

The closed-loop system in (6.41) can be rewritten in the following control-affine form

$$\dot{\zeta}(t) = f(\zeta(t)) + g(\zeta(t)) u(t), \quad (6.42)$$

where  $\zeta = [x \ y \ \theta \ \phi]^T : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^4$  is the state vector,  $u = [v_e \ w_e]^T : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^2$  is the control vector, and the locally Lipschitz functions  $f : \mathbb{R}^4 \rightarrow \mathbb{R}^4$  and  $g : \mathbb{R}^4 \rightarrow \mathbb{R}^{4 \times 2}$  are defined as

$$f(\zeta) \triangleq \begin{bmatrix} \kappa(\phi) y v_{des}(\phi) \cos \theta + k_1 \kappa(\phi) x y - k_1 x \\ v_{des}(\phi) \sin \theta - \kappa(\phi) x v_{des}(\phi) \cos \theta - k_1 \kappa(\phi) x^2 \\ \kappa(\phi) v_{des}(\phi) - \kappa(v_{des}(\phi) \cos \theta + k_1 x) \\ k_2(1 - \phi^2)(v_{des}(\phi) \cos \theta + k_1 x) \end{bmatrix},$$

$$g(\zeta) \triangleq \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}. \quad (6.43)$$

To facilitate the subsequent stability analysis, a subset of the state, denoted by  $e \in \mathbb{R}^3$ , is defined as  $e \triangleq [x \ y \ \theta]^T \in \mathbb{R}^3$ .

### 6.3.2 Optimal Control and Approximate Solution

The cost functional for the optimal control problem is selected as (6.15), where  $Q \in \mathbb{R}^{4 \times 4}$  is defined as

$$Q \triangleq \begin{bmatrix} \bar{Q} & 0_{3 \times 1} \\ 0_{1 \times 3} & 0 \end{bmatrix},$$

where  $\bar{Q} \in \mathbb{R}^{3 \times 3}$  is a user-defined positive definite matrix such that  $\underline{q} \|\xi_q\|^2 \leq \xi_q^T \bar{Q} \xi_q \leq \bar{q} \|\xi_q\|^2, \forall \xi_q \in \mathbb{R}^3$ , where  $\underline{q}$  and  $\bar{q}$  are positive constants.

The value function satisfies the Hamilton–Jacobi–Bellman equation [14]

$$0 = r(\zeta, u^*(\zeta)) + \nabla_\zeta V^*(\zeta) (f(\zeta) + g(\zeta) u^*(\zeta)), \quad (6.44)$$

Using (6.44) and the parametric approximation of the optimal value function and the optimal policy from (6.22) and (6.23), respectively, the Bellman error,  $\delta : \mathbb{R}^4 \times \mathbb{R}^L \times \mathbb{R}^L \rightarrow \mathbb{R}$  is defined as

$$\delta(\zeta, \hat{W}_c, \hat{W}_a) = r(\zeta, \hat{u}(\zeta, \hat{W}_a)) + \hat{W}_c^T \nabla_\zeta \sigma(\zeta) (f(\zeta) + g(\zeta) \hat{u}(\zeta, \hat{W}_a)). \quad (6.45)$$

The adaptive update laws for the critic weights and the actor weights are given by (6.26) and (6.28), respectively, with the regressor  $\omega_k$  defined as  $\omega_k(t) \triangleq \nabla_\zeta \sigma(\zeta_k) (f(\zeta_k) + g(\zeta_k) \hat{u}(\zeta_k, \hat{W}_a(t))) \in \mathbb{R}^L$  and the normalization factor defined as  $\rho_k(t) \triangleq \sqrt{1 + \omega_k^T(t) \omega_k(t)}$ . The adaptation gain  $\Gamma$  is held constant and it is assumed that the regressor satisfies the rank condition in (6.25).

### 6.3.3 Stability Analysis

To facilitate the subsequent stability analysis, an unmeasurable form of the Bellman error can be written as

$$\delta = -\tilde{W}_c^T \omega - \nabla_\zeta \epsilon f + \frac{1}{2} \nabla_\zeta \epsilon G \nabla_\zeta \sigma^T W + \frac{1}{4} \tilde{W}_a^T G_\sigma \tilde{W}_a + \frac{1}{4} \nabla_\zeta \epsilon G \nabla_\zeta \epsilon^T, \quad (6.46)$$

where  $G \triangleq g \mathcal{R}^{-1} g^T \in \mathbb{R}^{4 \times 4}$  and  $G_\sigma \triangleq \nabla_\zeta \sigma G \nabla_\zeta \sigma^T \in \mathbb{R}^{L \times L}$  are symmetric positive semi-definite matrices. Similarly, at the sampled points the Bellman error can be written as

$$\delta_j = -\tilde{W}_c^T \omega_j + \frac{1}{4} \tilde{W}_d^T G_{\sigma j} \tilde{W}_a + E_j, \quad (6.47)$$

where  $E_j \triangleq \frac{1}{2} \epsilon'_j G_j \sigma'^T W + \frac{1}{4} \epsilon'_j G_j \epsilon'^T - \epsilon'_j f_j \in \mathbb{R}$ ,  $\epsilon'_j \triangleq \nabla_\zeta \epsilon(\zeta_j)$ , and  $\sigma'_j \triangleq \nabla_\zeta \sigma(\zeta_j)$ .

The function  $f$  on any compact set  $\chi \subset \mathbb{R}^4$  is Lipschitz continuous, and bounded by

$$\|f(\zeta)\| \leq L_f \|\zeta\|, \quad \forall \zeta \in \chi,$$

where  $L_f$  is the positive Lipschitz constant. Furthermore, the normalized regressor can be upper bounded by  $\left\| \frac{\omega}{p} \right\| \leq 1$ .

The augmented equations of motion in (6.41) present a unique challenge with respect to the value function  $V$  which is utilized as a Lyapunov function in the stability analysis. To prevent penalizing the vehicle progression along the path, the path parameter  $\phi$  is removed from the cost function with the introduction of a positive semi-definite state weighting matrix  $Q$ . However, since  $Q$  is positive semi-definite, efforts are required to ensure the value function is positive definite. To address this challenge, the fact that the value function can be interpreted as a time-invariant map  $V^* : \mathbb{R}^4 \rightarrow [0, \infty)$  or a time-varying map  $V_t^* : \mathbb{R}^3 \times [0, \infty) \rightarrow [0, \infty)$ , defined as  $V_t^*(e, t) \triangleq V^*\begin{pmatrix} e \\ \phi(t) \end{pmatrix}$  is exploited. Lemma 3.14 is used to show that the time-varying map is a positive definite and decrescent function for use as a Lyapunov function. Hence, on any compact set  $\chi$  the optimal value function  $V_t^*$  satisfies the following properties

$$V_t^*(0, t) = 0,$$

$$\underline{v}(\|e\|) \leq V_t^*(e, t) \leq \bar{v}(\|e\|), \quad (6.48)$$

$\forall t \in [0, \infty)$  and  $\forall e \in \chi$  where  $\underline{v} : [0, \infty] \rightarrow [0, \infty)$  and  $\bar{v} : [0, \infty] \rightarrow [0, \infty)$  are class  $\mathcal{K}$  functions.

To facilitate the subsequent stability analysis, consider the candidate Lyapunov function  $V_L : \mathbb{R}^{4+2L} \times [0, \infty) \rightarrow [0, \infty)$  given as

$$V_L(Z, t) = V_t^*(e, t) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \Gamma^{-1} \tilde{W}_a. \quad (6.49)$$

Using (6.48), the candidate Lyapunov function can be bounded as

$$\underline{v}_L(\|Z\|) \leq V_L(Z, t) \leq \bar{v}_L(\|Z\|), \quad (6.50)$$

where  $\underline{v}_L, \bar{v}_L : [0, \infty) \rightarrow [0, \infty)$  are class  $\mathcal{K}$  functions and  $Z \triangleq [e^T \tilde{W}_c^T \tilde{W}_a^T]^T \in \beta \subset \chi \cup \mathbb{R}^{2L}$ .

**Theorem 6.6** *If Assumptions 6.3 and 6.5 are satisfied with the regressor  $\omega_k$  redefined as  $\omega_k(t) \triangleq \nabla_{\zeta} \sigma(\zeta_k) \left( f(\zeta_k) + g(\zeta_k) \hat{u}(\zeta_k, \hat{W}_a(t)) \right)$  and the normalization factor defined as  $\rho_k(t) \triangleq \sqrt{1 + \omega_k^T(t) \omega_k(t)}$ , the learning gains  $k_{c1}$  and  $k_{c2}$  in (6.26) are*

selected sufficiently small and large, respectively, and

$$K < \overline{v_L}^{-1}(\underline{v_L}(r)) \quad (6.51)$$

where  $K$  is an auxiliary positive constant (see Appendix A.3.4) and  $r \in \mathbb{R}$  is the radius of a selected compact set  $\beta$ , then the controller  $u(t) = \hat{u}(\zeta(t), \hat{W}_a(t))$  with the update laws in (6.26), (6.27), and (6.28) guarantees uniformly ultimately bounded convergence of the approximate policy to the optimal policy and of the vehicle to the virtual target.

*Proof* The time derivative of the candidate Lyapunov function in (6.49) is

$$\dot{V}_L = \nabla_{\zeta} V^* f + \nabla_{\zeta} V^* g \hat{u} - \tilde{W}_c^T \Gamma^{-1} \dot{\hat{W}}_c - \tilde{W}_a^T \dot{\hat{W}}_a.$$

Substituting (6.44), (6.26), and (6.28) yields

$$\begin{aligned} \dot{V}_L = & -e^T \overline{Q} e - u^* \mathcal{R} u^* + \nabla_{\zeta} V^* g \hat{u} - \nabla_{\zeta} V^* g u^* + \tilde{W}_c^T \left[ k_{c1} \frac{\omega}{p} \delta + \frac{k_{c2}}{N} \sum_{j=1}^N \frac{\omega_j}{p_j} \delta_j \right] \\ & + \tilde{W}_a^T k_a (\hat{W}_a - \hat{W}_c). \end{aligned}$$

Using Young's inequality,

$$\dot{V}_L \leq -\varphi_e \|e\|^2 - \varphi_c \|\tilde{W}_c\|^2 - \varphi_a \|\tilde{W}_a\|^2 + \iota_c \|\tilde{W}_c\| + \iota_a \|\tilde{W}_{a1}\| + \iota, \quad (6.52)$$

where  $\varphi_e, \varphi_c, \varphi_a, \iota_c, \iota_a, \iota$  are auxiliary positive constants (see Appendix A.3.4). Completing the squares, (6.52) can be upper bounded by

$$\dot{V}_L \leq -\varphi_e \|e\|^2 - \frac{\varphi_c}{2} \|\tilde{W}_c\|^2 - \frac{\varphi_a}{2} \|\tilde{W}_a\|^2 + \frac{\iota_c^2}{2\varphi_c} + \frac{\iota_a^2}{2\varphi_a} + \iota,$$

which can be further upper bounded as

$$\dot{V}_L \leq -\alpha \|Z\|^2, \quad \forall \|Z\| \geq K > 0 \quad (6.53)$$

$\forall Z \in \beta$ , where  $\alpha$  is an auxiliary positive constant (see Appendix A.3.4).

Using (6.50), (6.51), and (6.53), [16, Theorem 4.18] is invoked to conclude that  $Z$  is uniformly ultimately bounded. The sufficient condition in (6.51) requires the compact set  $\beta$  to be large enough based on the constant  $K$ . The constant  $K$  for a given  $\beta$  can be reduced to satisfy the sufficient condition by reducing the function approximation error in (6.20) and (6.21). The function approximation error can be decreased by increasing the number of neurons in the neural network.  $\square$

To demonstrate the performance of the developed approximate dynamic programming-based guidance law, simulation and experimental results are presented. The simulation allows the developed method to be compared to a numerical offline optimal solution, whereas experimental results demonstrate the real-time optimal performance.

### 6.3.4 Simulation Results

To illustrate the ability of the proposed method to approximate the optimal solution, a simulation is performed where the developed method's policy and value function neural network weight estimates are initialized to ideal weights identified on a previous trial. The true values of the ideal neural network weights are unknown. However, initializing the actor and critic neural network weights to the ideal weights determined offline, the accuracy of the approximation can be compared to the optimal solution. Since an analytical solution is not feasible for this problem, the simulation results are directly compared to results obtained by the offline numerical optimal solver GPOPS [19].

The simulation result utilize the kinematic model in (6.36) as the simulated mobile robot. The vehicle is commanded to follow a figure eight path with a desired speed of  $v_{des} = 0.25 \frac{\text{m}}{\text{s}}$ . The virtual target is initially placed at the position corresponding to an initial path parameter of  $s_p(0) = 0 \text{ m}$ , and the initial error state is selected as  $e(0) = [0.5 \text{ m } 0.5 \text{ m } \frac{\pi}{2\text{rad}}]^T$ . Therefore, the initial augmented state is  $\zeta(0) = [0.5 \text{ m } 0.5 \text{ m } \frac{\pi}{2\text{rad}} \ 0 \text{ m }]^T$ . The basis for the value function approximation is selected as

$$\sigma = [\zeta_1 \zeta_2, \zeta_1 \zeta_3, \zeta_2 \zeta_3, \zeta_1^2, \zeta_2^2, \zeta_3^2, \zeta_4^2]^T.$$

The sampled data points are selected on a  $5 \times 5 \times 3 \times 3$  grid about the origin. The quadratic cost weighting matrices are selected as  $\bar{Q} = \text{diag}([2, 2, 0.25])$  and  $\mathcal{R} = I_2$ . The learning gains are selected by trial and error as

$$\Gamma = \text{diag}([1, 2.5, 1, 0.125, 2.5, 25, 0.5]),$$

$$k_{c1} = 1, k_{c2} = 1, k_a = 1.25.$$

Additionally, systematic gain tuning methods may be used (e.g., a genetic algorithm approach similar to [20] may be used to minimize a desired performance criteria such as weight settling time).

The auxiliary gains in (6.37) and (6.39) are selected as  $k_1 = 0.5$  and  $k_2 = 0.005$ . Determined from a previous trial, the actor and critic neural network weight estimates are initialized to

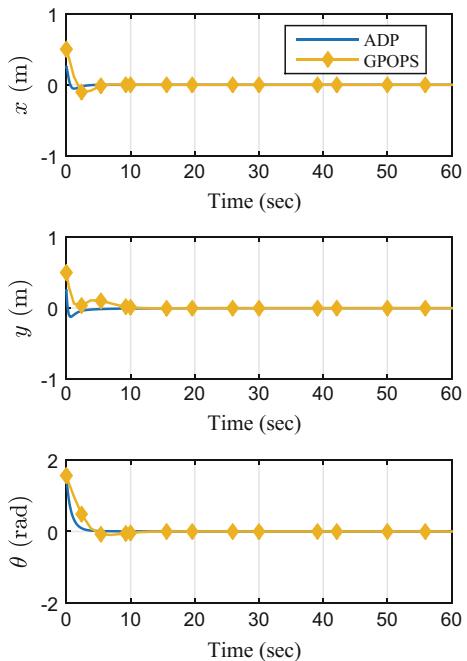
$$\hat{W}_c(0) = [2.8 \times 10^{-2}, -3.3 \times 10^{-2}, 4.0, 1.2, 2.7, 2.9, 1.0]^T$$

and

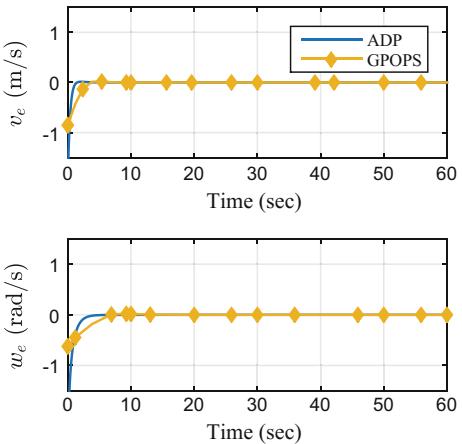
$$\hat{W}_a(0) = [2.8 \times 10^{-2}, -3.3 \times 10^{-2}, 4.0, 1.2, 2.7, 2.9, 1.0]^T$$

Figures 6.9 and 6.10 illustrate that the state and control trajectories approach the solution found using the offline optimal solver, and Fig. 6.11 shows that the neural network critic and actor weight estimates remain at their steady state values. The system trajectories and control values obtained using the developed method approximate the system trajectories and control value of the offline optimal solver. It takes approximately 125 s for the mobile robot to traverse the desired path. However, all figures with the exception of the vehicle trajectory are plotted only for 60 s to provide clarity on the transient response. The steady-state response remains the same after the initial transient (20 s).

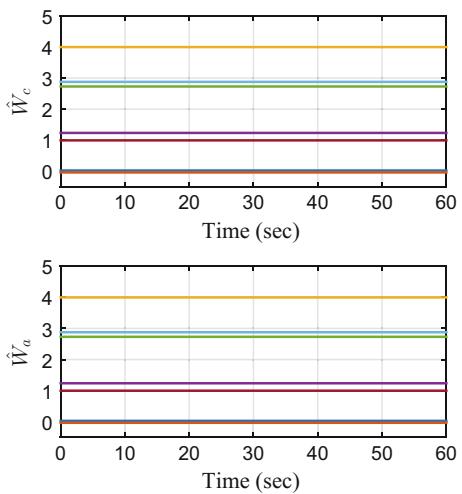
**Fig. 6.9** The error state trajectory generated by the developed method compared to an offline numerical optimal solver



**Fig. 6.10** The control trajectory generated by the developed method compared to an offline numerical optimal solver



**Fig. 6.11** The estimated neural network weight trajectories generated by the developed method in simulation

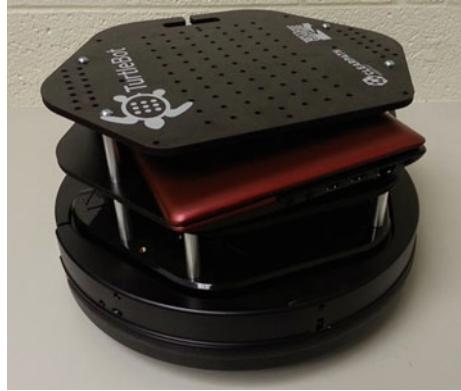


### 6.3.5 Experiment Results

Experimental results demonstrate the ability of the developed controller to simultaneously identify and implement an approximate optimal controller. The approximate dynamic programming-based guidance law is implemented on a Turtlebot wheeled mobile robot depicted in Fig. 6.12. Computation of the optimal guidance law takes place on an on-board ASUS Eee PC netbook with 1.8 GHz Intel Atom processor.<sup>3</sup> The Turtlebot is provided velocity commands from the guidance law where the existing low-level controller on the Turtlebot minimizes the velocity tracking error.

<sup>3</sup>The algorithm complexity is linear with respect to the number of sampled points and cubic with respect to the number of basis functions for each control iteration.

**Fig. 6.12** The Turtlebot wheeled mobile robot



As with the simulation results, the vehicle is commanded to follow a figure eight path with a desired speed of  $v_{des} = 0.25 \frac{\text{m}}{\text{s}}$ . The basis for the value function approximation is selected as

$$\sigma = [\zeta_1 \zeta_2, \zeta_1 \zeta_3, \zeta_1 \zeta_4, \zeta_2 \zeta_3, \zeta_2 \zeta_4, \zeta_3 \zeta_4, \zeta_1^2, \zeta_2^2, \zeta_3^2, \zeta_4^2]^T.$$

The sampled data points are selected on a  $5 \times 5 \times 3 \times 3$  grid about the origin. The quadratic cost weighting matrices are selected as  $\bar{Q} = \text{diag}([2, 2, 0.25])$  and  $R = I_2$ . The learning gains are selected by trial and error as

$$\Gamma = \text{diag}([1, 2.5, 2.5, 1, 0.25, 1, 0.125, 2.5, 7.5, 0.5]),$$

$$k_{c1} = 1, k_{c2} = 1, k_a = 1.25.$$

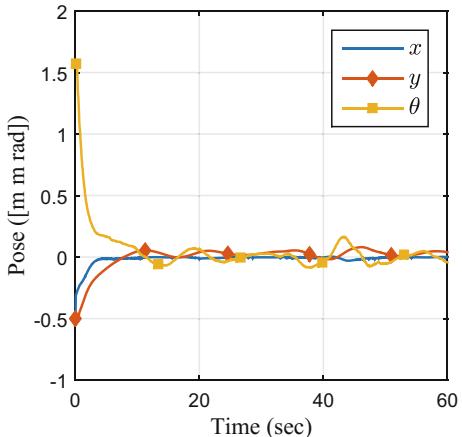
The auxiliary gains in (6.37) and (6.39) are selected as  $k_1 = 0.5$  and  $k_2 = 0.005$ . The initial augmented state is  $\zeta(0) = [-0.5 \text{ m} \ -0.5 \text{ m} \ \frac{\pi}{2\text{rad}} \ 0 \text{ m}]^T$ . The actor and critic neural network weight estimates are arbitrarily initialized to

$$\hat{W}_c(0) = [0, 0, 0, 0.5, 0, 0, 0.5, 0, 1, 0]^T$$

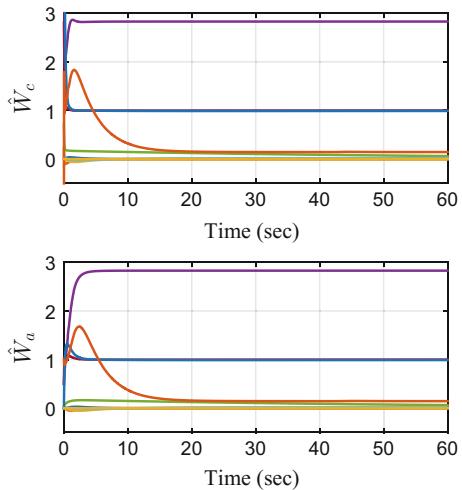
and

$$\hat{W}_a(0) = [0, 0, 0, 0.5, 0, 0, 0.5, 0, 1, 0]^T.$$

**Fig. 6.13** The error state trajectory generated by the developed method implemented on the Turtlebot



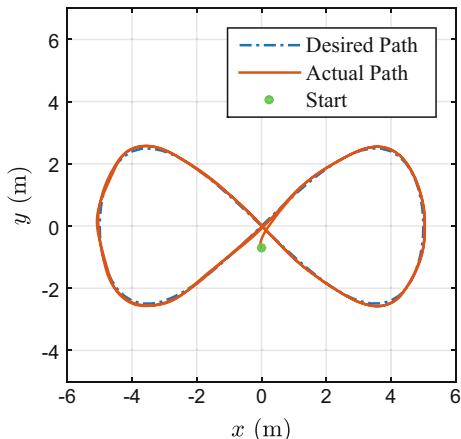
**Fig. 6.14** The estimated neural network weight trajectories generated by the developed method implemented on the Turtlebot



For the given basis, the actor and critic neural network weight estimates may also be initialized such that the value function approximation is equivalent to the solution to the algebraic Riccati equation corresponding to the kinematic model linearized about the initial conditions.

Figure 6.13 shows convergences of the error state to a ball about the origin. Figure 6.14 shows the neural network critic and actor weight estimates converge to steady state values. The ability of the mobile robot to track the desired path is demonstrated in Fig. 6.15.

**Fig. 6.15** The planar trajectory achieved by the developed method implemented on the Turtlebot



## 6.4 Background and Further Reading

Precise station keeping of a marine craft is challenging because of nonlinearities in the dynamics of the vehicle. A survey of station keeping for autonomous surface vehicles can be found in [21]. Common approaches employed to control a marine craft include robust and adaptive control methods [18, 22–24]. These methods provide robustness to disturbances and/or model uncertainty; however, they do not explicitly attempt to reduce energy expenditure. Motivated by the desire to balance energy expenditure and the accuracy of the vehicle’s station, optimal control methods where the performance criterion is a function of the total control effort (energy expended) and state error (station accuracy) are examined. Because of the difficulties associated with finding closed-form analytical solutions to optimal control problems for marine craft, [25] numerically approximates the solution to the Hamilton–Jacobi–Bellman equation using an iterative application of Galerkin’s method, and [26] implements a model-predictive control policy.

Guidance laws of a mobile robot are typically divided into three categories: point regulation, trajectory tracking, and path-following. Path-following refers to a class of problems where the control objective is to converge to and remain on a desired geometric path without the requirement of temporal constraints (cf. [6, 27, 28]). Path-following is ideal for applications intolerant of spatial error (e.g., navigating cluttered environments, executing search patterns). Heuristically, path-following yields smoother convergence to a desired path and reduces the risk of control saturation. A path-following control structure can also alleviate difficulties in the control of nonholonomic vehicles (cf. [28, 29]).

Additional optimal control techniques have been applied to path-following to improve performance. In [30], model-predictive control is used to develop a controller for an omnidirectional robot with dynamics linearized about the desired path. In [31], an adaptive optimal path-following feedback policy is determined by iteratively

solving the algebraic Riccati equation corresponding to the linearized error dynamics about the desired heading. Nonlinear model-predictive control is used in [32] to develop an optimal path-following controller over a finite time horizon. Dynamic programming was applied to the path-following problem in [33] to numerically determine an optimal path-following feedback policy offline. The survey in [34] cites additional examples of model-predictive control and dynamic programming applied to path-following. Unlike approximate dynamic programming, model-predictive control does not guarantee optimality of the implemented controller and dynamic programming does not accommodate simultaneous online learning and utilization of the feedback policy.

## References

1. Walters P, Kamalapurkar R, Voight F, Schwartz E, Dixon WE (to appear) Online approximate optimal station keeping of a marine craft in the presence of an irrotational current. *IEEE Trans Robot*
2. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92
3. Vrabie D, Lewis FL (2009) Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Netw* 22(3):237–246
4. Chowdhary G, Johnson E (2010) Concurrent learning for convergence in adaptive control without persistency of excitation. In: *Proceedings of the IEEE Conference on Decision and Control*, pp 3674–3679
5. Walters P, Kamalapurkar R, Andrews L, Dixon WE (2014) Online approximate optimal path-following for a mobile robot. In: *Proceedings of the IEEE Conference on Decision and Control*, pp 4536–4541
6. Lapierre L, Soetanto D, Pascoal A (2003) Non-singular path-following control of a unicycle in the presence of parametric modeling uncertainties. *Int J Robust Nonlinear Control* 16:485–503
7. Egerstedt M, Hu X, Stotsky A (2001) Control of mobile platforms using a virtual vehicle approach. *IEEE Trans Autom Control* 46(11):1777–1782
8. Dixon WE, Dawson DM, Zergeroglu E, Behal A (2000) Nonlinear control of wheeled mobile robots, vol 262. *Lecture notes in control and information sciences*, Springer, London
9. Fossen TI (2011) *Handbook of marine craft hydrodynamics and motion control*. Wiley, New York
10. Sastry S, Isidori A (1989) Adaptive control of linearizable systems. *IEEE Trans Autom Control* 34(11):1123–1131
11. Ioannou P, Sun J (1996) *Robust adaptive control*. Prentice Hall, Upper Saddle River
12. Chowdhary GV, Johnson EN (2011) Theory and flight-test validation of a concurrent-learning adaptive controller. *J Guid Control Dyn* 34(2):592–607
13. Chowdhary G, Yucelen T, Mühlegg M, Johnson EN (2013) Concurrent learning adaptive control of linear systems with exponentially convergent bounds. *Int J Adapt Control Signal Process* 27(4):280–301
14. Kirk D (2004) *Optimal control theory: an introduction*. Dover, Mineola
15. Dixon WE, Behal A, Dawson DM, Nagarkatti S (2003) *Nonlinear control of engineering systems: a Lyapunov-based approach*. Birkhauser, Boston
16. Khalil HK (2002) *Nonlinear systems*, 3rd edn. Prentice Hall, Upper Saddle River
17. Schmidt W (2004) Springs of Florida. *Bulletin* 66, Florida Geological Survey

18. Fischer N, Hughes D, Walters P, Schwartz E, Dixon WE (2014) Nonlinear RISE-based control of an autonomous underwater vehicle. *IEEE Trans Robot* 30(4):845–852
19. Rao AV, Benson DA, Darby CL, Patterson MA, Francolin C, Huntington GT (2010) Algorithm 902: GPOPS, a MATLAB software for solving multiple-phase optimal control problems using the Gauss pseudospectral method. *ACM Trans Math Softw* 37(2):1–39
20. Otsuka A, Nagata F (2013) Application of genetic algorithms to fine-gain tuning of improved the resolved acceleration controller. *Procedia Comput Sci* 22:50–59
21. Sørensen AJ (2011) A survey of dynamic positioning control systems. *Annu Rev Control* 35:123–136
22. Fossen T, Grovlen A (1998) Nonlinear output feedback control of dynamically positioned ships using vectorial observer backstepping. *IEEE Trans Control Syst Technol* 6:121–128
23. Sebastian E, Sotelo MA (2007) Adaptive fuzzy sliding mode controller for the kinematic variables of an underwater vehicle. *J Intell Robot Syst* 49(2):189–215
24. Tannuri E, Agostinho A, Morishita H, Moratelli L Jr (2010) Dynamic positioning systems: an experimental analysis of sliding mode control. *Control Eng Pract* 18:1121–1132
25. Beard RW, McLain TW (1998) Successive Galerkin approximation algorithms for nonlinear optimal and robust control. *Int J Control* 71(5):717–743
26. Fannemel ÅV (2008) Dynamic positioning by nonlinear model predictive control. Master's thesis, Norwegian University of Science and Technology
27. Morro A, Sgorbissa A, Zaccaria R (2011) Path following for unicycle robots with an arbitrary path curvature. *IEEE Trans Robot* 27(5):1016–1023
28. Morin P, Samson C (2008) Motion control of wheeled mobile robots. Springer handbook of robotics. Springer, Berlin, pp 799–826
29. Dacic D, Nesić D, Kokotovic P (2007) Path-following for nonlinear systems with unstable zero dynamics. *IEEE Trans Autom Control* 52(3):481–487
30. Kanjanawanishkul K, Zell A (2009) Path following for an omnidirectional mobile robot based on model predictive control. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp 3341–3346
31. Ratnoo A, Pb S, Kothari M (2011) Optimal path following for high wind flights. In: IFAC world congress, Milano, Italy 18:12985–12990
32. Faulwasser T, Findeisen R (2009) Nonlinear model predictive path-following control. In: Magni L, Raimondo D, Allgöwer F (eds) Nonlinear model predictive control, vol 384. Springer, Berlin, pp 335–343
33. da Silva JE, de Sousa JB (2011) A dynamic programming based path-following controller for autonomous vehicles. *Control Intell Syst* 39:245–253
34. Sujit P, Saripalli S, Borges Sousa J (2014) Unmanned aerial vehicle path following: a survey and analysis of algorithms for fixed-wing unmanned aerial vehicles. *IEEE Control Syst Mag* 34(1):42–59

# Chapter 7

## Computational Considerations



### 7.1 Introduction

Efficient methods for the approximation of the optimal value function are essential, since an increase in dimension can lead to a exponential increase in the number of required basis functions necessary to achieve an accurate approximation. This is known as the “curse of dimensionality”. To set the stage for the approximation methods of this chapter, the first half of the introduction outlines a problem that arises in the real time application of optimal control theory. Sufficiently accurate approximation of the value function over a sufficiently large neighborhood often requires a large number of basis functions, and hence, introduces a large number of unknown parameters. One way to achieve accurate function approximation with fewer unknown parameters is to use prior knowledge about the system to determine the basis functions. However, for general nonlinear systems, prior knowledge of the features of the optimal value function is generally not available; hence, a large number of generic basis functions is often the only feasible option.

