Learning Techniques in Receding Horizon Control and Cooperative Control

ZHANG, Hongwei

A Dissertation Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Automation and Computer-Aided Engineering

The Chinese University of Hong Kong
June 2010

Thesis/Assessment Committee

Professor Liao, Wei-Hsin (Chair)

Professor Huang, Jie (Thesis Supervisor)

Professor Liu, Yunhui (Committee Member)

Professor Lewis, Frank L (External Examiner)

Abstract

Two topics of modern control are investigated in this dissertation, namely receding horizon control (RHC) and cooperative control of networked systems. We apply learning techniques to these two topics. Specifically, we incorporate the reinforcement learning concept into the standard receding horizon control, yielding a new RHC algorithm, and relax the stability constraints required for standard RHC. For the second topic, we apply neural adaptive control in synchronization of the networked nonlinear systems and propose distributed robust adaptive controllers such that all nodes synchronize to a leader node.

Receding horizon control (RHC), also called model predictive control (MPC), is a suboptimal control scheme over an infinite horizon that is determined by solving a finite horizon open-loop optimal control problem repeatedly. It has widespread applications in industry. Reinforcement learning (RL) is a computational intelligence method in which an optimal control policy is learned over time by evaluating the performance of suboptimal control policies. In this dissertation it is shown that reinforcement learning techniques can significantly improve the behavior of RHC. Specifically, RL methods are used to add a learning feature to RHC. It is shown that keeping track of the value learned at the previous iteration and using it as the new terminal cost for RHC can overcome traditional strong requirements for RHC stability, such as that the terminal cost be a control Lyapunov function, or that the horizon length be greater than some bound. We propose improved RHC algorithms, called updated terminal cost receding horizon control (UTC-RHC), first in the framework of discrete-time linear systems and then in the framework of continuous-time linear systems. For both cases, we show the uniform exponential stability of the closed-loop system can be guaranteed under very mild conditions. Moreover, unlike RHC, the UTC-RHC control gain approaches the optimal policy associated with the infinite horizon optimal control problem. To show these properties, non-standard Lyapunov functions are introduced for both discrete-time case and continuous-time case.

Cooperative control of networked systems (or multi-agent systems) has attracted much attention during the past few years. But most of the existing results focus on first order and second order leaderless consensus problems with linear dynamics. The second part of this dissertation solves a higher-order synchronization problem for cooperative nonlinear systems with an active leader. The communication network considered is a weighted directed graph with fixed topology. Each agent is modeled by a higher-order nonlinear system with the nonlinear dynamics unknown. External unknown disturbances perturb each agent. The leader agent is modeled as a higher-order non-autonomous nonlinear system. It acts as a command generator and can only give commands to a small portion of the networked group. A robust adaptive neural network controller is designed for each agent. Neural network learning algorithms are given such that all nodes ultimately synchronize to the leader node with a small residual error. Moreover, these controllers are totally distributed in the sense that each controller only requires its own information and its neighbors' information.

摘 要

本文研究了现代控制理论中的两个专题,即滚动时域控制和多智能体系统的协同控制。我们将学习算法引入到以上两种控制中。具体来讲，我们将强化学习的概念引入到滚动时域控制，提出了一种新的控制算法，该算法减少了传统滚动时域控制的稳定性限制条件。对于协同跟踪控制，我们引入了神经网络自适应控制，提出了一种分布式鲁棒自适应控制算法，使得所有智能体成功地跟踪了目标智能体。

滚动时域控制又称模型预测控制，是一种无穷时域上的次优化控制算法。它的本质是在无穷时域上反复求解一个有限时域优化问题。它在工业上有着广泛的应用。强化学习是一种计算智能方法，其最优控制律通过评价次优控制律而得。该论文阐述了如何通过强化学习来改善传统的滚动时域控制。也就是说，利用强化学习的技术，将学习的概念引入到了传统的滚动时域控制中。具体来讲，我们利用上次求解有限时域优化问题所得的成本函数值来更新本次优化问题的终端成本函数。事实证明该算法减少了传统滚动时域控制所需要的稳定性终端约束。我们将提出的新算法命名为更新终端成本函数的滚动时域控制。我们分别将该算法运用到了离散时间线性控制系统和连续时间线性控制系统中。在两种情况下，闭环系统均为一致指数稳定。并且，其控制律最终趋于最优值。在证明中，我们运用了非常规的李亚普诺夫函数。

多智能体系统的协同控制近些年得到了广泛的关注。但多数现有结果仅考虑了一阶或二阶线性系统的协调问题。该论文的第二部分研究了高阶非线性多智能体系统的跟踪问题。所研究的多智能体网络可表示为具有固定拓扑结构的加权有向图。每个智能体均为高阶非线性系统，并且其非线性函数未知，此外还有外部扰动。目标智能体是高阶非自制非线性系统。目标智能体仅能向其他部分智能体传达命令。我们针对每个智能体，设计了其独有的神经网络鲁棒自适应控制器。其中神经网络的引入抵消了智能体未知的非线性作用，从而使得每个智能体最终均可很好的跟踪目标智能体。该控制器属于分布式控制器，即所有智能体的控制器均只用到该智能体本身以及它的邻居的信息。

Acknowledgement

I would like to express my sincere gratitude to people who made this dissertation possible.

First of all, I am indebted to my supervisor, Prof. Jie Huang, for his guidance and supervision. I am fortunate to be a PhD student of Prof. Huang. During these years, Prof. Huang not only teaches us how to do research but also influences us with his serious attitude towards doing research. I can never forget what he told us many times in the weekly meetings. He said that doing research requires an attitude of careful scrutiny on minute details in order to maintain the research quality and integrity (in Chinese: 做研究须战战兢兢，如临深渊，如履薄冰). Even after a paper is published, cautious examinations on unrealized hidden errors are still needed. These words are always my guidelines when I am doing research. I am also thankful to Prof. Huang for introducing me to Prof. Frank Lewis, my co-supervisor. He said Prof. Lewis is the best teacher in the world he could find for me. That is definitely true.

I always feel fortunate to work with Prof. Lewis, ever since May 2007 when we first met during his visit to CUHK. His enthusiasm, his insight, his wide area of research interests, and his strong sense of responsibility to his students impress me deeply. Prof. Lewis not only helped me to pin down the topics of my research, but also painstakingly led me step by step into these wonderful fields. For the first two years, Prof. Lewis had supervised me through emails and met me once or twice a year during his visits to CUHK. Many results in this dissertation were made born from our email communication. I am grateful to Prof. Lewis for supporting me to work with him at ARRI of UTA, for the last year of my PhD study. Working at ARRI is one of my best experiences. I benefit a lot from the weekly forums and from a bunch of great guys in Prof. Lewis' ACS group. I am also grateful to Prof. Lewis who will fly all the way to CUHK to attend my defense.

I am thankful to my defense committee: Prof. Wei-Hsin Liao and Prof. Yunhui Liu, for their time and invaluable suggestions to help improve this dissertation.

My thanks also go to my friends both in Hong Kong and in Texas, who have colored my life during the past four years. I would like to thank my labmates in CUHK: Lu Liu, Tianshi Chen, Renxin Zhong, Dabo Xu, Jin Zhao, Xi Yang, Zhaowu Ping, Youfeng Su, Yi Dong and Qiping Han. I would never forget the time we spent together. Special thanks go to Dabo, Tianshi and Youfeng, who helped me a lot for my research. Special thanks also go to Zhaowu for helping me handle all the paper works in CUHK during my stay at Texas. I also feel lucky to make a lot of good friends in Hong Kong, they are Changqing Yan, Chengguo Zhang, Junqiang Liu, Yiping Yang, Yinghui Zhou, Xiaoliu Wang, Bo Yang, Jane, Jenny,

Simon, Kenny, Alice, Annie, etc. They enriched my life in Hong Kong. I also have a wonderful time in Texas. I would like to thank the ACS group, in particularly, Draguna Vrabie, Abhijit Das and Mohammed Abouheaf. Draguna spent a lot of time helping me settle down when I first arrived at Texas and she is always willing to give me rides. I also enjoy a lot of discussions with Draguna and Abhijit. In fact, some results of Chapter 5 are benefited from discussions with Abhijit. Thanks to Mohammed for sharing his course videos with me. Isaac Weintraub offered me many rides during the cold winter and hot summer. I am also thankful to the friendship of my officemates Mooyoung Lee, Hsiang-Hsi Huang, Changbin Li and Gang Chen at ARRI. Every raining day when we need a ride, Mooyoung is always a phone call away. I also had a wonderful time with Uncle Ken, Aunt Cathy and Aunt Gretchen. They are like my family in Texas.

My sincere gratitude also goes to Joyce, Maggie and Winnie in the Department of MAE, and Kathleen Elfrink and Adrianna Watson at ARRI for helping me with all the administrative business during theses years. They saved me a lot of time from those kind of things.

Most importantly, I am indebted to my family. Without their unconditional love and constant support, I can never go to college, go to graduate school, and finally finish my PhD study. This dissertation is dedicated to them. Their love is worth far more than any degree. This dissertation is also dedicated to my fiancee, Lin. Without her constant encouragement and understanding, I might never finish this dissertation. Meeting my fiancee is my greatest achievement during my life.

Finally, I would like to acknowledge the financial support from CUHK and UTA. This dissertation will never be successfully made without the funding provided by the Research Grants Council of Hong Kong (Project no. 412408), the National Science Foundation under grant ECCS-0801330 and grant IIS-0326505, the Army Research Office under grant W91NF-05-1-0314, and the Air Force Office of Scientific Research under grant FA9550-09-1-0278.

Hongwei Zhang
June 2010

Contents

Notations and Symbols

$\square$	end of proof 22
$\blacksquare$	end of Definition, Lemma, Remark, Assumption, Algorithm $\cdots$ 6
$\forall$	for any 10
$\exists$	there exists 10
$\Rightarrow$	implies that 10
$\mathbb{R}$	set of real numbers 5
$\mathbb{R}^n$	real column vector spaces 16
$\mathbb{R}^{n\times m}$	set of real matrices with dimensions n-by-m 16
$\in$	is an element of 5
$\subseteq$	is a subset of 10
$\subset$	is a proper subset of 70
I	identity matrix with appropriate dimensions 8
A^T	transpose of matrix A 16
x^T	transpose of vector x 8
$>$	$A > 0$ means matrix A is positive definite 12
$\geq$	$A \geq 0$ means matrix A is positive semidefinite 8
$\lvert\cdot\rvert$	absolute value of a real number $a \in \mathbb{R}$; determinant of a matrix $A \in \mathbb{R}^{n\times n}$ 5
$\lVert x\rVert_p$	Hölder norm of vector $x \in \mathbb{R}^n$ 6
$\lVert x\rVert_1$	sum norm of vector $x \in \mathbb{R}^n$ 6
$\lVert x\rVert_2$	Euclidean norm of vector $x \in \mathbb{R}^n$ 7
$\lVert x\rVert_\infty$	max norm of vector $x \in \mathbb{R}^n$ 7
$\lVert A\rVert_F$	Frobenius norm of matrix A 7
$\lVert\cdot\rVert$	Euclidean norm of vectors; Frobenius norm of matrices 7
$[a_{ij}]$	matrix with a_{ij} being the element in the i-th row and j-th column 16
$\lambda_i(A)$	i-th largest eigenvalue of matrix $A \in \mathbb{R}^{n\times n}$ which has real eigenvalues 7
$\sigma_i(A)$	i-th largest singular value of matrix $A \in \mathbb{R}^{n\times m}$ 8
$\bar{\sigma}(A)$	the largest singular value of matrix $A \in \mathbb{R}^{n\times m}$ 8
$\underline{\sigma}(A)$	the smallest singular value of matrix $A \in \mathbb{R}^{n\times m}$ 8
$rank(A)$	rank of matrix $A \in \mathbb{R}^{n\times m}$ 8

List of Figures

Chapter 1

Introduction

1.1 Motivations and contributions

Two topics of modern control are investigated in this dissertation, namely receding horizon control (RHC) and synchronization of networked systems. We apply learning techniques to these two topics. Specifically, we incorporate the reinforcement learning (RL) concept into the standard receding horizon control, yielding a new RHC algorithm, called updated terminal cost receding horizon control (UTC-RHC). With UTC-RHC, the stability constraints for standard RHC algorithms are relaxed. For the second topic, we apply neural adaptive control to synchronization of the networked nonlinear systems and propose distributed robust adaptive controllers such that all nodes synchronize to a leader node.

1.1.1 Receding horizon control

Receding horizon control (RHC), also known as model predictive control (MPC), is a suboptimal control scheme that solves a finite horizon open-loop optimal control problem repeatedly in an infinite horizon context. This is an on-line method that effectively provides a measured state feedback control law. RHC is popular in industry, especially in the process control industries. Many stories have been told about its successful applications in the past three decades, see [61] for an excellent survey of industrial MPC technology.

Extensive attention has been paid to the stability issues of RHC (see [50] for a survey), leading to various stability conditions involving constraints on either the terminal state, or the terminal cost, or the horizon size, or their different combinations. Some approaches require the terminal state to be zero ([34], [35], [49]); some approaches impose constraints on the terminal cost ([7], [11], [15], [32], [38]); it is also shown that by taking sufficiently large horizon size, the stability of RHC algorithm is guaranteed without constraints on the terminal cost ([8], [17], [23]). To our best knowledge, the standard RHC contains no ingredient of learning. Most existing efforts have been devoted to finding a suitable terminal cost function such that the resulting RHC algorithm is stabilizing.

Reinforcement Learning (RL) is a computational approach which learns through interaction with the

environment [73]. Generally speaking, at a given state, the agent takes an action, observes a reward, updates the value function and then use the new value function to search for a better action. RL is well known in the computational intelligence community and has a wide applications from computer game playing to robotics. The applications of RL to feedback control were highlighted in a recent special issue of IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics (SMCB) [42]. Recently, many RL algorithms have also been developed to solve the optimal control problems. Such algorithms are known generally as approximate/adaptive dynamic programming (ADP) ([83], [84], [85], etc), also known as neuro-dynamic programming (NDP) [6]. This includes a school of methods, to name a few: policy iteration (PI), value iteration (VI) or heuristic dynamic programming (HDP), generalized policy iteration (GPI) and Q-learning [82] or action dependent learning [84], etc. ADP has gained a lot of attention for the past few decades; interested readers are referred to the SMCB special issue on ADP for feedback control ([2], [39], [42], [85], etc.), the survey paper [79], and the references therein.

Inspired by the key idea of RL, i.e. the agent keeps interacting with the environment to get enough information to update the value function, and then constructs a better policy using the updated value function, we incorporate this learning feature into the standard RHC and propose a modified RHC algorithm, which we call updated terminal cost RHC (UTC-RHC). In this algorithm, instead of using the fixed terminal cost function, we update it at each stage by the optimal cost function obtained from the previous stage. It should be pointed out that the updated cost function is "learned" from the system model, instead of the environment (i.e. the system). By integrating this "learning" ingredient into the standard RHC scheme, it can be shown that the performance of RHC is significantly improved. Specifically, the UTC-RHC algorithm ensures the closed-loop stability while imposing constraints on neither terminal state, nor terminal cost function, nor the horizon size. Also, the resulting feedback control law approaches the optimal value corresponding to the infinite horizon optimal control problem.

In this dissertation, we apply the UTC-RHC algorithm to two classes of systems, i.e. discrete time (DT) linear time invariant system, continuous time (CT) linear time invariant system. Two specific algorithms are proposed respectively.

1.1.2 Synchronization of networked systems

In the past few years, researchers in control community are getting more and more interested in systems consist of a group of agents, called multi-agent systems or networked systems. The group of agents are not isolated, but form a communication network. Some agents can pass or receive information to or from other agents. This research is motivated by widespread applications which include the spacecraft [37], unmanned air vehicles (UAVs) ([3] ,[64]), mobile robots [89], sensor networks [12], etc. Some seminal works are ([13], [24], [55], [65], [75], [76]), to name a few.

Considerable effort has focused on two subjects of the networked systems, i.e. cooperative regulator problem and tracking problem. For cooperative regulator problem, controllers are designed to drive all the agents / nodes to a common value, i.e. consensus equilibrium, which is not prescribed and depends on initial conditions. To be precise, the consensus equilibrium depends on the initial states values of

those agents who have a directed path to all the other agents [67]. This is also known as the leaderless consensus or synchronization problem in literature [24]. As for the tracking problem, there is a leader node and it acts as a command generator, ignoring all information of the other nodes. The leader node only gives commands to a small portion of the other nodes, which means only that small portion of agents can get information from the leader node. All the nodes are trying to track the trajectory of the leader node. This is called consensus with a leader or synchronization to a leader in literature ([18], [52], [68]). Numerous results on these two topics have been published in the past few years and readers are referred to survey papers ([54], [66], [67]) and the references therein.

Our research focuses on the tracking problem of higher-order nonlinear dynamics and is motivated by several points. First, most existing work on networked systems studies the first order or second order dynamics. So far, only a few results exist for the general higher-order (*greater than two*) systems ([26], [68], [80], [86], etc.), but most of them only consider linear dynamics. For example, [26] and [80] study the leaderless consensus problem of higher-order linear dynamics. Although [80] also considers the nonlinear case, they deal with that by transforming it into a linear system using coordinate transformation. [68] presents the idea of higher-order consensus with a leader, called model reference consensus algorithm, also for linear dynamics.

Second, even for the first order or second order synchronization problems, dynamics are often chosen to be single integrators or double integrators (see [66] for a survey), while synchronization of multi-agent system with complicated nonlinear dynamics has not been fully investigated yet. A recent work [90] studies the second order leaderless consensus problem of nonlinear dynamics, which is a double integrator incorporated with a nonlinear term. They introduce a new concept about the generalized algebraic connectivity and provide sufficient conditions for consensus to be reached. But they only analyze the frequently used second-order consensus protocol, instead of designing a protocol. Pinning control has been introduced for controlled synchronization of interconnected dynamical systems with identical nonlinear dynamics ([45], [46], [47], [81]). A control or leader node is connected (pinned) into a small percentage of nodes in the network. Analysis shows that with suitable design, all nodes converge to the state of the control node, which may be time-varying. Analysis has been done using Lyapunov techniques by assuming either a Jacobian linearization of the nonlinear node dynamics or a Lipschitz condition. But in practice, the node dynamics may not be identical, or even may be unknown. Many factors result in imprecision of the models of the networked agents, e.g. modeling the friction as a linear model for design purpose; system parameters drifting with time; and imperfect plant data.