For some problems, such as the linear quadratic regulator problem, the optimal value function takes a particular form which makes the choice of basis functions trivial. In the case of the linear quadratic regulator, the optimal value function is of the form  $\sum_{i,j=1}^n w_{i,j} x_j x_i$  (c.f., [1, 2]), so basis functions of the form  $\sigma_{i,j} = x_j x_i$  will provide an accurate estimation of the optimal value function. However, in most cases, the form of the optimal value function is unknown, and generic basis functions are employed to parameterize the problem.

Often, kernel functions from reproducing kernel Hilbert spaces are used as generic basis functions, and the approximation problem is solved over a (preferably large) compact domain of  $\mathbb{R}^n$  [3–5]. An essential property of reproducing kernel Hilbert spaces is given a collection of basis functions in the Hilbert space, there is a unique set of weights that minimize the error in the Hilbert space norm, the so called ideal

weights [6]. The model choice of kernel is the Gaussian radial basis function given by  $K(x, y) = \exp(-\|x - y\|^2/\mu)$  where  $x, y \in \mathbb{R}^n$  and  $\mu > 0$  [5, 7]. For the approximation of a function over a large compact domain, a large number of basis functions is required, which leads to an intractable computational problem for online control applications.

Thus, approximation methodologies for the reduction of the number of basis functions required to achieve accurate function approximation are well motivated. In particular, the aim of this chapter is the development of an efficient scheme for the approximation of continuous functions via state and time varying basis functions that maintain the approximation of a function in a local neighborhood of the state, deemed the state following (StaF) method. The method developed in this chapter is presented as a general strategy for function approximation, and can be implemented in contexts outside of optimal control.

The particular basis functions that will be employed throughout this chapter are derived from kernel functions corresponding to reproducing kernel Hilbert spaces. In particular, the centers are selected to be continuous functions of the state variable bounded by a predetermined value. That is, given a compact set  $D \subset \mathbb{R}^n$ ,  $\epsilon > 0$ ,  $r > 0$ , and  $L \in \mathbb{N}$ ,  $c_i(x) \triangleq x + d_i(x)$ , where  $d_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is continuously differentiable and  $\sup_{x \in D} \|d_i(x)\| < r$  for  $i = 1, \dots, L$ . The parameterization of a function  $V$  in terms of StaF kernel functions is given by

$$\hat{V}(y; x(t), t) = \sum_{i=1}^L w_i(t) K(y, c_i(x(t))),$$

where  $w_i(t)$  is a weight selected to satisfy

$$\limsup_{t \rightarrow \infty} E_r(x(t), t) < \epsilon,$$

where  $E_r$  is a measure of the accuracy of an approximation in a neighborhood of  $x(t)$ , such as that of the supremum norm:

$$E_r(x(t), t) = \sup_{y \in B_r(x(t))} |V(y) - \hat{V}(y; x(t), t)|.$$

The goal of the StaF method is to establish and maintain an approximation of a function in a neighborhood of the state. Justification for this approach stems from the observation that the optimal controller only requires the value of the estimation of the optimal value function to be accurate at the current system state. Thus, when computational resources are limited, computational efforts should be focused on improving the accuracy of approximations near the system state.

Previous nonlinear approximation methods focus on adjusting of the centers of radial basis functions (c.f. [8–10]). These efforts are focused on offline techniques, which allow the selection of optimal centers for a global approximation. Since computational resources are considered to be limited in the present context, global approximations become less feasible as the dimension of the system increases.

Section 7.3 lays the foundation for the establishment and maintenance of a real-time moving local approximation of a continuous function. Section 7.3.1 of this chapter frames the particular approximation problem of the StaF method. Section 7.3.2 demonstrates that the achievement of an accurate approximation with a fixed number of moving basis functions is possible (Theorem 7.1), and also demonstrates the existence of an ideal weight function (Theorem 7.3). Section 7.3.3 demonstrates an explicit bound on the number of required StaF basis functions for the case of the exponential kernel function. Section 7.3.4 provides a proof of concept demonstrating the existence of weight update laws to maintain an accurate approximation of a function in a local neighborhood, ultimately establishing a uniform ultimate bounded result. The remaining sections demonstrate the developed method through numerical experiments. Section 7.3.5 gives the results of a *gradient chase* algorithm.

In Sect. 7.4, a novel model-based reinforcement learning technique is developed to achieve sufficient excitation without causing undesirable oscillations and expenditure of control effort like traditional approximate dynamic programming techniques and at a lower computational cost than state-of-the-art data-driven approximate dynamic programming techniques. Motivated by the fact that the computational effort required to implement approximate dynamic programming and the data-richness required to achieve convergence both decrease with decreasing number of basis functions, this chapter focuses on reduction of the number of basis functions used for value function approximation.

A key contribution of this chapter is the observation that online implementation of an approximate dynamic programming-based approximate optimal controller does not require an estimate of the optimal value function over the entire domain of operation of the system. Instead, only an estimate of the slope of the value function evaluated at the current state is required for feedback. Hence, estimation of the value function over a small neighborhood of the current state should be sufficient to implement an approximate dynamic programming-based approximate optimal controller. Since it is reasonable to postulate that approximation of the value function over a local domain would require fewer basis functions than approximation over the entire domain of operation, reduction of the size of the approximation domain is motivated.

Unlike traditional value function approximation, where the unknown parameters are constants, the unknown parameters corresponding to the StaF kernels are functions of the system state. The Lyapunov-based stability analysis presented in Sect. 7.4.3 is

facilitated by the fact that the ideal weights are continuously differentiable functions of the system state. To facilitate the proof of continuous differentiability, the StaF kernels are selected from a reproducing kernel Hilbert space. Other function approximation methods, such as radial basis functions, sigmoids, higher order neural networks, support vector machines, etc., can potentially be utilized in a state-following manner to achieve similar results provided continuous differentiability of the ideal weights can be established. An examination of smoothness properties of the ideal weights resulting from a state-following implementation of the aforementioned function approximation methods is out of the scope of this chapter.

Another key contribution of this chapter is the observation that model-based reinforcement learning techniques can be implemented without storing any data if the available model is used to simulate persistent excitation. In other words, an excitation signal added to the simulated system, instead of the actual physical system, can be used to learn the value function. Excitation via simulation is implemented using Bellman error extrapolation (cf. [11–13]); however, instead of a large number of autonomous extrapolation functions employed in the previous chapters, a single time-varying extrapolation function is selected, where the time-variation of the extrapolation function simulates excitation. The use of a single extrapolation point introduces a technical challenge since the Bellman error extrapolation matrix is rank deficient at each time instance. The aforementioned challenge is addressed in Sect. 7.4.3 by modifying the stability analysis to utilize persistent excitation of the extrapolated regressor matrix. Simulation results including comparisons with state-of-the-art model-based reinforcement learning techniques are presented to demonstrate the effectiveness of the developed technique.

## 7.2 Reproducing Kernel Hilbert Spaces

A reproducing kernel Hilbert space,  $H$ , is a Hilbert space with inner product  $\langle \cdot, \cdot \rangle_H$  of functions  $f : X \rightarrow \mathbb{F}$  (where  $\mathbb{F} = \mathbb{C}$  or  $\mathbb{R}$ ) for which given any  $x \in X$ , the functional  $E_x f := f(x)$  is bounded. By the Reisz representation theorem, for each  $x \in X$  there is a unique function  $k_x \in H$  for which  $\langle f, k_x \rangle_H = f(x)$ . Each function  $k_x$  is called a reproducing kernel for the point  $x \in X$ . The function  $K(x, y) = \langle k_y, k_x \rangle_H$  is called the kernel function for  $H$  [7]. The norm corresponding to  $H$  will be denoted as  $\|\cdot\|_H$ , and the subscript will be suppressed when the Hilbert space is understood. Kernel functions are dense in  $H$  under the reproducing kernel Hilbert space norm.

Kernel functions have the property that for each collection of points  $\{x_1, \dots, x_m\} \subset X$ , the matrix  $(K(x_i, x_j))_{i,j=1}^m$  is positive semi-definite. The Aronszajn–Moore theorem states that there is a one to one correspondence between kernel functions with

this property and reproducing kernel Hilbert spaces. In fact, starting with a kernel function having the positive semi-definite property, there is an explicit construction for its reproducing kernel Hilbert space. Generally, the norm for the reproducing kernel Hilbert space is given by

$$\|f\|_H := \sup\{\|P_{c_1, \dots, c_M} f\|_H : M \in \mathbb{N} \text{ and } c_1, \dots, c_M \in X\}, \quad (7.1)$$

where  $P_{c_1, \dots, c_M} f$  is the projection of  $f$  onto the subspace of  $H$  spanned by the kernel function  $K(\cdot, c_i)$  for  $i = 1, \dots, M$ .  $P_{c_1, \dots, c_M} f$  is computed by interpolating the points  $(c_i, f(c_i))$  for  $i = 1, \dots, M$  with a function of the form  $\sum_{i=1}^M w_i K(\cdot, c_i)$ . The norm of the projection then becomes<sup>1</sup>  $\|P_{c_1, \dots, c_M} f\| = \left( \sum_{i,j=1}^M c_i \bar{c}_j K(c_j, c_i) \right)^{1/2}$ . In practice, the utility of computing the norm of  $f$  as (7.1) is limited, and alternate forms of the norm are sought for specific reproducing kernel Hilbert spaces.

Unlike  $L^2$  spaces, norm convergence in a reproducing kernel Hilbert space implies pointwise convergence. This follows since if  $f_n \rightarrow f$  in the reproducing kernel Hilbert space norm, then

$$\begin{aligned} |f(x) - f_n(x)| &= |\langle f - f_n, k_x \rangle| \leq \\ \|f - f_n\| \|k_x\| &= \|f - f_n\| \sqrt{K(x, x)}. \end{aligned}$$

When  $K$  is a continuous function of  $X$ , the term  $\sqrt{K(x, x)}$  is bounded over compact sets, and thus, norm convergence implies uniform convergence over compact sets. Therefore, the problem of establishing an accurate approximation in the supremum norm of a function is often relaxed to determining an accurate approximation of a function in the reproducing kernel Hilbert space norm.

Given a reproducing kernel Hilbert space  $H$  over a set  $X$  and  $Y \subset X$ , the space  $H_Y$  obtained by restricting each function  $f \in H$  to the set  $Y$  is itself a reproducing kernel Hilbert space where the kernel function is given by restricting the original kernel function to the set  $Y \times Y$ . The resulting Hilbert space norm is given by

$$\|g\|_{H_Y} = \inf\{\|f\|_H : f \in H \text{ and } f|_Y = g\}.$$

Therefore, the map  $f \mapsto f|_Y$  is norm decreasing from  $H$  to  $H_Y$  [7]. For the purposes of this paper, the norm obtained by restricting a reproducing kernel Hilbert space  $H$  over  $\mathbb{R}^n$  to a closed neighborhood  $\overline{B_r(x)}$  where  $r > 0$  and  $x \in \mathbb{R}^n$  will be denoted as  $\|\cdot\|_{r,x}$ .

---

<sup>1</sup>For  $z \in \mathbb{C}$  the quantity  $Re(z)$  is the real part of  $z$ , and  $\bar{z}$  represents the complex conjugate of  $z$ .

## 7.3 StaF: A Local Approximation Method

### 7.3.1 The StaF Problem Statement

Given a continuous function  $V : \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $\epsilon, r > 0$ , and a dynamical system  $\dot{x}(t) = f(x(t), u(t))$ , where  $f$  is regular enough for the system to be well defined, the goal of the StaF approximation method is to select a number, say  $L \in \mathbb{N}$ , of state and time varying basis functions  $\sigma_i : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$  for  $i = 1, 2, \dots, L$  and weight signals  $w_i : \mathbb{R}_+ \rightarrow \mathbb{R}$  for  $i = 1, 2, \dots, L$  such that

$$\limsup_{t \rightarrow \infty} \sup_{y \in B_r(x(t))} \left| V(y) - \sum_{i=1}^L w_i(t) \sigma_i(y; x(t), t) \right| < \epsilon. \quad (7.2)$$

Central problems to the StaF method are those of determining the basis functions and the weight signals. When reproducing kernel Hilbert spaces are used for basis functions, (7.2) can be relaxed so that the supremum norm is replaced with the Hilbert space norm. Since the Hilbert space norm of a reproducing kernel Hilbert space dominates the supremum norm (cf. [7, Corollary 4.36]), (7.2) with the supremum norm is simultaneously satisfied. Moreover, when using a reproducing kernel Hilbert space, the basis functions can be selected to correspond to centers placed in a moving neighborhood of the state. In particular, given a kernel function  $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$  corresponding to a (universal) reproducing kernel Hilbert space,  $H$ , and center functions  $c_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$  for which  $c_i(x) - x = d_i(x)$  is a continuous function bounded by  $r$  then the StaF problem becomes the determination of weight signals  $w_i : \mathbb{R}_+ \rightarrow \mathbb{R}$  for  $i = 1, \dots, L$  such that

$$\limsup_{t \rightarrow \infty} \left\| V(\cdot) - \sum_{i=1}^L w_i(t) K(\cdot, c_i(x(t))) \right\|_{r,x(t)} < \epsilon, \quad (7.3)$$

where  $\|\cdot\|_{r,x(t)}$  is the norm of the reproducing kernel Hilbert space obtained by restricting functions in  $H$  to  $B_r(x(t))$  [7, 14].

Since (7.3) implies (7.2), the focus of this section is to demonstrate the feasibility of satisfying (7.3). Theorem 7.1 demonstrates that under a certain continuity assumption a bound on the number of kernel functions necessary for the maintenance of an approximation throughout a compact set can be determined, and Theorem 7.3 shows that a collection of continuous ideal weight functions can be determined to satisfy (7.3). Theorem 7.3 justifies the use of weight update laws for the maintenance of an accurate function approximation, and this is demonstrated by Theorem 7.5.

The choice of reproducing kernel Hilbert space for Sect. 7.3.5 is that which corresponds to the exponential kernel  $K(x, y) = \exp(x^T y)$ , where  $x, y \in \mathbb{R}^n$ . The reproducing kernel Hilbert space will be denoted by  $F^2(\mathbb{R}^n)$  since it is closely connected to the Bergmann–Fock space [15]. The reproducing kernel Hilbert space corresponding to the exponential kernel is a universal reproducing kernel

Hilbert space [7, 16], which means that given any compact set  $D \subset \mathbb{R}^n$ ,  $\epsilon > 0$  and continuous function  $f : D \rightarrow \mathbb{R}$ , there exists a function  $\hat{f} \in F^2(\mathbb{R}^n)$  for which  $\sup_{x \in D} |f(x) - \hat{f}(x)| < \epsilon$ .

### 7.3.2 Feasibility of the StaF Approximation and the Ideal Weight Functions

The first theorem concerning the StaF method demonstrates that if the state variable is constrained to a compact subset of  $\mathbb{R}^n$ , then there is a finite number of StaF basis functions required to establish the accuracy of an approximation.

**Theorem 7.1** *Suppose that  $K : X \times X \rightarrow \mathbb{C}$  is a continuous kernel function corresponding to a reproducing kernel Hilbert space,  $H$ , over a set  $X$  equipped with a metric topology. If  $V \in H$ ,  $D$  is a compact subset of  $X$  with infinite cardinality,  $r > 0$ , and  $\|V\|_{x,r}$  is continuous with respect to  $x$ , then for all  $\epsilon > 0$  there is an  $L \in \mathbb{N}$  such that for each  $x \in D$  there are centers  $c_1, c_2, \dots, c_L \in B_r(x)$  and weights  $w_i \in \mathbb{C}$  such that*

$$\left\| V(\cdot) - \sum_{i=1}^L w_i K(\cdot, c_i) \right\|_{r,x} < \epsilon.$$

*Proof* Let  $\epsilon > 0$ . For each neighborhood  $B_r(x)$  with  $x \in D$ , there exists a finite number of centers  $c_1, \dots, c_L \in B_r(x)$ , and weights  $w_1, \dots, w_L \in \mathbb{C}$ , such that

$$\left\| V(\cdot) - \sum_{i=1}^L w_i K(\cdot, c_i) \right\|_{r,x} < \epsilon.$$

Let  $L_{x,\epsilon}$  be the minimum such number. The claim of the proposition is that the set  $Q_\epsilon \triangleq \{L_{x,\epsilon} : x \in D\}$  is bounded. Assume by way of contradiction that  $Q_\epsilon$  is unbounded, and take a sequence  $\{x_n\} \subset D$  such that  $L_{x_n,\epsilon}$  is a strictly increasing sequence (i.e., an unbounded sequence of integers) and  $x_n \rightarrow x$  in  $D$ . It is always possible to find such a convergent sequence, since every compact subset of metric space is sequentially compact. Let  $c_1, \dots, c_{L_{x,\epsilon/2}} \in B_r(x)$  and  $w_1, \dots, w_{L_{x,\epsilon/2}} \in \mathbb{C}$  be centers and weights for which

$$\left\| V(\cdot) - \sum_{i=1}^{L_{x,\epsilon/2}} w_i K(\cdot, c_i) \right\|_{r,x} < \epsilon/2. \quad (7.4)$$

For convenience, let each  $c_i \in B_r(x)$  be expressed as  $x + d_i$  for  $d_i \in B_r(0)$ . The norm in (7.4), which will be denoted by  $E(x)$ , can be written as

$$E(x) \triangleq \left( \|V\|_{r,x} - 2\operatorname{Re} \left( \sum_{i=1}^{L_{x,\epsilon/2}} w_i V(x + d_i) \right) + \sum_{i,j=1}^{L_{x,\epsilon/2}} w_i \overline{w_j} K(x + d_i, x + d_j) \right)^{1/2}.$$

By the hypothesis,  $K$  is continuous with respect to  $x$ , which implies that  $V$  is continuous [3], and  $\|V\|_{r,x}$  is continuous with respect to  $x$  by the hypothesis. Hence, there exists  $\eta > 0$  for which  $|E(x) - E(x_n)| < \epsilon/2$ ,  $\forall x_n \in B_\eta(x)$ . Thus  $E(x_n) < E(x) + \epsilon/2 < \epsilon$  for sufficiently large  $n$ . By minimality  $L_{x_n,\epsilon} < L_{x,\epsilon/2}$  for sufficiently large  $n$ . This is a contradiction.  $\square$

The assumption of the continuity of  $\|V\|_{r,x}$  in Theorem 7.1 is well founded. There are several examples where the assumption is known to hold. For instance, if the reproducing kernel Hilbert space is a space of real entire functions, as it is for the exponential kernel, then  $\|V\|_{r,x}$  is not only continuous, but it is constant.

Using a similar argument as that in Theorem 7.1, the theorem can be shown to hold when the restricted Hilbert space norm is replaced by the supremum norm over  $\overline{B_r(x)}$ . The proof of the following theorem can be found in [17].

**Proposition 7.2** *Let  $D$  be a compact subset of  $\mathbb{R}^n$ ,  $V : \mathbb{R}^n \rightarrow \mathbb{R}$  be a continuous function, and  $K : \mathbb{R}^n \rightarrow \mathbb{R}^n \rightarrow \mathbb{R}$  be a continuous and universal kernel function. For all  $\epsilon, r > 0$ , there exists  $L \in \mathbb{N}$  such that for each  $x \in D$ , there is a collection of centers  $c_1, \dots, c_L \in \overline{B_r(x)}$  and weights  $w_1, \dots, w_L \in \mathbb{R}$  such that  $\sup_{y \in \overline{B_r(x)}} \left| V(y) - \sum_{i=1}^L K(y, c_i) \right| < \epsilon$ .*

Now that it has been demonstrated that only a finite number of moving centers is required to maintain an accurate approximation, it will now be demonstrated that the ideal weights corresponding to the moving centers change continuously or smoothly with the corresponding change in centers. In traditional adaptive control applications, it is assumed that there is a collection of constant ideal weights, and much of the theory is in the demonstration of the convergence of approximate weights to the ideal weights. Since the ideal weights are no longer constant, it is necessary to show that the ideal weights change smoothly as the system progresses. The smooth change in centers will allow the proof of uniform ultimately bounded results through the use of weight update laws. One such result will be demonstrated in Sect. 7.3.4, in particular a uniformly ultimately bounded result is proven in Theorem 7.5.

**Theorem 7.3** *Let  $H$  be a reproducing kernel Hilbert space over a set  $X \subset \mathbb{R}^n$  with a strictly positive kernel  $K : X \times X \rightarrow \mathbb{C}$  such that  $K(\cdot, c) \in C^m(\mathbb{R}^n)$  for all  $c \in X$ . Suppose that  $V \in H$ . Let  $C$  be an ordered collection of  $L$  distinct centers,  $C = (c_1, c_2, \dots, c_L) \in X^L$ , with the associated ideal weights*

$$W_H(C) = \arg \min_{a \in \mathbb{C}^L} \left\| \sum_{i=1}^L a_i K(\cdot, c_i) - V(\cdot) \right\|_H. \quad (7.5)$$

*The function  $W_H$  is  $m$ -times continuously differentiable with respect to each component of  $C$ .*

*Proof* The determination of  $W_H(C)$  is equivalent to computing the projection of  $V$  onto the space  $Y = \text{span}\{K(\cdot, c_i) : i = 1, \dots, L\}$ . To compute the projection, a Gram-Schmidt algorithm may be employed. The Gram-Schmidt algorithm is most easily expressed in its determinant form. Let  $D_0 = 1$  and  $D_m = \det(K(c_j, c_i))_{i,j=1}^m$ , then for  $m = 1, \dots, L$  the functions

$$u_m(x) := \frac{1}{\sqrt{D_{m-1} D_m}} \det \begin{pmatrix} K(c_1, c_1) & K(c_1, c_2) & \cdots & K(c_1, c_m) \\ K(c_2, c_1) & K(c_2, c_2) & \cdots & K(c_2, c_m) \\ \vdots & \vdots & \ddots & \vdots \\ K(c_{m-1}, c_1) & K(c_{m-1}, c_2) & \cdots & K(c_{m-1}, c_m) \\ K(x, c_1) & K(x, c_2) & \cdots & K(x, c_m) \end{pmatrix}$$

constitute an orthonormal basis for  $Y$ . Since  $K$  is strictly positive definite,  $D_m$  is positive for each  $m$  and every  $C$ . The coefficient for each  $K(x, c_l)$  with  $l = 1, \dots, m$  in  $u_m$  is a sum of products of the terms  $K(c_i, c_j)$  for  $i, j = 1, \dots, m$ . Each such coefficient is  $m$ -times differentiable with respect to each  $c_i$ ,  $i = 1, \dots, L$ . Finally, each term in  $W_H(C)$  is a linear combination of the coefficients determined by  $u_m$  for  $m = 1, \dots, L$ , and thus is continuously differentiable with respect to each  $c_i$  for  $i = 1, \dots, L$ .  $\square$

### 7.3.3 Explicit Bound for the Exponential Kernel

Theorem 7.1 demonstrated a bound on the number of kernel functions required for the maintenance of the accuracy of a moving local approximation. However, the proof does not provide an algorithm to computationally determine the upper bound. Indeed, even when the approximation with kernel functions is performed over a fixed compact set, a general bound for the number of collocation nodes required for accurate function approximation is unknown.

Thus, it is desirable to have a computationally determinable upper bound to the number of StaF basis functions required for the maintenance of an accurate function approximation. Theorem 7.4 provides a calculable bound on the number of exponential functions required for the maintenance of an approximation with respect to the supremum norm.

While such error bounds have been computed for the exponential function before (cf. [18]), current literature lets the “frequencies” or centers of the exponential kernel functions to be unconstrained. The contribution of Theorem 7.4 is the development of an error bound while constraining the size of the centers.

**Theorem 7.4** *Let  $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$  given by  $K(x, y) = \exp(x^T y)$  be the exponential kernel function. Let  $D \subset \mathbb{R}^n$  be a compact set,  $V : D \rightarrow \mathbb{R}$  continuous, and  $\epsilon, r > 0$ . For each  $x \in D$ , there exists a finite number of centers  $c_1, \dots, c_{L_{x,\epsilon}} \in B_r(x)$  and weights  $w_1, w_2, \dots, w_{L_{x,\epsilon}} \in \mathbb{R}$ , such that*

$$\sup_{y \in \overline{B}_r(x)} \left| V(y) - \sum_{i=1}^{L_{x,\epsilon}} w_i K(y, c_i) \right| < \epsilon.$$

If  $p$  is an approximating polynomial that achieves the same accuracy over  $\overline{B}_r(x)$  with degree  $N_{x,\epsilon}$ , then an asymptotically similar bound can be found with  $L_{x,\epsilon}$  kernel functions, where  $L_{x,\epsilon} < \binom{n+N_{x,\epsilon}+S_{x,\epsilon}}{N_{x,\epsilon}+S_{x,\epsilon}}$  for some constant  $S_{x,\epsilon}$ . Moreover,  $N_{x,\epsilon}$  and  $S_{x,\epsilon}$  can be bounded uniformly over  $D$ , and thus, so can  $L_{x,\epsilon}$ .

*Proof* For notational simplicity, the quantity  $\|f\|_{D,\infty}$  denotes the supremum norm of a function  $f : D \rightarrow \mathbb{R}$  over the compact set  $D$  throughout the proof of Theorem 7.4.

First, consider the ball of radius  $r$  centered at the origin. The statement of the theorem can be proven by finding an approximation of monomials by a linear combination of exponential kernel functions.

Let  $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$  be a multi-index, and define  $|\alpha| = \sum \alpha_i$ . Note that

$$m^{|\alpha|} \prod_{i=1}^n (\exp(y_i/m) - 1)^{\alpha_i} = y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_n^{\alpha_n} + O\left(\frac{1}{m}\right)$$

which leads to the sum

$$\begin{aligned} m^{|\alpha|} \sum_{l_i \leq \alpha_i, i=1,2,\dots,n} & \binom{\alpha_1}{l_1} \binom{\alpha_2}{l_2} \cdots \binom{\alpha_n}{l_n} (-1)^{|\alpha| - \sum_i l_i} \exp\left(\sum_{i=1}^n y_i \left(\frac{l_i}{m}\right)\right) \\ &= y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_n^{\alpha_n} + O\left(\frac{1}{m}\right), \end{aligned} \quad (7.6)$$

where the notation  $g_m(x) = O(f(m))$  means that for sufficiently large  $m$ , there is a constant  $C$  for which  $g_m(x) < C f(m)$ ,  $\forall y \in \overline{B}_r(0)$ . The big-oh constant indicated by  $O(1/m)$  can be computed in terms of the derivatives of the exponential function via Taylor's Theorem. The centers corresponding to this approximation are of the form  $l_i/m$  where  $l_i$  is a non-negative integer satisfying  $l_i < \alpha_i$ . Hence, for  $m$  sufficiently large, the centers reside in  $B_r(0)$ .

To shift the centers so that they reside in  $B_r(y)$ , let  $x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$ , and multiply both sides of (7.6) by  $\exp(y^T x)$  to get

$$\begin{aligned} m^{|\alpha|} \sum_{l_i \leq \alpha_i, i=1,2,\dots,n} & \binom{\alpha_1}{l_1} \binom{\alpha_2}{l_2} \cdots \binom{\alpha_n}{l_n} (-1)^{|\alpha| - \sum_i l_i} \exp\left(\sum_{i=1}^n y_i \left(\frac{l_i}{m} + x_i\right)\right) \\ &= e^{y^T x} (y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_n^{\alpha_n}) + O\left(\frac{1}{m}\right). \end{aligned}$$

For each multi-index,  $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ , the centers for the approximation of the corresponding monomial are of the form  $x_i + l_i/m$  for  $0 \leq l_i \leq \alpha_i$ . Thus, by linear

combinations of these kernel functions, a function of the form  $e^{y^T x} g(y)$ , with  $g$  a multivariate polynomial, can be uniformly approximated by exponential functions over  $B_r(x)$ . Moreover if  $g$  is a polynomial of degree  $\beta$ , then this approximation can be a linear combination of  $\binom{n+\beta}{\beta}$  kernel functions.

Let  $\epsilon' > 0$  and suppose that  $p_x$  is polynomial with degree  $N_{x,\epsilon'}$  such that  $p_x(y) = V(y) + \epsilon_1(y)$  where  $|\epsilon_1(y)| < \|e^{y^T x}\|_{D,\infty}^{-1} \epsilon'/2 \forall y \in B_r(x)$ . Let  $q_x(y)$  be a polynomial in  $\mathbb{R}^n$  variables of degree  $S_{x,\epsilon}$  such that  $q_x(y) = e^{-y^T x} + \epsilon_2(y)$  where  $|\epsilon_2(y)| < \|V\|_{D,\infty}^{-1} \|e^{y^T x}\|_{D,\infty}^{-1} \epsilon'/2, \forall y \in B_r(x)$ .

The above construction indicates that there is a sequence of linear combinations of exponential kernel functions,  $F_m(y)$ , with a fixed number of centers inside  $B_r(x)$ , for which

$$\begin{aligned} F_m(y) &= e^{y^T x} q_x(y) p_x(y) + O\left(\frac{1}{m}\right) \\ &= e^{y^T x} \left( e^{-y^T x} + \epsilon_2(y) \right) (V(y) + \epsilon_1(y)) + O\left(\frac{1}{m}\right). \end{aligned}$$

After multiplication and an application of the triangle inequality, the following is established:

$$|F_m(y) - V(y)| < \epsilon' + \left( \frac{\|V\|_{D,\infty}^{-1} \|e^{y^T x}\|_{D,\infty}^{-1}}{4} \right) \epsilon'^2 + O\left(\frac{1}{m}\right)$$

for all  $y \in B_r(x)$ . The degree of the polynomial  $q_x$ ,  $S_{x,\epsilon}$ , can be uniformly bounded in terms of the modulus of continuity of  $e^{y^T x}$  over  $D$ . Similarly, the uniform bound on the degree of  $p_x$ ,  $N_{x,\epsilon'}$ , can be described in terms of the modulus of continuity of  $V$  over  $D$ . The number of centers required for  $F_m(y)$  is determined by the degree of the polynomial  $q \cdot p$  (treating the  $x$  terms of  $q$  as constant), which is sum of the two polynomial degrees. Finally for  $m$  large enough and  $\epsilon'$  small enough,  $|F_m(y) - V(y)| < \epsilon$ , and the proof is complete.  $\square$

Theorem 7.4 demonstrates an upper bound required for the accurate approximation of a function through the estimation of approximating polynomials. Moreover, the upper bound is a function of the polynomial degrees. The exponential kernel will be used for simulations in Sect. 7.3.5.

### 7.3.4 The Gradient Chase Theorem

As mentioned before, the theory of adaptive control is centered around the concept of weight update laws. Weight update laws are a collection of rules that the approximating weights must obey which lead to convergence to the ideal weights. In the case of the StaF approximation framework, the ideal weights are replaced with ideal

weight functions. Theorem 7.3 showed that if the moving centers of the StaF kernel functions are selected in such a way that the centers adjust smoothly with respect to the state  $x$ , then the ideal weight functions will also change smoothly with respect to  $x$ . Thus, in this context, weight update laws of the StaF approximation framework aim to achieve an estimation of the ideal weight function at the current state.

Theorem 7.5 provides an example of such weight update laws that achieve a uniformly ultimately bounded result. The theorem takes advantage of perfect samples of a function in the reproducing kernel Hilbert space  $H$  corresponding to a real valued kernel function. The proof of the theorem follows the standard proof for the convergence of the gradient descent algorithm for a quadratic programming problem [19].

**Theorem 7.5** *Let  $H$  be a real valued reproducing kernel Hilbert space over  $\mathbb{R}^n$  with a continuously differentiable strictly positive definite kernel function  $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ . Let  $V \in H$ ,  $D \subset \mathbb{R}^n$  be a compact set, and  $x : \mathbb{R} \rightarrow \mathbb{R}^n$  be a state variable subject to the dynamical system  $\dot{x} = q(x, t)$ , where  $q : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$  is a bounded locally Lipschitz continuous function. Further suppose that  $x(t) \in D \ \forall t > 0$ . Let  $c : \mathbb{R}^n \rightarrow \mathbb{R}^L$ , where for each  $i = 1, \dots, L$ ,  $c_i(x) = x + d_i(x)$  where  $d_i \in C^1(\mathbb{R}^n)$ , and let  $a \in \mathbb{R}^L$ . Consider the function*

$$F(a, c) = \left\| V - \sum_{i=1}^L a_i K(\cdot, c_i(x)) \right\|_H^2.$$

*At each time instance  $t > 0$ , there is a unique  $W(t)$  for which  $W(t) = \arg \min_{a \in \mathbb{R}^L} F(a, c(x(t)))$ . Given any  $\epsilon > 0$  and initial value  $a^0$ , there is a frequency  $\tau > 0$ , where if the gradient descent algorithm (with respect to  $a$ ) is iterated at time steps  $\Delta t < \tau^{-1}$ , then  $F(a^k, c^k) - F(w^k, c^k)$  will approach a neighborhood of radius  $\epsilon$  as  $k \rightarrow \infty$ .*

*Proof* Let  $\bar{\epsilon} > 0$ . By the Hilbert space structure of  $H$ :

$$F(a, c) = \|V\|_H^2 - 2V(c)^T a + a^T K(c)a,$$

where  $V(c) = (V(c_1), \dots, V(c_L))^T$  and  $K(c) = (K(c_i, c_j))_{i,j=1}^L$  is the symmetric strictly positive kernel matrix corresponding to  $c$ . At each time iteration  $t^k$ ,  $k = 0, 1, 2, \dots$ , the corresponding centers and weights will be written as  $c^k \in \mathbb{R}^{nL}$  and  $a^k \in \mathbb{R}^L$ , respectively. The ideal weights corresponding to  $c^k$  will be denoted by  $w^k$ . It can be shown that  $w^k = K(c^k)^{-1}V(c^k)$  and  $F(w^k, c^k) = \|V\|_H^2 - V(c^k)^T K(c^k)V(c^k)$ . Theorem 7.3 ensures that the ideal weights change continuously with respect to the centers which remain in a compact set  $\tilde{D}^L$ , where  $\tilde{D} = \{x \in \mathbb{R}^L : \|x - D\| \leq \max_{i=1, \dots, L} (\sup_{x \in D} |d_i(x)|)\}$ , so the collection of ideal weights is bounded. Let  $R > \bar{\epsilon}$  be large enough so that  $B_R(0)$  contains both the initial value  $a^0$  and the set of ideal weights. To facilitate the subsequent analysis, consider the constants

$$\begin{aligned} R_0 &= \max_{x \in D, t > 0} |q(x, t)|, & R_1 &= \max_{a \in \overline{B_r(0)}, c \in \tilde{D}} |\nabla_a F(a, c)|, \\ R_2 &= \max_{c \in \tilde{D}} |\nabla_c F(w(c), c)|, & R_3 &= \max_{c \in \tilde{D}} |\dot{d}_i(x(t))|, \\ R_4 &= \max_{c \in \tilde{D}} \left\| \frac{d}{dc} w(c) \right\|, \end{aligned}$$

and let  $\Delta t < \tau^{-1} \triangleq \bar{\epsilon}(2(R_0 + R_3)(R_1 R_4 (R_0 + R_3) + R_2 + 1))^{-1}$ . The proof aims to show that by using the gradient descent law for choosing  $a^k$ , the inequality

$$\frac{F(a^{k+1}, c^{k+1}) - F(w^{k+1}, c^{k+1})}{F(a^k, c^k) - F(w^k, c^k)} < \delta + \frac{\bar{\epsilon}}{F(a^k, c^k) - F(w^k, c^k)}$$

can be achieved for some  $0 < \delta < 1$ . Set

$$a^{k+1} = a^k + \lambda g, \quad (7.7)$$

where  $g = -\nabla_a F(a^k, c^k) = 2V(c^k) - 2K(c^k)a^k$  and  $\lambda$  is selected so that the quantity  $F(a^k + \lambda g, c^k)$  is minimized. The  $\lambda$  that minimizes this quantity is  $\lambda = \left(\frac{g^T g}{2g^T K(c^k)g}\right)$  which yields  $F(a^{k+1}, c^k) = F(a^k, c^k) - \frac{(g^T g)^2}{4g^T K(c^k)g}$ . Since  $F(a^{k+1}, c^{k+1})$  is continuously differentiable in the second variable, we have  $F(a^{k+1}, c^{k+1}) = F(a^{k+1}, c^k) + \nabla_c F(a^{k+1}, \eta) \cdot (c^{k+1} - c^k)$ . Since  $|\dot{c}(x(t))| < R_0 + R_3$ , an application of the mean value theorem demonstrates that  $\|c^{k+1} - c^k\| < (R_0 + R_3)\Delta t$ . Thus

$$F(a^{k+1}, c^{k+1}) = F(a^{k+1}, c^k) + \epsilon_1(t^k),$$

where  $|\epsilon_1(t^k)| < \bar{\epsilon}/2, \forall k$ . The quantity  $F(w^{k+1}, c^{k+1})$  is continuously differentiable in both variables. Thus, by the multi-variable chain rule and another application of the mean value theorem

$$F(w^{k+1}, c^{k+1}) = F(w^k, c^k) + \epsilon_2(t^k),$$

for  $|\epsilon_2(t^k)| < \bar{\epsilon}/2 \forall k$ . Therefore, the following is established:

$$\begin{aligned} \frac{F(a^{k+1}, c^{k+1}) - F(w^{k+1}, c^{k+1})}{F(a^k, c^k) - F(w^k, c^k)} &= \frac{F(a^{k+1}, c^k) - F(w^k, c^k) + (\epsilon_1(t^k) - \epsilon_2(t^k))}{F(a^k, c^k) - F(w^k, c^k)} \\ &= 1 - \frac{(g^T g)^2}{(g^T K(c^k)g)(g^T K(c^k)^{-1}g)} + \frac{\epsilon_1(t^k) - \epsilon_2(t^k)}{F(a^k, c^k) - F(w^k, c^k)}. \end{aligned}$$

The Kantorovich inequality [19, p. 77] yields

$$1 - \frac{(g^T g)^2}{(g^T K(c^k)g)(g^T K(c^k)^{-1}g)} \leq \left( \frac{A_{c^k}/a_{c^k} - 1}{A_{c^k}/a_{c^k} + 1} \right)^2, \quad (7.8)$$

where  $A_{c^k}$  is the largest eigenvalue of  $K(c^k)$  and  $a_{c^k}$  is the smallest eigenvalue of  $K(c^k)$ . The quantity on the right of (7.8) is continuous with respect to  $A_{c^k}$  and  $a_{c^k}$ . In turn,  $A_{c^k}$  and  $a_{c^k}$  are continuous with respect to  $K(c^k)$  (c.f. Exercise 4.1.6 [20]) which is continuous with respect to  $c^k$ . Therefore, there is a largest value,  $\delta$ , that the right hand side of (7.8) obtains on the compact set  $\tilde{D}$  and this value is less than 1. Moreover,  $\delta$  is independent of  $\bar{\epsilon}$ , so it may be declared that  $\bar{\epsilon} = \epsilon(1 - \delta)$ . Finally,

$$\frac{F(a^{k+1}, c^{k+1}) - F(w^{k+1}, c^{k+1})}{F(a^k, c^k) - F(w^k, c^k)} \leq \delta + \frac{(\epsilon_1(t^k) - \epsilon_2(t^k))}{F(a^k, c^k) - F(w^k, c^k)}.$$

Therefore, setting  $e(k) = F(a^k, c^k) - F(w^k, c^k)$ , it can be shown that  $e(k+1) \leq \delta e(k) + \epsilon(1 - \delta)$  and the conclusion of the theorem follows.  $\square$

### 7.3.5 Simulation for the Gradient Chase Theorem

To demonstrate the effectiveness of the gradient chase theorem, a simulation is performed on a two-dimensional linear system is presented below. The system dynamics are given by

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

which is the dynamical system corresponding to a circular trajectory. The state dependent function to be approximated is

$$V(x_1, x_2) = x_1^2 + 5x_2^2 + \tanh(x_1 x_2), \quad (7.9)$$

where exponential kernels,  $K(x, y) = \exp(x^T y)$ , are used to approximate (7.9). The centers are arranged in an equilateral triangle centered about the state. In particular, each center resides on a circle of radius 0.1 centered at the state as

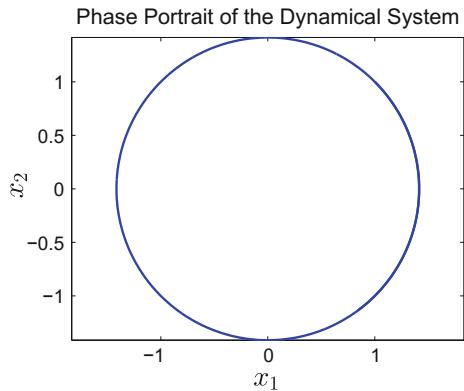
$$c_i(x) = x + 0.1 \begin{pmatrix} \sin((i-1)2\pi/3) \\ \cos((i-1)2\pi/3) \end{pmatrix}$$

for  $i = 1, 2, 3$ .

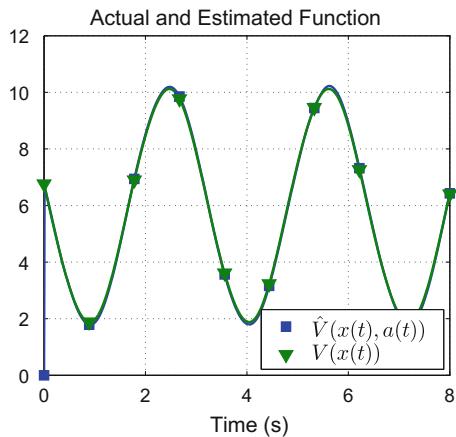
The initial values selected for the weights are  $a^0 = [0 \ 0 \ 0]^T$ . The gradient descent weight update law, given by (7.7), is applied 10 iterations per time-step and the time-steps incremented every 0.01 s. Figures 7.1, 7.2, 7.3 and 7.4 present the results of the simulation.

Figure 7.4 demonstrates that the function approximation error is driven to a small neighborhood of zero as the gradient chase theorem is implemented, which numerically validates the claim of the uniformly ultimately bounded result of Theorem 7.5. Approximations of the ideal weight function depicted in Fig. 7.3, are periodic and

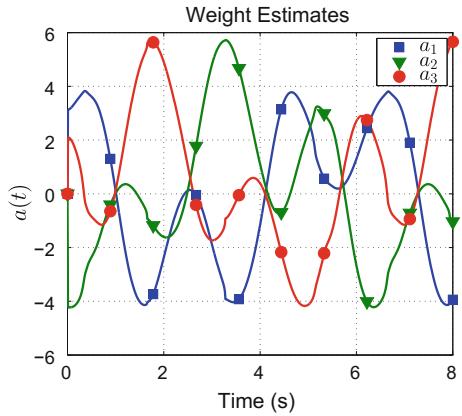
**Fig. 7.1** Trajectory of the state vector



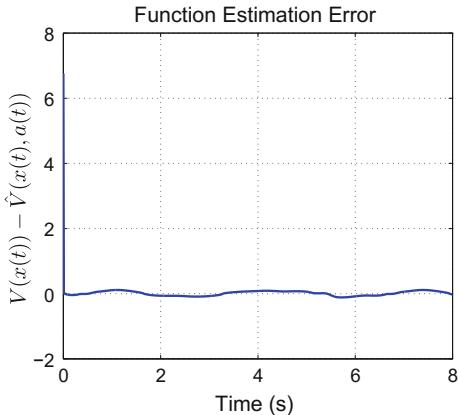
**Fig. 7.2** Comparison of  $V$  and the approximation  $\hat{V}$



**Fig. 7.3** The values of the weight function estimates



**Fig. 7.4** Error committed by the approximation



smooth. Smoothness of the ideal weight function itself is given in Theorem 7.3, and the periodicity of the approximation follows from the periodicity of the selected dynamical system, Fig. 7.1. Finally, Fig. 7.2 shows that along the system trajectory, the approximation  $\hat{V}$  rapidly converges to the true function  $V$ . Approximation of the function is maintained as the system state moves through its domain as anticipated.

## 7.4 Local Approximation for Efficient Model-Based Reinforcement Learning<sup>2</sup>

In the following, Sect. 7.4.1 summarizes key results from the previous section in the context of model-based reinforcement learning. In Sect. 7.4.2, the StaF-based function approximation approach is used to approximately solve an optimal regulation problem online using exact model knowledge via value function approximation. Section 7.4.3 is dedicated to Lyapunov-based stability analysis of the developed technique. Section 7.4.4 extends the developed technique to systems with uncertain drift dynamics and Sect. 7.4.5 presents comparative simulation results.

### 7.4.1 StaF Kernel Functions

Let  $H$  be a universal reproducing kernel Hilbert space over a compact set  $\chi \subset \mathbb{R}^n$  with a continuously differentiable positive definite kernel  $K : \chi \times \chi \rightarrow \mathbb{R}$ . Let  $\bar{V}^* : \chi \rightarrow \mathbb{R}$  be a function such that  $\bar{V}^* \in H$ . Let  $C \triangleq [c_1, c_2, \dots, c_L]^T \in \chi^L$  be a set of distinct centers, and let  $\sigma : \chi \times \chi^L \rightarrow \mathbb{R}^L$  be defined as  $\sigma(x, C) \triangleq$

---

<sup>2</sup>Parts of the text in this section are reproduced, with permission, from [21], ©2016, Elsevier.

$[K(x, c_1), \dots, K(x, c_L)]^T$ . Then, there exists a unique set of weights  $W_H$  such that

$$W_H(C) = \arg \min_{a \in \mathbb{R}^L} \|a^T \sigma(\cdot, C) - \bar{V}^*\|_H, \quad (7.10)$$

where  $\|\cdot\|_H$  denotes the Hilbert space norm. Furthermore, for any given  $\epsilon > 0$ , there exists a constant  $L \in \mathbb{N}$ , a set of centers,  $C \in \chi^L$ , and a set of weights,  $W_H \in \mathbb{R}^L$ , such that  $\|W_H^T \sigma(\cdot, C) - \bar{V}^*\|_H \leq \epsilon$ . On compact sets, the Hilbert space norm corresponding to a Hilbert space with continuously differentiable kernels dominates the supremum norm of functions and their derivatives [7, Corollary 4.36]. Hence, the function can be approximated as well as its derivative, that is, there exists centers and weights for which,  $\|W_H^T \sigma(\cdot, C) - \bar{V}^*\|_{\chi, \infty} < \epsilon$  and  $\|W_H^T \nabla \sigma(\cdot, C) - \nabla \bar{V}^*\|_{\chi, \infty} < \epsilon$ . The notation  $\nabla f$  denotes the gradient of  $f$  with respect to the first argument and the notation  $\|f\|_{A, \infty}$  denotes the supremum of the absolute value (or the pointwise norm, if  $f$  is vector-valued) of  $f$  over the set  $A$ .

Let  $H_{x,r}$  denote the restriction of the Hilbert space  $H$  to  $\overline{B_r(x)} \subset \chi$ . Then,  $H_{x,r}$  is a Hilbert space with the restricted kernel  $K_{x,r} : B_r(x) \times B_r(x) \rightarrow \mathbb{R}$  defined as  $K_{x,r}(y, z) = K(y, z)$ ,  $\forall (y, z) \in B_r(x) \times B_r(x)$ . The Weierstrass Theorem indicates that as  $r$  decreases, the degree  $N_{x,\epsilon}$  of the polynomial needed to achieve the same error  $\epsilon$  over  $B_r(x)$  decreases [22]. Hence, by Theorem 7.4, approximation of a function over a smaller domain requires a smaller number of exponential kernels. Furthermore, provided the region of interest is small enough, the number of kernels required to approximate continuous functions with arbitrary accuracy can be reduced to  $\binom{n+2}{2}$ .

In the StaF approach, the centers are selected to follow the current state  $x$  (i.e., the locations of the centers are defined as a function of the system state). Since the system state evolves in time, the ideal weights are not constant. To approximate the ideal weights using gradient-based algorithms, it is essential that the weights change smoothly with respect to the system state. Theorem 7.3 establishes differentiability of the ideal weights as a function of the centers to facilitate implementation of gradient-based update laws to learn the time-varying ideal weights in real-time.

#### 7.4.2 StaF Kernel Functions for Online Approximate Optimal Control

Consider the Bolza problem introduced in Sect. 1.5 where the functions  $f$  and  $g$  are assumed to be known and locally Lipschitz continuous. Furthermore, assume that  $f(0) = 0$  and that  $\nabla f : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$  is continuous. The selection of an optimal regulation problem and the assumption that the system dynamics are known are motivated by ease of exposition. Using the concurrent learning-based adaptive system identifier and the state augmentation technique described in Sects. 3.3 and 4.4, the approach developed in this section can be extended to a class of trajectory tracking

problems in the presence of uncertainties in the system drift dynamics. For a detailed description of StaF-based online approximate optimal control under uncertainty, see Sect. 7.4.4. Simulation results in Sect. 7.4.5 demonstrate the performance of such an extension.

The expression for the optimal policy in (1.13) indicates that to compute the optimal action when the system is at any given state  $x$ , one only needs to evaluate the gradient  $\nabla V^*$  at  $x$ . Hence, to compute the optimal policy at  $x$ , one only needs to approximate the value function over a small neighborhood around  $x$ . Furthermore, as established in Theorem 7.4, the number of basis functions required to approximate the value function is smaller if the region for the approximation is smaller (with respect to the ordering induced by set containment). Hence, in this result, the aim is to obtain a uniform approximation of the value function over a small neighborhood around the current system state.

StaF kernels are employed to achieve the aforementioned objective. To facilitate the development, let  $x$  be in the interior of  $\chi$ . Then, for all  $\epsilon > 0$ , there exists a function  $\bar{V}^* \in H_{x,r}$  such that  $\sup_{y \in \overline{B_r(x)}} |V^*(y) - \bar{V}^*(y)| < \epsilon$ , where  $H_{x,r}$  is a restriction of a universal reproducing kernel Hilbert space,  $H$ , introduced in Sect. 7.4.1, to  $\overline{B_r(x)}$ . In the developed StaF-based method, a small compact set  $\overline{B_r(x)}$  around the current state  $x$  is selected for value function approximation by selecting the centers  $C \in B_r(x)$  such that  $C = c(x)$  for some continuously differentiable function  $c : \chi \rightarrow \chi^L$ . Using StaF kernels centered at a point  $x$ , the value function can be represented as

$$V^*(y) = W(x)^T \sigma(y, c(x)) + \varepsilon(x, y), \quad y \in \overline{B_r(x)}$$

where  $\varepsilon(x, y)$  denotes the function approximation error.

Since the centers of the kernel functions change as the system state changes, the ideal weights also change as the system state changes. The state-dependent nature of the ideal weights differentiates this approach from state-of-the-art approximate dynamic programming methods in the sense that the stability analysis needs to account for changing ideal weights. Based on Theorem 7.3, it can be established that the ideal weight function  $W : \chi \rightarrow \mathbb{R}^L$  defined as  $W(x) \triangleq W_{H_{x,r}}(c(x))$ , where  $W_{H_{x,r}}$  was introduced in (7.10), is continuously differentiable, provided the functions  $\sigma$  and  $c$  are continuously differentiable.

The approximate value function  $\hat{V} : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^L \rightarrow \mathbb{R}$  and the approximate policy  $\hat{u} : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^L \rightarrow \mathbb{R}^m$ , evaluated at a point  $y \in \overline{B_r(x)}$ , using StaF kernels centered at  $x$ , can then be expressed as

$$\begin{aligned} \hat{V}(y, x, \hat{W}_c) &\triangleq \hat{W}_c^T \sigma(y, c(x)), \\ \hat{u}(y, x, \hat{W}_a) &\triangleq -\frac{1}{2} R^{-1} g^T(y) \nabla \sigma(y, c(x))^T \hat{W}_a, \end{aligned} \quad (7.11)$$

where  $\sigma$  denotes the vector of basis functions, introduced in Sect. 7.4.1.

The objective of the critic is to learn the ideal parameters  $W(x)$ , and the objective of the actor is to implement a stabilizing controller based on the parameters learned by the critic. Motivated by the stability analysis, the actor and the critic maintain separate estimates  $\hat{W}_a$  and  $\hat{W}_c$ , respectively, of the ideal parameters  $W(x)$ . Using the estimates  $\hat{V}$  and  $\hat{u}$  for  $V^*$  and  $u^*$ , respectively, the Bellman error is  $\delta : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^L \times \mathbb{R}^L \rightarrow \mathbb{R}$ , is computed as

$$\delta(y, x, \hat{W}_c, \hat{W}_a) \triangleq r(y, \hat{u}(y, x, \hat{W}_a)) + \nabla \hat{V}(y, x, \hat{W}_c)(f(y) + g(y) \hat{u}(y, x, \hat{W}_a)). \quad (7.12)$$

To solve the optimal control problem, the critic aims to find a set of parameters  $\hat{W}_c$  and the actor aims to find a set of parameters  $\hat{W}_a$  such that  $\delta(y, x, \hat{W}_c, \hat{W}_a) = 0, \forall x \in \mathbb{R}^n, \forall y \in \overline{B_r(x)}$ . Since an exact basis for value function approximation is generally not available, an approximate set of parameters that minimizes the Bellman error is sought.

To learn the ideal parameters online, the critic evaluates a form  $\delta_t : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  of the Bellman error at each time instance  $t$  as

$$\delta_t(t) \triangleq \delta(x(t), x(t), \hat{W}_c(t), \hat{W}_a(t)), \quad (7.13)$$

where  $\hat{W}_a(t)$  and  $\hat{W}_c(t)$  denote the estimates of the actor and the critic weights, respectively, at time  $t$ , and the notation  $x(t)$  is used to denote the state the system in (1.9), at time  $t$ , when starting from initial time  $t_0$ , initial state  $x_0$ , and under the feedback controller

$$u(t) = \hat{u}(x(t), x(t), \hat{W}_a(t)). \quad (7.14)$$

Since (1.14) constitutes a necessary and sufficient condition for optimality, the Bellman error serves as an indirect measure of how close the critic parameter estimates  $\hat{W}_c$  are to their ideal values; hence, in the context of reinforcement learning, each evaluation of the Bellman error is interpreted as gained experience. Since the Bellman error in (7.13) is evaluated along the system trajectory, the experience gained is along the system trajectory.

Learning based on simulation of experience is achieved by extrapolating the Bellman error to unexplored areas of the state-space. The critic selects a set of functions  $\{x_i : \mathbb{R}^n \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}^n\}_{i=1}^N$  such that each  $x_i$  maps the current state  $x(t)$  to a point  $x_i(x(t), t) \in B_r(x(t))$ . The critic then evaluates a form  $\delta_{ti} : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  of the Bellman error for each  $x_i$  as

$$\delta_{ti}(t) = \delta(x_i(x(t), t), x(t), \hat{W}_c(t), \hat{W}_a(t)). \quad (7.15)$$

The critic then uses the Bellman errors from (7.13) and (7.15) to improve the estimate  $\hat{W}_c(t)$  using the recursive least-squares-based update law

$$\dot{\hat{W}}_c(t) = -k_{c1}\Gamma(t) \frac{\omega(t)}{\rho(t)} \delta_t(t) - \frac{k_{c2}}{N} \Gamma(t) \sum_{i=1}^N \frac{\omega_i(t)}{\rho_i(t)} \delta_{ti}(t), \quad (7.16)$$

where  $\rho_i(t) \triangleq \sqrt{1 + \gamma_1 \omega_i^T(t) \omega_i(t)}$ ,  $\rho(t) \triangleq \sqrt{1 + \gamma_1 \omega^T(t) \omega(t)}$ ,

$$\begin{aligned} \omega(t) &\triangleq \nabla \sigma(x(t), c(x(t))) f(x(t)) + \nabla \sigma(x(t), c(x(t))) g(x(t)) \hat{u}(x(t), x(t), \hat{W}_a(t)), \\ \omega_i(t) &\triangleq \nabla \sigma(x_i(x(t)), c(x(t))) g(x_i(x(t), t)) \hat{u}(x_i(x(t), t), x(t), \hat{W}_a(t)) \\ &\quad + \nabla \sigma(x_i(x(t)), c(x(t))) f(x_i(x(t), t)), \end{aligned}$$

and  $k_{c1}, k_{c2}, \gamma_1 \in \mathbb{R}_{>0}$  are constant learning gains. In (7.16),  $\Gamma(t)$  denotes the least-squares learning gain matrix updated according to

$$\begin{aligned} \dot{\Gamma}(t) &= \beta \Gamma(t) - k_{c1} \Gamma(t) \frac{\omega(t) \omega^T(t)}{\rho^2(t)} \Gamma(t) - \frac{k_{c2}}{N} \Gamma(t) \sum_{i=1}^N \frac{\omega_i(t) \omega_i^T(t)}{\rho_i^2(t)} \Gamma(t), \\ \Gamma(t_0) &= \Gamma_0, \end{aligned} \quad (7.17)$$

where  $\beta \in \mathbb{R}_{>0}$  is a constant forgetting factor. Motivated by a Lyapunov-based stability analysis, the update law for the actor is designed as

$$\begin{aligned} \dot{\hat{W}}_a(t) &= -k_{a1} (\hat{W}_a(t) - \hat{W}_c(t)) - k_{a2} \hat{W}_a(t) + \frac{k_{c1} G_\sigma^T(t) \hat{W}_a(t) \omega(t)^T}{4\rho(t)} \hat{W}_c(t) \\ &\quad + \sum_{i=1}^N \frac{k_{c2} G_{\sigma i}^T(t) \hat{W}_a(t) \omega_i^T(t)}{4N\rho_i(t)} \hat{W}_c(t), \end{aligned} \quad (7.18)$$

where

$$\begin{aligned} G_\sigma(t) &\triangleq \nabla \sigma(x(t), c(x(t))) g(x(t)) R^{-1} g^T(x(t)) \nabla \sigma^T(x(t), c(x(t))), \\ G_{\sigma i}(t) &\triangleq \nabla \sigma(x_i(x(t), t), c(x(t))) g(x_i(x(t), t)) R^{-1} g^T(x_i(x(t), t)) \\ &\quad \cdot \nabla \sigma^T(x_i(x(t), t), c(x(t))), \end{aligned}$$

and  $k_{a1}, k_{a2} \in \mathbb{R}_{>0}$  are learning gains.