Third, external disturbances are often neglected for the current research. However, disturbances exist almost in every practical application, such as white noise, gust to the aircraft.

Motivated by the above facts, we consider the higher-order synchronization problem of networked nonlinear systems with a time-varying active leader agent. The communication network considered is a weighted directed graph with fixed topology. Each agent is modeled by a higher-order nonlinear system with the nonlinear dynamics unknown. External unknown disturbances perturb each agent. The leader agent is modeled as a higher-order non-autonomous nonlinear system. It acts as a command generator

and only give commands to a small portion of the networked group. To the best of our knowledge, this problem has not been investigated in the literature. A robust adaptive neural network controller is designed for each agent. Neural network learning algorithms are given such that all nodes ultimately synchronize to the leader node with a small residual error. Moreover, these controllers are totally distributed in the sense that each controller only requires information of itself and its neighbors.

1.2 Organization of the dissertation

The rest of the dissertation is organized as follows.

Chapter 2:

To make the dissertation self contained, in this chapter, some background of matrix theory and some results of stability theory are briefly introduced.

Chapter 3:

Integrating the learning feature in the standard receding horizon control, this chapter proposes an improved algorithm, called updated terminal cost receding horizon control (UTC-RHC) for unconstrained discrete linear time invariant systems. Stability constraints are relaxed. Rigorous proof and the simulation examples show the advantages of UTC-RHC against the standard RHC. The results of this chapter have been published in [91] and [92].

Chapter 4:

UTC-RHC algorithm is extended to the unconstrained continuous linear time invariant systems, yielding a sampled-date UTC-RHC. Proofs and examples are provided. This chapter is based on the work [93].

Chapter 5:

This chapter considers a synchronization problem for a higher-order networked nonlinear system with an active leader. The system dynamics are nonlinear and unknown. External disturbances are considered. The leader agent has a non-autonomous nonlinear dynamics which is unknown to all the follower nodes. Using neural adaptive control technique, we propose robust adaptive control laws, which are totally distributed and can be implemented locally. This chapter is based on the work [94] and [95].

Chapter 6:

A conclusion is drawn and future works are discussed in this chapter.

□ **End of chapter.**

Chapter 2

Mathematical background and preliminaries

To be self contained of this dissertation, in this chapter, some basic matrix properties that are frequently used in the control theory, are presented. Also we briefly overview some fundamental stability concepts and results for both linear systems and nonlinear systems, which will be used in this dissertation. The contents in this chapter are standard and fundamental, and can be found in many textbook, such as [4], [10], [20], [28], [70], [71].

2.1 Some basic results of matrix theories

We begin with the norms of vectors and matrices. Like the absolute value for a real number, norms are used to measure the magnitude of a vector or matrix, they maps a vector or matrix to a positive real number. Norms are used in control theories to describe the concepts of stabilities.

Definition 2.1 [20] (vector norm)
Let $\mathbb{R}^n$ be a vector space over the field $\mathbb{R}$. The mapping $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ is a vector norm, if $\forall x, y \in R^n$, the following properties hold.

V1) $\|x\| \geq 0$. Nonnegative

V2) $\|x\| = 0$ if and only if $x = 0$. Positive

V3) $\|ax\| = |a| \, \|x\|, \forall a \in \mathbb{R}$. Homogeneous

V4) $\|x + y\| \leq \|x\| + \|y\|$. Triangle inequality

∎

A mapping that only satisfies properties (V1), (V3) and (V4), but not necessarily (V2), is called a *seminorm*. A seminorm allows some nonzero vectors to have zero magnitude.

A very useful class of vector norms are the Hölder norms.

Definition 2.2 [4] (Hölder norms)
$\forall x = [x_1, x_2, \cdots, x_n]^T \in \mathbb{R}^n$, the Hölder norm is defined by

$$\|x\|_p = \begin{cases} (\sum_{i=1}^{n}(x_i)^p)^{1/p}, & 1 \leq p < \infty; \\ \max_{i=\{1,2,\cdots,n\}} |x_i|, & p = \infty. \end{cases}$$

∎

The most frequently used vector norms are Hölder norm when $p = 1, 2, \infty$. When $p = 1$, Hölder norm is known as the *sum norm* (or l_1 norm), i.e.

$$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|. \tag{2.1}$$

When $p = 2$, Hölder norm is known as the *Euclidean norm* (or l_2 norm), i.e.

$$\|x\|_2 = \sqrt{|x_1|^2 + |x_2|^2 + \cdots + |x_n|^2} = \sqrt{x^T x}. \tag{2.2}$$

When $p = \infty$, Hölder norm is known as the *max norm* (or l_∞ norm), i.e.

$$\|x\|_\infty = max\{|x_1|, |x_2|, \cdots, |x_n|\}. \tag{2.3}$$

Definition 2.3 [4] (matrix norm)
A function $\|\cdot\| : \mathbb{R}^{m\times n} \to \mathbb{R}$ is called a matrix norm if it satisfies the following conditions:

M1) $\|A\| \geq 0$ for all $A \in \mathbb{R}^{m\times n}$. Nonnegative

M2) $\|A\| = 0$ if and only if $A = 0$. Positive

M3) $\|aA\| = |a|\,\|A\|$ for all $a \in \mathbb{R}$ and $A \in \mathbb{R}^{m\times n}$. Homogeneous

M4) $\|A + B\| \leq \|A\| + \|B\|$, for all $A, B \in \mathbb{R}^{m\times n}$. Triangle inequality

∎

Different matrix norms may be defined as long as they satisfy the matrix definition 2.3. Similar to vector norms, matrix norms can be defined entrywisely, such as the matrix Hölder norms.

Definition 2.4 [4] (Hölder norms)
$\forall A = [a_{ij}] \in \mathbb{R}^{n\times m}$, the Hölder norm is defined by

$$\|A\|_p = \begin{cases} \left(\sum_{i=1}^{n}\sum_{j=1}^{m}(a_{ij})^p\right)^{1/p}, & 1 \leq p < \infty; \\ \max_{i=\{1,2,\cdots,n\},j=\{1,2,\cdots,m\}} |a_{ij}|, & p = \infty. \end{cases}$$

∎

Note we use the same notation $\|\cdot\|_p$ for both vector Hölder norms and matrix Hölder norms. The context will make it clear. The most familiar matrix Hölder norms are when $p = 1, 2, \infty$. When $p = 1$, the Hölder norm is known as the l_1 norm

$$\|A\|_1 = \sum_{i=1}^{n}\sum_{j=1}^{m}|a_{ij}|. \tag{2.4}$$

When $p = 2$, the Hölder norm is known as the *Euclidean norm* (or l_2 norm, or *Frobenius norm*), i.e.

$$\|A\|_F = \|A\|_2 = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m}|a_{ij}|^2}. \tag{2.5}$$

When $p = \infty$, the Hölder norm is known as the *max norm* (or l_∞ norm), i.e.

$$\|A\|_\infty = \max\{|a_{ij}|\}. \tag{2.6}$$

Another way to define the matrix norm is to derive it from the vector norms. This is called the induced norm, or operator norm.

Definition 2.5 [4] (induced norms)
Let $\|\cdot\|_p$ be a vector norm on both $\mathbb{R}^n$ and $\mathbb{R}^m$, then $\forall A \in \mathbb{R}^{n\times m}$ and $\forall x \in \mathbb{R}^m$, the induced norms on $\mathbb{R}^{n\times m}$ is defined as

$$\|A\|_p = \max_{\|x\|_p=1}\|Ax\|_p, \tag{2.7}$$

or equivalently,

$$\|A\|_p = \max_{x\neq 0}\frac{\|Ax\|_p}{\|x\|_p}. \tag{2.8}$$

∎

Since both Hölder norm and induced norm use the same notation $\|\cdot\|_p$, to eliminate the confusion, throughout this dissertation we shall use $\|\cdot\|_F$ for the Frobenius norm (2.5), and $\|\cdot\|_2$ for the induced norm (2.7) when $p = 2$. We also use the simplified notation $\|\cdot\|$ for the Frobenius norm of a matrix or a Euclidean norm of a vector.

Denote the eigenvalues of a matrix $A \in \mathbb{R}^{n\times n}$ (who has real eigenvalues) as $\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_n(A)$. Then the singular values of a matrix is defined accordingly.

Definition 2.6 [4] (singular value)
Let $A \in \mathbb{R}^{n\times m}$. Then the singular values of A are $\min\{n, m\}$ nonnegative numbers, denoted by $\sigma_1(A), \sigma_2(A), \cdots, \sigma_{\min\{n,m\}}(A)$, where

$$\sigma_i(A) = \sqrt{\lambda_i(A^TA)} = \sqrt{\lambda_i(AA^T)}$$

and $\sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_{\min\{n,m\}}(A) \geq 0$. We also denote the largest singular value as $\bar{\sigma} = \sigma_1$ and the smallest singular value as $\underline{\sigma} = \sigma_{\min\{m,n\}}$. ∎

Some useful results of the matrix theory, which are used in this dissertation, will be listed as follows.

Lemma 2.1 [4] Let $A^T = A \in \mathbb{R}^{n\times n}$, then $\lambda_{min}(A)I \leq A \leq \lambda_{max}(A)I$, where $I \in \mathbb{R}^{n\times n}$ is the identity matrix. ∎

Lemma 2.2 $\forall x \in \mathbb{R}^n$ and let $A^T = A \in \mathbb{R}^{n\times n}$ be positive definite, then $\lambda_{min}(A)\|x\|^2 \leq x^T Ax \leq \lambda_{max}(A)\|x\|^2$. ∎

Lemma 2.3 [4] $\forall x \in \mathbb{R}^n$, the following monotonicity property of vector norms holds,

$$\|x\|_\infty \leq \cdots \leq \|x\|_2 \leq \|x\|_1 . \tag{2.9}$$

∎

Lemma 2.4 [4] (Hölder inequality)
$\forall x, y \in \mathbb{R}^n$ and $\forall p, q \in [1, 2, \cdots, \infty]$, if $1/p + 1/q = 1$, then

$$|x^T y| \leq \|x\|_p \|y\|_q . \tag{2.10}$$

Note $1/\infty = 0$. When $p = q = 2$, this is known as *Cauchy-Schwarz inequality.* ∎

Lemma 2.5 [4] Let $A \in \mathbb{R}^{n\times m}$ and $x \in \mathbb{R}^m$, then $\|Ax\| \leq \|A\|_F \|x\|$. ∎

Lemma 2.6 [4] $\forall A \in \mathbb{R}^{n\times m}$, we have

$$\|A\|_F = \sqrt{tr(A^T A)} = \sqrt{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2}$$

. where $tr(\cdot)$ is the trace of a square matrix. ∎

Lemma 2.7 $tr(A) = \sum_{i=1}^n \lambda_i(A), \forall A \in \mathbb{R}^{n\times n}$. ∎

Lemma 2.8 [4] Let $A \in \mathbb{R}^{n\times m}$, then $\bar{\sigma}(A) \leq \|A\|_F \leq \sqrt{rank(A)}\bar{\sigma}(A)$. ∎

Lemma 2.9 [4] $\forall x \in \mathbb{R}^n$ and $\forall y \in \mathbb{R}^m$, then $\bar{\sigma}(xy^T) = \|xy^T\|_F = \|x\|\,\|y\|$. ∎

Lemma 2.10 [4] $\forall A \in \mathbb{R}^{n\times m}$, denote $\langle A\rangle \triangleq \sqrt{A^T A}$, then

$$tr\langle A\rangle = \sigma_1(A) + \sigma_2(A) + \sigma_{\min\{m,n\}}(A).$$

∎

Proof. Let $B = \langle A\rangle = \sqrt{A^T A}$, then $B \geq 0$ (i.e. B is positive semidefinite) and $B^2 = A^T A$. Then $\lambda_i(A^T A) = \lambda_i(B^2) = (\lambda_i(B))^2$, i.e. $\sigma_i(A) = |\lambda_i(B)| = \lambda_i(B)$. Thus $tr(B) = tr\langle A\rangle = \lambda_1(B) + \cdots + \lambda_{\min\{m,n\}}(B) = \sigma_1(A) + \cdots + \sigma_{\min\{m,n\}}(A)$. □

Lemma 2.11 If $A^T = A \in \mathbb{R}^{n\times n}$, then $\sigma_i(A) = |\lambda_i(A)|$ and $tr(A) \leq tr\langle A\rangle$. ∎

Lemma 2.12 [4] If $A \in \mathbb{R}^{n\times m}$, then $\|A\|_2 = \bar{\sigma}(A)$. ∎

Lemma 2.13 [4] Let $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{m\times l}$, then

$$\bar{\sigma}(AB) \leq \|AB\|_F \leq \left\{ \begin{array}{l} \bar{\sigma}(A)\|B\|_F \\ \|A\|_F \bar{\sigma}(B) \\ tr\langle AB\rangle \end{array} \right\} \leq \|A\|_F \|B\|_F .$$

∎

Lemma 2.14 [4] Let $A, B \in \mathbb{R}^{n\times m}$, then $\bar{\sigma}(AB) \leq \bar{\sigma}(A)\bar{\sigma}(B)$. ∎

Lemma 2.15 [4] Let $A, B \in \mathbb{R}^{n\times m}$, then $tr(A^T B) \leq \|A\|_F \|B\|_F$. ∎

Lemma 2.16 [4] Let $A \in \mathbb{R}^{n\times n}$ is nonsingular, then $\underline{\sigma}(A) = \frac{1}{\bar{\sigma}(A)}$. ∎

Lemma 2.17 [4] Let $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{m\times l}$. Then

- If $m \leq n$, then $\sigma_m(A)\|B\|_F \leq \|AB\|_F$
- If $m \leq l$, then $\|A\|_F \sigma_m(B) \leq \|AB\|_F$.
- if $m = n = l$, then $\|AB\|_F \geq \left\{ \begin{array}{l} \|A\|_F \underline{\sigma}(B) \\ \underline{\sigma}(A)\|B\|_F \end{array} \right.$

∎

2.2 Fundamental stability theory

Stability is a major concern for a control system. Many stability concepts are defined to describe every detail of the stability properties of systems. We only list some stability concepts and results that will be used in this dissertation. These are classical materials and can be found in many textbooks, e.g. [10], [28], [70], [71], etc.

2.2.1 Definitions of stability

We put the definitions of stability in the context of the most general case, i.e. continuous time non-autonomous nonlinear system. Linear systems and autonomous nonlinear systems are special cases of non-autonomous nonlinear systems. Stabilities of discrete-time systems are conceptually the same with stabilities of continuous-time systems.

Consider the continuous-time non-autonomous nonlinear system

$$\dot{x} = f(t, x), \quad x \in \mathbb{R}^n, \tag{2.11}$$

where $f : [0, \infty) \times D \to \mathbb{R}$ is piecewise continuous in t and locally Liptschitz in x on $[0, \infty) \times D$ and $D \subseteq \mathbb{R}^n$ is a domain that contains the origin $x = 0$.

Definition 2.7 [71] (equilibrium point)
A state x_e is an equilibrium point of the system (2.11), if once $x(t)$ is equal to x_e, it remains equal to x_e for all the future time. This implies $f(x_e, t) = 0$ for all $t \geq 0$. ∎

The origin $x = 0$ is an equilibrium point of system (2.11) at $t = 0$ if $f(t, 0) = 0$ for all $t \geq 0$. In this dissertation, we only consider the case when $x = 0$ is an equilibrium point. Stability is defined with respect to the equilibrium point of system (2.11).

Definition 2.8 [28] The equilibrium point $x = 0$ of system (2.11) is

- stable if, $\forall \epsilon > 0$, $\exists \delta = \delta(\epsilon, t_0) > 0$, such that $\|x(t_0)\| \leq \delta \Rightarrow \|x(t)\| \leq \epsilon$, $\forall t \geq t_0 \geq 0$
- asymptotically stable if, it is stable and $\exists c = c(t_0)$, such that $\|x(t_0)\| \leq c \Rightarrow \|x(t)\| \to 0$ as $t \to \infty$
- uniformly stable if, $\forall \epsilon > 0$, $\exists \delta = \delta(\epsilon) > 0$ which is independent of t_0, such that $\|x(t_0)\| \leq \delta \Rightarrow \|x(t)\| \leq \epsilon$, $\forall t \geq t_0 \geq 0$
- uniformly asymptotically stable if, $\forall \epsilon > 0$, $\exists \delta = \delta(\epsilon)$ and $T = T(\epsilon)$, such that $\|x(t_0)\| \leq \delta \Rightarrow \|x(t)\| \leq \epsilon$, $\forall t \geq t_0 + T$
- globally uniformly asymptotically stable if, $\forall \epsilon > 0$ and $\forall \delta > 0$ (δ can be arbitrarily large), $\exists T = T(\epsilon, \delta)$, such that $\|x(t_0)\| \leq \delta \Rightarrow \|x(t)\| \leq \epsilon$, $\forall t \geq t_0 + T$
- exponentially stable if, $\exists c, k, \lambda > 0$ such that $\|x(t)\| \leq k \|x(t_0)\| e^{-\lambda(t-t_0)}$, $\forall x(t_0) < c$
- globally exponentially stable if, $\exists k, \lambda > 0$ such that $\|x(t)\| \leq k \|x(t_0)\| e^{-\lambda(t-t_0)}$, $\forall x(t_0) \in \mathbb{R}^n$

∎

Remove t_0 and "uniformly" in Definition 2.9, similar stabilities (i.e. stable, asymptotically stable, globally asymptotically stable) can be defined for autonomous system $\dot{x} = f(x)$.

A more advanced stability concept is uniformly ultimately boundedness, which describes the property of the solutions of the system (2.11).

Definition 2.9 [28] The solutions of (2.11) is

- uniformly bounded if $\exists c > 0$, independent of t_0, and $\forall \alpha \in (0, c)$, $\exists \beta = \beta(\alpha) > 0$, such that $\|x(t_0)\| \leq \alpha \Rightarrow \|x(t)\| \leq \beta$, $\forall t \geq t_0$.

- uniformly ultimately bounded by b if $\exists b, c > 0$ which are independent of t_0, and $\forall \alpha \in (0, c)$, $\exists T = T(\alpha, b)$, such that $\|x(t_0)\| \leq \alpha \Rightarrow \|x(t)\| \leq b, \forall t \geq T + t_0$.

∎

Remark 2.1 For linear time invariant (LTI) systems, under either discrete time or continuous time frameworks, when we say stable, it always means asymptotically stable (or equivalently exponentially stable). Moreover, it is always uniform and global for LTI system. ∎

Stability criteria for a LTI system can be described by the eigenvalues of the system matrix.

Lemma 2.18 [10] Consider continuous time LTI system $\dot{x} = Ax$, where $x \in \mathbb{R}^n$ and A is nonsingular. The equilibrium point $x = 0$ is said to be asymptotically stable (or exponentially stable) if and only if all the eigenvalues of A have negative real parts. ∎

Lemma 2.19 [10] Consider discrete time LTI system $x(k+1) = Ax(k)$, where $x(k) \in \mathbb{R}^n$ and A is nonsingular. The equilibrium point $x = 0$ is said to be asymptotically stable (or exponentially stable) if and only if all the eigenvalues of A are within the unit disc. ∎

2.2.2 Lyapunov function

The eigenvalue criteria do not apply to linear time-varying systems nor nonlinear systems. In these cases, we have to resort to a more general stability analysis approach, called Lyapunov function approach, which is named after the Russian mathematician Alexandr Mikhailovich Lyapunov. Two Lyapunov's methods are used in the stability analysis, i.e. Lyapunov direct method and Lyapunov indirect method. Lyapunov indirect method is also known as Lyapunov linearization method, which analyze the local stability of a nonlinear system. Lyapunov direct method is an energy like stability analysis method and is most widely used in analyzing the stability of nonlinear systems, which include linear systems as special cases. We shall focus on the Lyapunov direct method in this dissertation. An introduction to Lyapunov direct method can be found in almost every modern control or nonlinear control textbook, such as [10], [28], [70], [71]. We only list some Lyapunov stability results relevant to this dissertation. Before proceeding, some definitions are needed.

Definition 2.10 [28] Let $V : D \to \mathbb{R}$ be a continuous differential function, where $D \subseteq \mathbb{R}^n$ is a domain containing $x = 0$. Then $V(x)$ is said to be positive semi-definition if $V(x) \geq 0$, $\forall x \in D$; it is said to be positive definite if $V(x) \geq 0$, $\forall x \in D$ and $V(x) = 0$ if and only if $x = 0$; it is said to be negative semidefinite (negative definite) if $-V(x)$ is positive semidefinite (positive definite); it is said to be radially unbounded if $x \to \infty \Rightarrow V(x) \to \infty$. ∎

Definition 2.11 [28] Let $V : [0, \infty) \times D \to \mathbb{R}$ be a continuous differential function, where $D \subseteq \mathbb{R}^n$ is a domain containing $x = 0$. Then the time-varying function $V(t, x)$ is said to be positive semi-definition if $V(t, x) \geq 0, \forall t \geq 0$ and $\forall x \in D$; it is said to be positive definite if $V(t, x) \geq W_1(x)$ for some positive definite function $W_1(x)$; it is said to be radially unbounded if $W_1(x)$ is radially unbounded; it is said to be decrescent if $V(t, x) \leq W_2(x)$ for some positive definite function $W_2(x)$; it is said to be negative semidefinite (negative definite) if $-V(t, x)$ is positive semidefinite (positive definite). ∎

Note Definition 2.10 and 2.11 are defined locally on a domain D. They can also be defined globally by making $D = \mathbb{R}^n$. Throughout this dissertation, we do not indicate whether the positive definiteness of $V(x)$ (or $V(t, x)$) is defined locally or globally, for the context will make it clear.

Definition 2.12 [28] A continuous function $\alpha : [0, a) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$. Moreover if $a = \infty$ and $\alpha(r) \to \infty$ as $r \to \infty$, it is a class $\mathcal{K}_\infty$ function. ∎

Theorem 2.1 [28] The equilibrium point $x = 0$ of system (2.11) is uniformly stable, if there exists a positive definite function $V(t, x)$ which is also decrescent, and $\dot{V}(t, x)$ is negative semidefinite, $\forall t \geq 0$ and $\forall x \in D$. ∎

Theorem 2.2 [28] The equilibrium point $x = 0$ of system (2.11) is uniformly asymptotically stable, if there exists a positive definite function $V(t, x)$ which is also decrescent, and $\dot{V}(t, x)$ is negative definite, $\forall t \geq 0$ and $\forall x \in D$. ∎

Theorem 2.3 [28] The equilibrium point $x = 0$ of system (2.11) is globally uniformly asymptotically stable, if there exists a positive definite, decrescent and radially unbounded function $V(t, x)$, and $\dot{V}(t, x)$ is negative definite, $\forall t \geq 0$ and $\forall x \in \mathbb{R}^n$. ∎

Theorem 2.4 [28] The solution of system (2.11) is uniformly ultimately bounded if there exists a continuous differentiable function $V : [0, \infty) \times D \to \mathbb{R}$, such that $\forall t \geq 0$ and $\forall x \in D$

- $\alpha_1(\|x\|) \leq V(t, x) \leq \alpha_2(\|x\|)$,
- $\dot{V}(t, x) \leq -W_3(x), \forall \|x\| \geq \mu > 0$,

where α_1 and α_2 are class $\mathcal{K}$ functions and $W_3(x)$ is a positive definite function. ∎

For LTI systems, the Lyapunov theory can also be expressed by the following lemmas, where $V(x) = x^T P x$ is the Lyapunov function.

Lemma 2.20 [10] Consider the continuous LTI system $\dot{x} = Ax$. Its equilibrium point $x = 0$ is asymptotically stable, or all the eigenvalues of system matrix A have negative real parts, if for any positive definite matrix $Q > 0$, the Lyapunov equation

$$A^T P + PA = -Q \tag{2.12}$$

has a unique solution P and P is positive definite. ∎

Lemma 2.21 [10] Consider the discrete LTI system $x(k+1) = Ax(k)$. Its equilibrium point $x = 0$ is asymptotically stable, or all the eigenvalues of system matrix A are within the unit circle, if for any positive definite matrix Q, the Lyapunov equation

$$A^T PA - P = -Q \tag{2.13}$$

has a unique solution P and P is positive definite. ∎

Remark 2.2 Note that all the Lyapunov stability results only provide sufficient conditions. If a suitable Lyapunov function is found, stability of the system is guaranteed by the above results. Otherwise, nothing can be said about the stability property of the system. Non-existence of a suitable Lyapunov function not necessarily implies instability. ∎

2.2.3 Control Lyapunov function

While Lyapunov function techniques are used to analysis the stability of a given system (or closed-loop control system), the control Lyapunov function (CLF) techniques are used to design the controller for a control system. CLF technique is used in Chapter 5 when we design the distributed adaptive controllers.

Consider the system

$$\dot{x} = f(x, u), \tag{2.14}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}$ and $f(0,0) = 0$. Our goal is to find a feedback control law $u = k(x)$ such that the equilibrium point $x = 0$ of the closed-loop system $\dot{x} = f(x, k(x))$ is globally asymptotically stable. This goal can be achieved if we can find a control Lyapunov function for system (2.14).

Definition 2.13 [30] A smooth positive definite and radially bounded function $V : \mathbb{R}^n \to \mathbb{R}$ is called a control Lyapunov function for system (2.14) if

$$\inf_{u \in \mathbb{R}} \left\{ \frac{\partial V}{\partial x}(x) f(x, u) \right\} < 0, \forall x \neq 0. \tag{2.15}$$

∎

From the definition, it is clear that the existence of a CLF is equivalent to the statement that there exists a positive definite Lyapunov function $V(x)$, whose derivative $\dot{V}(x)$ is negative definite, thence the existence of a CLF implies system (2.14) is globally asymptotically stable.

□ **End of chapter.**

Chapter 3

UTC-RHC for discrete time linear systems

In this chapter, inspired by reinforcement learning (RL), we propose a learning featured receding horizon control algorithm, named updated terminal cost receding horizon control (UTC-RHC). As the beginning of our journey, we develop this algorithm in the framework of unconstrained discrete linear time-invariant (LTI) systems. It is not just because this system is easier to begin with, but because a great deal of analysis for standard RHC has been performed for such systems, allowing for a meaningful comparison of the new algorithm and a clear understanding of its improved performance. Existing work on RHC for LTI systems includes, to name a few, Kwon et al. ([31][32][34]), Quan et al. [63] and Bitmead et al. ([7][8]). These important contributions clarified the structure of RHC, including terminal requirements on cost or state, and/or requirements for a long enough horizon.

This chapter is organized as follows. Section 3.1 presents the standard algorithm of receding horizon control, together with some relevant stability results and performance bound; Then we review in Section 3.2 the general concepts of reinforcement learning and lists some basic algorithms of RL. Section 3.3 describes the new algorithm, i.e., UTC-RHC, and provides its stability and convergence proof, and also investigate the relation between UTC-RHC and value iteration; Simulation results comparing the performance of RHC and UTC-RHC are shown in Section 3.4; Finally we end this chapter with a conclusion in Section 3.5.

3.1 Receding horizon control

To be self contained and for the convenience of illustration, we start with the linear quadratic (LQ) optimal control or LQR [44], which is probably the most fundamental approach to modern control design, to which RHC is closely related. The LQ optimal control problem is often put into either infinite horizon or finite horizon framework.

Consider the LTI dynamical system

$$x_{k+1} = Ax_k + Bu_k, \;\; x(k_0) = x_0 \tag{3.1}$$

where $x_k \in \mathbb{R}^n$ is the state vector, k_0 is the initial time / stage, $u_k \in \mathbb{R}^m$ is the control input, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are constant matrices. We also make the following assumptions.

Assumption 3.1

A1) $Q \in \mathbb{R}^{n\times n}, R \in \mathbb{R}^{m\times m}, Q = Q^T \geq 0,\ R = R^T > 0$

A2) (A, B) is stabilizable

A3) $(A, \sqrt{Q})$ is detectable

∎

To this dynamical system associate the cost function

$$V(x_k) = \sum_{i=k}^{\infty} r(x_i, u_i), \tag{3.2}$$

where $r(x, u) = x^T Q x + u^T R u$ is the stage cost, also known as the reward function in computer science or utility in economics. This is an infinite horizon quadratic energy-based cost function that quantifies the performance of any control sequence. Good control sequences result in smaller costs. The cost function provides a basis for comparison of the goodness of different control inputs, and so forms the basis for applying reinforcement learning methods to (3.1).

3.1.1 Infinite horizon linear quadratic optimal control

The infinite horizon LQ optimal control problem can be formulated as

Problem $\mathscr{P}(x_k, k + k_0)$:

Find an infinite control sequence $\{u_i\}_{i=k}^{\infty}$ such that (3.2) is minimized where x_k is the current state at the current stage $k + k_0$. ∎

By LQR theory [44], the infinite horizon optimal control is a stationary feedback control law given by

$$u^{IH}(x_k) = -(B^T P_\infty B + R)^{-1} B^T P_\infty A x_k = L_\infty x_k, \tag{3.3}$$

where P_∞ is the unique maximal positive semidefinite solution of the associated algebraic Riccati equation (ARE)

$$P = A^T P A + Q - A^T P B (B^T P B + R)^{-1} B^T P A. \tag{3.4}$$

The optimal cost is

$$V^{IH}(x_k, k + k_0) = x_k^T P_\infty x_k. \tag{3.5}$$

Under control law (3.3), the closed-loop system is

$$x_{k+1} = (A - (B^T P_\infty B + R)^{-1} B^T P_\infty A) x_k. \tag{3.6}$$

The following Lemma is an important property of the stability of ARE.

Lemma 3.1 (Theorem 4.1 in[7] or Theorem 2.4-2 in [44])
Suppose Assumption 3.1 holds. Then

1) There exists a unique positive semidefinite solution P_∞ to the algebraic Riccati equation (3.4).

2) The closed-loop system (3.6) is asymptotically stable, i.e.,

$$A - (B^T P_\infty B + R)^{-1} B^T P_\infty A$$

has all its eigenvalues strictly within the unit circle.

∎

3.1.2 Finite horizon linear quadratic optimal control

The finite horizon LQ optimal control problem can be formulated as

Problem $\mathscr{P}(x_k, k + k_0, N, P_0)$**:**

Find a control sequence $\{u_i\}_{i=k}^{k+N-1}$ such that the following summation is minimized

$$\sum_{i=k}^{k+N-1} (x_i^T Q x_i + u_i^T R u_i) + x_{k+N}^T P_0 x_{k+N}, \tag{3.7}$$

where x_k is the current state at the current stage $k + k_0$, N is the horizon length and $P_0 \geq 0$ is the terminal cost weighting matrix. ∎

Denote the finite horizon optimal control sequence as $\{u^{FH}(i; x_k, k + k_0, N, P_0)\}_{i=k}^{k+N-1}$, then

$$\begin{aligned} & u^{FH}(k + j; x_k, k + k_0, N, P_0) \\ = & -(B^T P_{N-j-1} B + R)^{-1} B^T P_{N-j-1} A x_{k+j}, \ \ j = 0, 1, \cdots, N-1, \end{aligned} \tag{3.8}$$

where P_t is the t-th term in the solution sequence of the Riccati difference equation (RDE) (3.9) with initial condition P_0.

$$P_{t+1} = A^T P_t A + Q - A^T P_t B (B^T P_t B + R)^{-1} B^T P_t A \tag{3.9}$$

Denote the optimal cost as $V^{FH}(x_k, k + k_0, N, P_0)$, then

$$\begin{aligned} & V^{FH}(x_k, k + k_0, N, P_0) \\ = & \min_{\bar{u}_i} \{ \sum_{i=k}^{k+N-1} (x_i^T Q x_i + u_i^T R u_i) + x_{k+N}^T P_0 x_{k+N} \} = x_k^T P_N x_k, \end{aligned} \tag{3.10}$$

where $\bar{u}_i = \{u_k, u_{k+1}, \cdots, u_{k+N-1}\}$.

Two well known properties of the RDE, i.e., convergence property and monotonicity property, are presented below.

Lemma 3.2 [7] Consider the RDE (3.9). Suppose Assumption 3.1 holds and $P_0 \geq 0$. Then $P_t \to P_\infty$ as $t \to \infty$, where P_∞ is the unique maximal positive semidefinite solution to the ARE (3.4). ∎

Lemma 3.3 [7] If the positive semidefinite solution P_t of the RDE (3.9) is monotonically non-increasing (non-decreasing) at one time, then P_t is monotonically non-increasing (non-decreasing) for all subsequent times. ∎

3.1.3 Receding horizon control

The finite horizon optimal control can deal with the input and state constraints by using mathematical programming, such as quadratic programming and semidefinite programming. However, the industry processes, especially the chemical reaction processes, are often slow and run for hours, days or even weeks. In these cases, infinite horizon optimal control is more practical than finite horizon optimal control. Although the infinite horizon optimal control algorithm ensures stability for any linear system under mild conditions (stability and detectability), it can not deal with constraints, model uncertainty, etc., which are common in industry [61]. This leads to a mixed-mode optimal control problem between finite horizon and infinite horizon, i.e., receding horizon control.

Receding horizon control is a suboptimal control strategy for an infinite horizon optimal control problem. It applies the finite horizon optimal control repeatedly in an infinite time context. At each stage, an open-loop finite horizon, say N, optimal control problem is solved, yielding an optimal control sequence. Only the first control in the resulting control sequence is applied at the current stage. At the next stage, the horizon is shifted forward and the same procedure is repeated. The detailed RHC scheme under LQ framework is stated as follows:

At the current stage $k + k_0$, a finite horizon LQ optimal control problem $\mathscr{P}(x_k, k + k_0, N, P_0)$ is solved, yielding the optimal control sequence (3.8). Taking the first control in the sequence (3.8), the receding horizon optimal control law at current stage $k + k_0$ is

$$\begin{aligned} & u^{RH}(x_k, k + k_0, N, P_0) \\ & = u^{FH}(k; x_k, k + k_0, N, P_0) \\ & = -(B^T P_{N-1} B + R)^{-1} B^T P_{N-1} A x_k = L_N^{RH} x_k. \end{aligned} \tag{3.11}$$

The controller (3.11) drives the system (3.1) to the next stage $k + k_0 + 1$.

At the next stage $k+k_0+1$, the N-horizon LQ optimal control problem $\mathscr{P}(x_{k+1}, k+1+k_0, N, P_0)$ is re-solved for the *measured* new state x_{k+1}, and again the first control in the control sequence is applied. Thus

$$\begin{aligned} & u^{RH}(x_{k+1}, k + 1 + k_0, N, P_0) \\ =& u^{FH}(k + 1; x_{k+1}, k + 1 + k_0, N, P_0) \\ =& -(B^T P_{N-1} B + R)^{-1} B^T P_{N-1} A x_{k+1} = L_N^{RH} x_{k+1}. \end{aligned} \tag{3.12}$$

For fixed horizon N and P_0, the receding horizon control gain is constant, as shown by Eqs. (3.11) and (3.12). The RHC closed-loop system takes the following form

$$x_{k+1} = (A - B(B^T P_{N-1} B + R)^{-1} B^T P_{N-1} A) x_k. \tag{3.13}$$

By Lemma 3.2, $P_{N-1} \to P_\infty$ as $N \to \infty$, thus $L_N^{RH} \to L_\infty$ as $N \to \infty$. This also means the receding horizon control is not optimal unless $N \to \infty$.

3.1.4 Stability of RHC

Closed-loop stability has been an important issue in the RHC literature. Almost all the stability results put requirements on the terminal state x_{k+N} and/or the terminal cost weighting matrix P_0 and/or the horizon length N. This chapter has no intention to give a thorough review of the existing stability results. Readers may refer to [50] for an excellent survey about the RHC stability issue. Here we only list some of the stability approaches that are related to our algorithm and put them into the following three categories.