### 7.4.3 Analysis

The computational cost associated with the implementation of the developed method can be computed as  $O(N(L^3 + mnL + Lm^2 + n^2 + m^2))$ . Since local approximation is targeted, the StaF kernels result in a reduction in the number of required basis functions (i.e.,  $L$ ). Since the computational cost has a cubic relationship with the number of basis functions, the StaF methodology results in a significant computa-

tional benefit. The computational cost grows linearly with the number of extrapolation points (i.e.,  $N$ ). If the points are selected using grid-based methods employed in results such as [11], the number  $N$  increases geometrically with respect to the state dimension,  $n$ . On the other hand, if the extrapolation points are selected to be time varying, then even a single point is sufficient, provided the time-trajectory of the point contains enough information to satisfy the subsequent Assumption 7.6.

In the following, Assumption 7.6 formalizes the conditions under which the trajectories of the closed-loop system can be shown to be ultimately bounded, and Lemma 7.7 facilitates the analysis of the closed-loop system when time-varying extrapolation trajectories are utilized.

For notational brevity, time-dependence of all the signals is suppressed hereafter. Let  $\chi$  denote the projection of  $B_\zeta$  onto  $\mathbb{R}^n$ . To facilitate the subsequent stability analysis, the Bellman errors in (7.13) and (7.15) are expressed in terms of the weight estimation errors  $\tilde{W}_c \triangleq W - \hat{W}_c$  and  $\tilde{W}_a = W - \hat{W}_a$  as

$$\begin{aligned}\delta_t &= -\omega^T \tilde{W}_c + \frac{1}{4} \tilde{W}_a G_\sigma \tilde{W}_a + \Delta(x), \\ \delta_{ti} &= -\omega_i^T \tilde{W}_c + \frac{1}{4} \tilde{W}_a^T G_{\sigma i} \tilde{W}_a + \Delta_i(x),\end{aligned}\quad (7.19)$$

where the functions  $\Delta, \Delta_i : \mathbb{R}^n \rightarrow \mathbb{R}$  are uniformly bounded over  $\chi$  such that the bounds  $\|\Delta\|$  and  $\|\Delta_i\|$  decrease with decreasing  $\|\nabla \varepsilon\|$  and  $\|\nabla W\|$ . Let a candidate Lyapunov function  $V_L : \mathbb{R}^{n+2L} \times \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  be defined as

$$V_L(Z, t) \triangleq V^*(x) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1}(t) \tilde{W}_c + \frac{1}{2} \tilde{W}_a^T \tilde{W}_a,$$

where  $V^*$  is the optimal value function, and

$$Z = \left[ x^T, \tilde{W}_c^T, \tilde{W}_a^T \right]^T.$$

To facilitate learning, the system states  $x$  and the selected functions  $x_i$  are assumed to satisfy the following.

**Assumption 7.6** There exist constants  $T \in \mathbb{R}_{>0}$  and  $c_1, c_2, c_3 \in \mathbb{R}_{\geq 0}$ , such that

$$\begin{aligned}c_1 I_L &\leq \int_t^{t+T} \left( \frac{\omega(\tau) \omega^T(\tau)}{\rho^2(\tau)} \right) d\tau, \forall t \in \mathbb{R}_{\geq t_0}, \\ c_2 I_L &\leq \inf_{t \in \mathbb{R}_{\geq t_0}} \left( \frac{1}{N} \sum_{i=1}^N \frac{\omega_i(t) \omega_i^T(t)}{\rho_i^2(t)} \right), \\ c_3 I_L &\leq \frac{1}{N} \int_t^{t+T} \left( \sum_{i=1}^N \frac{\omega_i(\tau) \omega_i^T(\tau)}{\rho_i^2(\tau)} \right) d\tau, \forall t \in \mathbb{R}_{\geq t_0},\end{aligned}$$

where at least one of the constants  $c_1, c_2$ , and  $c_3$  is strictly positive.

Unlike typical approximate dynamic programming literature that assumes  $\omega$  is persistently exciting, Assumption 7.6 only requires either the regressor  $\omega$  or the regressor  $\omega_i$  to be persistently exciting. The regressor  $\omega$  is completely determined by the system state  $x$ , and the weights  $\hat{W}_a$ . Hence, excitation in  $\omega$  vanishes as the system states and the weights converge. Hence, in general, it is unlikely that  $\underline{c}_1 > 0$ . However, the regressor  $\omega_i$  depends on  $x_i$ , which can be designed independent of the system state  $x$ . Hence,  $\underline{c}_3$  can be made strictly positive if the signal  $x_i$  contains enough frequencies, and  $\underline{c}_2$  can be made strictly positive by selecting a sufficient number of extrapolation functions.

Intuitively, selection of a single time-varying Bellman error extrapolation function results in virtual excitation. That is, instead of using input-output data from a persistently excited system, the dynamic model is used to simulate persistent excitation to facilitate parameter convergence. The performance of the developed extrapolation method is demonstrated using comparative simulations in Sect. 7.4.5, where it is demonstrated that the developed method using a single time-varying extrapolation point results in improved computational efficiency when compared to a large number of fixed extrapolation functions.

The following lemma facilitates the stability analysis by establishing upper and lower bounds on the eigenvalues of the least-squares learning gain matrix,  $\Gamma$ .

**Lemma 7.7** *Provided Assumption 7.6 holds and  $\lambda_{\min}\{\Gamma_0^{-1}\} > 0$ , the update law in (7.17) ensures that the least-squares gain matrix satisfies*

$$\underline{\Gamma} \mathbf{I}_L \leq \Gamma(t) \leq \overline{\Gamma} \mathbf{I}_L, \quad (7.20)$$

where

$$\begin{aligned} \overline{\Gamma} &= \frac{1}{\min\{k_{c1}\underline{c}_1 + k_{c2}\max\{\underline{c}_2 T, \underline{c}_3\}, \lambda_{\min}\{\Gamma_0^{-1}\}\} e^{-\beta T}}, \\ \underline{\Gamma} &= \frac{1}{\lambda_{\max}\{\Gamma_0^{-1}\} + \frac{(k_{c1}+k_{c2})}{\beta \gamma_1}}. \end{aligned}$$

Furthermore,  $\overline{\Gamma} > 0$ .

*Proof* The proof closely follows the proof of [23, Corollary 4.3.2]. The update law in (7.17) implies that

$$\frac{d}{dt} \Gamma^{-1}(t) = -\beta \Gamma^{-1}(t) + k_{c1} \frac{\omega(t) \omega^T(t)}{\rho^2(t)} + \frac{k_{c2}}{N} \sum_{i=1}^N \frac{\omega_i(t) \omega_i^T(t)}{\rho_i^2(t)}.$$

Hence,

$$\begin{aligned} \Gamma^{-1}(t) &= k_{c1} \int_0^t e^{-\beta(t-\tau)} \frac{\omega(\tau) \omega^T(\tau)}{\rho^2(\tau)} d\tau + \frac{k_{c2}}{N} \int_0^t e^{-\beta(t-\tau)} \sum_{i=1}^N \frac{\omega_i(\tau) \omega_i^T(\tau)}{\rho_i^2(\tau)} d\tau \\ &\quad + e^{-\beta t} \Gamma_0^{-1}. \end{aligned}$$

To facilitate the proof, let  $t < T$ . Then,

$$\Gamma^{-1}(t) \geq e^{-\beta t} \Gamma_0^{-1} \geq e^{-\beta T} \Gamma_0^{-1} \geq \lambda_{\min}\{\Gamma_0^{-1}\} e^{-\beta T} I_L.$$

Since the integrands are positive, it follows that if  $t \geq T$ , then  $\Gamma^{-1}$  can be bounded as

$$\Gamma^{-1}(t) \geq k_{c1} \int_{t-T}^t e^{-\beta(t-\tau)} \frac{\omega(\tau) \omega^T(\tau)}{\rho^2(\tau)} d\tau + \frac{k_{c2}}{N} \int_{t-T}^t e^{-\beta(t-\tau)} \sum_{i=1}^N \frac{\omega_i(\tau) \omega_i^T(\tau)}{\rho_i^2(\tau)} d\tau.$$

Therefore,

$$\Gamma^{-1}(t) \geq k_{c1} e^{-\beta T} \int_{t-T}^t \frac{\omega(\tau) \omega^T(\tau)}{\rho^2(\tau)} d\tau + \frac{k_{c2}}{N} e^{-\beta T} \int_{t-T}^t \sum_{i=1}^N \frac{\omega_i(\tau) \omega_i^T(\tau)}{\rho_i^2(\tau)} d\tau.$$

Using Assumption 7.6,

$$\frac{1}{N} \int_{t-T}^t \sum_{i=1}^N \frac{\omega_i(\tau) \omega_i^T(\tau)}{\rho_i^2(\tau)} d\tau \geq \max\{\underline{c}_2 T, \underline{c}_3\} I_L, \quad \int_{t-T}^t \frac{\omega(\tau) \omega^T(\tau)}{\rho^2(\tau)} d\tau \geq \underline{c}_1 I_L.$$

A lower bound for  $\Gamma^{-1}$  is thus obtained as,

$$\Gamma^{-1}(t) \geq \min\left\{k_{c1}\underline{c}_1 + k_{c2} \max\{\underline{c}_2 T, \underline{c}_3\}, \lambda_{\min}\{\Gamma_0^{-1}\}\right\} e^{-\beta T} I_L. \quad (7.21)$$

Provided Assumption 7.6 holds, the lower bound in (7.21) is strictly positive. Furthermore, using the facts that  $\frac{\omega(t)\omega^T(t)}{\rho^2(t)} \leq \frac{1}{\gamma_1}$  and  $\frac{\omega_i(t)\omega_i^T(t)}{\rho_i^2(t)} \leq \frac{1}{\gamma_1}$ ,  $\forall t \in \mathbb{R}_{\geq t_0}$ ,

$$\begin{aligned} \Gamma^{-1}(t) &\leq \int_0^t e^{-\beta(t-\tau)} \left( k_{c1} \frac{1}{\gamma_1} + \frac{k_{c2}}{N} \sum_{i=1}^N \frac{1}{\gamma_1} \right) I_L d\tau + e^{-\beta t} \Gamma_0^{-1} \\ &\leq \left( \lambda_{\max}\{\Gamma_0^{-1}\} + \frac{(k_{c1} + k_{c2})}{\beta \gamma_1} \right) I_L. \end{aligned}$$

Since the inverse of the lower and upper bounds on  $\Gamma^{-1}$  are the upper and lower bounds on  $\Gamma$ , respectively, the proof is complete.  $\square$

Since the optimal value function is positive definite, (7.20) and [24, Lemma 4.3] can be used to show that the candidate Lyapunov function satisfies the following bounds

$$\underline{v}_l(\|Z\|) \leq V_L(Z, t) \leq \bar{v}_l(\|Z\|), \quad (7.22)$$

$\forall t \in \mathbb{R}_{\geq 0}$  and  $\forall Z \in \mathbb{R}^{2+2L}$ . In (7.22),  $v_l, \bar{v}_l : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions. To facilitate the analysis, let  $\underline{c} \in \mathbb{R}_{>0}$  be a constant defined as

$$\underline{c} \triangleq \frac{\beta}{2\Gamma k_{c2}} + \frac{c_2}{2}, \quad (7.23)$$

and let  $\iota \in \mathbb{R}_{>0}$  be a constant defined as

$$\begin{aligned} \iota &\triangleq \frac{3 \left( \frac{(k_{c1}+k_{c2})\|\Delta\|}{\sqrt{\nu}} + \frac{\|\nabla Wf\|}{\Gamma} + \frac{\|\Gamma^{-1}G_{W\sigma}W\|}{2} \right)^2}{4k_{c2}\underline{c}} + \frac{1}{2}\frac{\|G_{VW\sigma}\|}{\|G_{V\epsilon}\|} + \frac{1}{2}\frac{\|G_{V\epsilon}\|}{\|G_{VW}\|} \\ &+ \frac{\left( \frac{\|G_{W\sigma}W\| + \|G_{V\sigma}\|}{2} + k_{a2}\|W\| + \|\nabla Wf\| + \frac{(k_{c1}+k_{c2})\|G_\sigma\|\|W\|^2}{4\sqrt{\nu}} \right)^2}{(k_{a1} + k_{a2})}, \end{aligned}$$

where  $G_{W\sigma} \triangleq \nabla W G \nabla \sigma^T$ ,  $G_{V\sigma} \triangleq \nabla V^* G \nabla \sigma^T$ ,  $G_{VW} \triangleq \nabla V^* G \nabla W^T$ , and  $G_{V\epsilon} \triangleq \nabla V^* G \nabla \epsilon^T$ . Let  $v_l : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  be a class  $\mathcal{K}$  function such that

$$v_l(\|Z\|) \leq \frac{Q(x)}{2} + \frac{k_{c2}\underline{c}}{6} \left\| \tilde{W}_c \right\|^2 + \frac{(k_{a1} + k_{a2})}{8} \left\| \tilde{W}_a \right\|^2.$$

The sufficient conditions for the subsequent Lyapunov-based stability analysis are given by

$$\frac{k_{c2}\underline{c}}{3} \geq \frac{\left( \frac{\|G_{W\sigma}\|}{2\Gamma} + \frac{(k_{c1}+k_{c2})\|W^T G_\sigma\|}{4\sqrt{\nu}} + k_{a1} \right)^2}{(k_{a1} + k_{a2})}, \quad (7.24)$$

$$\frac{(k_{a1} + k_{a2})}{4} \geq \left( \frac{\|G_{W\sigma}\|}{2} + \frac{(k_{c1}+k_{c2})\|W\|\|G_\sigma\|}{4\sqrt{\nu}} \right), \quad (7.25)$$

$$v_l^{-1}(\iota) < \bar{v}_l^{-1}(v_l(\zeta)). \quad (7.26)$$

The sufficient condition in (7.24) can be satisfied provided the points for Bellman error extrapolation are selected such that the minimum eigenvalue  $\underline{c}$ , introduced in (7.23) is large enough. The sufficient condition in (7.25) can be satisfied without affecting (7.24) by increasing the gain  $k_{a2}$ . The sufficient condition in (7.26) can be satisfied provided  $\underline{c}$ ,  $k_{a2}$ , and the state penalty  $Q(x)$  are selected to be sufficiently large and the StaF kernels for value function approximation are selected such that  $\|\nabla W\|$ ,  $\|\epsilon\|$ , and  $\|\nabla \epsilon\|$  are sufficiently small.

Similar to neural network-based approximation methods such as [25–32], the function approximation error,  $\epsilon$ , is unknown, and in general, infeasible to compute for a given function, since the ideal neural network weights are unknown. Since a bound on  $\epsilon$  is unavailable, the gain conditions in (7.24)–(7.26) cannot be formally verified. However, they can be met using trial and error by increasing the gain  $k_{a2}$ ,

the number of StaF basis functions, and  $\underline{c}$  by selecting more points to extrapolate the Bellman error.

To improve computational efficiency, the size of the domain around the current state where the StaF kernels provide good approximation of the value function is desired to be small. Smaller approximation domain results in almost identical extrapolated points, which in turn, results in smaller  $\underline{c}$ . Hence, the approximation domain cannot be selected to be arbitrarily small and needs to be large enough to meet the sufficient conditions in (7.24)–(7.26).

**Theorem 7.8** *Provided Assumption 7.6 holds and the sufficient gain conditions in (7.24)–(7.26) are satisfied, the controller in (7.14) and the update laws in (7.16)–(7.18) ensure that the state  $x$  and the weight estimation errors  $\tilde{W}_c$  and  $\tilde{W}_a$  are ultimately bounded.*

*Proof* The time-derivative of the Lyapunov function is given by

$$\dot{V}_L = \dot{V}^* + \tilde{W}_c^T \Gamma^{-1} (\dot{W} - \dot{\tilde{W}}_c) + \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \tilde{W}_c + \tilde{W}_a^T (\dot{W} - \dot{\tilde{W}}_a).$$

Using Theorem 7.3, the time derivative of the ideal weights can be expressed as

$$\dot{W} = \nabla W(x) (f(x) + g(x) u). \quad (7.27)$$

Using (7.16)–(7.19) and (7.27), the time derivative of the Lyapunov function is expressed as

$$\begin{aligned} \dot{V}_L &= \nabla V^*(x) (f(x) + g(x) u) + \tilde{W}_c^T \Gamma^{-1} \nabla W(x) (f(x) + g(x) u) \\ &\quad - \tilde{W}_c^T \Gamma^{-1} \left( -k_{c1} \Gamma \frac{\omega}{\rho} \left( -\omega^T \tilde{W}_c + \frac{1}{4} \tilde{W}_a G_\sigma \tilde{W}_a + \Delta(x) \right) \right) \\ &\quad - \tilde{W}_c^T \Gamma^{-1} \left( -\frac{k_{c2}}{N} \Gamma \sum_{i=1}^N \frac{\omega_i}{\rho_i} \frac{1}{4} \tilde{W}_a^T G_{\sigma i} \tilde{W}_a \right) \\ &\quad - \tilde{W}_c^T \Gamma^{-1} \left( -\frac{k_{c2}}{N} \Gamma \sum_{i=1}^N \frac{\omega_i}{\rho_i} \left( -\omega_i^T \tilde{W}_c + \Delta_i(x) \right) \right) \\ &\quad - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \left( \beta \Gamma - k_{c1} \Gamma \frac{\omega \omega^T}{\rho} \Gamma \right) \Gamma^{-1} \tilde{W}_c \\ &\quad - \frac{1}{2} \tilde{W}_c^T \Gamma^{-1} \left( -\frac{k_{c2}}{N} \Gamma \sum_{i=1}^N \frac{\omega_i \omega_i^T}{\rho_i} \Gamma \right) \Gamma^{-1} \tilde{W}_c \\ &\quad + \tilde{W}_a^T \left( \nabla W(x) (f(x) + g(x) u) - \dot{\tilde{W}}_a \right). \end{aligned}$$

Provided the sufficient conditions in (7.24)–(7.26) hold, the time derivative of the candidate Lyapunov function can be bounded as

$$\dot{V}_L \leq -v_l(\|Z\|), \quad \forall \zeta > \|Z\| > v_l^{-1}(\iota). \quad (7.28)$$

Using (7.22), (7.26), and (7.28), [24, Theorem 4.18] can be invoked to conclude that  $Z$  is ultimately bounded, in the sense that

$$\limsup_{t \rightarrow \infty} \|Z(t)\| \leq \underline{v}_l^{-1}(\bar{v}_l(v_l^{-1}(\iota))).$$

Since  $Z(\cdot) \in \mathcal{L}_\infty$ ,  $x(\cdot) \in \mathcal{L}_\infty$ ,  $\tilde{W}_a(\cdot) \in \mathcal{L}_\infty$ , and  $\tilde{W}_c(\cdot) \in \mathcal{L}_\infty$ . Since  $x(\cdot) \in \mathcal{L}_\infty$  and  $W \in C^0(\chi, \mathbb{R}^L)$ ,  $t \mapsto W(x(t)) \in \mathcal{L}_\infty$ . Hence,  $\hat{W}_a(\cdot) \in \mathcal{L}_\infty$  and  $\hat{W}_c(\cdot) \in \mathcal{L}_\infty$ , which implies  $u(\cdot) \in \mathcal{L}_\infty$ .  $\square$

#### 7.4.4 Extension to Systems with Uncertain Drift Dynamics

If the drift dynamics are uncertain, a parametric approximation of the dynamics can be employed for Bellman error extrapolation. On any compact set  $\mathcal{C} \subset \mathbb{R}^n$  the function  $f$  can be represented using a neural network as  $f(x) = \theta^T \sigma_f(Y^T x_1(x)) + \varepsilon_\theta(x)$ , where  $x_1(x) \triangleq [1, x^T]^T \in \mathbb{R}^{n+1}$ ,  $\theta \in \mathbb{R}^{p+1 \times n}$ , and  $Y \in \mathbb{R}^{n+1 \times p}$  denote the constant unknown output-layer and hidden-layer neural network weights,  $\sigma_f : \mathbb{R}^p \rightarrow \mathbb{R}^{p+1}$  denotes a bounded neural network basis function,  $\varepsilon_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^n$  denotes the function reconstruction error, and  $p \in \mathbb{N}$  denotes the number of neural network neurons. Using the universal function approximation property of single layer neural networks, given a constant matrix  $Y$  such that the rows of  $\sigma_f(Y^T x_1)$  form a proper basis (cf. [33]), there exist constant ideal weights  $\theta$  and known constants  $\bar{\theta}$ ,  $\bar{\varepsilon}_\theta$ , and  $\bar{\varepsilon}'_\theta \in \mathbb{R}$  such that  $\|\theta\| \leq \bar{\theta} < \infty$ ,  $\sup_{x \in \mathcal{C}} \|\varepsilon_\theta(x)\| \leq \bar{\varepsilon}_\theta$ , and  $\sup_{x \in \mathcal{C}} \|\nabla_x \varepsilon_\theta(x)\| \leq \bar{\varepsilon}'_\theta$ . Using an estimate  $\hat{\theta} \in \mathbb{R}^{p+1 \times n}$  of the weight matrix  $\theta$ , the function  $f$  can be approximated by the function  $\hat{f} : \mathbb{R}^n \times \mathbb{R}^{p+1 \times n} \rightarrow \mathbb{R}^n$  defined as  $\hat{f}(x, \hat{\theta}) \triangleq \hat{\theta}^T \sigma_\theta(x)$ , where  $\sigma_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^{p+1}$  is defined as  $\sigma_\theta(x) \triangleq \sigma_f(Y^T [1, x^T]^T)$ . Using  $\hat{f}$ , the Bellman error in (7.12) can be approximated by  $\hat{\delta} : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^L \times \mathbb{R}^L \times \mathbb{R}^{p+1 \times n} \rightarrow \mathbb{R}^n$  as

$$\hat{\delta}(y, x, \hat{W}_c, \hat{W}_a, \hat{\theta}) \triangleq r(y, \hat{u}(y, x, \hat{W}_a)) + \nabla \hat{V}(y, x, \hat{W}_c)(\hat{f}(y, \hat{\theta}) + g(y) \hat{u}(y, x, \hat{W}_a)). \quad (7.29)$$

Using  $\hat{\delta}$ , the instantaneous Bellman errors in (7.13) and (7.15) are redefined as

$$\delta_t(t) \triangleq \hat{\delta}(x(t), x(t), \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t)), \quad (7.30)$$

and

$$\delta_{ti}(t) \triangleq \hat{\delta}(x_i(x(t), t), x(t), \hat{W}_c(t), \hat{W}_a(t), \hat{\theta}(t)), \quad (7.31)$$

respectively, where  $\omega$  and  $\omega_i$  are redefined as

$$\begin{aligned}\omega(t) &\triangleq \nabla\sigma(x(t), c(x(t))) \hat{f}(x(t), \hat{\theta}(t)) \\ &+ \nabla\sigma(x(t), c(x(t))) g(x(t)) \hat{u}(x(t), x(t), \hat{W}_a(t)),\end{aligned}\quad (7.32)$$

$$\begin{aligned}\omega_i(t) &\triangleq \nabla\sigma(x_i(x(t), t), c(x(t))) f(x_i(x(t), t), \hat{\theta}(t)) \\ &+ \nabla\sigma(x_i(x(t), t), c(x(t))) g(x_i(x(t), t)) \cdot \hat{u}(x_i(x(t), t), x(t), \hat{W}_a(t)).\end{aligned}\quad (7.33)$$

Assumption 4.1 describes the characteristic of a parameter estimator required to achieve closed-loop stability. The main result for uncertain drift dynamics is summarized in the following theorem.

**Theorem 7.9** *Provided a parameter estimator that satisfies Assumption 4.1 is available, the StaF kernels and the basis functions for system identification are selected such that  $\nabla W$  and the approximation errors  $\varepsilon$ ,  $\nabla\varepsilon$ ,  $\varepsilon_\theta$ , and  $\nabla\varepsilon_\theta$  are sufficiently small, and provided the points for Bellman error extrapolation are selected such that the minimum eigenvalue  $c$ , introduced in (7.23) is sufficiently large, then the update laws given by (7.16)–(7.18), with the renewed definitions in (7.29)–(7.33) ensure that the state  $x$  and the weight estimation errors  $\tilde{\theta}$ ,  $\tilde{W}_c$ , and  $\tilde{W}_a$  are ultimately bounded.*

*Proof* The proof is a trivial combination of the proofs of Theorems 7.8 and 4.3.  $\square$

#### 7.4.5 Simulation

##### Optimal Regulation Problem with Exact Model Knowledge

To demonstrate the effectiveness of the StaF kernels, simulations are performed on a two-state nonlinear dynamical system. The system dynamics are given by (1.9), where  $x = [x_1, x_2]^T$ ,

$$\begin{aligned}f(x) &= \begin{bmatrix} -x_1 + x_2 \\ -\frac{1}{2}x_1 - \frac{1}{2}x_2(\cos(2x_1) + 2)^2 \end{bmatrix}, \\ g(x) &= \begin{bmatrix} 0 \\ \cos(2x_1) + 2 \end{bmatrix}.\end{aligned}\quad (7.34)$$

The control objective is to minimize the cost

$$\int_0^\infty (x^T(\tau)x(\tau) + u^2(\tau)) d\tau. \quad (7.35)$$

The system in (7.34) and the cost in (7.35) are selected because the corresponding optimal control problem has a known analytical solution. The optimal value function is  $V^*(x) = \frac{1}{2}x_1^{o2} + x_2^{o2}$ , and the optimal control policy is  $u^*(x) = -(\cos(2x_1) + 2)x_2$ .

To apply the developed technique to this problem, the value function is approximated using three exponential StaF kernels (i.e.,  $\sigma(x, C) = [\sigma_1(x, c_1), \sigma_2(x, c_2), \sigma_3(x, c_3)]^T$ ). The kernels are selected to be  $\sigma_i(x, c_i) = e^{x^T c_i} - 1$ ,  $i = 1, \dots, 3$ . The centers  $c_i$  are selected to be on the vertices of a shrinking equilateral triangle around the current state (i.e.,  $c_i = x + d_i(x)$   $i = 1, \dots, 3$ ), where  $d_1(x) = 0.7v^o(x) \cdot [0, 1]^T$ ,  $d_2(x) = 0.7v^o(x) \cdot [0.87, -0.5]^T$ , and  $d_3(x) = 0.7v^o(x) \cdot [-0.87, -0.5]^T$ , and  $v^o(x) \triangleq \left( \frac{x^T x + 0.01}{1 + \gamma_2 x^T x} \right)$  denotes the shrinking function, where  $\gamma_2 \in \mathbb{R}_{>0}$  is a constant normalization gain. To ensure sufficient excitation, a single point for Bellman error extrapolation is selected at random from a uniform distribution over a  $2.1v^o(x(t)) \times 2.1v^o(x(t))$  square centered at the current state  $x(t)$  so that the function  $x_i$  is of the form  $x_i(x, t) = x + a_i(t)$  for some  $a_i(t) \in \mathbb{R}^2$ . For a general problem with an  $n$ -dimensional state, exponential kernels can be utilized with the centers placed at the vertices of an  $n$ -dimensional simplex with the current state as the centroid. The extrapolation point can be sampled at each iteration from a uniform distribution over an  $n$ -dimensional hypercube centered at the current state.

The system is initialized at  $t_0 = 0$  and the initial conditions

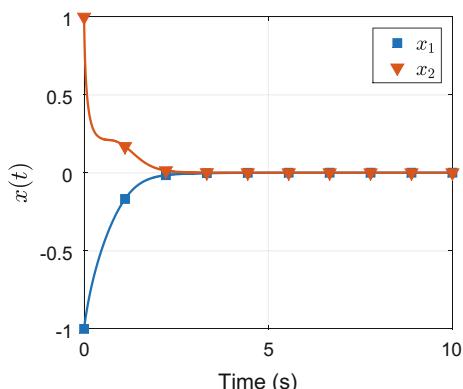
$$x(0) = [-1, 1]^T, \hat{W}_c(0) = 0.4 \times \mathbf{1}_{3 \times 1}, \Gamma(0) = 500\mathbf{I}_3, \hat{W}_a(0) = 0.7\hat{W}_c(0),$$

and the learning gains are selected as

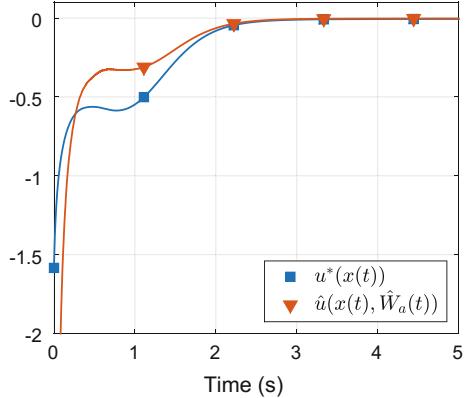
$$k_{c1} = 0.001, k_{c2} = 0.25, k_{a1} = 1.2, k_{a2} = 0.01, \beta = 0.003, \gamma_1 = 0.05, \gamma_2 = 1.$$

Figure 7.5 shows that the developed StaF-based controller drives the system states to the origin while maintaining system stability. Figure 7.6 shows the implemented control signal compared with the optimal control signal. It is clear that the imple-

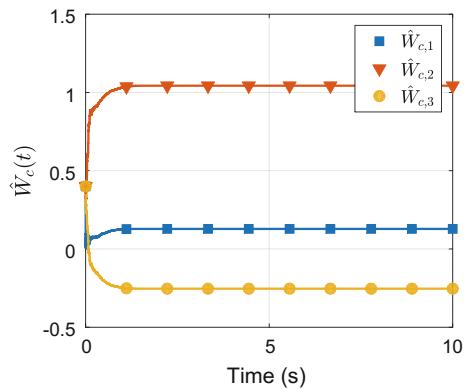
**Fig. 7.5** State trajectories generated using StaF kernel-based approximate dynamic programming (reproduced with permission from [21], ©2016, Elsevier)



**Fig. 7.6** kernel-based approximate dynamic programming compared with the optimal control trajectory (reproduced with permission from [21], ©2016, Elsevier)



**Fig. 7.7** Trajectories of the estimates of the unknown parameters in the value function generated using StaF kernel-based approximate dynamic programming. The ideal weights are unknown and time-varying; hence, the obtained weights can not be compared with their ideal values (reproduced with permission from [21], ©2016, Elsevier)

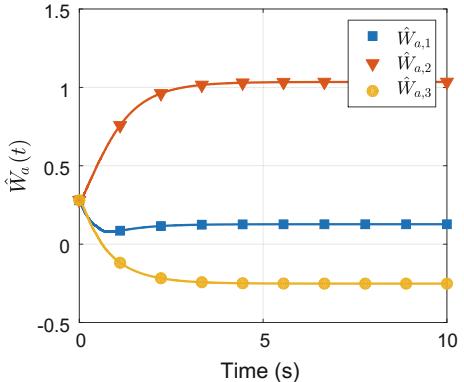


mented control converges to the optimal controller. Figures 7.7 and 7.8 shows that the weight estimates for the StaF-based value function and policy approximation remain bounded and converge as the state converges to the origin. Since the ideal values of the weights are unknown, the weights can not directly be compared with their ideal values. However, since the optimal solution is known, the value function estimate corresponding to the weights in Fig. 7.7 can be compared to the optimal value function at each time  $t$ . Figure 7.9 shows that the error between the optimal and the estimated value functions rapidly decays to zero.

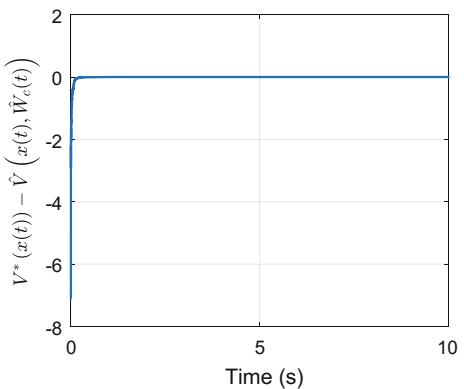
### Optimal Tracking Problem with Parametric Uncertainties in the Drift Dynamics

This simulation demonstrates the effectiveness of the extension developed in Sect. 7.4.4. The drift dynamics in the two-state nonlinear dynamical system in (7.34) are assumed to be linearly parameterized as

**Fig. 7.8** Trajectories of the estimates of the unknown parameters in the policy generated using StaF kernel-based approximate dynamic programming. The ideal weights are unknown and time-varying; hence, the obtained weights can not be compared with their ideal weights (reproduced with permission from [21], ©2016, Elsevier)



**Fig. 7.9** The error between the optimal and the estimated value function (reproduced with permission from [21], ©2016, Elsevier)

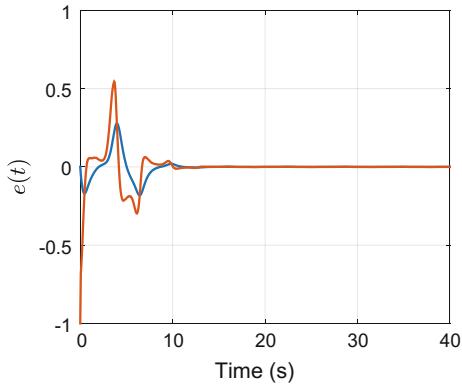


$$f(x) = \underbrace{\begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \\ \theta_4 & \theta_5 & \theta_6 \end{bmatrix}}_{\theta^T} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_2(\cos(2x_1) + 2) \end{bmatrix}}_{\sigma_\theta(x)},$$

where  $\theta \in \mathbb{R}^{3 \times 2}$  is the matrix of unknown parameters, and  $\sigma_\theta$  is the known vector of basis functions. The ideal values of the unknown parameters are  $\theta_1 = -1$ ,  $\theta_2 = 1$ ,  $\theta_3 = 0$ ,  $\theta_4 = -0.5$ ,  $\theta_5 = 0$ , and  $\theta_6 = -0.5$ . Let  $\hat{\theta}$  denote an estimate of the unknown matrix  $\theta$ . The control objective is to drive the estimate  $\hat{\theta}$  to the ideal matrix  $\theta$ , and to drive the state  $x$  to follow a desired trajectory  $x_d$ . The desired trajectory is selected to be the solution to the initial value problem

$$\dot{x}_d(t) = \begin{bmatrix} -1 & 1 \\ -2 & 1 \end{bmatrix} x_d(t), \quad x_d(0) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad (7.36)$$

**Fig. 7.10** Tracking error trajectories generated using the proposed method for the trajectory tracking problem (reproduced with permission from [21], ©2016, Elsevier)



and the cost functional is selected to be  $\int_0^\infty (e^T(t) \text{diag}(10, 10)e(t) + (\mu(t))^2) dt$ , where  $e(t) = x(t) - x_d(t)$ ,  $\mu$  is an auxiliary controller designed using the developed method, and the tracking controller is designed as

$$u(t) = g^+(x_d(t)) \left( \begin{bmatrix} -1 & 1 \\ -2 & 1 \end{bmatrix} x_d(t) - \hat{\theta}^T \sigma_\theta(x_d(t)) \right) + \mu(t),$$

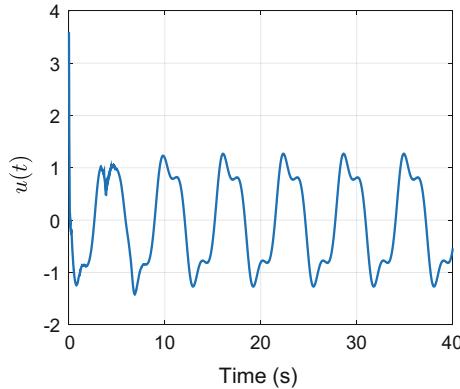
where  $g^+(x)$  denotes the pseudoinverse of  $g(x)$ .

The value function is a function of the concatenated state  $\zeta \triangleq [e^T \ x_d^T]^T \in \mathbb{R}^4$ . The value function is approximated using five exponential StaF kernels given by  $\sigma_i(\zeta^o, C)$ , where the five centers are selected according to  $c_i = \zeta^o + d_i(\zeta^o)$  to form a regular five dimensional simplex around the current state with  $v^o(\zeta^o) \equiv 1$ . Learning gains for system identification and value function approximation are selected as

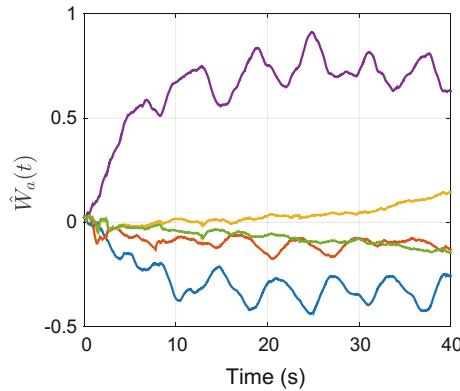
$$\begin{aligned} k_{c1} &= 0.001, \ k_{c2} = 2, \ k_{a1} = 2, \ k_{a2} = 0.001, \ \beta = 0.01, \ \gamma_1 = 0.1, \ \gamma_2 = 1, \ k = 500, \\ \Gamma_\theta &= I_3, \ \Gamma(0) = 50I_5, \ k_\theta = 20. \end{aligned}$$

Sufficient excitation is ensured by selecting a single state trajectory  $\zeta_i(\zeta^o, t) \triangleq \zeta^o + a_i(t)$  for Bellman error extrapolation, where  $a_i(t)$  is sampled at each  $t$  from a uniform distribution over the  $2.1 \times 2.1 \times 2.1 \times 2.1$  hypercube centered at the origin. The history stack required for concurrent learning contains ten points, and is recorded online using a singular value maximizing algorithm (cf. [34]), and the required state derivatives are computed using a fifth order Savitzky–Golay smoothing filter (cf. [35]).

The initial values for the state and the state estimate are selected to be  $x(0) = [0, 0]^T$  and  $\hat{x}(0) = [0, 0]^T$ , respectively. The initial values for the neural network weights for the value function, the policy, and the drift dynamics are selected to be  $0.025 \times \mathbf{1}_{5 \times 1}$ ,  $0.025 \times \mathbf{1}_{5 \times 1}$ , and  $\mathbf{0}_{3 \times 2}$ , respectively. Since the system in (7.34) has no stable equilibria, the initial policy  $\hat{\mu}(\zeta, \mathbf{0}_{3 \times 2})$  is not stabilizing. The stabilization demonstrated in Fig. 7.10 is achieved via fast simultaneous learning of the system dynamics and the value function.



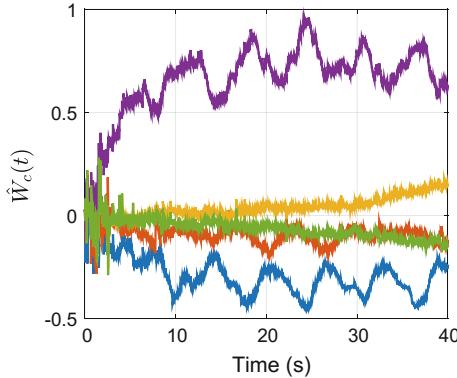
**Fig. 7.11** Control signal generated using the proposed method for the trajectory tracking problem (reproduced with permission from [21], ©2016, Elsevier)



**Fig. 7.12** Actor weight trajectories generated using the proposed method for the trajectory tracking problem. The weights do not converge to a steady-state value because the ideal weights are not constant, they are functions of the time-varying system state. Since an analytical solution to the optimal tracking problem is not available, weights cannot be compared against their ideal values (reproduced with permission from [21], ©2016, Elsevier)

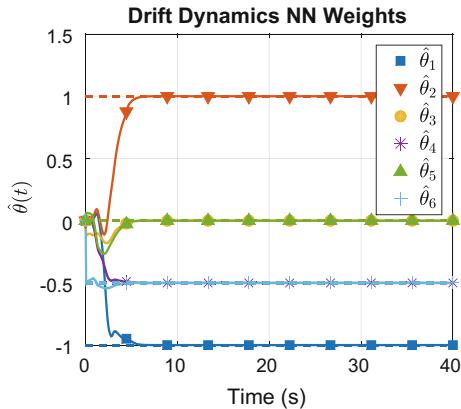
Figures 7.10 and 7.11 demonstrate that the controller remains bounded and the tracking error is regulated to the origin. The neural network weights are functions of the system state  $\zeta$ . Since  $\zeta$  converges to a periodic orbit, the neural network weights also converge to a periodic orbit (within the bounds of the excitation introduced by the Bellman error extrapolation signal), as demonstrated in Figs. 7.12 and 7.13. Figure 7.14 demonstrates that the unknown parameters in the drift dynamics, represented by solid lines, converge to their ideal values, represented by dashed lines.

The developed technique is compared with the model-based reinforcement learning method developed in [11] for regulation and [12] for tracking, respectively. The simulations are performed in MATLAB® Simulink® at 1000 Hz on the same



**Fig. 7.13** Critic function weight trajectories generated using the proposed method for the trajectory tracking problem. The weights do not converge to a steady-state value because the ideal weights not constant, they are functions of the time-varying system state. Since an analytical solution to the optimal tracking problem is not available, weights cannot be compared against their ideal values (reproduced with permission from [21], ©2016, Elsevier)

**Fig. 7.14** Trajectories of the unknown parameters in the system drift dynamics for the trajectory tracking problem. The dotted lines represent the true values of the parameters (reproduced with permission from [21], ©2016, Elsevier)



**Table 7.1** Simulation results for 2, 3, and 4 dimensional nonlinear systems

Problem description	Regulation (2-states)		Regulation (3-states)		Tracking (4-states)	
Controller	StaF	[11]	StaF	[11]	StaF	[12]
Running time (seconds)	6.5	17	9.5	62	12	260
Total cost	2.8	1.8	9.3	12.3	3.9	3.4
RMS steady-state error	2.5E - 6	0	4.3E - 6	4.5E - 6	3.5E - 4	2.5E - 4

machine. The simulations run for 100s of simulated time. Since the objective is to compare computational efficiency of the model-based reinforcement learning method, exact knowledge of the system model is used. Table 7.1 shows that the

developed controller requires significantly fewer computational resources than the controllers from [11, 12]. Furthermore, as the system dimension increases, the developed controller significantly outperforms the controllers from [11, 12] in terms of computational efficiency.

Since the optimal solution for the regulation problem is known to be quadratic, the model-based reinforcement learning method from [11] is implemented using three quadratic basis functions. Since the basis used is exact, the method from [11] yields a smaller steady-state error than the developed method, which uses three generic StaF kernels. For the three-state regulation problem and the tracking problem, the methods from [11, 12] are implemented using polynomial basis functions selected based on a trial-and-error approach. The developed technique is implemented using generic StaF kernels. In this case, since the optimal solution is unknown, both the methods use generic basis functions, resulting in similar steady-state errors.

The two main advantages of StaF kernels are that they are universal, in the sense that they can be used to approximate a large class of value functions, and that they target local approximation, resulting in a smaller number of required basis functions. However, the StaF kernels trade optimality for universality and computational efficiency. The kernels are generic, and the weight estimates need to be continually adjusted based on the system trajectory. Hence, as shown in Table 7.1, the developed technique results in a higher total cost than state-of-the-art model-based reinforcement learning techniques.

## 7.5 Background and Further Reading

Reinforcement learning has become a popular tool for determining online solutions of optimal control problems for systems with finite state and action-spaces [30, 36–40]. Due to various technical and practical difficulties, implementation of reinforcement learning-based closed-loop controllers on hardware platforms remains a challenge. Approximate dynamic programming-based controllers are void of pre-designed stabilizing feedback and are completely defined by the estimated parameters. Hence, the error between the optimal and the estimated value function is required to decay to a sufficiently small bound sufficiently fast to establish closed-loop stability. The size of the error bound is determined by the selected basis functions, and the convergence rate is determined by richness of the data used for learning.

Fast approximation of the value function over a large neighborhood requires sufficiently rich data to be available for learning. In traditional approximate dynamic programming methods such as [31, 41, 42], richness of data manifests itself as the amount of excitation in the system. In experience replay-based techniques such as [34, 43–45], richness of data is quantified by eigenvalues of a recorded history stack. In model-based reinforcement learning techniques such as [11–13], richness of data corresponds to the eigenvalues of a learning matrix. As the dimension of the system and the number of basis functions increases, the richer data is required to achieve learning. In traditional approximate dynamic programming methods, the demand

for rich data is met by adding excitation signals to the controller, thereby causing undesirable oscillations. In experience replay-based approximate dynamic programming methods and in model-based reinforcement learning, the demand for richer data causes exponential growth in the required data storage. Hence, experimental implementations of traditional approximate dynamic programming techniques such as [25–32, 41, 42, 46, 47] and data-driven approximate dynamic programming techniques such as [11–13, 45, 48, 49] in high dimensional systems are scarcely found in the literature.

The control design in (7.11) exploits the fact that given a basis  $\sigma$  for approximation of the value function, the basis  $\frac{1}{2}R^{-1}g^T\nabla\sigma^T$  approximates the optimal controller, provided the dynamics control-affine. As a part of future research, possible extensions to nonaffine systems could potentially be explored by approximating the controller using an independent basis (cf. [50–57]).

## References

1. Kirk D (2004) Optimal control theory: an introduction. Dover, Mineola
2. Liberzon D (2012) Calculus of variations and optimal control theory: a concise introduction. Princeton University Press, Princeton
3. Christmann A, Steinwart I (2010) Universal kernels on non-standard input spaces. In: Advances in neural information processing, pp 406–414
4. Micchelli CA, Xu Y, Zhang H (2006) Universal kernels. J Mach Learn Res 7:2651–2667
5. Park J, Sanberg I (1991) Universal approximation using radial-basis-function networks. Neural Comput 3(2):246–257
6. Folland GB (1999) Real analysis: modern techniques and their applications, 2nd edn. Pure and applied mathematics, Wiley, New York
7. Steinwart I, Christmann A (2008) Support vector machines. Information science and statistics, Springer, New York
8. Gaggero M, Gnecco G, Sanguineti M (2013) Dynamic programming and value-function approximation in sequential decision problems: error analysis and numerical results. J Optim Theory Appl 156
9. Gaggero M, Gnecco G, Sanguineti M (2014) Approximate dynamic programming for stochastic n-stage optimization with application to optimal consumption under uncertainty. Comput Optim Appl 58(1):31–85
10. Zoppoli R, Sanguineti M, Parisini T (2002) Approximating networks and extended Ritz method for the solution of functional optimization problems. J Optim Theory Appl 112(2):403–440
11. Kamalapurkar R, Walters P, Dixon WE (2013) Concurrent learning-based approximate optimal regulation. In: Proceedings of the IEEE conference on decision and control, Florence, IT, pp 6256–6261
12. Kamalapurkar R, Andrews L, Walters P, Dixon WE (2014) Model-based reinforcement learning for infinite-horizon approximate optimal tracking. In: Proceedings of the IEEE conference on decision and control, Los Angeles, CA, pp 5083–5088
13. Kamalapurkar R, Klotz J, Dixon WE (2014) Concurrent learning-based online approximate feedback Nash equilibrium solution of N-player nonzero-sum differential games. IEEE/CAA J Autom Sin 1(3):239–247
14. Aronszajn N (1950) Theory of reproducing kernels. Trans Am Math Soc 68:337–404
15. Zhu K (2012) Analysis on fock spaces, vol 263. Graduate texts in mathematics, Springer, New York

16. Pinkus A (2004) Strictly positive definite functions on a real inner product space. *Adv Comput Math* 20:263–271
17. Rosenfeld JA, Kamalapurkar R, Dixon WE (2015) State following (StaF) kernel functions for function approximation part I: theory and motivation. In: Proceedings of the American control conference, pp 1217–1222
18. Beylkin G, Monzon L (2005) On approximation of functions by exponential sums. *Appl Comput Harmon Anal* 19(1):17–48
19. Bertsekas DP (1999) Nonlinear programming. Athena Scientific, Belmont
20. Pedersen GK (1989) Analysis now, vol 118. Graduate texts in mathematics, Springer, New York
21. Kamalapurkar R, Rosenfeld J, Dixon WE (2016) Efficient model-based reinforcement learning for approximate online optimal control. *Automatica* 74:247–258
22. Lorentz GG (1986) Bernstein polynomials, 2nd edn. Chelsea Publishing Co., New York
23. Ioannou P, Sun J (1996) Robust adaptive control. Prentice Hall, Upper Saddle River
24. Khalil HK (2002) Nonlinear systems, 3rd edn. Prentice Hall, Upper Saddle River
25. Doya K (2000) Reinforcement learning in continuous time and space. *Neural Comput* 12(1):219–245
26. Padhi R, Unnikrishnan N, Wang X, Balakrishnan S (2006) A single network adaptive critic (SNAC) architecture for optimal control synthesis for a class of nonlinear systems. *Neural Netw* 19(10):1648–1660
27. Al-Tamimi A, Lewis FL, Abu-Khalaf M (2008) Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *IEEE Trans Syst Man Cybern Part B Cybern* 38:943–949
28. Lewis FL, Vrabie D (2009) Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits Syst Mag* 9(3):32–50
29. Dierks T, Thumati B, Jagannathan S (2009) Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. *Neural Netw* 22(5–6):851–860
30. Mehta P, Meyn S (2009) Q-learning and pontryagin's minimum principle. In: Proceedings of the IEEE conference on decision and control, pp 3598–3605
31. Vamvoudakis KG, Lewis FL (2010) Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica* 46(5):878–888
32. Zhang H, Cui L, Zhang X, Luo Y (2011) Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method. *IEEE Trans Neural Netw* 22(12):2226–2236
33. Sadegh N (1993) A perceptron network for functional identification and control of nonlinear systems. *IEEE Trans Neural Netw* 4(6):982–988
34. Chowdhary G, Yucelen T, Mühlegg M, Johnson EN (2013) Concurrent learning adaptive control of linear systems with exponentially convergent bounds. *Int J Adapt Control Signal Process* 27(4):280–301
35. Savitzky A, Golay MJE (1964) Smoothing and differentiation of data by simplified least squares procedures. *Anal Chem* 36(8):1627–1639
36. Bertsekas D, Tsitsiklis J (1996) Neuro-dynamic programming. Athena Scientific, Belmont
37. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge
38. Konda V, Tsitsiklis J (2004) On actor-critic algorithms. *SIAM J Control Optim* 42(4):1143–1166
39. Bertsekas D (2007) Dynamic programming and optimal control, vol 2, 3rd edn. Athena Scientific, Belmont
40. Szepesvári C (2010) Algorithms for reinforcement learning. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool Publishers, San Rafael
41. Vamvoudakis KG, Lewis FL (2009) Online synchronous policy iteration method for optimal control. In: Yu W (ed) Recent advances in intelligent control systems. Springer, Berlin, pp 357–374

42. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92
43. Chowdhary G (2010) Concurrent learning for convergence in adaptive control without persistency of excitation. Ph.D. thesis, Georgia Institute of Technology
44. Chowdhary G, Johnson E (2011) A singular value maximizing data recording algorithm for concurrent learning. In: Proceedings of the American control conference, pp 3547–3552
45. Modares H, Lewis FL, Naghibi-Sistani MB (2014) Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica* 50(1):193–202
46. Zhang H, Cui L, Luo Y (2013) Near-optimal control for nonzero-sum differential games of continuous-time nonlinear systems using single-network ADP. *IEEE Trans Cybern* 43(1):206–216
47. Zhang H, Liu D, Luo Y, Wang D (2013) Adaptive dynamic programming for control algorithms and stability. *Communications and control engineering*, Springer, London
48. Luo B, Wu HN, Huang T, Liu D (2014) Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design. *Automatica*
49. Yang X, Liu D, Wei Q (2014) Online approximate optimal control for affine non-linear systems with unknown internal dynamics using adaptive dynamic programming. *IET Control Theory Appl* 8(16):1676–1688
50. Ge SS, Zhang J (2003) Neural-network control of nonaffine nonlinear system with zero dynamics by state and output feedback. *IEEE Trans Neural Netw* 14(4):900–918
51. Wang D, Liu D, Wei Q, Zhao D, Jin N (2012) Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. *Automatica* 48(8):1825–1832
52. Zhang X, Zhang H, Sun Q, Luo Y (2012) Adaptive dynamic programming-based optimal control of unknown nonaffine nonlinear discrete-time systems with proof of convergence. *Neurocomputing* 91:48–55
53. Liu D, Huang Y, Wang D, Wei Q (2013) Neural-network-observer-based optimal control for unknown nonlinear systems using adaptive dynamic programming. *Int J Control* 86(9):1554–1566
54. Bian T, Jiang Y, Jiang ZP (2014) Adaptive dynamic programming and optimal control of nonlinear nonaffine systems. *Automatica* 50(10):2624–2632
55. Yang X, Liu D, Wei Q, Wang D (2015) Direct adaptive control for a class of discrete-time unknown nonaffine nonlinear systems using neural networks. *Int J Robust Nonlinear Control* 25(12):1844–1861
56. Kiumarsi B, Kang W, Lewis FL (2016)  $H_{\infty}$  control of nonaffine aerial systems using off-policy reinforcement learning. *Unmanned Syst* 4(1):1–10
57. Song R, Wei Q, Xiao W (2016) Off-policy neuro-optimal control for unknown complex-valued nonlinear systems based on policy iteration. *Neural Comput Appl* 46(1):85–95

# Appendix A

## Supplementary Lemmas and Definitions

### A.1 Chapter 3 Supplementary Material

#### A.1.1 Derivation of the Sufficient Conditions in (3.19)

Integrating (3.18) yields

$$\begin{aligned} \int_0^t L(\tau) d\tau &= \int_0^t \left\{ r^T [N_{B1}(\tau) - \beta_1 sgn(\tilde{x})] + \dot{\tilde{x}}(\tau)^T N_{B2}(\tau) - \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}\| \right\} d\tau \\ &= \tilde{x}^T N_B - \tilde{x}^T(0) N_B(0) - \int_0^t \tilde{x}^T \dot{N}_B d\tau + \beta_1 \sum_{i=1}^n |\tilde{x}_i(0)| - \beta_1 \sum_{i=1}^n |\tilde{x}_i(t)| \\ &\quad + \int_0^t \alpha \tilde{x}^T (N_{B1} - \beta_1 sgn(\tilde{x})) d\tau - \int_0^t \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}\| d\tau, \end{aligned}$$

where (3.9) is used. Using the fact that  $\|\tilde{x}\|_2 \leq \sum_{i=1}^n |\tilde{x}_i|$ , and using the bounds in (3.14), yields

$$\begin{aligned} \int_0^t L(\tau) d\tau &\leq \beta_1 \sum_{i=1}^n |\tilde{x}_i(0)| - \tilde{x}^T(0) N_B(0) - (\beta_1 - \zeta_1 - \zeta_2) \|\tilde{x}\| \\ &\quad - \int_0^t \alpha (\beta_1 - \zeta_1 - \frac{\zeta_3}{\alpha}) \|\tilde{x}\| d\tau - \int_0^t (\beta_2 - \zeta_4) \rho_2(\|z\|) \|z\| \|\tilde{x}\| d\tau. \end{aligned}$$

If the sufficient conditions in (3.19) are satisfied, then the following inequality holds

$$\int_0^t L(\tau) d\tau \leq \beta_1 \sum_{i=1}^n |\tilde{x}_i(0)| - \tilde{x}^T(0) N_B(0) = P(0). \quad (\text{A.1})$$

Using (3.17) and (A.1), it can be shown that  $P(t) \geq 0$ .

### A.1.2 Proof of Theorem 3.4

*Proof* Let  $y(t)$  for  $t \in [t_0, \infty)$  denote a Filippov solution to the differential equation in (3.22) that satisfies  $y(t_0) \in \mathcal{S}$ . Using Filippov's theory of differential inclusions [1, 2], the existence of solutions can be established for  $\dot{y} \in K[h](y, t)$ , where  $K[h](y, t) \triangleq \bigcap_{\delta>0} \bigcap_{\mu S_m=0} \overline{\text{coh}}(B_\delta(y) \setminus S_m, t)$ , where  $\bigcap_{\mu S_m=0}$  denotes the intersection of all sets  $S_m$  of Lebesgue measure zero,  $\overline{\text{co}}$  denotes convex closure [3, 4]. The time derivative of (3.20) along the Filippov trajectory  $y(\cdot)$  exists almost everywhere (a.e.), and  $\dot{V}_I \stackrel{\text{a.e.}}{\in} \dot{\tilde{V}}_I$  where

$$\dot{\tilde{V}}_I = \bigcap_{\xi \in \partial V_I(y)} \xi^T K \left[ e_f^T \dot{\tilde{x}}^T \frac{1}{2} P^{-\frac{1}{2}} \dot{P} \frac{1}{2} Q^{-\frac{1}{2}} \dot{Q} \right]^T, \quad (\text{A.2})$$

where  $\partial V_I$  is the generalized gradient of  $V_I$  [5]. Since  $V_I$  is continuously differentiable, (A.2) can be simplified as [3]

$$\begin{aligned} \dot{\tilde{V}}_I &= \nabla V_I^T K \left[ e_f^T \dot{\tilde{x}}^T \frac{1}{2} P^{-\frac{1}{2}} \dot{P} \frac{1}{2} Q^{-\frac{1}{2}} \dot{Q} \right]^T \\ &= \left[ e_f^T \gamma \tilde{x}^T 2P^{\frac{1}{2}} 2Q^{\frac{1}{2}} \right] K \left[ e_f^T \dot{\tilde{x}}^T \frac{1}{2} P^{-\frac{1}{2}} \dot{P} \frac{1}{2} Q^{-\frac{1}{2}} \dot{Q} \right]^T. \end{aligned}$$

Using the calculus for  $K[\cdot]$  from [4] (Theorem 1, Properties 2, 5, 7), and substituting the dynamics from (3.9) and (3.17), yields

$$\begin{aligned} \dot{\tilde{V}}_I &\subset e_f^T (\tilde{N} + N_{B1} + \hat{N}_{B2} - k e_f - \beta_1 K[\text{sgn}](\tilde{x}) - \gamma \tilde{x}) - e_f^T (N_{B1} - \beta_1 K[\text{sgn}](\tilde{x})) \\ &\quad + \gamma \tilde{x}^T (e_f - \alpha \tilde{x}) - \dot{\tilde{x}}^T N_{B2} + \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}\| - \frac{1}{2} \alpha \left[ \text{tr}(\tilde{W}_f^T \Gamma_{wf}^{-1} \dot{W}_f) + \text{tr}(\tilde{V}_f^T \Gamma_{vf}^{-1} \dot{V}_f) \right] \\ &\quad - \frac{1}{2} \alpha \sum_{i=1}^m \left[ \text{tr}(\tilde{W}_{gi}^T \Gamma_{wgi}^{-1} \dot{W}_{gi}) + \text{tr}(\tilde{V}_{gi}^T \Gamma_{vgi}^{-1} \dot{V}_{gi}) \right]. \end{aligned} \quad (\text{A.3})$$

$$\begin{aligned} &\stackrel{\text{a.e.}}{\leq} -\alpha \gamma \|\tilde{x}\|^2 - k \|e_f\|^2 + \rho_1(\|z\|) \|z\| \|e_f\| + \zeta_5 \|\tilde{x}\|^2 + \zeta_6 \|e_f\|^2 \\ &\quad + \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}\|, \end{aligned} \quad (\text{A.4})$$

where (3.8), (3.13), and (3.15) are used,  $K[\text{sgn}](\tilde{x}) = \text{SGN}(\tilde{x})$  [4], such that  $\text{SGN}(\tilde{x}_i) = \{1\}$  if  $\tilde{x}_i > 0$ ,  $[-1, 1]$  if  $\tilde{x}_i = 0$ , and  $\{-1\}$  if  $\tilde{x}_i < 0$  (the subscript  $i$  denotes the  $i$ th element).