Constraint on the terminal state

This approach falls into two subcategories. One requires the terminal state to vanish at the origin, i.e., $x_{k+N} = 0$, which is also known as the terminal equality constraint (see, for example, [27], [34]). But this condition is somewhat strong. The other requires the terminal state to enter a neighborhood of the origin, and in this neighborhood, a local stabilizing controller is applied [51].

Constraint on the terminal cost function

The choice of terminal cost weighting matrix P_0 is important for the stability of RHC. Ideally, one should choose $P_0 = P_\infty$. In that case the finite horizon LQ optimal control problem $\mathscr{P}(x_k, k+k_0, N, P_0)$ would be the same as infinite horizon LQ optimal control problem $\mathscr{P}(x_k, k+k_0)$, therefore the closed-loop stability can be automatically guaranteed. When P_∞ is hard to get due to nonlinearities or constraints, researchers show that closed-loop stability can still be obtained by choosing the terminal cost properly ([7], [8], [23]). For discrete linear time-invariant systems, Bitmead et al. [7] obtained the following result:

Lemma 3.4 (Theorem 4.7 in [7]) Consider the RDE (3.9) and suppose Assumption 3.1 holds. If $P_{n+1} \leq P_n$ for some integer $n \geq 0$, then the closed-loop system (3.13) is stable for all $N \geq n+1$. ∎

A special case is when the terminal cost weighting matrix satisfies $P_0 \geq P_1$, then the RHC closed-loop system (3.13) is asymptotically stable for all $N \geq 1$.

Theorem 1 in [38] implies that if $Q > 0$ and P_0 satisfies

$$P_0 \geq (A + BH)^T P_0 (A + BH) + Q + H^T R H \tag{3.14}$$

for some $H \in \mathbb{R}^{m\times n}$, the stabilizing RHC controller can also be obtained for any $N \geq 1$. This is a more relaxed condition than $P_0 \geq P_1$. Condition (3.14) says the terminal cost $V(x) = x^T P_0 x$ is a control Lyapunov function (CLF) with the associated stabilizing feedback control gain H, which provides an incremental upper bound on the cost-to-go in the sense that $\Delta V(x_k) \leq -(x_k^T Q x_k + u_k^T R u_k)$.

Constraint on the horizon length

When there is no terminal cost (i.e., $P_0 = 0$) or for a general terminal cost (i.e., $P_0 \geq 0$), the relationship between stability and the horizon size is tricky. Since P_j will eventually approach P_∞, which leads to a stabilizing closed-loop system, one may conjecture that if for some N, P_N leads to a stable closed-loop system, then for any $n \geq N$, P_n will also lead to a stable closed-loop system. However this is not true. Readers may refer to [9] and [60] for some examples.

But research also shows that when the horizon size N is chosen to be long enough, the stability of RHC scheme can be guaranteed. Primbs and Nevistic [59] showed that, for constrained discrete LTI systems, picking the terminal cost as $P_0 = Q$, there exists a finite horizon length N^*, such that for all $N \geq N^*$, the RHC closed-loop system is asymptotically stable. They also indicate that the same result holds even for arbitrary $P_0 \geq Q$; Jadbabaie et al. [23] showed that, for input constrained nonlinear systems, exponential stability can be obtained for RHC scheme with general positive semidefinite terminal cost if a sufficiently long horizon is adopted; Grimm et al. [17] showed that, for both unconstrained discrete time nonlinear system and input-constraint linear system, a long enough horizon can ensure the exponential stability of the MPC scheme without particular requirements on the terminal cost.

For unconstrained discrete LTI systems, Bitmead et al. [8] obtained the following result:

Lemma 3.5 (Theorem 10.17 in [8]) Suppose $P_0 \geq 0$. Then there exits an $N^* < \infty$ such that the RHC closed-loop system (3.13) is asymptotically stable for all $N \geq N^*$. ∎

This result only guarantees the eventual stability of RHC scheme for arbitrary $P_0 \geq 0$, yet does not indicate how big N^* should be. Recently, an explicit expression of stability guaranteed RHC horizon size for discrete LTI system is given in [63].

3.1.5 Ties between RHC and rollout algorithm

One goal of this chapter is to relate RHC more closely to methods of approximate dynamic programming (ADP) and reinforcement learning (RL). In [5] and [6], Bertsekas presents a one-step lookahead algorithm known as rollout algorithm. Therefore in this section, we present the next result, inspired by Proposition 3.1 in [5], which shows a cost improvement property for rollout algorithms. The next result extends Bertsekas' Proposition to the case $N \geq 1$ for LQR problems, and also provides a bound for the performance of RHC.

First note that under Assumption 3.1, for any stabilizing feedback control law $u_i = Lx_i$, we claim that there exists a $P_0 \geq 0$ such that for all k and x_k,

$$x_k^T P_0 x_k = \sum_{i=k}^{\infty} (x_i^T Q x_i + u_i^T R u_i), \tag{3.15}$$

and P_0 can be computed using the Lyapunov equation

$$P_0 = (A + BL)^T P_0 (A + BL) + (Q + L^T R L). \tag{3.16}$$

This can be seen from the following derivation. Since L is a stabilizing control gain, i.e., $A + BL$ has all its eigenvalues strictly inside the unit disc, by Lyapunov theory, $P_0 \geq 0$ exists and is unique. Furthermore, the solution P_0 can be expressed as

$$P_0 = \sum_{j=0}^{\infty} ((A + BL)^T)^j (Q + L^T R L)(A + BL)^j. \tag{3.17}$$

Then $\forall x_k$, noting $x_{j+k} = (A + BL)^j x_k$ for $j \geq 0$, we have

$$\begin{aligned} x_k^T P_0 x_k &= \sum_{j=0}^{\infty} x_k^T ((A + BL)^T)^j (Q + L^T R L)(A + BL)^j x_k \\ &= \sum_{i=k}^{\infty} x_i^T (Q + L^T R L) x_i \\ &= \sum_{i=k}^{\infty} (x_i^T Q x_i + u_i^T R u_i). \end{aligned}$$

Lemma 3.6 *Consider system (3.1). Suppose Assumption 3.1 holds. Let $u_i = Lx_i$ be a stabilizing feedback control law and P_0 satisfies (3.15). Define $V^{RH}(x_k, k + k_0, N, P_0)$ as the cost-to-go by using RHC control, i.e.,*

$$V^{RH}(x_k, k + k_0, N, P_0) = \sum_{i=k}^{\infty} \{x_i^T Q x_i + (L_N^{RH} x_i)^T R (L_N^{RH} x_i)\}.$$

Then

$$V^{RH}(x_k, k + k_0, N, P_0) \leq x_k^T P_N x_k \leq \cdots \leq x_k^T P_0 x_k, \quad \forall N \geq 1, \tag{3.18}$$

where P_j ($j = 0, 1, \cdots, N$) is the j-th term in the solution sequence of RDE (3.9) with initial value P_0.

Proof:
Since (3.15) can be written in the recursive form

$$x_k^T P_0 x_k = x_k^T Q x_k + u_k^T R u_k + x_{k+1}^T P_0 x_{k+1}, \tag{3.19}$$

then for all k and x_k,

$$\min_u \{x_k^T Q x_k + u^T R u + x_{k+1}^T P_0 x_{k+1}\} \leq x_k^T P_0 x_k. \tag{3.20}$$

Also by LQR theory [44],

$$\min_u\{x_k^TQx_k + u^TRu + x_{k+1}^TP_jx_{k+1}\} = x_k^TP_{j+1}x_k,\ \ j = 0, 1, 2\cdots. \tag{3.21}$$

(3.20) and (3.21) imply $P_1 \leq P_0$, and it follows that $P_{j+1} \leq P_j$ for $j = 0, 1, 2, \cdots$, by the monotonicity property of RDE (e.g., Lemma 3.3). Thus $x_k^TP_{j+1}x_k \leq x_k^TP_jx_k$ for all k and x_k, i.e.,

$$\min_u\{x_k^TQx_k + u^TRu + x_{k+1}^TP_jx_{k+1}\} \leq x_k^TP_jx_k,\ \ j = 0, 1, 2, \cdots. \tag{3.22}$$

Denote the one-step lookahead policy

$$\begin{aligned} u_k^j &= \arg\min_u\{x_k^TQx_k + u^TRu + x_{k+1}^TP_jx_{k+1}\} \\ &= L^jx_k, \quad j = 0, 1, 2, \cdots. \end{aligned} \tag{3.23}$$

Then by Proposition 3.1 in [5], the cost-to-go corresponding to u_k^j satisfies for all x_k and k,

$$\begin{aligned} &\sum_{i=k}^{\infty}\{x_i^TQx_i + (u_i^j)^TRu_i^j\} \\ \leq &x_k^TP_{j+1}x_k \leq x_k^TP_jx_k,\ \ \forall j = 0, 1, 2, \cdots. \end{aligned} \tag{3.24}$$

Observing $u_k^j = u^{RH}(x_k, k + k_0, j + 1, P_0)$, we have

$$\sum_{i=k}^{\infty}\{x_i^TQx_i + (u_i^j)^TRu_i^j\} = V^{RH}(x_k, k + k_0, j + 1, P_0).$$

Thus (3.24) implies

$$V^{RH}(x_k, k + k_0, j + 1, P_0) \leq x_k^TP_{j+1}x_k \leq x_k^TP_jx_k,\ \ \forall j = 0, 1, 2, \cdots. \tag{3.25}$$

With non-increasing monotonicity of the sequence $\{P_j\}$, we can draw the conclusion that

$$V^{RH}(x_k, k + k_0, N, P_0) \leq x_k^TP_Nx_k \leq x_k^TP_{N-1}x_k \leq \cdots \leq x_k^TP_0x_k,\ \ \forall N \geq 1.$$

□

Remark 3.1 The importance of this result is that it shows RHC can always improve on, or at least is as good as, the performance of any initial stabilizing feedback control law $u_i = Lx_i$ for any $N \geq 1$, in the sense that $V^{RH}(x_k, k + k_0, N, P_0) \leq x_k^TP_0x_k$. ∎

Remark 3.2 One may conjecture from (3.18) that under the condition $P_1 \leq P_0$, RHC may perform better in the sense that $V^{RH}(x_k, k + k_0, N, P_0)$ gets smaller, as N gets bigger. However, this is not the case. In fact (3.18) only means that $V^{RH}(x_k, k + k_0, N, P_0)$ is upper bounded by $x_k^TP_Nx_k$ and says nothing about the monotonicity of $V^{RH}(x_k, k + k_0, N, P_0)$, although $\{P_j\}$ is monotonically non-increasing, as shown by the following counterexample. ∎

Counter example 3.1 Consider the open-loop unstable system [63]

$$x_{k+1} = \begin{bmatrix} 0 & 3 \\ -4 & 5 \end{bmatrix} x_k + \begin{bmatrix} 2 \\ 2 \end{bmatrix} u_k. \tag{3.26}$$

Let $Q = \begin{bmatrix} 1.3 & 0 \\ 0 & 1.4 \end{bmatrix}$, $R = 0.1$. Obviously, A, B, R and Q satisfy Assumption 3.1. The solution of the ARE (3.4) is $P_\infty = \begin{bmatrix} 114.6619 & -56.4269 \\ -56.4269 & 29.7112 \end{bmatrix} > 0$. Pick $P_0 = 2P_\infty$. Computing RDE (3.9) gives $P_1 = \begin{bmatrix} 227.7368 & -112.9642 \\ -112.9642 & 57.9799 \end{bmatrix}$. It is easy to check that $P_1 < P_0$. Picking an initial state $x_0 = [-10, 18]^T$, by applying the RHC control law from the beginning $k_0 = 0$, one obtains $V^{RH}(x_0, 0, 1, P_0) = 41407$, $V^{RH}(x_0, 0, 2, P_0) = 42369$ and $V^{RH}(x_0, 0, 3, P_0) = 41413$ for $N = 1$, $N = 2$ and $N = 3$ respectively. Obviously, $V^{RH}(x_0, 0, 2, P_0) > V^{RH}(x_0, 0, 1, P_0)$, so the performance does not necessarily improve as the horizon gets longer. ∎

3.2 Reinforcement learning

Reinforcement learning (RL) is a method of machine learning that is based on learning mechanisms observed in mammals [58][73]. Every living organism interacts with its environment and uses those interactions to improve its own actions in order to survive and increase. Reinforcement learning refers to an actor or agent that interacts with its environment and modifies its actions, or control policies, based on stimuli received in response to its actions. This is based on evaluative information from the environment and could be called *action-based learning*. RL is a means of *learning optimal behaviors by observing the response from the environment to nonoptimal control policies*. Note *control policy* is an alias of *control law* and it is frequently used in the computational intelligence community. We do not differentiate them in this dissertation.

Our objective is to apply RL techniques for feedback control of dynamic systems that can be described in terms of difference equations.

Define a control policy $u_k = h(x_k)$ as a mapping from the state space $\mathbb{R}^n$ to the control input space $\mathbb{R}^m$. For the LTI system one is concerned with linear state variable feedbacks so that $u_k = h(x_k) = Lx_k$ for some feedback gain matrix L. We say a policy is *admissible* if it stabilizes the system (3.1), i.e., closed-loop system $x_{k+1} = (A + BL)x_k$ has all its eigenvalues inside the unit circle, and also renders the cost function finite. If a policy $u_k = h(x_k)$ is admissible, then

$$V(x_k) = \sum_{i=k}^{\infty} (x_i^T Q x_i + h(x_i)^T R h(x_i)) = \sum_{i=k}^{\infty} r(x_i, h(x_i)) \tag{3.27}$$

is called the value of the policy. For admissible policies, one may write the difference equation equivalent to (3.2) as

$$V(x_k) = r(x_k, h(x_k)) + V(x_{k+1}). \tag{3.28}$$

That is, the value of the policy can be determined either by evaluating (3.27) along the system trajectories, or by solving (3.28) with the condition $V(0) = 0$. Equation (3.28) is known as the Bellman equation.

For the LQR, it is known that the value of an admissible control policy is quadratic in the state, that is

$$V(x_k) = \sum_{i=k}^{\infty} r(x_i, h(x_i)) = x_k^T P x_k, \tag{3.29}$$

for some kernel matrix (or cost weighting matrix) $P \in \mathbb{R}^{n \times n}$. Then the Bellman equation is a Lyapunov equation that can be written in the form

$$x_k^T P x_k = r(x_k, h(x_k)) + x_{k+1}^T P x_{k+1}. \tag{3.30}$$

According to Bellman's principle of optimality, the minimum value $V^*(x_k)$, corresponding to the optimal control policy that minimizes the cost (3.27), may be found by solving the Hamilton-Jacobi-Bellman (HJB) equation

$$V^*(x_k) = \min_{h(\cdot)} \{r(x_k, h(x_k)) + V^*(x_{k+1})\}. \tag{3.31}$$

Then, the optimal policy is given by

$$h^*(x_k) = \arg\min_{h(\cdot)} \{r(x_k, h(x_k)) + V^*(x_{k+1})\}. \tag{3.32}$$

For the LQR, the HJB equation (3.31) is a Riccati equation.

Determining the optimal value and optimal control using the HJB equation is an off-line procedure that requires knowledge of the system dynamics to solve (3.31). By contrast, in this chapter one is concerned with finding online methods of learning good control policies. The Bellman equation (3.28) or (3.30) provides the basis for developing RL methods for finding the optimal value and policy using online learning techniques. Here are given three standard RL methods ([58], [73], [83]). They give insight on how to improve RHC by adding a learning feature, as detailed in Section 3.3.

The HJB equation is a fixed-point equation. Therefore, it can be solved using the method of successive approximations based on contraction maps. Write the HJB equation in the form

$$0 = \min_{h(\cdot)} \{-V^*(x_k) + r(x_k, h(x_k)) + V^*(x_{k+1})\}. \tag{3.33}$$

This fixed-point equation can be used to define a contraction map that leads to a method of solution known as policy iteration (PI), as given in the next well known algorithm.

Policy iteration (PI) algorithm

Initialization. Select a stabilizing control policy $h_0(x_k) = L_0 x_k$

Policy evaluation step. Determine the value $V_{j+1}(\cdot)$ of the current control policy $h_j(\cdot)$ by

$$x_k^T P_{j+1} x_k = r(x_k, h_j(x_k)) + x_{k+1}^T P_{j+1} x_{k+1}, \ \ \forall x_k \in \mathbb{R}^n \tag{3.34}$$

Policy improvement step. Improve the policy using

$$h_{j+1}(x_k) = \arg\min_{h(\cdot)}(r(x_k, h(x_k)) + V_{j+1}(x_{k+1})), \forall x_k \in \mathbb{R}^n \tag{3.35}$$

then increase j by 1, go to the policy evaluation step and repeat until h_j converges. ∎

PI is an online reinforcement learning method, since at each step j one evaluates the current policy to determine its value using the Bellman equation (3.34). Then, based on the newly determined value, the control is updated using policy update (3.35). It is shown in [5] that (3.35) always leads to an improved policy in the sense that the value $V_{j+2}(x_k)$ of using the new policy $h_{j+1}(x_k)$ is less than or equal to the value $V_{j+1}(x_k)$ of using the former policy $h_j(x_k)$.

PI requires an initial stabilizing gain, since only for admissible policies do there exist meaningful solutions to (3.34). It has been shown by Hewer [19] that for the LQR this algorithm converges to the optimal policy $h^*(x_k)$ and value $V^*(x_k)$ under Assumption 3.1 if started with a stabilizing initial policy. That is, PI learns the optimal policy and value by evaluating the performance of intermediate suboptimal policies $h_j(x_k)$ through solving (3.34). It is well known how to solve (3.34) online using data measured along the system trajectories using methods such as recursive least-squares (RLS) or batch least-squares.

Considering the HJB equation in the form (3.31), one can define a different contraction map for solving the fixed-point HJB equation. This leads to a method of solution known as value iteration (VI), as given in the next well known algorithm.