The set in (A.3) reduces to the scalar inequality in (A.4) since the right hand side is continuous almost everywhere (i.e., the right hand side is continuous except for the Lebesgue negligible set of times when  $e_f^T K[\text{sgn}](\tilde{x}) - e_f^T K[\text{sgn}](\tilde{x}) \neq 0$ ). The set

of times  $\Lambda \triangleq \{t \in [0, \infty) \mid e_f(t)^T K[\text{sgn}](\tilde{x}(t)) - e_f(t)^T K[\text{sgn}](\tilde{x}(t)) \neq 0\} \subset [0, \infty)$  is equivalent to the set of times  $\{t \mid \tilde{x}(t) = 0 \wedge e_f(t) \neq 0\}$ . From (3.16), this set can also be represented by  $\{t \mid \tilde{x}(t) = 0 \wedge \dot{\tilde{x}}(t) \neq 0\}$ . Provided  $\tilde{x}$  is continuously differentiable, it can be shown that the set of time instances  $\{t \mid \tilde{x}(t) = 0 \wedge \dot{\tilde{x}}(t) \neq 0\}$  is isolated, and thus, measure zero. This implies that the set  $\Lambda$  is measure zero [6].

Substituting for  $k \triangleq k_1 + k_2$  and  $\gamma \triangleq \gamma_1 + \gamma_2$ , and completing the squares, the expression in (A.4) can be upper bounded as

$$\dot{\tilde{V}}_I \stackrel{\text{a.e.}}{\leq} -(\alpha\gamma_1 - \zeta_5) \|\tilde{x}\|^2 - (k_1 - \zeta_6) \|e_f\|^2 + \frac{\rho_1(\|z\|)^2}{4k_2} \|z\|^2 + \frac{\beta_2^2 \rho_2(\|z\|)^2}{4\alpha\gamma_2} \|z\|^2. \quad (\text{A.5})$$

Provided the sufficient conditions in (3.23) are satisfied, the expression in (A.5) can be rewritten as

$$\begin{aligned} \dot{\tilde{V}}_I &\stackrel{\text{a.e.}}{\leq} -\lambda \|z\|^2 + \frac{\rho(\|z\|)^2}{4\eta} \|z\|^2 \\ &\stackrel{\text{a.e.}}{\leq} -U(y) \quad \forall y \in \mathcal{D}, \end{aligned} \quad (\text{A.6})$$

where  $\lambda \triangleq \min\{\alpha\gamma_1 - \zeta_5, k_1 - \zeta_6\}$ ,  $\eta \triangleq \min\{k_2, \frac{\alpha\gamma_2}{\beta_2^2}\}$ ,  $\rho(\|z\|)^2 \triangleq \rho_1(\|z\|)^2 + \rho_2(\|z\|)^2$  is a positive strictly increasing function, and  $U(y) = c \|z\|^2$ , for some positive constant  $c$ , is a continuous positive semi-definite function defined on the domain  $\mathcal{D}$ . The size of the domain  $\mathcal{D}$  can be increased by increasing the gains  $k$  and  $\gamma$ . Using the inequalities in (3.21) and (A.6), [7, Corollary 1] can be invoked to show that  $y(\cdot) \in \mathcal{L}_\infty$ , provided  $y(0) \in \mathcal{S}$ . Furthermore,  $\|\tilde{x}(t)\|, \|\dot{\tilde{x}}(t)\|, \|e_f(t)\| \rightarrow 0$  as  $t \rightarrow \infty$  provided  $y(0) \in \mathcal{S}$ .

Since  $y(\cdot) \in \mathcal{L}_\infty$ ,  $\tilde{x}(\cdot), e_f(\cdot) \in \mathcal{L}_\infty$ . Using (3.6), standard linear analysis can be used to show that  $\dot{\tilde{x}}(\cdot) \in \mathcal{L}_\infty$ , and since  $\dot{x}(\cdot) \in \mathcal{L}_\infty$ ,  $\hat{x}(\cdot) \in \mathcal{L}_\infty$ . Since  $\hat{W}_f(\cdot) \in \mathcal{L}_\infty$  from the use of projection in (3.8),  $t \mapsto \sigma_f\left(\hat{V}_f^T(t) \hat{x}(t)\right) \in \mathcal{L}_\infty$  from Property 2.3,  $u(\cdot) \in \mathcal{L}_\infty$  from Assumption 3.3, and  $\mu(\cdot) \in \mathcal{L}_\infty$  from (3.3). Using the above bounds, it can be shown from (3.9) that  $\dot{e}_f(\cdot) \in \mathcal{L}_\infty$ .  $\square$

### A.1.3 Algorithm for Selection of Neural Network Architecture and Learning Gains

Since the gains depend on the initial conditions, the compact sets used for function approximation, and the Lipschitz bounds, an iterative algorithm is developed to select the gains. In Algorithm A.1, the notation  $\{\varpi\}_i$  for any parameter  $\varpi$  denotes the value of  $\varpi$  computed in the  $i$ th iteration. Algorithm A.1 ensures satisfaction of the sufficient condition in (3.75).

---

**Algorithm A.1** Gain Selection
 

---

First iteration:

Given  $Z_0 \in \mathbb{R}_{\geq 0}$  such that  $\|Z(t_0)\| < Z_0$ , let  $\mathcal{Z}_1 = \{\rho \in \mathbb{R}^{n+2\{L\}_1} \mid \|\rho\| \leq \beta_1 v_l^{-1}(\bar{v}_l(Z_0))\}$  for some  $\beta_1 > 1$ . Using  $\mathcal{Z}_1$ , compute the bounds in (3.67) and (3.73), and select the gains according to (3.74). If  $\{\bar{Z}\}_1 \leq \beta_1 v_l^{-1}(\bar{v}_l(\|Z_0\|))$ , set  $\mathcal{Z} = \mathcal{Z}_1$  and terminate.

Second iteration:

If  $\{\bar{Z}\}_2 > \beta_1 v_l^{-1}(\bar{v}_l(\|Z_0\|))$ , let  $\mathcal{Z}_2 \triangleq \{\rho \in \mathbb{R}^{n+2\{L\}_1} \mid \|\rho\| \leq \beta_2 \{\bar{Z}\}_1\}$ . Using  $\mathcal{Z}_2$ , compute the bounds in (3.67) and (3.73) and select the gains according to (3.74). If  $\{\bar{Z}\}_2 \leq \{\bar{Z}\}_1$ , set  $\mathcal{Z} = \mathcal{Z}_2$  and terminate.

Third iteration:

If  $\{\bar{Z}\}_2 > \{\bar{Z}\}_1$ , increase the number of neural network neurons to  $\{L\}_3$  to yield a lower function approximation error  $\{\bar{\epsilon}\}_3$  such that  $\{L_F\}_2 \{\bar{\epsilon}\}_3 \leq \{L_F\}_1 \{\bar{\epsilon}\}_1$ . The increase in the number of neural network neurons ensures that  $\{\iota\}_3 \leq \{\iota\}_1$ . Furthermore, the assumption that the persistence of excitation interval  $\{T\}_3$  is small enough such that  $\{L_F\}_2 \{T\}_3 \leq \{T\}_1 \{L_F\}_1$  and  $\{L\}_3 \{T\}_3 \leq \{T\}_1 \{L\}_1$  ensures that  $\left\{\frac{\varpi_{10}}{\varpi_{11}}\right\}_3 \leq \left\{\frac{\varpi_{10}}{\varpi_{11}}\right\}_1$ , and hence,  $\{\bar{Z}\}_3 \leq \beta_2 \{\bar{Z}\}_1$ . Set  $\mathcal{Z} = \{\rho \in \mathbb{R}^{n+2\{L\}_3} \mid \|\rho\| \leq \beta_2 \{\bar{Z}\}_1\}$  and terminate.

---

#### A.1.4 Proof of Lemma 3.14

The following supporting technical lemma is used to prove Lemma 3.14.

**Lemma A.1** Let  $D \subseteq \mathbb{R}^n$  contain the origin and let  $\Xi : D \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  be positive definite. If  $t \mapsto \Xi(x, t)$  is uniformly bounded for all  $x \in D$  and if  $x \mapsto \Xi(x, t)$  is continuous, uniformly in  $t$ , then  $\Xi$  is decrescent in  $D$ .

*Proof* Since  $t \mapsto \Xi(x, t)$  is uniformly bounded, for all  $x \in D$ ,  $\sup_{t \in \mathbb{R}_{\geq 0}} \{\Xi(x, t)\}$  exists and is unique for all  $x \in D$ . Let the function  $\alpha : D \rightarrow \mathbb{R}_{\geq 0}$  be defined as

$$\alpha(x) \triangleq \sup_{t \in \mathbb{R}_{\geq 0}} \{\Xi(x, t)\}. \quad (\text{A.7})$$

Since  $x \rightarrow \Xi(x, t)$  is continuous, uniformly in  $t$ ,  $\forall \varepsilon > 0$ ,  $\exists \varsigma(x) > 0$  such that  $\forall y \in D$ ,

$$d_{D \times \mathbb{R}_{\geq 0}}((x, t), (y, t)) < \varsigma(x) \implies d_{\mathbb{R}_{\geq 0}}(\Xi(x, t), \Xi(y, t)) < \varepsilon, \quad (\text{A.8})$$

where  $d_M(\cdot, \cdot)$  denotes the standard Euclidean metric on the metric space  $M$ . By the definition of  $d_M(\cdot, \cdot)$ ,  $d_{D \times \mathbb{R}_{\geq 0}}((x, t), (y, t)) = d_D(x, y)$ . Using (A.8),

$$d_D(x, y) < \varsigma(x) \implies |\Xi(x, t) - \Xi(y, t)| < \varepsilon. \quad (\text{A.9})$$

Given the fact that  $\Xi$  is positive, (A.9) implies  $\Xi(x, t) < \Xi(y, t) + \varepsilon$  and  $\Xi(y, t) < \Xi(x, t) + \varepsilon$  which from (A.7) implies  $\alpha(x) < \alpha(y) + \varepsilon$  and  $\alpha(y) < \alpha(x) + \varepsilon$ , and hence, from (A.9),  $d_D(x, y) < \varsigma(x) \implies |\alpha(x) - \alpha(y)| < \varepsilon$ . Since  $\Xi$  is positive definite, (A.7) can be used to conclude  $\alpha(0) = 0$ . Thus,  $\Xi$  is bounded above by a continuous positive definite function; hence,  $\Xi$  is decreasing in  $D$ .  $\square$

*Proof* Based on the definitions in (3.51), (3.52) and (3.68),  $V_t^*(e, t) > 0, \forall t \in \mathbb{R}_{\geq 0}$  and  $\forall e \in \overline{B_a} \setminus \{0\}$ . The optimal value function  $V^*\left([0, x_d^T]^T\right)$  is the cost incurred when starting with  $e = 0$  and following the optimal policy thereafter for an arbitrary desired trajectory  $x_d$ . Substituting  $x(t_0) = x_d(t_0)$ ,  $\mu(t_0) = 0$  and (3.45) in (3.47) indicates that  $\dot{e}(t_0) = 0$ . Thus, when starting from  $e = 0$ , a policy that is identically zero satisfies the dynamic constraints in (3.47). Furthermore, the optimal cost is  $V^*\left([0, x_d^T(t_0)]^T\right) = 0, \forall x_d(t_0)$  which, from (3.68), implies (3.69b). Since the optimal value function  $V_t^*$  is strictly positive everywhere but at  $e = 0$  and is zero at  $e = 0$ ,  $V_t^*$  is a positive definite function. Hence, [8, Lemma 4.3] can be invoked to conclude that there exists a class  $\mathcal{K}$  function  $\underline{\gamma} : [0, a] \rightarrow \mathbb{R}_{\geq 0}$  such that  $\underline{\gamma}(\|e\|) \leq V_t^*(e, t), \forall t \in \mathbb{R}_{\geq 0}$  and  $\forall e \in \overline{B_a}$ .

Admissibility of the optimal policy implies that  $V^*(\zeta)$  is bounded over all compact subsets  $K \subset \mathbb{R}^{2n}$ . Since the desired trajectory is bounded,  $t \mapsto V_t^*(e, t)$  is uniformly bounded for all  $e \in \overline{B_a}$ . To establish that  $e \mapsto V_t^*(e, t)$  is continuous, uniformly in  $t$ , let  $\chi_{e_o} \subset \mathbb{R}^n$  be a compact set containing  $e_o$ . Since  $x_d$  is bounded,  $x_d \in \chi_{x_d}$ , where  $\chi_{x_d} \subset \mathbb{R}^n$  is compact. Since  $V^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}_{\geq 0}$  is continuous, and  $\chi_{e_o} \times \chi_{x_d} \subset \mathbb{R}^{2n}$  is compact,  $V^*$  is uniformly continuous on  $\chi_{e_o} \times \chi_{x_d}$ . Thus,  $\forall \varepsilon > 0, \exists \varsigma > 0$ , such that  $\forall [e_o^T, x_d^T]^T, [e_1^T, x_d^T]^T \in \chi_{e_o} \times \chi_{x_d}, d_{\chi_{e_o} \times \chi_{x_d}}\left([e_o^T, x_d^T]^T, [e_1^T, x_d^T]^T\right) < \varsigma \implies d_{\mathbb{R}}\left(V^*\left([e_o^T, x_d^T]^T\right), V^*\left([e_1^T, x_d^T]^T\right)\right) < \varepsilon$ . Thus, for each  $e_o \in \mathbb{R}^n$ , there exists a  $\varsigma > 0$  independent of  $x_d$ , that establishes the continuity of  $e \mapsto V^*\left([e^T, x_d^T]^T\right)$  at  $e_o$ . Thus,  $e \mapsto V^*\left([e^T, x_d^T]^T\right)$  is continuous, uniformly in  $x_d$ , and hence, using (3.68),  $e \mapsto V_t^*(e, t)$  is continuous, uniformly in  $t$ . Using Lemma A.1, (3.69a), and (3.69b), there exists a positive definite function  $\alpha : \mathbb{R}^n \rightarrow \mathbb{R}_{\geq 0}$  such that  $V_t^*(e, t) < \alpha(e), \forall (e, t) \in \mathbb{R}^n \times \mathbb{R}_{\geq 0}$ . Using [8, Lemma 4.3] it can be shown that there exists a class  $\mathcal{K}$  function  $\bar{\gamma} : [0, a] \rightarrow \mathbb{R}_{\geq 0}$  such that  $\alpha(e) \leq \bar{\gamma}(\|e\|)$ , which implies (3.69c).  $\square$

### A.1.5 Proof of Lemma 3.15

*Proof* Let the constants  $\varpi_0 - \varpi_6$  be defined as

$$\begin{aligned}
\varpi_0 &= \frac{(1 - 6nT^2 L_F^2)}{2}, \\
\varpi_1 &= \frac{3n}{4} \sup_t \|g R^{-1} G^T \nabla_\zeta \sigma^T\|^2, \\
\varpi_2 &= \frac{3n^2 T^2 (dL_F + \sup_t \|gg_d^+ (h_d - f_d) - \frac{1}{2} g R^{-1} G^T \nabla_\zeta \sigma^T W - h_d\|)^2}{n}, \\
\varpi_3 &= \frac{(1 - 6L (k_{a1} + k_{a2})^2 T^2)}{2}, \\
\varpi_4 &= \frac{6Lk_{a1}T^2}{\left(1 - 6L (k_c \bar{\varphi} T)^2 / (\nu \underline{\varphi})^2\right)}, \\
\varpi_5 &= \frac{18 (k_{a1} L k_c \bar{\varphi} \epsilon L_F T^2)^2}{\nu \underline{\varphi} \left(1 - 6L (k_c \bar{\varphi} T)^2 / (\nu \underline{\varphi})^2\right)}, \\
\varpi_6 &= \frac{18 (L k_{a1} k_c \bar{\varphi} (\bar{\epsilon} L_F d + \iota_5) T^2)^2}{\nu \underline{\varphi} \left(1 - 6L (k_c \bar{\varphi} T)^2 / (\nu \underline{\varphi})^2\right)} + 3L (k_{a2} \bar{W} T)^2.
\end{aligned}$$

Using the definition of the controller in (3.57), the tracking error dynamics can be expressed as

$$\dot{e} = f + \frac{1}{2} g R^{-1} G^T \sigma'^T \tilde{W}_a + gg_d^+ (h_d - f_d) - \frac{1}{2} g R^{-1} G^T \sigma'^T W - h_d.$$

On any compact set, the tracking error derivative can be bounded above as

$$\|\dot{e}\| \leq L_F \|e\| + L_W \|\tilde{W}_a\| + L_e,$$

where  $L_e = L_F \|x_d\| + \|gg_d^+ (h_d - f_d) - \frac{1}{2} g R^{-1} G^T \sigma'^T W - h_d\|$  and  $L_W = \frac{1}{2} \|g R^{-1} G^T \sigma'^T\|$ . Using the fact that  $e$  and  $\tilde{W}_a$  are continuous functions of time, on the interval  $[t, t + T]$ , the time derivative of  $e$  can be bounded as

$$\|\dot{e}\| \leq L_F \sup_{\tau \in [t, t+T]} \|e(\tau)\| + L_W \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\| + L_e.$$

Since the infinity norm is less than the 2-norm, the derivative of the  $j$ th component of  $\dot{e}$  is bounded as

$$\dot{e}_j \leq L_F \sup_{\tau \in [t, t+T]} \|e(\tau)\| + L_W \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\| + L_e.$$

Thus, the maximum and the minimum value of  $e_j$  are related as

$$\begin{aligned} \sup_{\tau \in [t, t+T]} |e_j(\tau)| &\leq \inf_{\tau \in [t, t+T]} |e_j(\tau)| \\ &+ \left( L_F \sup_{\tau \in [t, t+T]} \|e(\tau)\| + L_W \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\| + L_e \right) T. \end{aligned}$$

Squaring the above expression and using the inequality  $(x + y)^2 \leq 2x^2 + 2y^2$

$$\begin{aligned} \sup_{\tau \in [t, t+T]} |e_j(\tau)|^2 &\leq 2 \inf_{\tau \in [t, t+T]} |e_j(\tau)|^2 \\ &+ 2 \left( L_F \sup_{\tau \in [t, t+T]} \|e(\tau)\| + L_W \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\| + L_e \right)^2 T^2. \end{aligned}$$

Summing over  $j$ , and using the facts that  $\sup_{\tau \in [t, t+T]} \|e(\tau)\|^2 \leq \sum_{j=1}^n \sup_{\tau \in [t, t+T]} |e_j(\tau)|^2$  and  $\inf_{\tau \in [t, t+T]} \sum_{j=1}^n |e_j(\tau)|^2 \leq \inf_{\tau \in [t, t+T]} \|e(\tau)\|^2$ ,

$$\begin{aligned} \sup_{\tau \in [t, t+T]} \|e(\tau)\|^2 &\leq 2 \inf_{\tau \in [t, t+T]} \|e(\tau)\|^2 \\ &+ 2 \left( L_F \sup_{\tau \in [t, t+T]} \|e(\tau)\|^2 + L_W \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2 + L_e \right)^2 n T^2. \end{aligned}$$

Using the inequality  $(x + y + z)^2 \leq 3x^2 + 3y^2 + 3z^2$ , (3.70) is obtained.

Using a similar procedure on the dynamics for  $\tilde{W}_a$ ,

$$\begin{aligned} - \inf_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2 &\leq - \frac{(1 - 6N(\eta_{a1} + \eta_{a2})^2 T^2)}{2} \sup_{\tau \in [t, t+T]} \|\tilde{W}_a(\tau)\|^2 \\ &+ 3N\eta_{a1}^2 \sup_{\tau \in [t, t+T]} \|\tilde{W}_c(\tau)\|^2 T^2 + 3N\eta_{a2}^2 W^2 T^2. \quad (\text{A.10}) \end{aligned}$$

Similarly, the dynamics for  $\tilde{W}_c$  yield

$$\begin{aligned} \sup_{\tau \in [t, t+T]} \|\tilde{W}_c(\tau)\|^2 &\leq \frac{2}{\left(1 - \frac{6N\eta_c^2 \bar{\varphi}^2 T^2}{\nu^2 \underline{\varphi}^2}\right)} \inf_{\tau \in [t, t+T]} \|\tilde{W}_c(\tau)\|^2 \\ &+ \frac{6NT^2 \eta_c^2 \bar{\varphi}^2 \bar{\epsilon}^2 L_F^2}{\nu \underline{\varphi} \left(1 - \frac{6N\eta_c^2 \bar{\varphi}^2 T^2}{\nu^2 \underline{\varphi}^2}\right)} \sup_{\tau \in [t, t+T]} \|e(\tau)\|^2 \\ &+ \frac{6NT^2 \eta_c^2 \bar{\varphi}^2 (\bar{\epsilon}' L_F d + \iota_5)^2}{\nu \underline{\varphi} \left(1 - \frac{6N\eta_c^2 \bar{\varphi}^2 T^2}{\nu^2 \underline{\varphi}^2}\right)}. \quad (\text{A.11}) \end{aligned}$$

Substituting (A.11) into (A.10), (3.71) can be obtained.  $\square$

### A.1.6 Proof of Lemma 3.16

*Proof* Let the constants  $\varpi_7 - \varpi_9$  be defined as  $\varpi_7 = \frac{\nu^2 \varphi^2}{2(\nu^2 \varphi^2 + k_c \bar{\varphi}^2 T^2)}$ ,  $\varpi_8 = 3\bar{\epsilon}^2 L_F^2$ , and  $\varpi_9 = 2(\iota_5^2 + \bar{\epsilon}^2 L_F^2 d^2)$ . The integrand on the left hand side can be written as

$$\tilde{W}_c^T(\tau) \psi(\tau) = \tilde{W}_c^T(t) \psi(\tau) + (\tilde{W}_c^T(\tau) - \tilde{W}_c^T(t)) \psi(\tau).$$

Using the inequality  $(x + y)^2 \geq \frac{1}{2}x^2 - y^2$  and integrating,

$$\begin{aligned} \int_t^{t+T} (\tilde{W}_c^T(\tau) \psi(\tau))^2 d\tau &\geq \frac{1}{2} \tilde{W}_c^T(t) \left( \int_t^{t+T} (\psi(\tau) \psi(\tau)^T) d\tau \right) \tilde{W}_c(t) \\ &\quad - \int_t^{t+T} \left( \left( \int_t^\tau \dot{\tilde{W}}_c(\sigma) d\sigma \right)^T \psi(\tau) \right)^2 d\tau. \end{aligned}$$

Substituting the dynamics for  $\tilde{W}_c$  from (3.66) and using the persistence of excitation condition in Assumption 3.13,

$$\begin{aligned} \int_t^{t+T} (\tilde{W}_c^T(\tau) \psi(\tau))^2 d\tau &\geq \frac{1}{2} \underline{\psi} \tilde{W}_c^T(t) \tilde{W}_c(t) \\ &\quad - \int_t^{t+T} \left( \left( \int_t^\tau \left( \frac{\eta_c \Gamma(\sigma) \psi(\sigma) \Delta(\sigma)}{\sqrt{1 + \nu \omega(\sigma)^T \Gamma(\sigma) \omega(\sigma)}} \right. \right. \right. \\ &\quad \left. \left. \left. - \eta_c \Gamma(\sigma) \psi(\sigma) \psi^T(\sigma) \tilde{W}_c(\sigma) \right. \right. \right. \\ &\quad \left. \left. \left. + \frac{\eta_c \Gamma(\sigma) \psi(\sigma) \tilde{W}_a^T \mathcal{G}_\sigma \tilde{W}_a}{4\sqrt{1 + \nu \omega(\sigma)^T \Gamma(\sigma) \omega(\sigma)}} \right. \right. \right. \\ &\quad \left. \left. \left. - \frac{\eta_c \Gamma(\sigma) \psi(\sigma) \epsilon'(\sigma) F(\sigma)}{\sqrt{1 + \nu \omega(\sigma)^T \Gamma(\sigma) \omega(\sigma)}} \right) d\sigma \right)^T \psi(\tau) \right)^2, \end{aligned}$$

where  $\Delta \triangleq \frac{1}{4}\epsilon' \mathcal{G} \epsilon'^T + \frac{1}{2}W^T \sigma' \mathcal{G} \epsilon'^T$ . Using the inequality  $(x + y + w - z)^2 \leq 2x^2 + 6y^2 + 6w^2 + 6z^2$ ,

$$\int_t^{t+T} (\tilde{W}_c^T(\tau) \psi(\tau))^2 d\tau \geq \frac{1}{2} \underline{\psi} \tilde{W}_c^T(t) \tilde{W}_c(t)$$

$$\begin{aligned}
& - \int_t^{t+T} 2 \left( \int_t^\tau \eta_c \tilde{W}_c^T(\sigma) \psi(\sigma) \psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau) d\sigma \right)^2 d\tau \\
& - 6 \int_t^{t+T} \left( \int_t^\tau \frac{\eta_c \Delta^T(\sigma) \psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau)}{\sqrt{1 + \nu \omega(\sigma)^T \Gamma(\sigma) \omega(\sigma)}} d\sigma \right)^2 d\tau \\
& - 6 \int_t^{t+T} \left( \int_t^\tau \frac{\eta_c F^T(\sigma) \epsilon'^T(\sigma) \psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau)}{\sqrt{1 + \nu \omega(\sigma)^T \Gamma(\sigma) \omega(\sigma)}} d\sigma \right)^2 d\tau \\
& - 6 \int_t^{t+T} \left( \int_t^\tau \frac{\eta_c \tilde{W}_a^T(\sigma) \mathcal{G}_\sigma(\sigma) \tilde{W}_a(\sigma) \psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau)}{\sqrt{1 + \nu \omega(\sigma)^T \Gamma(\sigma) \omega(\sigma)}} d\sigma \right)^2 d\tau.
\end{aligned}$$

Using the Cauchy–Schwarz inequality, the Lipschitz property, the fact that  $\frac{1}{\sqrt{1 + \nu \omega^T \Gamma \omega}} \leq 1$ , and the bounds in (3.67),

$$\begin{aligned}
\int_t^{t+T} \left( \tilde{W}_c^T(\tau) \psi(\tau) \right)^2 d\tau & \geq \frac{1}{2} \underline{\psi} \tilde{W}_c^T(t) \tilde{W}_c(t) - 6 \int_t^{t+T} \left( \int_t^\tau \frac{\eta_c \iota_5 \bar{\varphi}}{\nu \underline{\varphi}} d\sigma \right)^2 d\tau \\
& - \int_t^{t+T} 2\eta_c^2 \left( \int_t^\tau \left( \tilde{W}_c^T(\sigma) \psi(\sigma) \right)^2 d\sigma \int_t^\tau (\psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau))^2 d\sigma \right) d\tau \\
& - \int_t^{t+T} 6\eta_c^2 \iota_2^2 \left( \int_t^\tau \left\| \tilde{W}_a(\sigma) \right\|^4 d\sigma \int_t^\tau (\psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau))^2 d\sigma \right) d\tau \\
& - \int_t^{t+T} 6\eta_c^2 \bar{\epsilon}^2 \left( \int_t^\tau \|F(\sigma)\|^2 d\sigma \int_t^\tau (\psi^T(\sigma) \Gamma^T(\sigma) \psi(\tau))^2 d\sigma \right) d\tau.
\end{aligned}$$

Rearranging,

$$\begin{aligned}
\int_t^{t+T} \left( \tilde{W}_c^T(\tau) \psi(\tau) \right)^2 d\tau & \geq \frac{1}{2} \underline{\psi} \tilde{W}_c^T(t) \tilde{W}_c(t) - 3\eta_c^2 A^4 \bar{\varphi}^2 \iota_5^2 T^3 \\
& - 2\eta_c^2 A^4 \bar{\varphi}^2 \int_t^{t+T} (\tau - t) \int_t^\tau \left( \tilde{W}_c^T(\sigma) \psi(\sigma) \right)^2 d\sigma d\tau - 3\eta_c^2 A^4 \bar{\varphi}^2 \bar{\epsilon}^2 L_F^2 d^2 T^3 \\
& - 6\eta_c^2 \iota_2^2 A^4 \bar{\varphi}^2 \int_t^{t+T} (\tau - t) \int_t^\tau \left\| \tilde{W}_a(\sigma) \right\|^4 d\sigma d\tau - 6\eta_c^2 \bar{\epsilon}^2 L_F^2 A^4 \bar{\varphi}^2 \int_t^{t+T} (\tau - t) \int_t^\tau \|e\|^2 d\sigma d\tau,
\end{aligned}$$

where  $A = \frac{1}{\sqrt{\nu\varphi}}$ . Changing the order of integration,

$$\begin{aligned} \int_t^{t+T} \left( \tilde{W}_c^T(\tau) \psi(\tau) \right)^2 d\tau &\geq \frac{1}{2} \underline{\psi} \tilde{W}_c^T(t) \tilde{W}_c(t) - \eta_c^2 A^4 \bar{\varphi}^2 T^2 \int_t^{t+T} \left( \tilde{W}_c^T(\sigma) \psi(\sigma) \right)^2 d\sigma \\ &\quad - 3\eta_c^2 A^4 \bar{\varphi}^2 \bar{\epsilon}^2 L_F^2 T^2 \int_t^{t+T} \|e(\sigma)\|^2 d\sigma - 3\eta_c^2 \iota_2^2 A^4 \bar{\varphi}^2 T^2 \int_t^{t+T} \|\tilde{W}_a(\sigma)\|^4 d\sigma \\ &\quad - 2\eta_c^2 A^4 \bar{\varphi}^2 T^3 \left( \iota_5^2 + \bar{\epsilon}^2 L_F^2 d^2 \right). \end{aligned}$$