Value iteration (VI) algorithm

Initialization. Select an arbitrary control policy $h_0(\cdot)$ and initial value function $V_0(\cdot)$

Value update step. Update the value function using

$$x_k^T P_{j+1} x_k = r(x_k, h_j(x_k)) + x_{k+1}^T P_j x_{k+1}, \ \forall x_k \in \mathbb{R}^n \tag{3.36}$$

Policy improvement step. Improve the policy using

$$h_{j+1}(x_k) = \arg\min_{h(\cdot)}(r(x_k, h(x_k)) + V_{j+1}(x_{k+1})), \ \forall x_k \in \mathbb{R}^n \tag{3.37}$$

then increase j by 1, go to the value update step and repeat until h_j converges. ∎

VI algorithm is based on performing (3.36) at each step, which is not an equation but simply a replacement iteration. Therefore, VI does not require an initial stabilizing gain. It has been shown by Lancaster and Rodman [36] that for the LQR this algorithm converges under Assumption 3.1. VI was also called heuristic dynamic programming (HDP) by Werbos ([83], [84]), who defined a family of optimal control learning algorithms for general nonlinear systems, based on the idea of VI, which are known generally as approximate dynamic programming (ADP).

Considering the fact that the Bellman equation (3.30) is also a fixed point equation, one can solve it also by using successive approximation using a contraction map. This allows us to replace the solution of (3.34) by an iterative procedure, as in the next generalized PI (GPI) algorithm.

Generalized policy iteration (GPI) algorithm

Initialization. Select an arbitrary control policy $h_0(x_k) = L_0 x_k$ and value function $V_0(x_k) = x_k^T P_0 x_k$

Value update step. Update the value function for all $x_k \in \mathbb{R}^n$ by iterating on

$$x_k^T P_j^{i+1} x_k = r(x_k, h_j(x_k)) + x_{k+1}^T P_j^i x_{k+1}, \;\; i = 0, 1, 2, \cdots, M-1 \tag{3.38}$$

for some finite positive integer M, where the initial condition is $P_j^0 = P_j$. Set $P_{j+1} = P_j^M$.

Policy improvement step. Improve the policy using

$$h_{j+1}(x_k) = \arg\min_{h(\cdot)} (r(x_k, h(x_k)) + V_{j+1}(x_{k+1})), \; \forall x_k \in \mathbb{R}^n \tag{3.39}$$

then increase j by 1, go to the value update step and repeat until h_j converges. ∎

Remark 3.3 PI, VI, and GPI have been discussed from a computational intelligence point of view in [6], [58], [73], and elsewhere. GPI provides an unified expression of PI and VI. When $M = 1$, (3.38) turns out to be (3.36), which relates to value iteration; On the other hand, when $N \to \infty$, (3.38) provides a method for solving (3.34), which associates with policy iteration. Note in the latter case, the initial control policy $h_0(x_k) = L_0 x_k$ should be stabilizing. ∎

To investigate the relation of RL and RHC, we present the following result, which provides an alternative way to compute the value function $V_{j+1}(\cdot)$ (or equivalently P_{j+1}) in (3.38).

Lemma 3.7 *Consider (3.38) and the following equation*

$$x_k^T W_{j+1} x_k = \sum_{l=k}^{k+M-1} r(x_l, h_j(x_l)) + x_{k+M}^T W_j x_{k+M}, \;\; \forall x_k \in \mathbb{R}^n. \tag{3.40}$$

If $W_j = P_j^0 = P_j$, then $W_{j+1} = P_j^M = P_{j+1}$. In other words, the value update step (3.38) in GPI can be replaced by

$$x_k^T P_{j+1} x_k = \sum_{l=k}^{k+M-1} r(x_l, h_j(x_l)) + x_{k+M}^T P_j x_{k+M}, \;\; \forall x_k \in \mathbb{R}^n \tag{3.41}$$

Proof:

This can be proved by mathematical induction as follows.

1) When $M = 1$, it is obvious that $W_{j+1} = P_j^1$.

2) Assume when $M = n \geq 1$, $W_{j+1} = P_j^n$, i.e.,

$$\sum_{l=k}^{k+n-1} r(x_l, h_j(x_l)) + x_{k+n}^T W_j x_{k+n} = r(x_k, h_j(x_k)) + x_{k+1}^T P_j^{n-1} x_{k+1}, \; \forall x_k \in \mathbb{R}^n. \tag{3.42}$$

This implies that

$$\sum_{l=k+1}^{k+n} r(x_l, h_j(x_l)) + x_{k+n+1}^T W_j x_{k+n+1} = r(x_{k+1}, h_j(x_{k+1})) + x_{k+2}^T P_j^{n-1} x_{k+2}. \tag{3.43}$$

Then when $M = n + 1$, for any $x_k \in \mathbb{R}^n$, we have

$$\begin{aligned} x_k^T P_j^{n+1} x_k &= r(x_k, h_j(x_k)) + x_{k+1}^T P_j^n x_{k+1} \\ &= r(x_k, h_j(x_k)) + [r(x_{k+1}, h_j(x_{k+1})) + x_{k+2}^T P_j^{n-1} x_{k+2}] \\ &= r(x_k, h_j(x_k)) + \sum_{l=k+1}^{k+n} r(x_l, h_j(x_l)) + x_{k+n+1}^T W_j x_{k+n+1} \\ &= \sum_{l=k}^{k+n} r(x_l, h_j(x_l)) + x_{k+n+1}^T W_j x_{k+n+1} \\ &= x_k^T W_{j+1} x_k. \end{aligned} \tag{3.44}$$

□

Remark 3.4 Comparison of Eq. (3.41) and Eq. (3.36) suggests that GPI can be regarded as multi-step VI. It is also easy to verify that Lemma 3.7 holds for $M \to \infty$ when GPI becomes PI. ∎

3.3 Updated terminal cost receding horizon control

In this section, we show how to integrate the concept of reinforcement learning into the standard RHC scheme, so that the performance of RHC is improved in two senses. First, the restrictive stability conditions in Section 3.1 are removed; and second, the modified algorithm converges after enough stages to the optimal infinite horizon control law.

The key idea for incorporating learning into RHC is to compare (3.10) with (3.41) which updates the terminal cost function at each step. To make this more specific, the RHC algorithm is described in a different way. This is justified by applying the principle of optimality [44] to (3.10). Suppose the initial terminal cost function is $V_0(x) = x^T P_0 x$, initial stage is k_0 and the horizon is N. At stage $k + k_0$ ($k \geq 0$), the following three steps are equivalent to equation (3.10):

S1) Solve for an $(N-1)$-horizon optimal cost function $V_1^*(x_k)$ for all $x_k \in \mathbb{R}^n$ by

$$V_1^*(x_k) = \begin{cases} \min_{\bar{u}_i} \{\sum_{i=k}^{k+N-2} r(x_i, u_i) + V_0(x_{k+N-1})\}, & N \geq 2; \\ V_0(x_k), & N = 1. \end{cases} \tag{3.45}$$

S2) Compute the RHC control law by

$$u^{RH}(x_k, k + k_0, N, P_0) = \arg\min_u \{r(x_k, u) + V_1^*(x_{k+1})\}. \tag{3.46}$$

S3) Compute the optimal cost function for the current state by

$$V_2^*(x_k) = r(x_k, u^{RH}(x_k, k+k_0, N, P_0)) + V_1^*(x_{k+1}). \tag{3.47}$$

Obviously, the RHC control law given by (3.46) is the same as (3.11). This standard RHC contains no ingredient of learning. It solves the same finite horizon optimal control problem at each stage (see (3.45) and (3.46)). On the other hand, the RL algorithm solves a different optimal control problem at each step. Specifically, the value $V_{j+1}(\cdot)$ learned from the previous policy $h_j(\cdot)$ is used as a basis for finding the new policy $h_{j+1}(\cdot)$, as in (3.35), (3.37) and (3.39). A comparison of (3.39) to (3.46), and (3.41) to (3.45) suggests a modified RHC scheme, which we call updated terminal cost RHC (UTC-RHC). In this modified scheme, the terminal cost in (3.45) (or equivalently in (3.7) and (3.10)) is updated at each step by the value learned in the previous stage. Again by using the principle of optimality, the UTC-RHC algorithm can be presented in a traditional way as follows.

3.3.1 Algorithm

At the initial stage k_0, the corresponding N-horizon LQ optimal control problem solved by UTC-RHC is $\mathscr{P}(x_0, k_0, N, P_0)$. Solving problem $\mathscr{P}(x_0, k_0, N, P_0)$ yields the optimal control sequence

$$\bar{u}^{FH}(\cdot; x_0, k_0, N, P_0) = \{u^{FH}(i; x_0, k_0, N, P_0)\}_{i=0}^{N-1} \tag{3.48}$$

and the optimal cost

$$V^{FH}(x_0, k_0, N, P_0) = x_0^T P_N x_0. \tag{3.49}$$

Then the UTC-RHC control law at stage k_0 is given by the first control in the sequence, i.e.,

$$\begin{aligned} u^{UR}(x_0, k_0, N, P_0) =& u^{FH}(0; x_0, k_0, N, P_0) \\ =& -(B^T P_{N-1} B + R)^{-1} B^T P_{N-1} A x_0. \end{aligned} \tag{3.50}$$

At the stage $k_0 + 1$, we modify the terminal cost weighting matrix of the corresponding N-horizon LQ optimal control problem to P_N, which is obtained at stage $k = k_0$ as in (3.49). Solving the problem $\mathscr{P}(x_1, k_0 + 1, N, P_N)$ yields an optimal control sequence

$$\bar{u}^{FH}(\cdot; x_1, k_0 + 1, N, P_N) = \{u^{FH}(i; x_1, k_0 + 1, N, P_N)\}_{i=1}^{N}, \tag{3.51}$$

and the optimal cost

$$V^{FH}(x_1, k_0 + 1, N, P_N) = x_1^T P_{2N} x_1. \tag{3.52}$$

Then the UTC-RHC control law at stage $k_0 + 1$ is given by

$$u^{UR}(x_1, k_0 + 1, N, P_N) = u^{FH}(1; x_1, k_0 + 1, N, P_N). \tag{3.53}$$

Similarly, at stage $k+k_0 (k \geq 1)$, the corresponding N-horizon LQ optimal control problem solved by UTC-RHC is $\mathscr{P}(x_k, k+k_0, N, P_{kN})$, i.e.,

$$x_k^T P_{(k+1)N} x_k = \min_{\bar{u}_i} \{ \sum_{i=k}^{k+N-1} (x_i^T Q x_i + u_i^T R u_i) + x_{k+N}^T P_{kN} x_{k+N} \}. \tag{3.54}$$

Note the terminal cost weighting matrix is *updated at each stage* $k+k_0$ to P_{kN} $(k \geq 1)$, which associates with the optimal cost function $V^{FH}(x_{k-1}, k-1+k_0, N, P_{(k-1)N})$ solved at the previous stage. Solving problem $\mathscr{P}(x_k, k+k_0, N, P_{kN})$ yields the following optimal control sequence

$$\bar{u}^{FH}(\cdot; x_k, k+k_0, N, P_{kN}) = \{u^{FH}(i; x_k, k+k_0, N, P_{kN})\}_{i=k}^{k+N-1}. \tag{3.55}$$

The UTC-RHC control law at stage $k+k_0$ $(k \geq 1)$ is

$$u^{UR}(x_k, k+k_0, N, P_{kN}) = u^{FH}(k; x_k, k+k_0, N, P_{kN}). \tag{3.56}$$

The following lemma shows that updating the terminal cost weighting matrix is equivalent to lengthening the horizon.

Lemma 3.8

$$V^{FH}(x_k, k+k_0, N, P_{kN}) = \quad V^{FH}(x_k, k+k_0, (k+1)N, P_0) \tag{3.57}$$

∎

Proof: By knowledge of LQR, it is straightforward to have the result that $V^{FH}(x_k, k+k_0, N, P_t) = x_k^T P_{t+N} x_k$ for all $k \geq 0$ and all $t \geq 0$. Then by principle of optimality [44], one obtains

$$\begin{aligned}
&V^{FH}(x_k, k+k_0, N, P_{kN}) \\
=&\min_{\bar{u}_i} \{ \sum_{i=k}^{k+N-1} (x_i^T Q x_i + u_i^T R u_i) + x_{k+N}^T P_{kN} x_{k+N} \} \\
=&\min_{\bar{u}_i} \{ \sum_{i=k}^{k+N-1} (x_i^T Q x_i + u_i^T R u_i) + V^{FH}(x_{k+N}, k+N+k_0, N, P_{(k-1)N}) \} \\
=&\min_{\bar{u}_i} \{ \sum_{i=k}^{k+N-1} (x_i^T Q x_i + u_i^T R u_i) \\
&+ \min_{\bar{u}_t} \{ \sum_{t=k+N}^{k+2N-1} (x_t^T Q x_t + u_t^T R u_t) + x_{k+2N}^T P_{(k-1)N} x_{k+2N} \} \} \\
=&\min_{\bar{u}_i} \{ \sum_{i=k}^{k+2N-1} (x_i^T Q x_i + u_i^T R u_i) + x_{k+2N}^T P_{(k-1)N} x_{k+2N} \} \\
=&V^{FH}(x_k, k+k_0, 2N, P_{(k-1)N}).
\end{aligned} \tag{3.58}$$

(3.57) follows by induction. □

Remark 3.5 The terminal cost weighting matrix is updated at each stage $k+k_0$ ($k \geq 1$) to the N horizon LQ optimal cost weighting matrix computed at the previous stage. Equation (3.57) shows that this is in fact equivalent to effectively making the horizon longer by N at each stage; that is, (3.54) is equivalent to

$$\min_{\tilde{u}_i}\{\sum_{i=k}^{(k+1)N-1}(x_i^TQx_i+u_i^TRu_i)+x_{(k+1)N}^TP_{kN}x_{(k+1)N}\}. \tag{3.59}$$

As will be seen in Theorem 3.2, for arbitrary initial terminal cost weighting matrix $P_0 \geq 0$, our algorithm generates a trajectory that converges to the origin exponentially. ∎

Noting equation (3.57), the UTC-RHC control law at stage $k+k_0$ ($k \geq 1$) can be written as

$$\begin{aligned} &u^{UR}(x_k, k+k_0, N, P_{kN}) \\ =&u^{FH}(k; x_k, k+k_0, (k+1)N, P_0) \\ =&-(B^TP_{(k+1)N-1}B+R)^{-1}B^TP_{(k+1)N-1}Ax_k = L_{(k+1)N}^{UR}x_k. \end{aligned} \tag{3.60}$$

Clearly, when $k=0$, equality (3.60) still holds.

Therefore, the UTC-RHC closed-loop system is

$$x_{k+1} = (A - B(B^TP_{(k+1)N-1}B+R)^{-1}B^TP_{(k+1)N-1}A)x_k, \quad k=0,1,2,\cdots, \tag{3.61}$$

where P_t is the t-th term in the solution sequence of the RDE (3.9) with the initial condition P_0. Note that system (3.61) is a linear time-varying system, and uniform in k_0.

3.3.2 Stability and convergence

In this section, it shows in Theorem 3.2 that the UTC-RHC algorithm has significant advantages over standard RHC for unconstrained LTI systems. The next result shows that UTC-RHC guarantees stability under a standard assumption made in RHC. This result is established using a proof technique adopted from [50] which shows the closed-loop stability under RHC scheme. The objective is to tie UTC-RHC firmly to the mode of thinking prevalent in standard RHC.

Theorem 3.1 Consider the UTC-RHC closed-loop system (3.61) with the initial state x_0. Suppose Assumption 3.1 holds with $Q > 0$. Let P_0 satisfy the following inequality

$$P_0 \geq A^TP_0A + Q - A^TP_0B(B^TP_0B+R)^{-1}B^TP_0A \tag{3.62}$$

Then $\forall N \geq 1$, the closed-loop system (3.61) is uniformly exponentially stable. ∎

Proof: Taking

$$\begin{aligned} \mathscr{J}(x_k) =&V^{FH}(x_k, k+k_0, N, P_{kN}) \\ =&V^{FH}(x_k, k+k_0, (k+1)N, P_0) = x_k^TP_{(k+1)N}x_k \end{aligned}$$

as a Lyapunov function candidate, where $P_{(k+1)N}$ is the $((k+1)N) - th$ term in the solution sequence of RDE (3.9) with initial condition P_0. By monotonicity of RDE (Theorem 4.4 in [7]), condition (3.62), i.e., $P_0 \geq P_1$, implies the non-increasing monotonicity of sequence $\{P_t\}$. Thus $\mathscr{J}(x_k) \leq x_k^T P_0 x_k$ for all $x_k \in \mathbb{R}^n$.

To simplify the notations, let $u(x_k) = u^{UR}(x_k, k+k_0, N, P_{kN})$. It is trivial to show that

$$\mathscr{J}(x_k) \geq r(x_k, u(x_k)) \geq x_k^T Q x_k, \quad \forall x_k \in \mathbb{R}^n.$$

Hence, $\mathscr{J}(x_k)$ is positive definite and decrescent.

Now we show $\Delta\mathscr{J}(x_k) = \mathscr{J}(x_{k+1}) - \mathscr{J}(x_k)$ is negative definite. By principle of optimality [44], we have

$$\mathscr{J}(x_k) = r(x_k, u(x_k)) + V^{FH}(x_{k+1}, k+1+k_0, (k+1)N-1, P_0). \tag{3.63}$$

Then

$$\begin{aligned} &\mathscr{J}(x_{k+1}) - \mathscr{J}(x_k) + r(x_k, u(x_k)) \\ =&\mathscr{J}(x_{k+1}) - V^{FH}(x_{k+1}, k+1+k_0, (k+1)N-1, P_0) \\ =&x_{k+1}^T (P_{(k+2)N} - P_{(k+1)N-1}) x_{k+1} \leq 0, \end{aligned} \tag{3.64}$$

since the sequence $\{P_t\}$ is monotonically non-increasing. Therefore,

$$\Delta\mathscr{J}(x_k) = \mathscr{J}(x_{k+1}) - \mathscr{J}(x_k) \leq -r(x_k, u(x_k)) \leq -x_k^T Q x_k, \tag{3.65}$$

i.e., $\Delta\mathscr{J}(x_k)$ is negative definite.

By Lyapunov stability criteria (Theorem 23.3 in [70]), the closed-loop system (3.61) is uniformly exponentially stable. □

Remark 3.6 Theorem 3.1 shows the stability of our algorithm under condition $P_0 \geq P_1$, which is a standard assumption in RHC [50]. One should note that, the condition $P_0 \geq P_1$ may not be replaced by $P_0 \geq P_\infty$. From the monotonicity and convergence properties of RDE (Theorem 4.2 in [7]), one may naturally conjecture that $P_0 \geq P_\infty$ implies $P_t \geq P_{t+1}$ for all $t \geq 0$. However, this is not the case [9]. Since the non-increasing monotonicity of the sequence $\{P_t\}$ is required in the proof, the condition $P_0 \geq P_1$ can not be replaced by $P_0 \geq P_\infty$. ∎

Further investigation reveals that UTC-RHC guarantees the uniform exponential stability under much more relaxed conditions, and it obtains the optimal performance eventually, as shown by the following Theorem.

Theorem 3.2 Consider the UTC-RHC closed-loop system (3.61) with initial state $x(k_0) = x_0$. Then $\forall N \geq 1$ and $\forall P_0 \geq 0$,

a) the closed-loop system (3.61) is uniformly exponentially stable in the sense that

$$\|x_k\| \leq \alpha\beta^k \|x_0\|, \forall k \geq k_0$$

where $0 \leq \beta < 1$ and α is a finite positive number.

b) the UTC-RHC state feedback control gain converges to that of the infinite horizon optimal control law (3.3), i.e., $L^{UR}_{(k+1)N} \to L_\infty$ as $k \to \infty$.

∎

Proof.

a) System (3.61) can be written as

$$x_{k+1} = (A_\infty + A_k)x_k, \;\; x_0 = x(k_0) \tag{3.66}$$

with

$$\begin{aligned} A_\infty =& A - B(B^T P_\infty B + R)^{-1} B^T P_\infty A, \\ A_k =& B(B^T P_\infty B + R)^{-1} B^T P_\infty A - B(B^T P_{(k+1)N-1} B + R)^{-1} B^T P_{(k+1)N-1} A, \end{aligned} \tag{3.67}$$

where P_∞ is the unique maximal positive semidefinite solution of the associated ARE (3.4).

It is well known that $x_{k+1} = A_\infty x_k$ is exponentially stable (Theorem 4.1 in [7]). By Lyapunov theory (Theorem 8-22 in [10]), there exists a unique positive definite symmetric matrix P^* such that

$$P^* = A_\infty^T P^* A_\infty + I, \tag{3.68}$$

where I is the identity matrix of appropriate dimensions. Thus we have

$$\eta I = \lambda_{min}(P^*) I \le P^* \le \lambda_{max}(P^*) I = \gamma I, \tag{3.69}$$

where η and γ are positive real numbers.

Choose $V(x) = x^T P^* x$ as a Lyapunov function candidate for system (3.66). Now we show $\Delta V(x_k)$ satisfies

$$\begin{aligned} \Delta V(x_k) =& V(x_{k+1}) - V(x_k) = x_{k+1}^T P^* x_{k+1} - x_k^T P^* x_k \\ =& [(A_\infty + A_k)x_k]^T P^* [(A_\infty + A_k)x_k] - x_k^T P^* x_k \\ =& x_k^T (A_\infty^T P^* A_\infty + A_\infty^T P^* A_k + A_k^T P^* A_\infty + A_k^T P^* A_k - P^*) x_k \\ =& x_k^T (-I + A_\infty^T P^* A_k + A_k^T P^* A_\infty + A_k^T P^* A_k) x_k \\ =& - x_k^T x_k + x_k^T M_k x_k \\ \le& - \|x_k\|^2 + \|M_k\| \, \|x_k\|^2 \\ =& - (1 - \|M_k\|) \, \|x_k\|^2 , \end{aligned} \tag{3.70}$$

where $M_k = A_\infty^T P^* A_k + A_k^T P^* A_\infty + A_k^T P^* A_k$.

By convergence property of RDE (Theorem 4.2 in [7]), $P_{(k+1)N-1} \to P_\infty$ as $k \to \infty$. Thus $A_k \to 0$, and thence $M_k \to 0$ as $k \to \infty$. Then $\forall\, 0 < \xi < 1$, there exists $0 < K < \infty$, which is only determined by A, B, R, Q, ξ and P_0, such that $\|M_k\| \le \xi$ for all $k \ge K$. Thus

$$\Delta V(x_k) \le -(1 - \xi) \, \|x_k\|^2 , \;\; \forall k \ge K. \tag{3.71}$$

Using essentially the same technique as in Theorem 23.3 in [70], one obtains

$$\|x_k\| \leq \sqrt{\frac{\gamma}{\eta}}\beta^{k-K}\|x_K\|, \forall k \geq K, \tag{3.72}$$

where $\beta = \sqrt{1-\frac{1-\xi}{\gamma}}$ and $0 \leq \beta < 1$. Since (3.66) is linear system, a few steps further yields

$$\|x_k\| \leq \alpha\beta^k\|x_0\|, \ \forall k \geq k_0, \tag{3.73}$$

where $\alpha = \frac{\gamma}{\eta}\omega\beta^{-K}$ is a finite positive number, $\omega = max\{\|\Phi(i+k_0,k_0)\|\}_{i=1}^{K}$ and $\Phi(j,k)$ is the transition matrix of system (3.66). Therefore, system (3.61) is uniformly exponentially stable.

b) This is implied immediately from the result that $P_{(k+1)N-1} \to P_\infty$ as $k \to \infty$, as shown in part a). □

Remark 3.7 In this proof, we do not use the finite horizon optimal cost $\mathscr{J}(x_k)$ as a Lyapunov function, which is prevalently adopted in the literature ([38], [50], etc.), as in the proof of Theorem 3.1. In fact, $\mathscr{J}(x_k)$ does not qualify as a Lyapunov function for Theorem 3.2, because the arbitrary $P_0 \geq 0$ cannot ensure the non-increasing monotonicity of the sequence $\{P_t\}$, which is a must for $\Delta\mathscr{J}(x_k)$ to be negative definite. By decomposing the system matrix into two parts (i.e., a stable part A_∞ and a perturbation part A_k), we find $V(x) = x^T P^* x$ as a Lyapunov function candidate. The difference of this function along the trajectory of the closed-loop system is negative, and hence guarantees the uniform exponential stability. This proof is rooted in the convergence property of RDE and the stability property of ARE. ∎

Remark 3.8 UTC-RHC can be viewed as a RHC algorithm with varing terminal cost weighting matrix. There is a generalized stability result for RHC with varying terminal cost weighting matrix by Lee *et al.* [38] (Theorem 1), which implies that if the terminal cost weighting matrix P_{kN} satisfies the following inequality for some $H_j \in \mathbb{R}^{m\times n}$,

$$P_{kN} \geq (A+BH_k)^T P_{(k+1)N}(A+BH_k) + Q + H_k^T R H_k, \ \ \forall k \geq 0 \text{ and } \forall N \geq 1 \tag{3.74}$$

where $Q > 0$, then UTC-RHC control law exponentially stabilizes the system (3.1). However, the exponential stability conditions for UTC-RHC scheme is more relaxed than the stability conditions required by [38] in the following aspects:

a) Matrix Q is required to be positive definite in [38] to ensure the positive definiteness of the Lyapunov function they choose. Since we choose $V(x) = x^T P^* x$ as the Lyapunov function, the positive definiteness of P^* is guaranteed by Assumption 3.1, in which $Q \geq 0$ and $(A, \sqrt{Q})$ is detectable.

b) Theorem 1 in [38] requires *all* the terminal cost weighting matrices P_{kN} ($\forall k \geq 0$, $\forall N \geq 1$) to be positive definite, more exactly, $P_{kN} \geq Q$ for condition (3.74) to hold. While our result holds for arbitrary $P_0 \geq 0$, even for $P_0 = 0$, and this benefits from the convergence property of RDE and the different Lyapunov function we choose.

c) Even when $Q > 0$ and $P_{kN} \geq Q$, to investigate the stability of UTC-RHC by [38], one has to check the condition (3.74) for every $k \geq 0$ and $N \geq 1$. This is not a trivial job, since for UTC-RHC both $P_0 > 0$ and $N \geq 1$ can be arbitrary. In this sense, our analysis is more germane.

∎

3.3.3 Relation between VI and UTC-RHC

In this section, we disclose the relation between value iteration (VI), also known as heuristic dynamic programming (HDP) by Werbos [84], and UTC-RHC by comparing their underlying iterations. Note that for system (3.1), equation (3.37) of VI is equivalent to

$$h_{j+1}(x_k) = -(B^T P_{j+1} B + R)^{-1} B^T P_{j+1} A x_k, \ \forall x_k \in \mathbb{R}^n. \tag{3.75}$$

Equation (3.36) of VI is equivalent to RDE

$$P_{j+1} = A^T P_j A + Q - A^T P_j B (B^T P_j B + R)^{-1} B^T P_j A \tag{3.76}$$

for $j \geq 1$. Note one can also start the VI algorithm from the policy improvement step, i.e., (3.37), then (3.36) is equivalent to RDE from $j = 0$.

On the other hand, when $N = 1$, the UTC-RHC control law (3.60) is

$$u^{UR}(x_k, k + k_0, 1, P_k) = -(B^T P_k B + R)^{-1} B^T P_k A x_k, \tag{3.77}$$

where P_k is computed from RDE with initial condition P_0. Therefore, the underlying iteration of UTC-RHC and VI are actually the same. However, UTC-RHC and VI differ in the way in which they are implemented. In VI, one takes enough samples along the system trajectory using the same control until (3.36) can be solved, e.g. if $x \in \mathbb{R}^n$, one requires at least $n(n+1)/2$ samples of triple x_k, x_{k+1} and $r(x_k, u_k)$, and a persistence excitation condition. The value update step does not require the knowledge of any system dynamics A or B [1], though A and B are needed in the policy improvement step. On the other hand, in UTC-RHC, one assumes full knowledge of the plant dynamics A and B and solves RDE with N iterations at each time k.

3.4 Simulation results

Two examples are presented to compare the stability and optimality performances of the standard RHC and UTC-RHC algorithm. The first example is a single input open-loop unstable LTI system which is adopted directly from [29]. In the second example, we apply our algorithm to a discretized two-mass translational mechanical system. This is a two-input two-output system.

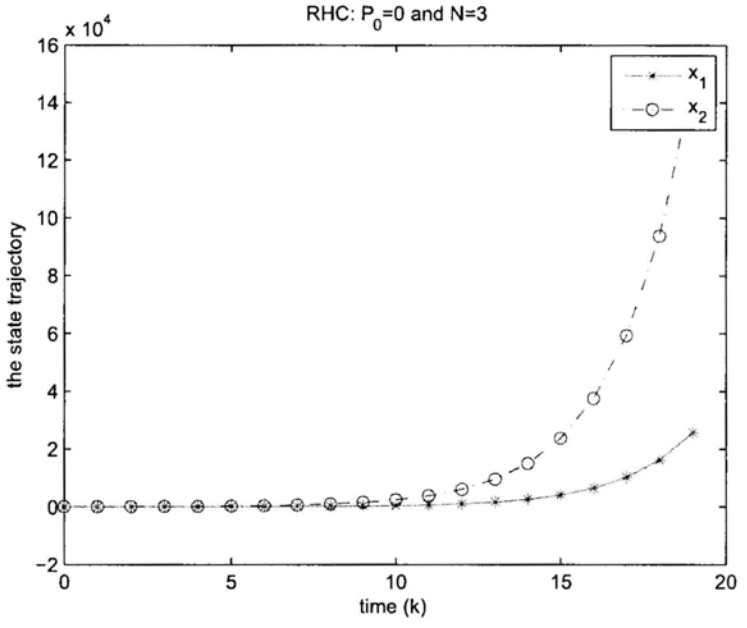

Figure 3.1: State trajectory of RHC closed-loop system for N=3

3.4.1 Example 1

Consider the single input open-loop unstable system (3.26). Q and R take the same values as those in Counterexample 3.1. P_∞ solves the ARE (3.4) and corresponds to the optimal infinite horizon cost, and L_∞ is the optimal control gain given by (3.3).

Then

$$P_\infty = \begin{bmatrix} 114.6619 & -56.4269 \\ -56.4269 & 29.7112 \end{bmatrix} \quad \text{and} \quad L_\infty = \begin{bmatrix} -1.6938 \\ -0.6519 \end{bmatrix}^T$$

For RHC, [63] shows that the stability guaranteed horizon size takes its maximal value when the terminal cost weighting matrix $P_0 = 0$. So for this example, we choose $P_0 = 0$. For simulation, we pick an initial state to be $x_0 = [-6, 4]^T$ while noting that x_0 can be chosen arbitrarily.

Stability of RHC and UTC-RHC

For this example, by standard RHC algorithm, [63] gives the smallest stability guaranteed horizon size N^* with $N^* \geq 5$ and it also points out that the proposed stabilizing horizon size may be conservative. In fact, this system is also stable for $N = 4$ but unstable for $N \leq 3$, as shown in Fig.3.1 and Fig.3.2.

On the other hand, by applying the UTC-RHC algorithm, the closed-loop system is always stable for arbitrary $N \geq 1$, as shown in Fig.3.3 and Fig.3.4.

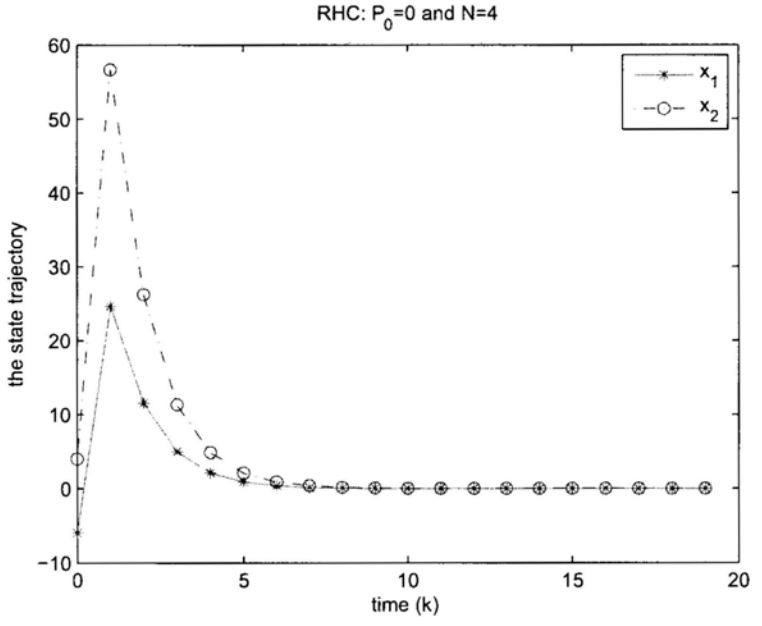

Figure 3.2: State trajectory of RHC closed-loop system for N=4

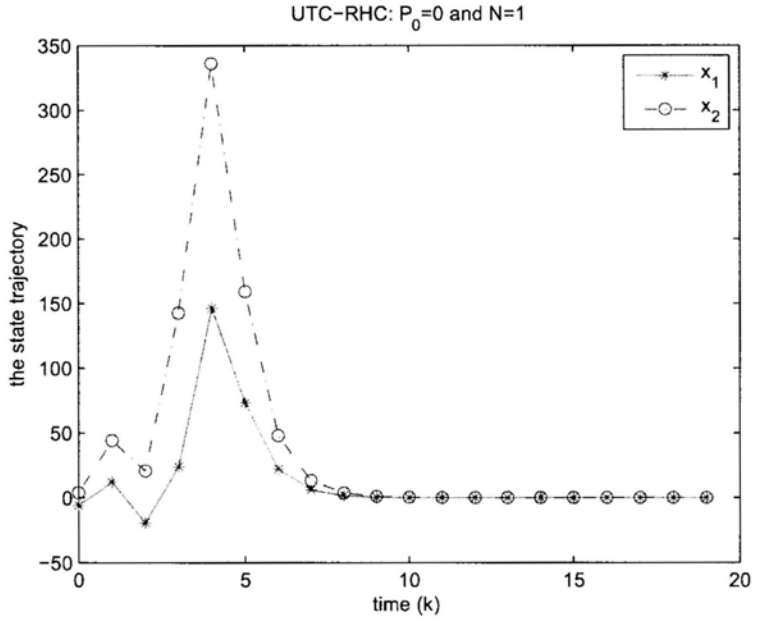

Figure 3.3: State trajectory of UTC-RHC closed-loop system for N=1

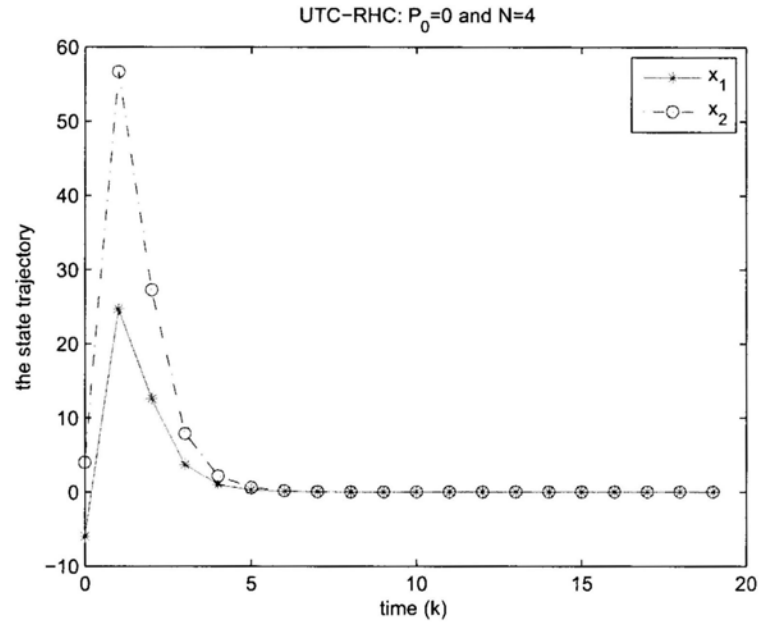

Figure 3.4: State trajectory of UTC-RHC closed-loop system for N=4

Convergence of RHC and UTC-RHC

When $N = 3$, the RHC control gain is given by (3.11) as $L = [L_{11}, L_{12}] = [-0.3988, -1.2938]$; when N = 4 , the RHC control gain is $L = [L_{11}, L_{12}] = [-1.5397, -0.7283]$. For a fixed horizon N , the RHC control gain is constant and not optimal unless $N \to \infty$. On the other hand, the UTC-RHC control gain eventually approaches its optimal value L_∞ for all $N \geq 1$, which is illustrated in Fig.3.5 and Fig.3.6.