Reordering the terms, the inequality in Lemma 3.16 is obtained.  $\square$

### A.1.7 Proof of Theorem 3.20

*Proof* To facilitate the subsequent development, let the gains  $k$  and  $\gamma$  be split as  $k \triangleq k_1 + k_2$  and  $\gamma \triangleq \gamma_1 + \gamma_2$ . Let  $\lambda \triangleq \min\{\alpha\gamma_1 - \zeta_5, k_1 - \zeta_6\}$ ,  $\rho(\|z\|)^2 \triangleq \rho_1(\|z\|)^2 + \rho_2(\|z\|)^2$ , and  $\eta \triangleq \min\{k_2, \frac{\alpha\gamma_2}{\beta_2^2}\}$ . Let  $y(t)$  for  $t \in [t_0, \infty)$  denote a Filippov solution to the differential equation in (3.111) that satisfies  $y(t_0) \in \mathcal{S}$ . Using Filippov's theory of differential inclusions [1, 2], the existence of solutions can be established for  $\dot{y} \in K[h](y, t)$ , where  $K[h](y, t) \triangleq \bigcap_{\delta>0} \bigcap_{\mu S_m=0} \overline{coh}(B_\delta(y) \setminus S_m, t)$ , where  $\bigcap_{\mu S_m=0}$  denotes the intersection of all sets  $S_m$  of Lebesgue measure zero [3, 4]. The time derivative of (3.109) along the Filippov trajectory  $y(\cdot)$  exists almost everywhere (a.e.), and  $\dot{\tilde{V}}_I \stackrel{\text{a.e.}}{\in} \tilde{V}_I$  where

$$\dot{\tilde{V}}_I = \bigcap_{\xi \in \partial V_I(y)} \xi^T K \left[ \dot{e}_f^T \dot{\tilde{x}}^T \frac{1}{2} P^{-\frac{1}{2}} \dot{P} \frac{1}{2} Q^{-\frac{1}{2}} \dot{Q} \right]^T, \quad (\text{A.12})$$

where  $\partial V_I$  is the generalized gradient of  $V_I$  [5]. Since  $V_I$  is continuously differentiable, (A.12) can be simplified as [3]

$$\begin{aligned} \dot{\tilde{V}}_I &= \nabla_x V_I^T K \left[ \dot{e}_f^T \dot{\tilde{x}}^T \frac{1}{2} P^{-\frac{1}{2}} \dot{P} \frac{1}{2} Q^{-\frac{1}{2}} \dot{Q} \right]^T \\ &= \left[ e_f^T \gamma \tilde{x}^T 2P^{\frac{1}{2}} 2Q^{\frac{1}{2}} \right] K \left[ \dot{e}_f^T \dot{\tilde{x}}^T \frac{1}{2} P^{-\frac{1}{2}} \dot{P} \frac{1}{2} Q^{-\frac{1}{2}} \dot{Q} \right]^T. \end{aligned}$$

Using the calculus for  $K[\cdot]$  from [4], and substituting the dynamics from (3.99) and (3.107), yields

$$\begin{aligned} \dot{\tilde{V}}_I &\subset e_f^T (\tilde{N} + N_{B1} + \hat{N}_{B2} - ke_f - \beta_1 K[\text{sgn}](\tilde{x}) - \gamma \tilde{x}^T (e_f - \alpha \tilde{x}) \\ &\quad - e_f^T (N_{B1} - \beta_1 K[\text{sgn}](\tilde{x})) - \tilde{x}^T N_{B2} + \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}\| \\ &\quad - \frac{1}{2} \alpha \left[ \text{tr}(\tilde{W}_f^T \Gamma_{wf}^{-1} \hat{W}_f) + \text{tr}(\tilde{V}_f^T \Gamma_{vf}^{-1} \hat{V}_f) \right], \end{aligned} \quad (\text{A.13})$$

where  $K[\text{sgn}](\tilde{x}) = \text{SGN}(\tilde{x})$ . Substituting (3.98), canceling common terms, and rearranging the expression yields

$$\begin{aligned} \dot{\tilde{V}}_I &\stackrel{\text{a.e.}}{\leq} -\alpha \gamma \tilde{x}^T \tilde{x} - ke_f^T e_f + e_f^T \tilde{N} + \frac{1}{2} \alpha \tilde{x}^T \tilde{W}_f^T \nabla_x \hat{\sigma}_f \hat{V}_f^T \dot{\tilde{x}} + \frac{1}{2} \alpha \tilde{x}^T \hat{W}_f^T \nabla_x \hat{\sigma}_f \tilde{V}_f^T \dot{\tilde{x}} \\ &\quad + \dot{\tilde{x}}^T (\hat{N}_{B2} - N_{B2}) + \beta_2 \rho_2(\|z\|) \|z\| \|\tilde{x}\| - \frac{1}{2} \alpha \text{tr}(\tilde{W}_f^T \nabla_x \hat{\sigma}_f \hat{V}_f^T \dot{\tilde{x}} \tilde{x}^T) \\ &\quad - \frac{1}{2} \alpha \text{tr}(\tilde{V}_f^T \dot{\tilde{x}} \tilde{x}^T \hat{W}_f^T \nabla_x \hat{\sigma}_f), \end{aligned} \quad (\text{A.14})$$

where  $\hat{\sigma}_f \triangleq \sigma_f(\hat{V}_f^T \dot{\tilde{x}})$ . The set inclusion in (A.13) reduces to the scalar inequality in (A.14) because the right hand side of (A.13) is set valued only on the Lebesgue negligible set of times when  $e_f^T K[\text{sgn}](\tilde{x}) - e_f^T K[\text{sgn}](\tilde{x}) \neq 0$ . The set of times  $\Lambda \triangleq \{t \in [0, \infty) \mid e_f(t)^T K[\text{sgn}](\tilde{x}(t)) - e_f(t)^T K[\text{sgn}](\tilde{x}(t)) \neq 0\} \subset [0, \infty)$  is equivalent to the set of times  $\{t \mid \tilde{x}(t) = 0 \wedge e_f(t) \neq 0\}$ . From (3.96), this set can also be represented by  $\{t \mid \tilde{x}(t) = 0 \wedge \dot{\tilde{x}}(t) \neq 0\}$ . Provided  $\tilde{x}$  is continuously differentiable, it can be shown that the set of time instances  $\{t \mid \tilde{x}(t) = 0 \wedge \dot{\tilde{x}}(t) \neq 0\}$  is isolated, and thus, measure zero. This implies that the set  $\Lambda$  is measure zero. [6]

Substituting for  $k = k_1 + k_2$  and  $\gamma = \gamma_1 + \gamma_2$ , using (3.98), (3.103), and (3.105), and completing the squares, the expression in (A.14) can be upper bounded as

$$\dot{\tilde{V}}_I \stackrel{\text{a.e.}}{\leq} -(\alpha \gamma_1 - \zeta_5) \|\tilde{x}\|^2 - (k_1 - \zeta_6) \|e_f\|^2 + \frac{\rho_1(\|z\|)^2}{4k_2} \|z\|^2 + \frac{\beta_2^2 \rho_2(\|z\|)^2}{4\alpha \gamma_2} \|z\|^2. \quad (\text{A.15})$$

Provided the sufficient conditions in (3.112) are satisfied, the expression in (A.15) can be rewritten as

$$\dot{\tilde{V}}_I \stackrel{\text{a.e.}}{\leq} -\lambda \|z\|^2 + \frac{\rho(\|z\|)^2}{4\eta} \|z\|^2 \stackrel{\text{a.e.}}{\leq} -U(y), \quad \forall y \in \mathcal{D}. \quad (\text{A.16})$$

In (A.16),  $U(y) = c \|z\|^2$  is a continuous positive semi-definite function defined on  $\mathcal{D}$ , where  $c$  is a positive constant.

The inequalities in (3.110) and (A.16) can be used to show that  $t \mapsto V_I(y(t)) \in \mathcal{L}_\infty$ ; hence,  $\tilde{x}(\cdot), r(\cdot) \in \mathcal{L}_\infty$ . Using (3.96), standard linear analysis can be used to show that  $\dot{\tilde{x}}(\cdot) \in \mathcal{L}_\infty$ , and since  $\dot{x}(\cdot) \in \mathcal{L}_\infty$ ,  $\dot{\tilde{x}}(\cdot) \in \mathcal{L}_\infty$ . Since  $\tilde{W}_f(\cdot), \tilde{V}_f(\cdot) \in \mathcal{L}_\infty$  from the use of projection in (3.98),  $t \mapsto \sigma_f(\hat{V}_f^T(t) \dot{\tilde{x}}(t)) \in \mathcal{L}_\infty$  from Property

**2.3**, and  $u(\cdot) \in \mathcal{L}_\infty$  from Assumption 3.19, (3.92) can be used to conclude that  $\mu(\cdot) \in \mathcal{L}_\infty$ . Using (3.97) and the above bounds it can be shown that  $\dot{e}_f(\cdot) \in \mathcal{L}_\infty$ . From (A.16), [7, Corollary 1] can be invoked to show that  $y(\cdot) \in \mathcal{L}_\infty$ , provided  $y(0) \in \mathcal{S}$ . Furthermore,

$$\|\tilde{x}(t)\|, \|\dot{\tilde{x}}(t)\|, \|e_f(t)\| \rightarrow 0 \text{ as } t \rightarrow \infty,$$

provided  $y(t_0) \in \mathcal{S}$ .  $\square$

## A.2 Chapter 4 Supplementary Material

### A.2.1 Algorithm for Gain Selection

In the following, the notation  $\{\varpi\}_i$  for any parameter  $\varpi$  denotes the value of  $\varpi$  computed in the  $i$ th iteration.

---

#### Algorithm A.2 Gain Selection

---

##### First iteration:

Given  $\bar{z} \in \mathbb{R}_{\geq 0}$  such that  $\|Z(t_0)\| < \bar{z}$ , let  $\mathcal{Z}_1 \triangleq \{\xi \in \mathbb{R}^{2n+2L+p} \mid \|\xi\| \leq \underline{v}^{-1}(\bar{v}(\bar{z}))\}$ . Using  $\mathcal{Z}_1$ , compute the bounds in (4.17) and select the gains according to (4.18). If  $\left\{\sqrt{\frac{\underline{v}}{v_l}}\right\}_1 \leq \bar{z}$ , set  $\mathcal{Z} = \mathcal{Z}_1$  and terminate.

##### Second iteration:

If  $\bar{z} < \left\{\sqrt{\frac{\underline{v}}{v_l}}\right\}_1$ , let  $\mathcal{Z}_2 \triangleq \left\{\xi \in \mathbb{R}^{2n+2L+p} \mid \|\xi\| \leq \underline{v}^{-1}\left(\bar{v}\left(\left\{\sqrt{\frac{\underline{v}}{v_l}}\right\}_1\right)\right)\right\}$ . Using  $\mathcal{Z}_2$ , compute the bounds in (4.17) and select the gains according to (4.17). If  $\left\{\frac{\underline{v}}{v_l}\right\}_2 \leq \left\{\frac{\underline{v}}{v_l}\right\}_1$ , set  $\mathcal{Z} = \mathcal{Z}_2$  and terminate.

##### Third iteration:

If  $\left\{\frac{\underline{v}}{v_l}\right\}_2 > \left\{\frac{\underline{v}}{v_l}\right\}_1$ , increase the number of neural network neurons to  $\{L\}_3$  to ensure  $\{L_Y\}_2 \{\bar{e}\}_3 \leq \{L_Y\}_2 \{\bar{e}\}_2, \forall i = 1, \dots, N$ , increase the constant  $\zeta_3$  to ensure  $\frac{\{L_Y\}_2}{\{\zeta_3\}_3} \leq \frac{\{L_Y\}_2}{\{\zeta_3\}_2}$ , and increase the gains  $K$  and  $\eta_{a1}$  to satisfy the gain conditions in (4.18). Provided the constant  $c$  is large enough and  $D$  is small enough, these adjustments ensure  $\{\iota\}_3 \leq \{\iota\}_2$ . Set  $\mathcal{Z} = \left\{\xi \in \mathbb{R}^{2n+2L+p} \mid \|\xi\| \leq \underline{v}^{-1}\left(\bar{v}\left(\left\{\sqrt{\frac{\underline{v}}{v_l}}\right\}_2\right)\right)\right\}$  and terminate.

---

### A.2.2 Algorithm for Gain Selection - N-Player Game

In the following, the notation  $\{\varpi\}_i$  for any parameter  $\varpi$  denotes the value of  $\varpi$  computed in the  $i$ th iteration.

---

**Algorithm A.3** Gain Selection
 

---

First iteration:

Given  $z \in \mathbb{R}_{\geq 0}$  such that  $\|Z(t_0)\| < z$ , let  $\mathcal{Z}_1 \triangleq \left\{ \xi \in \mathbb{R}^{2n+2N \sum_i \{L_i\}_1 + p_\theta} \mid \|\xi\| \leq \underline{v}^{-1}(\bar{v}(z)) \right\}$ .

Using  $\mathcal{Z}_1$ , compute the bounds in (4.60) and select the gains according to (4.61). If  $\left\{ \frac{\underline{v}}{v_l} \right\}_1 \leq z$ , set  $\mathcal{Z} = \mathcal{Z}_1$  and terminate.

Second iteration:

If  $z < \left\{ \frac{\underline{v}}{v_l} \right\}_1$ , let  $\mathcal{Z}_2 \triangleq \left\{ \xi \in \mathbb{R}^{2n+2N \sum_i \{L_i\}_1 + p_\theta} \mid \|\xi\| \leq \underline{v}^{-1}(\bar{v}\left(\left\{ \frac{\underline{v}}{v_l} \right\}_1\right)) \right\}$ . Using  $\mathcal{Z}_2$ , compute the bounds in (4.60) and select the gains according to (4.61). If  $\left\{ \frac{\underline{v}}{v_l} \right\}_2 \leq \left\{ \frac{\underline{v}}{v_l} \right\}_1$ , set  $\mathcal{Z} = \mathcal{Z}_2$  and terminate.

Third iteration:

If  $\left\{ \frac{\underline{v}}{v_l} \right\}_2 > \left\{ \frac{\underline{v}}{v_l} \right\}_1$ , increase the number of neural network neurons to  $\{p_{Wi}\}_3$  to ensure  $\{L_Y\}_2 \{\bar{\epsilon}_i\}_3 \leq \{L_Y\}_2 \{\bar{\epsilon}_i\}_2, \forall i = 1, \dots, N$ , decrease the constant  $\zeta_3$  to ensure  $\{L_Y\}_2 \{\zeta_3\}_3 \leq \{L_Y\}_2 \{\zeta_3\}_2$ , and increase the gain  $k_\theta$  to satisfy the gain conditions in (4.61). These adjustments ensure  $\{\underline{v}\}_3 \leq \{\underline{v}\}_2$ . Set  $\mathcal{Z} = \left\{ \xi \in \mathbb{R}^{2n+2N \sum_i \{L_i\}_3 + p_\theta} \mid \|\xi\| \leq \underline{v}^{-1}(\bar{v}\left(\left\{ \frac{\underline{v}}{v_l} \right\}_2\right)) \right\}$  and terminate.

---

### A.2.3 System Identification

#### Concurrent learning-based parameter update

In traditional adaptive control, convergence of the estimates  $\hat{\theta}$  to their true values  $\theta$  is ensured by assuming that a *persistent* excitation condition is satisfied [9–11]. To ensure convergence under a *finite* excitation condition, this result employs a concurrent learning-based approach to update the parameter estimates using recorded input-output data [12–14].

**Assumption A.2** ([13, 14]) A collection  $\mathcal{H}_{id}$  of triplets  $\{(a_j, b_j, c_j) \mid a_j \in \mathbb{R}^n, b_j \in \mathbb{R}^n, c_j \in \mathbb{R}^m\}_{j=1}^M$  that satisfies

$$\begin{aligned} \text{rank} \left( \sum_{j=1}^M Y^T(a_j) Y(a_j) \right) &= p, \\ \|b_j - f(a_j) + g(a_j)c_j\| &< \bar{d}, \quad \forall j, \end{aligned} \tag{A.17}$$

is available a priori, where  $\bar{d} \in \mathbb{R}_{\geq 0}$  is a positive constant. Since  $\theta \in \Theta$ , where  $\Theta$  is a compact set, the assumption that  $\bar{d}$  is independent of  $\theta$  is justified.

To satisfy Assumption A.2, data recorded in a previous run of the system can be utilized, or the data stack can be recorded by running the system using a different known stabilizing controller for a finite amount of time until the recorded data satisfies the rank condition (A.17).

In some cases, a data stack may not be available a priori. For such applications, the data stack can be recorded online (i.e., the points  $a_j$  and  $c_j$  can be recorded along the system trajectory as  $a_j = x(t_j)$  and  $c_j = u(t_j)$  for some  $t_j \in \mathbb{R}_{\geq t_0}$ ). Provided

the system states are exciting over a finite time interval  $t \in [t_0, t_0 + \bar{t}]$  (versus  $t \in [t_0, \infty)$  as in traditional persistence of excitation-based approaches) until the data stack satisfies (A.17), then a modified form of the controller developed in Sect. A.2.4 can be used over the time interval  $t \in [t_0, t_0 + \bar{t}]$ , and the controller developed in Sect. 4.3.3 can be used thereafter.

Based on Assumption A.2, the update law for the parameter estimates is designed as

$$\dot{\tilde{\theta}} = \frac{\Gamma_\theta k_\theta}{M} \sum_{j=1}^M Y^T(a_j) (b_j - g(a_j) c_j - Y(a_j) \hat{\theta}), \quad (\text{A.18})$$

where  $\Gamma_\theta \in \mathbb{R}^{p \times p}$  is a constant positive definite adaptation gain matrix and  $k_\theta \in \mathbb{R}$  is a constant positive concurrent learning gain. From (1.9) and the definition of  $\tilde{\theta}$ , the bracketed term in (A.18), can be expressed as  $b_j - g(a_j) c_j - Y(a_j) \hat{\theta} = Y(a_j) \tilde{\theta} + d_j$ , where  $d_j \triangleq b_j - f(a_j) + g(a_j) c_j \in \mathbb{R}^n$ , and the parameter update law in (A.18) can be expressed in the advantageous form

$$\dot{\tilde{\theta}} = \frac{\Gamma_\theta k_\theta}{M} \left( \sum_{j=1}^M Y^T(a_j) Y(a_j) \right) \tilde{\theta} + \frac{\Gamma_\theta k_\theta}{M} \sum_{j=1}^M Y^T(a_j) d_j. \quad (\text{A.19})$$

The rate of convergence of the parameter estimates to a neighborhood of their ideal values is directly (and the ultimate bound is inversely) proportional to the minimum singular value of the matrix  $\sum_{j=1}^M Y^T(a_j) Y(a_j)$ ; hence, the performance of the estimator can be improved online if a triplet  $(a_j, b_j, c_j)$  in  $\mathcal{H}_{id}$  is replaced with an updated triplet  $(a_k, b_k, c_k)$  that increases the singular value of  $\sum_{j=1}^M Y^T(a_j) Y(a_j)$ . The stability analysis in Sect. 4.3.4 allows for this approach through the use of a singular value maximizing algorithm (cf. [12, 14]).

### Convergence analysis

Let  $V_\theta : \mathbb{R}^{n+p} \rightarrow \mathbb{R}_{\geq 0}$  be a positive definite continuously differentiable candidate Lyapunov function defined as

$$V_\theta(\tilde{\theta}) \triangleq \frac{1}{2} \tilde{\theta}^T \Gamma_\theta^{-1} \tilde{\theta}.$$

The following bounds on the Lyapunov function can be established:

$$\frac{\gamma}{2} \|\tilde{\theta}\|^2 \leq V_\theta(\tilde{\theta}) \leq \frac{\bar{\gamma}}{2} \|\tilde{\theta}\|^2,$$

where  $\underline{\gamma}, \bar{\gamma} \in \mathbb{R}$  denote the minimum and the maximum eigenvalues of the matrix  $\Gamma_\theta^{-1}$ . Using (A.19), the Lyapunov derivative can be expressed as

$$\dot{V}_\theta = -\tilde{\theta}^T \frac{k_\theta}{M} \left( \sum_{j=1}^M Y^T(a_j) Y(a_j) \right) \tilde{\theta} - \frac{k_\theta}{M} \tilde{\theta}^T \sum_{j=1}^M Y^T(a_j) d_j.$$

Let  $\underline{y} \in \mathbb{R}$  be the minimum eigenvalue of  $\left( \frac{1}{M} \sum_{j=1}^M Y^T(a_j) Y(a_j) \right)$ . Since  $\left( \sum_{j=1}^M Y^T(a_j) Y(a_j) \right)$  is symmetric and positive semi-definite, (A.17) can be used to conclude that it is also positive definite, and hence  $\underline{y} > 0$ . Hence, the Lyapunov derivative can be bounded as

$$\dot{V}_0 \leq -\underline{y} k_\theta \|\tilde{\theta}\|^2 + k_\theta d_\theta \|\tilde{\theta}\|,$$

where  $d_\theta = \overline{dY}$ ,  $\overline{Y} = \max_{j=1,\dots,M} (\|Y(a_j)\|)$ . Hence,  $\|\tilde{\theta}\|$  exponentially decays to an ultimate bound as  $t \rightarrow \infty$ . If  $\mathcal{H}_{id}$  is updated with new data, the update law (A.19) forms a switched system. Provided (A.17) holds, and  $\mathcal{H}_{id}$  is updated using a singular value maximizing algorithm,  $V_\theta$  is a common Lyapunov function for the switched system (cf. [14]). The concurrent learning-based system identifier satisfies Assumption 4.1 with  $K = \underline{y} k_\theta$  and  $D = k_\theta d_\theta$ . To satisfy the last inequality in (4.18), the quantity  $\frac{\underline{v}}{v_l}$  needs to be small. Based on the definitions in (4.17), the quantity  $\frac{\underline{v}}{v_l}$  is proportional to  $\frac{D^2}{K^2}$ , which is proportional to  $\frac{d_\theta^2}{\underline{y}^2}$ . From the definitions of  $d_\theta$  and  $\underline{y}$ ,

$$\frac{d_\theta^2}{\underline{y}^2} = \bar{d}^2 \frac{\left( \sum_{j=1}^M \|Y(a_j)\| \right)^2}{\left( \lambda_{\min} \left\{ \sum_{j=1}^M Y^T(a_j) Y(a_j) \right\} \right)^2}.$$

Thus, in general, a small  $\bar{d}$  (i.e., accurate numerical differentiation) is required to obtain the result in Theorem 4.3.

#### A.2.4 Online Data Collection for System Identification

A data stack  $\mathcal{H}_{id}$  that satisfies conditions in (A.17) can be collected online provided the controller in (4.6) results in the system states being sufficiently exciting over a finite time interval  $[t_0, t_0 + \bar{t}] \subset \mathbb{R}$ . To collect the data stack, the first  $M$  values of the state, the control, and the corresponding numerically computed state derivative are added to the data stack. Then, the existing values are progressively replaced with new values using a singular value maximization algorithm. During this finite time interval, since a data stack is not available, an adaptive update law that ensures fast convergence of  $\tilde{\theta}$  to zero without persistence of excitation can not be developed. Hence, the system dynamics can not be directly estimated without persistence of excitation. Since extrapolation of the Bellman error to unexplored areas of the state-

space requires estimates of the system dynamics, without persistence of excitation, such extrapolation is not feasible during the time interval  $[t_0, t_0 + \bar{t}]$ .

However, evaluation of the Bellman error along the system trajectories does not explicitly depend on the parameters  $\theta$ . Estimation of the state derivative is enough to evaluate the Bellman error along system trajectories. This motivates the development of the following state derivative estimator

$$\begin{aligned}\dot{\hat{x}}_f &= gu + k_f \tilde{x}_f + \mu_f, \\ \dot{\mu}_f &= (k_f \alpha_f + 1) \tilde{x}_f,\end{aligned}\quad (\text{A.20})$$

where  $\hat{x}_f \in \mathbb{R}^n$  is an estimate of the state  $x$ ,  $\tilde{x}_f \triangleq x - \hat{x}_f$ , and  $k_f, \alpha_f, \gamma_f \in \mathbb{R}_{>0}$  are constant estimation gains. To facilitate the stability analysis, define a filtered error signal  $r \in \mathbb{R}^n$  as  $r \triangleq \dot{\tilde{x}}_f + \alpha_f \tilde{x}_f$ , where  $\dot{\tilde{x}}_f \triangleq \dot{x} - \dot{\hat{x}}_f$ . Using (1.9) and (A.20), the dynamics of the filtered error signal can be expressed as  $\dot{r} = -k_f r + \tilde{x}_f + f' f + f' g u + \alpha \dot{\tilde{x}}_f$ . The instantaneous Bellman error in (2.3) can be approximated along the state trajectory using the state derivative estimate as

$$\hat{\delta}_f = \omega_f^T \hat{W}_{cf} + x^T Q x + \hat{u}^T \left( x, \hat{W}_{af} \right) R \hat{u} \left( x, \hat{W}_{af} \right), \quad (\text{A.21})$$

where  $\omega_f \in \mathbb{R}^L$  is the regressor vector defined as  $\omega_f \triangleq \sigma'(x) \dot{\hat{x}}_f$ . During the interval  $[t_0, t_0 + \bar{t}]$ , the value function and the actor weights can be learned based on the approximate Bellman error in (A.21) provided the system states are exciting (i.e., if the following assumption is satisfied).

**Assumption A.3** There exists a time interval  $[t_0, t_0 + \bar{t}] \subset \mathbb{R}$  and positive constants  $\underline{\psi}, T \in \mathbb{R}$  such that closed-loop trajectories of the system in (1.9) with the controller  $u = \hat{u}^T \left( x, \hat{W}_{af} \right)$  along with the weight update laws

$$\begin{aligned}\dot{\hat{W}}_{cf} &= -\eta_{cf} \Gamma_f \frac{\omega_f}{\rho_f} \delta_f, \quad \dot{\Gamma}_f = \lambda_f \Gamma_f - \eta_{cf} \Gamma_f \frac{\omega_f \omega_f^T}{\rho_f} \Gamma_f, \\ \dot{\hat{W}}_{af} &= -\eta_{a1f} \left( \hat{W}_a - \hat{W}_c \right) - \eta_{a2f} \hat{W}_a,\end{aligned}\quad (\text{A.22})$$

and the state derivative estimator in (A.20) satisfy

$$\underline{\psi} I_L \leq \int_t^{t+T} \psi_f(\tau) \psi_f(\tau)^T d\tau, \quad \forall t \in [t_0, t_0 + \bar{t}], \quad (\text{A.23})$$

where  $\rho_f \triangleq 1 + \nu_f \omega_f^T \omega_f$  is the normalization term,  $\eta_{a1f}, \eta_{a2f}, \eta_{cf}, \nu_f \in \mathbb{R}$  are constant positive gains,  $\Gamma_f \in \mathbb{R}^{L \times L}$  is the least-squares gain matrix, and  $\psi_f \triangleq \frac{\omega_f}{\sqrt{1 + \nu_f \omega_f^T \omega_f}} \in \mathbb{R}^N$  is the regressor vector. Furthermore, there exists a set of time

instances  $\{t_1 \cdots t_M\} \subset [t_0, t_0 + \bar{t}]$  such that the data stack  $\mathcal{H}_{id}$  containing the values of state-action pairs and the corresponding numerical derivatives recorded at  $\{t_1 \cdots t_M\}$  satisfies the conditions in Assumption A.2.

Conditions similar to (A.23) are ubiquitous in online approximate optimal control literature. In fact, Assumption A.3 requires the regressor  $\psi_f$  to be exciting over a finite time interval, whereas the persistence of excitation conditions used in related results such as [15–19] require similar regressor vectors to be exciting over all  $t \in \mathbb{R}_{\geq t_0}$ .