3.4.2 Example 2

In this example, we consider a two-mass translational mechanical system [87] shown in Fig.3.7, where m_1 and m_2 are two masses; k_1 and k_2 are spring coefficients; c_1 and c_2 are damping coefficients; u_1 and u_2 are force inputs; y_1 and y_2 are displacement outputs of the two masses.

Define the states as $x_c = [x_{c1}, x_{c2}, x_{c3}, x_{c4}]^T$, where $x_{c1} = y_1$, $x_{c2} = \dot{y}_1 = \dot{x}_{c1}$, $x_{c3} = y_2$ and $x_{c4} = \dot{y}_2 = \dot{x}_{c3}$. Define the output as $y = [y_1, y_2]^T$. The state space representation of this mechanical system is

$$\begin{aligned} \dot{x}_c &= A_c x_c + B_c u_c \\ y &= C_c x_c + D_c u_c \end{aligned} \tag{3.78}$$

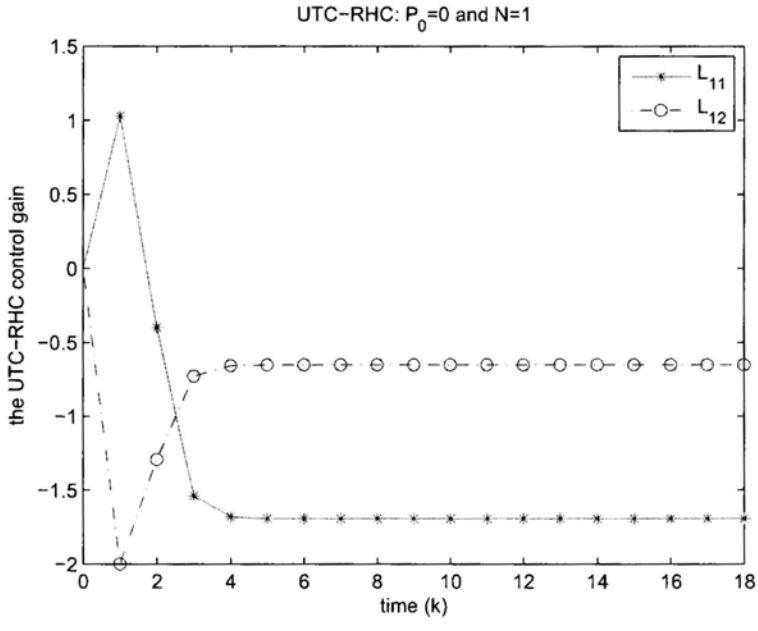

Figure 3.5: Control gain of UTC-RHC algorithm for N=1

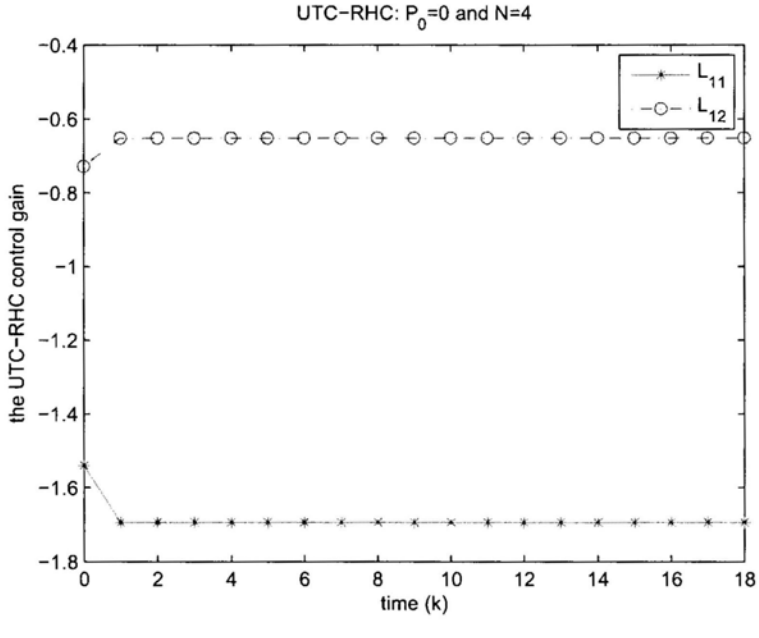

Figure 3.6: Control gain of UTC-RHC algorithm for N=4

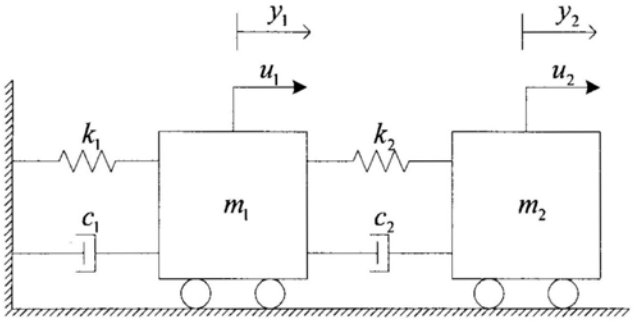

Figure 3.7: The two-mass translational mechanical system

where $A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{-(k_1+k_2)}{m_1} & \frac{-(c_1+c_2)}{m_1} & \frac{k_2}{m_1} & \frac{c_2}{m_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & \frac{c_2}{m_2} & \frac{-k_2}{m_2} & \frac{-c_2}{m_2} \end{bmatrix}$, $B_c = \begin{bmatrix} 0 & 0 \\ \frac{1}{m_1} & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix}$, $C_c = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}^T$ and $D_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$.

This is a 4th-order two-input two-output LTI system. For the convenience of illustration, we let the system parameters be $c_1 = c_2 = 0$, $m_1 = 1$ kg, $m_2 = 1$ kg, $k_1 = 1$ N/m and $k_2 = 1$ N/m, then

$$A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 \end{bmatrix}, \text{ and } B_c = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

The open-loop system is marginally stable, for all the eigenvalues of A_c are on the imaginary axis.

Picking a sampling time $T_S = 0.1s$, the discrete time system is obtained by using the zero-order-hold technique as follows $A_d = \begin{bmatrix} 0.9900 & 0.0997 & 0.0050 & 0.0002 \\ -0.1992 & 0.9900 & 0.0995 & 0.0050 \\ 0.0050 & 0.0002 & 0.9950 & 0.0998 \\ 0.0995 & 0.0050 & -0.0997 & 0.9950 \end{bmatrix}$, $B_d = \begin{bmatrix} 0.0005 & 0 \\ 0.0997 & 0.0002 \\ 0 & 0.0050 \\ 0.0002 & 0.0998 \end{bmatrix}$, $C_d = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ and $D_d = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$.

We consider the above discrete-time system. Choose Q and R to be the identity matrices with appropriate dimensions. P_∞ and L_∞ are obtained similarly as in Example 3.4.1,

$$P_\infty = \begin{bmatrix} 28.8229 & 3.2635 & -7.2967 & 2.2451 \\ 3.2635 & 13.2579 & 2.2451 & 1.6630 \\ -7.2967 & 2.2451 & 21.5262 & 5.5086 \\ 2.2451 & 1.6630 & 5.5086 & 14.9209 \end{bmatrix}, L_\infty = \begin{bmatrix} -0.1868 & -1.2057 & -0.2610 & -0.1568 \\ -0.2610 & -0.1568 & -0.4478 & -1.3625 \end{bmatrix}.$$

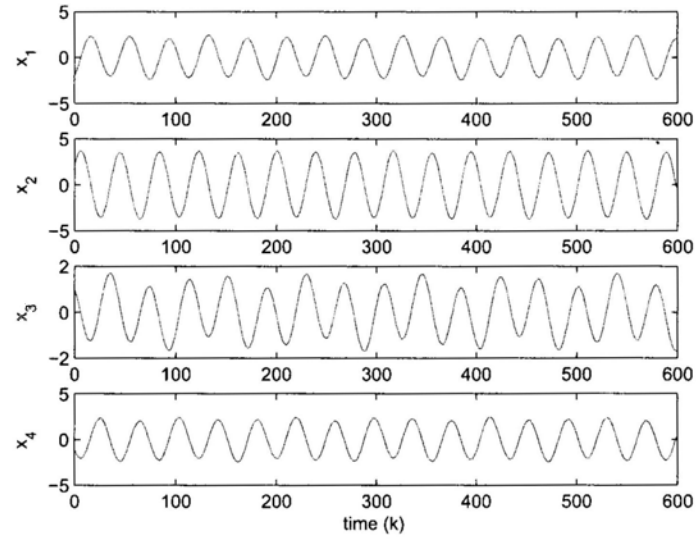

Figure 3.8: State trajectory of the open-loop system

Open-loop system

Obviously, the discrete-time open-loop system is marginally stable, for the eigenvalues of the system matrix A are on the unit circle. Given the initial state $x_0 = [-2, 2, 1, -1]^T$, the state trajectory of the open-loop system is shown in Fig.3.8. This is a periodic oscillation movement.

Stability of RHC and UTC-RHC

Let the terminal cost weighting matrix $P_0 = 0$. We plot the state trajectory of the RHC closed-loop system and UTC-RHC closed-loop system for $N = 1$ and $N = 2$ respectively, as shown in Fig.3.9-Fig.3.12.

As illustrated by these figures, the RHC closed-loop system is not stable when $N = 1$, while the UTC-RHC closed-loop system is stable. In fact, when $N = 1$, the RHC control gain is 0, which will be shown in Fig.3.13, and in this case the closed-loop system is the same as the open-loop system, see Fig.3.8 and Fig.3.9. When $N = 2$, both the RHC closed-loop system and UTC-RHC closed-loop system are stable, but the state approaches the origin much faster for UTC-RHC closed-loop system than for RHC closed-loop system.

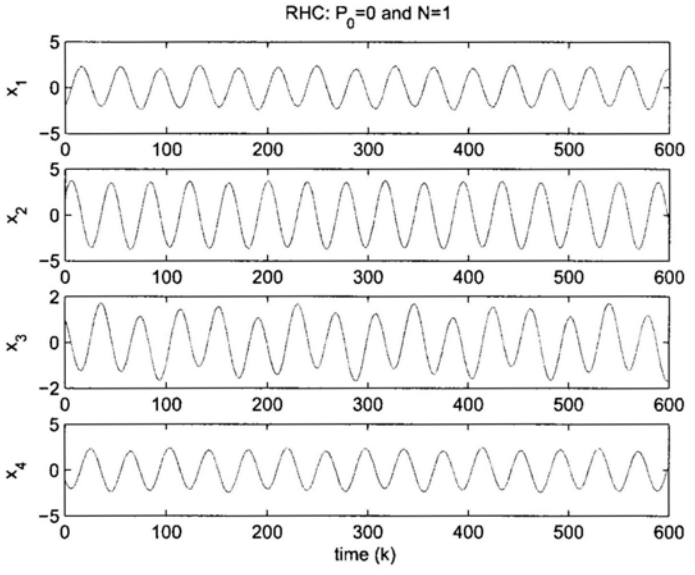

Figure 3.9: State trajectory of the RHC closed-loop system for N=1

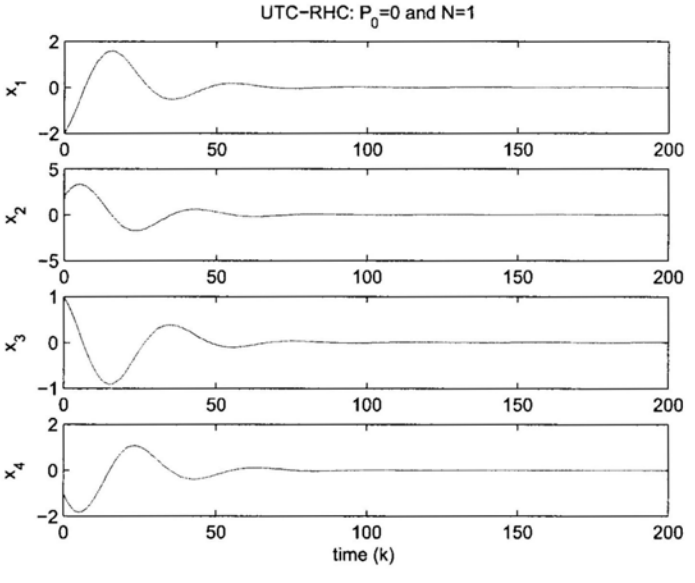

Figure 3.10: State trajectory of the UTC-RHC closed-loop system for N=1

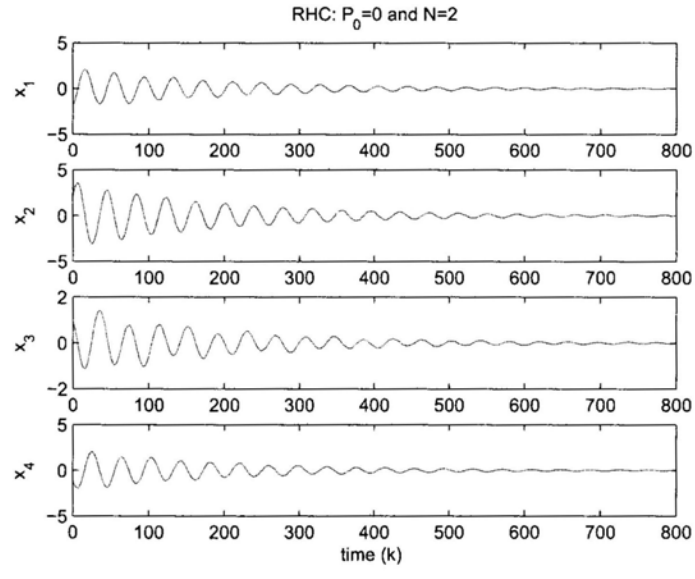

Figure 3.11: State trajectory of the RHC closed-loop system for N=2

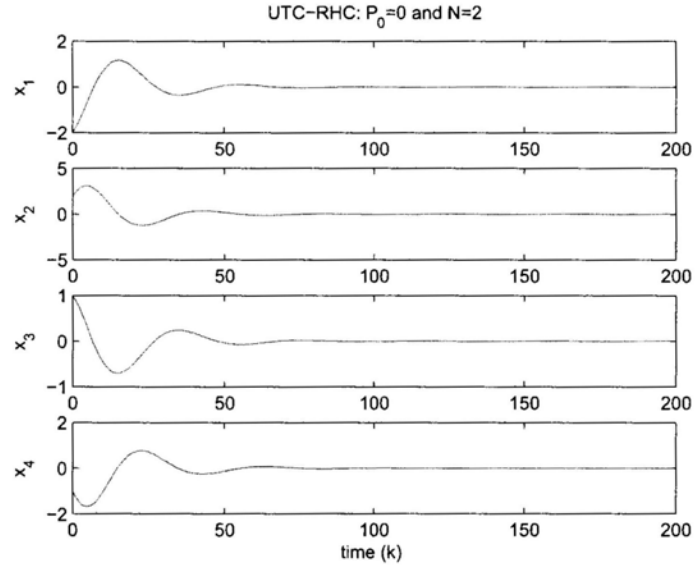

Figure 3.12: State trajectory of the UTC-RHC closed-loop system for N=2

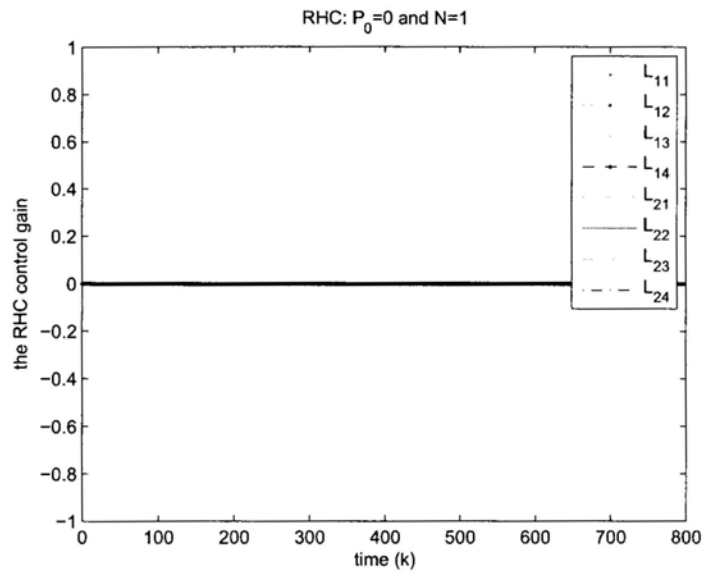

Figure 3.13: Control gain of RHC algorithm for N=1

Convergence of RHC and UTC-RHC

As shown in Fig.3.13 and Fig.3.14, the control gain of the RHC algorithm is constant and not optimal for $N = 1$ and $N = 2$. But for any positive integer N, the control gain of the UTC-RHC algorithm is time varying and converges to the optimal value L_∞ as $k \to \infty$, which is illustrated by Fig.3.15 and Fig.3.16 for the case $N = 1$ and $N = 2$.