On any compact set  $\chi \subset \mathbb{R}^n$  the function  $f$  is Lipschitz continuous; hence, there exist positive constants  $L_f, L_{df} \in \mathbb{R}$  such that

$$\|f(x)\| \leq L_f \|x\| \text{ and } \|f'(x)\| \leq L_{df}, \quad (\text{A.24})$$

$\forall x \in \chi$ . The update laws in (A.22) along with the excitation condition in (A.23) ensure that the adaptation gain matrix is bounded such that

$$\underline{\Gamma}_f \leq \|\Gamma_f\| \leq \bar{\Gamma}_f, \quad \forall t \in \mathbb{R}_{\geq t_0}, \quad (\text{A.25})$$

where (cf. [11, Proof of Corollary 4.3.2])

$$\underline{\Gamma}_f = \min \left\{ \eta_{cf} \underline{\psi} T, \lambda_{\min} \{ \Gamma_f(t_0) \} \right\} e^{-\lambda_f T}. \quad (\text{A.26})$$

The following positive constants are defined for brevity of notation.

$$\begin{aligned} \vartheta_8 &\triangleq \frac{L_{df}}{2} \overline{\|g R^{-1} g^T \sigma'^T\|}, \quad \vartheta_{10} \triangleq \frac{\overline{\|2W^T \sigma' G \epsilon'^T + G_\epsilon\|}}{4}, \\ \vartheta_9 &\triangleq \frac{\overline{\|W^T G_\sigma + \frac{1}{2} \epsilon' G^T \sigma'^T\|}}{2} + \eta_{a2f} \overline{W}, \\ \vartheta_{10} &\triangleq \frac{\overline{\|2W^T \sigma' G \epsilon'^T + G_\epsilon\|}}{4}, \\ \iota_f &\triangleq 2\eta_{cf}\vartheta_{10} + \frac{3\vartheta_9}{4(\eta_{a1f} + \eta_{a2f})} + \vartheta_4 + \frac{5\vartheta_8^2 \overline{W}^2}{4k_f}, \\ v_{lf} &= \frac{1}{2} \min \left( \frac{q}{2}, \frac{\beta \underline{\Gamma}_f}{4}, \frac{(\eta_{a1f} + \eta_{a2f})}{3}, \frac{\alpha_f}{3}, \frac{k_f}{5} \right). \end{aligned} \quad (\text{A.27})$$

To facilitate the stability analysis, let  $V_{Lf} : \mathbb{R}^{3n+2L} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  be a continuously differentiable positive definite candidate Lyapunov function defined as

$$V_{Lf}(Z_f, t) \triangleq V^*(x) + \frac{1}{2} \tilde{W}_{cf}^T \Gamma_f^{-1} \tilde{W}_{cf} + \frac{1}{2} \tilde{W}_{af}^T \tilde{W}_{af} + \frac{1}{2} \tilde{x}_f^T \tilde{x}_f + \frac{1}{2} r^T r. \quad (\text{A.28})$$

Using the fact that  $V^*$  is positive definite, (A.25) and [8, Lemma 4.3] can be used to establish the bound

$$\underline{v}_{lf}(\|Z_f\|) \leq V_{Lf}(Z_f, t) \leq \overline{v}_{lf}(\|Z_f\|), \quad (\text{A.29})$$

$\forall t \in \mathbb{R}_{\geq 0}$  and  $\forall Z_f \in \mathbb{R}^{3n+2L}$ . In (A.29),  $\underline{v}_{lf}, \overline{v}_{lf} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  are class  $\mathcal{K}$  functions and  $Z \triangleq \left[ x^T, \tilde{W}_{cf}^T, \tilde{W}_{af}^T, \tilde{x}_f^T, r^T \right]^T$ . The sufficient conditions for uniformly ultimately bounded convergence are derived based on the subsequent stability analysis as

$$\begin{aligned} (\eta_{a1f} + \eta_{a2f}) &> \frac{3\eta_{a1f}}{2\zeta_4} - \frac{3\vartheta_8\zeta_5}{2} - \frac{3\eta_{cf}\overline{\|G_\sigma\|}}{4\sqrt{\nu_f\Gamma_f}}\overline{Z}_f \\ k_f &> 5 \max \left( \frac{\vartheta_8}{2\zeta_5} + \alpha_f + 2\eta_{cf}\overline{W}^2\overline{\|\sigma'\|^2}, \frac{3\alpha_f^3}{4} \right) \\ \underline{q} &> 2L_f^2 \left( 2\eta_{cf}\bar{\epsilon}'^2 + \frac{5L_{af}^2}{4k_f} \right) \\ \frac{1}{\alpha_f} &> 6\eta_{cf}\overline{W}^2\overline{\|\sigma'\|^2}, \quad \beta\underline{\Gamma}_f > 2\eta_{a1f}\zeta_4, \end{aligned} \quad (\text{A.30})$$

where  $\overline{Z}_f \triangleq \underline{v}_f^{-1} \left( \overline{v}_f \left( \max \left( \|Z_f(t_0)\|, \sqrt{\frac{\iota_f}{v_{lf}}} \right) \right) \right)$  and  $\zeta_4, \zeta_5 \in \mathbb{R}$  are known positive adjustable constants. An algorithm similar to Algorithm A.2 is employed to select the gains and a compact set  $\mathcal{Z}_f \subset \mathbb{R}^{3n+2L}$  such that

$$\sqrt{\frac{\iota_f}{v_{lf}}} \leq \frac{1}{2} \text{diam}(\mathcal{Z}_f). \quad (\text{A.31})$$

**Theorem A.4** *Provided the gains are selected to satisfy the sufficient conditions in (A.30) based on an algorithm similar to Algorithm A.2, the controller in (4.6), the weight update laws in (A.22), the state derivative estimator in (A.20), and the excitation condition in (A.23) ensure that the state trajectory  $x$ , the state estimation error  $\tilde{x}_f$ , and the parameter estimation errors  $\tilde{W}_{cf}$ , and  $\tilde{W}_{af}$  remain bounded such that*

$$\|Z_f(t)\| \leq \overline{Z}_f, \quad \forall t \in [t_0, t_0 + \bar{t}].$$

*Proof* Using techniques similar to the proof of Theorem 4.3, the time derivative of the candidate Lyapunov function in (A.28) can be bounded as

$$\dot{V}_{Lf} \leq -v_{lf} \|Z_f\|^2, \quad \forall \|Z_f\| \geq \sqrt{\frac{\iota_f}{v_{lf}}}, \quad (\text{A.32})$$

in the domain  $\mathcal{Z}_f$ . Using (A.29), (A.31), and (A.32), [8, Theorem 4.18] is used to show that  $Z_f$  is uniformly ultimately bounded, and that  $\|Z_f(t)\| \leq \bar{Z}_f$ ,  $\forall t \in [t_0, t_0 + \bar{t}]$ .  $\square$

During the interval  $[t_0, t_0 + \bar{t}]$ , the controller in (4.6) is used along with the weight update laws in Assumption A.3. When enough data is collected in the data stack to satisfy the rank condition in (A.17), the update laws from Sect. 4.3.3 are used. The bound  $\bar{Z}_f$  is used to compute gains for Theorem 4.3 using Algorithm A.2.

## A.3 Chapter 6 Supplementary Material

### A.3.1 Auxiliary Constants and Sufficient Gain Conditions for the Station-Keeping Problem

The constants  $\varphi_\zeta$ ,  $\varphi_c$ ,  $\varphi_a$ ,  $\varphi_\theta$ ,  $\kappa_c$ ,  $\kappa_a$ ,  $\kappa_\theta$ , and  $\kappa$  are defined as

$$\varphi_\zeta = \underline{q} - \frac{k_{c1} \sup_{Z \in \beta} \|\nabla_\zeta \epsilon\| (L_{Y_{res}} \|\theta\| + L_{f_{0res}})}{2} - \frac{L_{Y_c} \|g\| (\|W\| \sup_{Z \in \beta} \|\nabla_\zeta \sigma\| + \sup_{Z \in \beta} \|\nabla_\zeta \epsilon\|)}{2},$$

$$\varphi_c = \frac{k_{c2}}{N} \underline{c} - \frac{k_a}{2} - \frac{k_{c1} \sup_{Z \in \beta} \|\nabla_\zeta \epsilon\| (L_{Y_{res}} \|\theta\| + L_{f_{0res}})}{2} - \frac{k_{c1} L_Y \sup_{Z \in \beta} \|\zeta\| \sup_{Z \in \beta} \|\nabla_\zeta \sigma\| \|W\|}{2} - \frac{\frac{k_{c2}}{N} \sum_{j=1}^n (\|Y_{res_j} \sigma'_j\|) \|W\|}{2},$$

$$\varphi_a = \frac{k_a}{2},$$

$$\varphi_\theta = k_\theta \underline{y} - \frac{\frac{k_{c2}}{N} \sum_{k=1}^N (\|Y_{res_k} \sigma'_k\|) \|W\|}{2} - \frac{L_{Y_c} \|g\| (\|W\| \sup_{Z \in \beta} \|\nabla_\zeta \sigma\| + \sup_{Z \in \beta} \|\nabla_\zeta \epsilon\|)}{2} - \frac{k_{c1} L_{Y_{res}} \|W\| \sup_{Z \in \beta} \|\zeta\| \sup_{Z \in \beta} \|\nabla_\zeta \sigma\|}{2},$$

$$\kappa_c = \sup_{Z \in \beta} \left\| \frac{k_{c2}}{4N} \sum_{j=1}^N \tilde{W}_a^T G_{\sigma_j} \tilde{W}_a + \frac{k_{c1}}{4} \tilde{W}_a^T G_{\sigma} \tilde{W}_a \right. \\ \left. + k_{c1} \nabla_{\zeta} \epsilon G \nabla_{\zeta} \sigma^T W + \frac{k_{c1}}{4} \nabla_{\zeta} \epsilon G \nabla_{\zeta} \epsilon + \frac{k_{c2}}{N} \sum_{k=1}^N E_k \right\|,$$

$$\kappa_a = \sup_{Z \in \beta} \left\| \frac{1}{2} W^T G_{\sigma} + \frac{1}{2} \nabla_{\zeta} \epsilon G \nabla_{\zeta} \sigma \right\|,$$

$$\kappa_{\theta} = k_{\theta} d_{\theta},$$

$$\kappa = \sup_{Z \in \beta} \left\| \frac{1}{4} \nabla_{\zeta} \epsilon G \nabla_{\zeta} \epsilon \right\|.$$

The sufficient gain conditions utilized in Theorem 6.4 are

$$\underline{q} > \frac{k_{c1} \sup_{Z \in \beta} \|\nabla_{\zeta} \epsilon\| (L_{Y_{res}} \|\theta\| + L_{f_{0res}})}{2}, \\ + \frac{L_{Y_c} \|g\| (\|W\| \sup_{Z \in \beta} \|\nabla_{\zeta} \sigma\| + \sup_{Z \in \beta} \|\nabla_{\zeta} \epsilon\|)}{2}, \quad (\text{A.33})$$

$$\underline{c} > \frac{N}{k_{c2}} \left( \frac{k_{c1} \sup_{Z \in \beta} \|\nabla_{\zeta} \epsilon\| (L_{Y_{res}} \|\theta\| + L_{f_{0res}})}{2} + \frac{k_a}{2} \right. \\ \left. + \frac{k_{c1} L_Y \sup_{Z \in \beta} \|\zeta\| \sup_{Z \in \beta} \|\nabla_{\zeta} \sigma\| \|W\|}{2} + \frac{\frac{k_{c2}}{N} \sum_{k=1}^N (\|Y_{resk} \sigma'_k\|) \|W\|}{2} \right), \quad (\text{A.34})$$

$$\underline{y} > \frac{1}{k_{\theta}} \left( \frac{\frac{k_{c2}}{N} \sum_{k=1}^N (\|Y_{resk} \sigma'_k\|) \|W\|}{2} \right. \\ \left. + \frac{L_{Y_c} \|g\| (\|W\| \sup_{Z \in \beta} \|\nabla_{\zeta} \sigma\| + \sup_{Z \in \beta} \|\nabla_{\zeta} \epsilon\|)}{2} \right. \\ \left. + \frac{k_{c1} L_{Y_{res}} \|W\| \sup_{Z \in \beta} \|\zeta\| \sup_{Z \in \beta} \|\nabla_{\zeta} \sigma\|}{2} \right), \quad (\text{A.35})$$

### A.3.2 Extension to Constant Earth-Fixed Current

In the case where the earth-fixed current is constant, the effects of the current may be included in the development of the optimal control problem. The body-relative

current velocity  $\nu_c (\zeta)$  is state dependent and may be determined from

$$\dot{\eta}_c = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \nu_c,$$

where  $\dot{\eta}_c \in \mathbb{R}^n$  is the known constant current velocity in the inertial frame. The functions  $Y_{res}\theta$  and  $f_{0_{res}}$  in (6.11) can then be redefined as

$$Y_{res}\theta \triangleq \begin{bmatrix} 0 \\ -M^{-1}C_A(-\nu_c)\nu_c - M^{-1}D(-\nu_c)\nu_c \dots \\ -M^{-1}C_A(\nu_r)\nu_r - M^{-1}D(\nu_r)\nu_r \end{bmatrix},$$

$$f_{0_{res}} \triangleq \begin{bmatrix} J_E\nu \\ -M^{-1}C_{RB}(\nu)\nu - M^{-1}G(\eta) \end{bmatrix},$$

respectively. The control vector  $u$  is

$$u = \tau_b - \tau_c$$

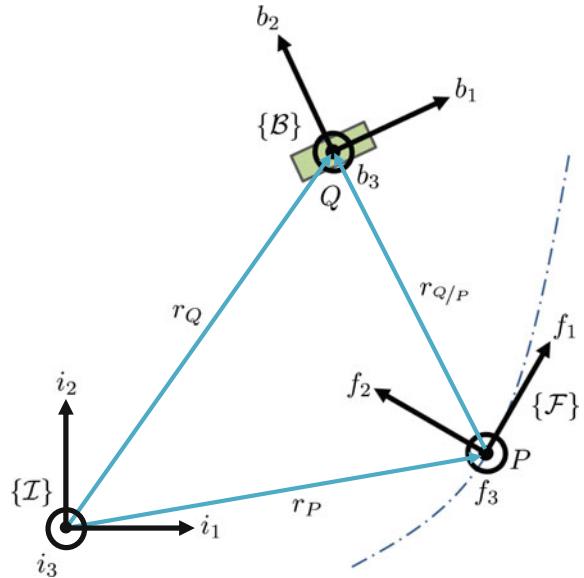
where  $\tau_c (\zeta) \in \mathbb{R}^n$  is the control effort required to keep the vehicle on station given the current and is redefined as

$$\tau_c \triangleq -M_A\dot{\nu}_c - C_A(-\nu_c)\nu_c - D(-\nu_c)\nu_c.$$

### A.3.3 Derivation of Path Following Error Dynamics

The geometry of the path-following problem is depicted in Fig. A.1. Let  $\mathcal{I}$  denote an inertial frame. Consider the coordinate system  $i$  in  $\mathcal{I}$  with its origin and the basis vectors  $i_1 \in \mathbb{R}^3$  and  $i_2 \in \mathbb{R}^3$  in the plane of vehicle motion and  $i_3 \triangleq i_1 \times i_2$ . The point  $P(t) \in \mathbb{R}^3$  on the desired path represents the location of the virtual target at time  $t$ . The location of the virtual target is determined by the path parameter  $s_p(t) \in \mathbb{R}$ . In the controller development, the path parameter is defined as the arc length along the desired path from some arbitrary initial position on the path to the point  $P(t)$ . It is convenient to select the arc length as the path parameter for a mobile robot, since the desired speed can be defined as unit length per unit time. Let  $\mathcal{F}$  denote a frame fixed to the virtual target with the origin of the coordinate system  $f$  fixed in  $\mathcal{F}$  at point  $P(t)$ . The basis vectors  $f_1(t), f_2(t) \in \mathbb{R}^3$  are the unit tangent and normal vectors of the path at  $P(t)$ , respectively, in the plane of vehicle motion and  $f_3(t) \triangleq f_1(t) \times f_2(t)$ . Let  $\mathcal{B}$  denote a frame fixed to the vehicle with the origin of its coordinate system  $b$  at the center of mass  $Q(t) \in \mathbb{R}^3$ . The basis vectors  $b_1(t), b_2(t) \in \mathbb{R}^3$  are the unit tangent and normal vectors of the vehicle motion at  $Q(t)$ , and  $b_3(t) \triangleq b_1(t) \times b_2(t)$ . Note, the bases  $\{i_1, i_2, i_3\}, \{f_1(t), f_2(t), f_3(t)\}$ , and  $\{b_1(t), b_2(t), b_3(t)\}$  form standard bases.

**Fig. A.1** The frame  $\mathcal{F}$  is attached to the virtual target at a distance of  $s_p$  on the desired path. The frame  $\mathcal{B}$  is fixed to the mobile robot, and the frame  $\mathcal{I}$  is inertially fixed (reproduced with permission from [20], ©2014, IEEE)



Consider the following vector equation from Fig. A.1,

$$\underline{r}_{\underline{P}}(t) = \underline{r}_Q(t) - \underline{r}_P(t),$$

where  $\underline{r}_Q(t) \in \mathbb{R}^3$  and  $\underline{r}_P(t) \in \mathbb{R}^3$  are the position vectors of points  $Q$  and  $P$ , from the origin of the inertial coordinate system, respectively, at time  $t$ . The rate of change of  $\underline{r}_{\underline{P}}$  as viewed by an observer in  $\mathcal{I}$  and expressed in the coordinate system  $f$  is

$$\underline{v}_{\underline{P}}^f(t) = \underline{v}_Q^f(t) - \underline{v}_P^f(t). \quad (\text{A.36})$$

The velocity of point  $P$  as viewed by an observer in  $\mathcal{I}$  and expressed in  $f$  is

$$\underline{v}_P^f(t) = [\dot{s}_p(t) \ 0 \ 0]^T, \quad (\text{A.37})$$

where  $\dot{s}_p : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the velocity of the virtual target along the path. The velocity of point  $Q$  as viewed by an observer in  $\mathcal{I}$  and expressed in  $f$  is

$$\underline{v}_Q^f(t) = R_b^f(\theta(t)) \underline{v}_Q^b(t), \quad (\text{A.38})$$

where  $\theta : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the angle between  $f_1$  and  $b_1$  and  $R_b^f : \mathbb{R} \rightarrow \mathbb{R}^{3 \times 3}$  is a transformation from  $b$  to  $f$ , defined as

$$R_b^f(\theta) \triangleq \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The velocity of the vehicle as viewed by an observer in  $\mathcal{I}$  expressed in  $b$  is  $v_Q^b(t) = [v(t) \ 0 \ 0]^T$  where  $v : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the velocity of the vehicle. The velocity between points  $P$  and  $Q$  as viewed by an observer in  $\mathcal{I}$  and expressed in  $f$  is

$$v_{Q/P}^f(t) = {}^{\mathcal{F}} \frac{d}{dt} r_{Q/P}^f(t) + {}^{\mathcal{I}} \rightarrow {}^{\mathcal{F}}(t) \times r_{Q/P}^f(t). \quad (\text{A.39})$$

The angular velocity of  $\mathcal{F}$  as viewed by an observer in  $\mathcal{I}$  expressed in  $f$  is given as  ${}^{\mathcal{I}}\omega^{\mathcal{F}}(t) = [0 \ 0 \ \kappa(t) \dot{s}_p(t)]^T$  where  $\kappa : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the path curvature, and the relative position of the vehicle with respect to the virtual target expressed in  $f$  is  $r_{Q/P}^f(t) = [x(t) \ y(t) \ 0]^T$ . Substituting (A.37)–(A.39) into (A.36) the planar positional error dynamics are given as

$$\begin{aligned} \dot{x}(t) &= v(t) \cos \theta(t) + (\kappa(t) y(t) - 1) \dot{s}_p(t) \\ \dot{y}(t) &= v(t) \sin \theta(t) - \kappa(t) x(t) \dot{s}_p(t). \end{aligned}$$

The angular velocity of  $\mathcal{B}$  as viewed by an observer in  $\mathcal{F}$  is

$${}^{\mathcal{F}}\omega^{\mathcal{B}}(t) = {}^{\mathcal{F}}\omega^{\mathcal{I}}(t) + {}^{\mathcal{I}}\omega^{\mathcal{B}}(t). \quad (\text{A.40})$$

From (A.40), the planar rotational error dynamic expressed in  $f$  is

$$\dot{\theta}(t) = w(t) - \kappa(t) \dot{s}_p(t),$$

where  $w : \mathbb{R}_{\geq t_0} \rightarrow \mathbb{R}$  is the angular velocity of the vehicle.

### A.3.4 Auxiliary Signals

The constants  $\varphi_e, \varphi_c, \varphi_a, \iota_c, \iota_a, \iota, K \in \mathbb{R}$  are defined as

$$\varphi_e \triangleq \underline{q} - \frac{\eta_{c1} \sup_{\zeta \in \chi} \|e'\| L_f}{2}, \quad \varphi_c \triangleq \frac{\eta_{c2}}{N} \underline{c} - \frac{\eta_a}{2} - \frac{\eta_{c1} \sup_{\zeta \in \chi} \|e'\| L_f}{2},$$

$$\begin{aligned}
\iota_c &\triangleq \sup_{\zeta \in \chi} \left\| \frac{\eta_{c2}}{4N} \sum_{j=1}^N \tilde{W}_a^T G_{\sigma j} \tilde{W}_a + \frac{\eta_{c1}}{4} \tilde{W}_a^T G_\sigma \tilde{W}_a + \frac{\eta_{c1}}{2} \epsilon' G \sigma'^T W + \frac{\eta_{c1}}{4} \epsilon' G \epsilon'^T \right. \\
&\quad \left. + \frac{\eta_{c2}}{N} \sum_{j=1}^N E_j + \eta_{c1} \epsilon' L_f \right\|, \\
\iota_a &\triangleq \sup_{\zeta \in \chi} \left\| \frac{1}{2} G_\sigma W + \frac{1}{2} \sigma' G \epsilon'^T \right\|, \quad \varphi_a \triangleq \frac{\eta_a}{2}, \quad \iota \triangleq \sup_{\zeta \in \chi} \left\| \frac{1}{4} \epsilon' G \epsilon'^T \right\|, \\
K &\triangleq \sqrt{\frac{\iota_c^2}{2\alpha\varphi_c} + \frac{\iota_a^2}{2\alpha\varphi_a} + \frac{\iota}{\alpha}}, \quad \alpha \triangleq \frac{1}{2} \min \left\{ \varphi_e, \frac{\varphi_c}{2}, \frac{\varphi_a}{2} \right\}.
\end{aligned}$$

When Assumption 6.3 and the sufficient gain conditions

$$\eta_{c1} < \frac{2q}{\sup_{\zeta \in \chi} \|\epsilon'\| L_f}, \quad (\text{A.41})$$

$$\eta_{c2} > \frac{N\eta_a}{2\underline{c}} + \frac{N\eta_{c1} \sup_{\zeta \in \chi} \|\epsilon'\| L_f}{2\underline{c}} \quad (\text{A.42})$$

are satisfied, the constants  $\varphi_e$ ,  $\varphi_c$ ,  $\varphi_a$ ,  $\iota_c$ ,  $\iota_a$ ,  $\iota$ ,  $K \in \mathbb{R}$  are positive.

## References

1. Filippov AF (1988) Differential equations with discontinuous right-hand sides. Kluwer Academic Publishers, Dordrecht
2. Aubin JP, Frankowska H (2008) Set-valued analysis. Birkhäuser
3. Shevitz D, Paden B (1994) Lyapunov stability theory of nonsmooth systems. IEEE Trans Autom Control 39(9):1910–1914
4. Paden BE, Sastry SS (1987) A calculus for computing Filippov’s differential inclusion with application to the variable structure control of robot manipulators. IEEE Trans Circuits Syst 34(1):73–82
5. Clarke FH (1990) Optimization and nonsmooth analysis. SIAM
6. Kamalapurkar R, Rosenfeld JA, Klotz J, Downey RJ, Dixon WE (2014) Supporting lemmas for RISE-based control methods. [arXiv:1306.3432](https://arxiv.org/abs/1306.3432)
7. Fischer N, Kamalapurkar R, Dixon WE (2013) LaSalle-Yoshizawa corollaries for nonsmooth systems. IEEE Trans Autom Control 58(9):2333–2338
8. Khalil HK (2002) Nonlinear systems, 3rd edn. Prentice Hall, Upper Saddle River
9. Sastry S, Bodson M (1989) Adaptive control: stability, convergence, and robustness. Prentice-Hall, Upper Saddle River
10. Narendra K, Annaswamy A (1989) Stable adaptive systems. Prentice-Hall Inc, Upper Saddle River
11. Ioannou P, Sun J (1996) Robust adaptive control. Prentice Hall, Upper Saddle River

12. Chowdhary G (2010) Concurrent learning for convergence in adaptive control without persistence of excitation. PhD thesis, Georgia Institute of Technology
13. Chowdhary GV, Johnson EN (2011) Theory and flight-test validation of a concurrent-learning adaptive controller. *J Guid Control Dynam* 34(2):592–607
14. Chowdhary G, Yucelen T, Mühlegg M, Johnson EN (2013) Concurrent learning adaptive control of linear systems with exponentially convergent bounds. *Int J Adapt Control Signal Process* 27(4):280–301
15. Dierks T, Jagannathan S (2009) Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics. In: Proceedings of the IEEE conference on decision and control. Shanghai, CN, pp 6750–6755
16. Vamvoudakis KG, Lewis FL (2010) Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica* 46(5):878–888
17. Vrabie D, Lewis FL (2010) Integral reinforcement learning for online computation of feedback nash strategies of nonzero-sum differential games. In: Proceedings of the IEEE conference on decision and control, pp 3066–3071
18. Bhasin S, Kamalapurkar R, Johnson M, Vamvoudakis KG, Lewis FL, Dixon WE (2013) A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica* 49(1):89–92
19. Zhang H, Cui L, Luo Y (2013) Near-optimal control for nonzero-sum differential games of continuous-time nonlinear systems using single-network ADP. *IEEE Trans Cybern* 43(1):206–216
20. Walters P, Kamalapurkar R, Andrews L, Dixon WE (2014) Online approximate optimal path-following for a mobile robot. In: Proceedings of the IEEE conference on decision and control, pp 4536–4541

# Index

## A

Actor, xv, 25, 26, 28, 30, 31, 34, 35, 43–45, 50, 51, 55, 58, 73, 75, 76, 80, 81, 90–92, 94, 162, 165, 185, 219, 258  
Actor-critic, 25, 26, 28, 31, 33, 35, 52, 56, 85, 90, 94  
Actor-critic-identifier, 33, 43–45, 73, 75, 76, 102, 103, 131  
Adjacency matrix, 151, 180  
Advantage updating, 26, 35  
Algebraic Riccati equation, 125, 209, 222, 224  
Autonomous surface vehicles, 223  
Autonomous underwater vehicle, 195, 196, 208–210

## B

Bellman error, 24, 25, 28–33, 43, 45, 50, 51, 56, 62, 63, 73, 76, 81, 99–105, 107, 110, 118–120, 122, 125, 130, 131, 133, 135, 138, 144, 150, 157–159, 161–163, 185–187, 203–205, 209, 210, 215, 245, 247, 251, 252  
Bellman error extrapolation, 166, 169, 174, 230, 248, 250, 252–254, 257, 258  
Bellman’s principle of optimality, 3  
Bergmann–Fock space, 232  
Bolza problem, 1–3, 5, 7, 18, 26, 243  
Brachistochrone problem, 3

## C

Carathéodory solutions, 2  
Common Lyapunov function, 110, 140

Concurrent learning, 100, 101, 103, 104, 107, 111, 114, 118, 120, 127, 130–134, 140, 141, 150, 158, 159, 182, 183, 186–189, 195, 199, 243  
Converse Lyapunov Theorem, 84  
Cooperative control, 150, 167, 172, 177, 189, 190  
Cost, Lagrange, 1, 2  
Cost, Mayer, 1, 2  
Cost-to-go, 3, 18, 25  
Critic, 25, 26, 28–35, 43–45, 49–51, 55–58, 62, 65, 73, 75, 76, 80–83, 89–92, 94, 101, 102, 141, 161, 162, 165, 219, 245, 259

## D

Differential game, 11, 12, 44, 74, 75, 94, 189, 190  
Differential game, closed-loop, 44  
Differential game, graphical, 150, 190  
Differential game, nonzero-sum, 44, 73, 75, 89, 90, 92–94, 101, 131, 140  
Differential game, zero-sum, 94  
Dynamic neural network, 43, 46, 56, 73, 75, 77, 78, 90

## E

$\epsilon$ —modification, 33  
Existence, 2  
Experience replay, 30, 99, 102, 260, 261  
Exponential kernel function, 229, 232, 234–237, 240

**F**

- Filippov solution, 46, 48, 49, 77, 79  
 Finite excitation, 102  
 Finite time horizon, 224  
 Finite-horizon, 71, 94, 185  
 Formation tracking, 149, 153, 154, 189

**G**

- Galerkin's method, 223  
 Galerkin's spectral method, 35, 93  
 GPOPS, 70, 72, 73  
 Gram-Schmidt algorithm, 235  
 Graph, directed, 151, 180  
 Graph Laplacian, 151

**H**

- Hamiltonian, 5, 6, 9, 10, 61, 62, 72, 76, 185, 189  
 Hamilton–Jacobi–Bellman equation, 5–8, 10–13, 18–23, 27, 35, 36, 43, 44, 51, 53, 61, 93, 99, 104, 119, 181, 202, 215, 223  
 Hamilton–Jacobi equation, 75, 182, 184  
 Hamilton–Jacobi–Isaacs equation, 94  
 Heuristic dynamic programming, 25, 34–36, 43, 94  
 Hopfield, 43

**I**

- Identifier, 43, 45, 46, 49, 50, 52, 55–57, 73, 76, 80, 85, 90  
 Infinite-horizon, 5, 35, 44, 45, 71, 73, 74, 94, 95, 100, 101, 117, 118, 131, 149, 153

**K**

- Kalman filter, 209  
 Kantorovich inequality, 239

**L**

- Least-squares, 28, 31, 33, 35, 43, 49–51, 62, 63, 73, 81, 91, 93, 100, 101, 105, 106, 111, 114, 134, 140, 162, 186, 187, 245, 246, 248

Levenberg–Marquardt, 51

Linear quadratic regulator, 227

**M**

- Model-predictive control, 11, 36, 223, 224

**N**

- Nash equilibrium, 11, 44, 45, 74, 89, 91, 101, 131, 138–141, 149, 150, 156–158, 161, 166, 181, 190

Nash policy, 165

Network, leader-follower, 188

Network systems, 149–152, 167, 172, 176, 180, 181, 184, 185, 189, 190

Nonholonomic system, 172

Nonholonomic vehicle, 223

**P**

- Path-following, 196, 213, 223, 224  
 Persistence of excitation, 24, 31, 33, 43, 52, 55, 56, 63, 66–68, 83, 85, 89–91, 93, 99, 101–103, 107, 111, 118, 120, 131–133, 143, 145, 183, 187, 190, 195, 199, 204, 230, 248

Policy evaluation, 18, 35, 55

Policy gradient, 35

Policy improvement, 18, 35, 55, 93

Policy iteration, 17–19, 22–25, 34–36, 55, 94, 95

Policy iteration, synchronous, 94

Pontryagin's maximum principle, 2, 3, 9, 10, 22, 34

Prediction error, 107

Projection operator, 50, 78, 82, 84, 114, 204, 205

Pseudospectral, Gauss, 117

Pseudospectral, Radau, 70

**Q**

- Q-learning, 17, 22, 26, 34–36, 94

**R**

- Radial basis function, 228–230  
 Randomized stationary policy, 105  
 Receding horizon, 36  
 Reinforcement learning, 12, 13, 17, 29, 30, 33, 35–37, 43, 45, 55, 60, 91, 94, 99–103, 105, 118, 144, 149, 150, 158, 161, 195, 229, 230, 242, 245, 258–261

Reproducing kernel Hilbert space, 227, 228, 230–234, 238

Reproducing kernel Hilbert space, universal, 233, 242, 244

Riccati equation, 11

RISE, 43, 46, 56, 77, 209

R–learning, 34–36

**S**

- Saddle point, 94
- SARSA, 34–36
- $\sigma$ —modification, 33
- Sigmoid, 230
- Simulation of experience, 100–103, 105, 118, 122, 131, 161–163, 245
- Single network adaptive critic, 36
- Singular value maximization algorithm, 111, 128, 133, 140, 141
- Spanning tree, 151, 152, 164
- State following (StaF), 228–230, 232, 233, 235, 237, 238, 242–244, 246, 250, 251, 253–258, 260
- Station keeping, 195, 211, 223
- Stone-Weierstrass Theorem, 23
- Successive approximation, 18, 35, 36
- Support vector machines, 230
- Switched subsystem, 104, 110, 120, 133, 139, 140

**T**

- Temporal difference, 13, 17, 25, 35, 93

**U**

- Uniqueness, 2, 12
- Universal Approximation, 23, 63, 101, 104, 158, 160, 184, 185, 252

**V**

- Value iteration, 17, 22, 23, 34–36
- Viscosity solutions, 12

**W**

- Weierstrass Theorem, 243
- Wheeled mobile robot, 172, 196, 218–223