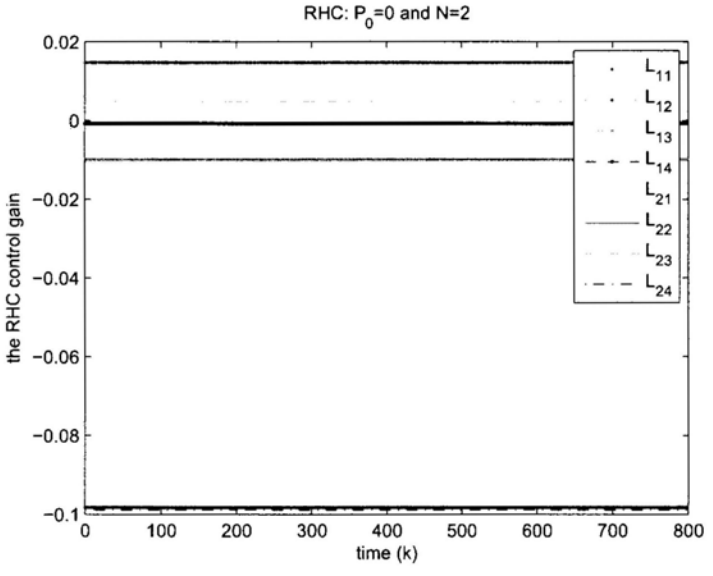

Figure 3.14: Control gain of RHC algorithm for N=2

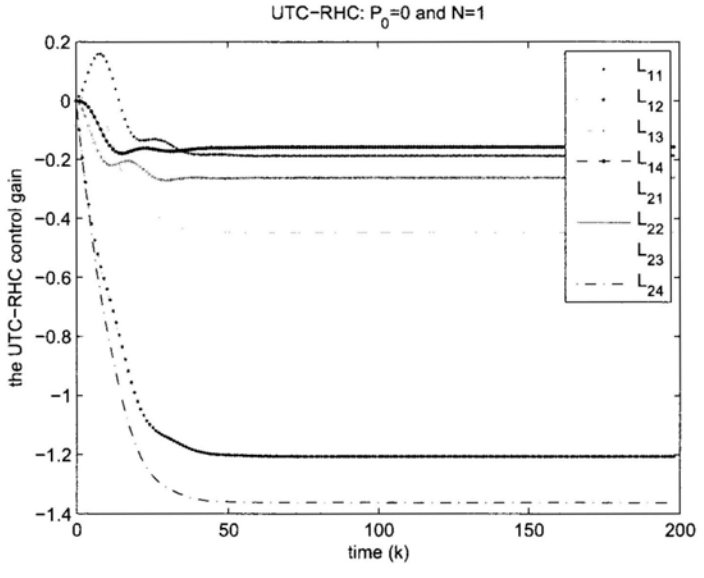

Figure 3.15: Control gain of UTC-RHC algorithm for N=1

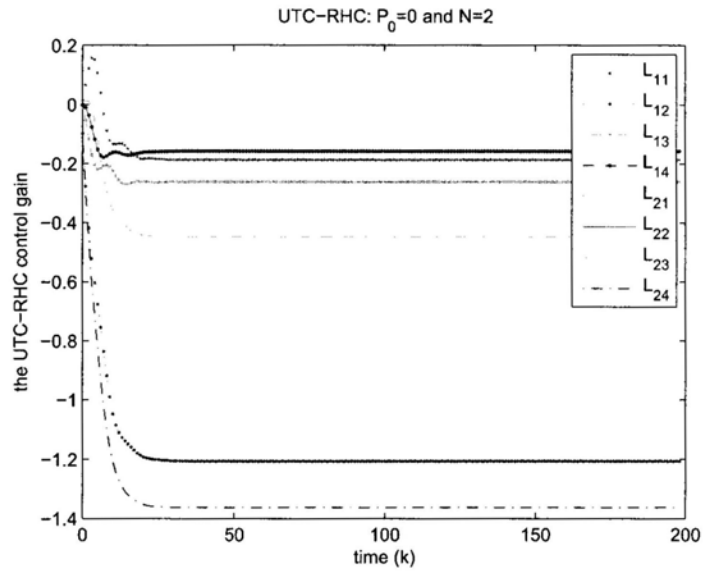

Figure 3.16: Control gain of UTC-RHC algorithm for N=2

3.5 Conclusion

In this chapter, we incorporate a learning feature in standard RHC, proposing an updated terminal cost RHC (UTC-RHC) algorithm under the framework of discrete linear time-invariant system. The closed-loop system under the UTC-RHC scheme is uniformly exponentially stable without imposing any constraints on terminal state, or terminal cost, or the horizon size. Thus the design of a stable RHC closed-loop system becomes more flexible. Moreover, the UTC-RHC control gain converges to the optimal value.

□ **End of chapter.**

Chapter 4

UTC-RHC for continuous time linear systems

In practical applications, most physical systems are inherently continuous time systems, for example, mechanical systems, electrical systems and thermodynamic systems. They are described by differential equations, instead of difference equations. Therefore, in this chapter, we extend the discrete time UTC-RHC algorithm, developed in Chapter 3, to continuous linear time invariant systems.

4.1 Introduction

In the framework of continuous-time (CT) systems, two categories of RHC schemes are investigated in the literature, namely instantaneous RHC (e.g., [8], [33], [50]) and sampled-data RHC (e.g., [11], [14], [15], [23]). The instantaneous RHC solves *instantaneously* an associated finite horizon optimal control problem at *any* time t and applies the *first* control continuously, thus yielding a state feedback control law. On the other hand, the sampled-data RHC only measures the state information at discrete sampling instants, and solves an open-loop optimal control problem with the measured state as the initial condition, and then repeatedly applies the *first portion* of the resulting control trajectory in the sampling intervals. We focus on the sampled-data RHC for CT systems, since sampled-data RHC is more amenable to practical applications where computation times are non-negligible and the states are sampled and not measured continuously.

As for the stability property, similar with the case for discrete time (DT) RHC, most existing continuous time RHC schemes also put constraints on either the terminal state, or the terminal cost or the horizon size. To name a few, some results require the terminal state to be zero [49] (which is often referred to as the zero terminal constraint); some require the terminal state to enter into a neighborhood of the origin [51]; some put constraints on the terminal cost [11][15], and usually the terminal cost is chosen to be a proper control Lyapunov function [25]; some researchers also point out that the stability can also be obtained given a enough long horizon, even without the use of terminal cost or terminal constraints [23].

To relax the various stability constraints proposed in the continuous RHC literature, we extend the discrete time version of UTC-RHC to continuous linear time invariant systems. Sampled-data UTC-RHC requires treatment of a certain hybrid CT/DT system as well as a consideration of the inter sampling behavior, as detailed in the proof of Proposition 4.1. This is inspired by [11] and [15]. By choosing a novel Lyapunov function, we show in Theorem 4.1 that exponential stability of the UTC-RHC closed-loop system is guaranteed for arbitrary horizon length T with $T > \delta > 0$ (δ is the constant inter-sampling time) and arbitrary terminal cost weighting matrix $P_0 \geq 0$. Moreover, the control gain converges to the optimal value corresponding to the infinite horizon optimal control problem.

The chapter is organized as follows. Section 4.2 reviews the algorithm of receding horizon control and this provides a conceptual and notational basis. Section 4.3 presents the sampled-data UTC-RHC algorithm and its stability and convergence results, and this is the main body of this chapter. Simulation results are given in Section 4.4 to show the properties of the algorithm. Section 4.5 ends this chapter with a conclusion.

4.2 Sampled-data receding horizon control

The principle of sampled-data receding horizon control (RHC) is presented in [14] and [15] in the framework of general nonlinear systems. In this section, we briefly revisit the sampled-data RHC in the framework of CT linear systems.

Consider the CT linear time-invariant (LTI) system

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{4.1}$$

where $x(t) \in \mathbb{R}^n$ is the state vector with the initial state $x(t_0) = x_0$, $u(t) \in \mathbb{R}^m$ is the control input, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are constant matrices. For convenience, we denote $x_t = x(t)$ and $u_t = u(t)$ in the sequel.

The following assumptions are supposed to hold throughout the chapter.

Assumption 4.1

A1) $Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{m\times m}$, $Q = Q^T \geq 0$, $R = R^T > 0$

A2) (A, B) is stabilizable

A3) $(A, \sqrt{Q})$ is detectable

∎

For sampled-data RHC schemes, the states are only measured at discrete sampling instants. At each sampling instant, an open-loop optimal control problem is solved with the current measured state as the initial condition, yielding an open-loop optimal control trajectory. The *first portion* of the optimal control

trajectory is applied to the system until the next sampling instant. In this chapter, the inter-sampling time δ is supposed to be a positive constant, which is frequently assumed in the literature.

Let $t_i = t_0 + i\delta$ $(i = 0, 1, 2, \cdots)$ be the sampling instants; x_{t_i} be the actual state measured at each sampling instant t_i; $(\bar{x}_t^*, \bar{u}_t^*)$ be the open-loop optimal trajectory pair; (x_t^*, u_t^*) be the closed-loop trajectory pair resulting from the sampled-data RHC algorithm. For convenience, we denote $r(x, u) = x^T Q x + u^T R u$ and $M(x) = x^T Q x$ in the sequel. The algorithm of sampled-data RHC in the framework of continuous LTI systems can be illustrated as follows.

Algorithm of sampled-data RHC

Step 1) At the current sampling instant t_i, measure the state x_{t_i}

Step 2) Solve the open-loop optimal control problem $\mathscr{P}(x_{t_i}, t_i, T, W)$:

$$\min_{\substack{\bar{u}_\tau \\ t_i \le \tau \le t_i+T}} \{ \int_{t_i}^{t_i+T} r(\bar{x}_\tau, \bar{u}_\tau) d\tau + W(\bar{x}_{t_i+T}) \} \tag{4.2}$$

subject to

$$\dot{\bar{x}}_t = A\bar{x}_t + B\bar{u}_t, \quad \bar{x}_{t_i} = x_{t_i}, \tag{4.3}$$

where T is the horizon length; $W(x) = x^T P_0 x$ is the terminal cost function with P_0 being the terminal cost weighting matrix; the bar as in $\bar{x}$ and $\bar{u}$ denotes predicted variables. The solution to the problem $\mathscr{P}(x_{t_i}, t_i, T, W)$ is denoted as $\bar{u}^*(t; x_{t_i}, t_i, T, W)$ and the open-loop optimal state trajectory $\bar{x}^*(t; x_{t_i}, t_i, T, W)$ is generated by the model (4.3) with $\bar{u}^*(t; x_{t_i}, t_i, T, W)$ being the control input. The open-loop optimal cost is

$$V_i^*(x_{t_i}) = V^*(x_{t_i}, t_i, T, W) = \int_{t_i}^{t_i+T} r(\bar{x}_\tau^*, \bar{u}_\tau^*) d\tau + W(\bar{x}_{t_i+T}^*). \tag{4.4}$$

Step 3) Apply $\bar{u}^*(t; x_{t_i}, t_i, T, W)$ to the plant (4.1) in the interval $t \in [t_i, t_{i+1})$ and drive the system to the next sampling instant t_{i+1}. Then go to Step 1 and repeat.

∎

By standard theory of linear quadratic regulator (LQR) [44], this control is given by

$$\bar{u}^*(t; x_{t_i}, t_i, T, W) = -R^{-1} B^T P(t_i + T - t) \bar{x}_t^*, \ \ t \in [t_i, t_i + T), \ \ \bar{x}_{t_i}^* = x_{t_i}, \tag{4.5}$$

where $P(t)$ is the solution of the Riccati differential equation (RDE) (4.6) with initial condition $P(0) = P_0 \ge 0$.

$$\dot{P}(t) = A^T P(t) + P(t) A + Q - P(t) B R^{-1} B^T P(t) \tag{4.6}$$

The open-loop state trajectory is generated by the following system

$$\dot{\bar{x}}_t^* = (A - B R^{-1} B^T P(t_i + T - t)) \bar{x}_t^*, \ \ t \in [t_i, t_i + T), \ \ \bar{x}_{t_i}^* = x_{t_i}. \tag{4.7}$$

For general cases, the sampled-data RHC closed-loop trajectory x_t^* does not necessarily coincide with the predicted open-loop optimal trajectory $\bar{x}_t^*$. But for nominal case (i.e., the plant model is exact and there is no disturbance) which is considered in this chapter, $x_t^* = \bar{x}_t^*$ during the sampling intervals $[t_i, t_{i+1})$ and $x_{t_i}^* = x_{t_i} = \bar{x}_{t_i}^*$. Therefore, the sampled-data RHC closed-loop system can also be written as

$$\dot{x}_t^* = (A - BR^{-1}B^T P(t_i + T - t))x_t^*, \quad t \in [t_i, t_{i+1}], \quad i = 0, 1, 2, \cdots . \tag{4.8}$$

A general framework to design a stabilizing sampled-data RHC is provided by [15], where the terminal cost function $W(x)$ is required to be a control Lyapunov function (CLF). Similar constraints on the terminal cost function are required by most of the DT RHC schemes and instantaneous RHC schemes (see [50] and the references therein). In the next section, we extend the DT UTC-RHC scheme to sampled-data continuous time systems and propose a sampled-date UTC-RHC (or UTC-RHC for short). This allows one to obtain uniform exponential stability without imposing constraints on the terminal state, or the terminal cost function, or the horizon length.

4.3 Sampled-data UTC-RHC

4.3.1 Algorithm

Different from that of the *standard* sampled-data RHC presented in Section 4.2 (here by *standard* we mean the terminal cost function W and the horizon length T are fixed), the corresponding open-loop optimal control problem solved by UTC-RHC scheme keeps the horizon length T fixed, but has its terminal cost function *updated at each sampling instant* by the open-loop optimal cost function computed at the last sampling instant.

More exactly, at the initial time t_0, the associated open-loop optimal control problem is $\mathscr{P}(x_0, t_0, T, W_0)$ with $W_0(x) = x^T P_0 x$, and the optimal cost is computed by

$$V_0^*(x_0) = \min_{\substack{\bar{u}_\tau \\ t_0 \le \tau \le t_0+T}} \{ \int_{t_0}^{t_0+T} r(\bar{x}_\tau, \bar{u}_\tau) d\tau + W_0(\bar{x}_{t_0+T}) \}. \tag{4.9}$$

At sampling instant t_i $(i \ge 1)$, we solve the corresponding open-loop optimal control problem $\mathscr{P}(x_{t_i}, t_i, T, W_i)$ with *modified terminal cost function* $W_i(x) = V_{i-1}^*(x)$, and compute the new optimal cost function $V_i^*(x)$ by

$$V_i^*(x_{t_i}) = \min_{\substack{\bar{u}_\tau \\ t_i \le \tau \le t_i+T}} \{ \int_{t_i}^{t_i+T} r(\bar{x}_\tau, \bar{u}_\tau) d\tau + W_i(\bar{x}_{t_i+T}) \}. \tag{4.10}$$

Comparing (4.10) to (4.4), it is seen that the terminal cost function is updated at each sampling instant.

Lemma 4.1

$$V^*(x_{t_i}, t_i, T, W_i) = V^*(x_{t_i}, t_i, (i+1)T, W_0) \tag{4.11}$$

where $W_i(x) = V_{i-1}^*(x)$. ∎

Proof:
Consider (4.9) and (4.10), by LQR theory, $W_i(x) = x^T P(iT)x$. By the principle of optimality [44], we have

$$\begin{aligned}
&V^*(x_{t_i}, t_i, T, W_i) \\
=&\min_{\substack{\bar{u}_\tau \\ t_i \le \tau \le t_i+T}} \{\int_{t_i}^{t_i+T} r(\bar{x}_\tau, \bar{u}_\tau)d\tau + \bar{x}_{t_i+T}^T P(iT)\bar{x}_{t_i+T}\} \\
=&\min_{\substack{\bar{u}_\tau \\ t_i \le \tau \le t_i+T}} \{\int_{t_i}^{t_i+T} r(\bar{x}_\tau, \bar{u}_\tau)d\tau \\
&+ \min_{\substack{\bar{u}_\tau \\ t_i+T \le \tau \le t_i+2T}} \{\int_{t_i+T}^{t_i+2T} r(\bar{x}_\tau, \bar{u}_\tau)d\tau + \bar{x}_{t_i+2T}^T P((i-1)T)\bar{x}_{t_i+2T}\}\} \\
=&\min_{\substack{\bar{u}_\tau \\ t_i \le \tau \le t_i+2T}} \{\int_{t_i}^{t_i+2T} r(\bar{x}_\tau, \bar{u}_\tau)d\tau + \bar{x}_{t_i+2T}^T P((i-1)T)\bar{x}_{t_i+2T}\} \\
=&V^*(x_{t_i}, t_i, 2T, W_{i-1}).
\end{aligned} \tag{4.12}$$

By induction, (4.11) is straightforward. □

Remark 4.1 Lemma 4.1 shows that the optimal control problem using the varying terminal cost function $W_i(x)$ and the fixed horizon T is equivalent to that using the fixed terminal cost function $W_0(x)$ and the lengthened horizon $(i+1)T$. In this sense, our algorithm effectively lengthens the horizon by T incrementally at each sampling instant, compared with the standard sampled-data RHC. As will be seen in Theorem 4.1, for arbitrary terminal cost weighting matrix $P_0 \ge 0$, our algorithm always generates a state trajectory that uniformly converges to the origin exponentially. ∎

Noting equation (4.11), the UTC-RHC control trajectory in the interval $t \in [t_i, t_{i+1})$ is given by

$$\begin{aligned}
&\bar{u}^*(t; x_{t_i}, t_i, T, W_i) = \bar{u}^*(t; x_{t_i}, t_i, (i+1)T, W_0) \\
=&-R^{-1}B^T P(t_i + (i+1)T - t)\bar{x}_t^*, \quad t \in [t_i, t_{i+1}), \quad i = 0, 1, 2, \cdots,
\end{aligned} \tag{4.13}$$

where $P(t_i + (i+1)T - t)$ is computed from RDE (4.6) with initial condition $P_0 \ge 0$.

Since we consider the nominal case, similar with the development in Section 4.2, we can write the UTC-RHC closed-loop system as

$$\dot{x}_t^* = (A - BR^{-1}B^T P(t_i + (i+1)T - t))x_t^* = L_t^{UR} x_t^*, \quad t \in [t_i, t_{i+1}], \ i = 0, 1, 2, \cdots, \tag{4.14}$$

while keeping in mind that x_t^* is generated by the real plant which is driven by the open-loop control trajectory (4.13).

4.3.2 Stability and convergence

In this section we demonstrate the stability and convergence properties of UTC-RHC. We follow two lines of thoughts. First, we show that UTC-RHC algorithm guarantees $x_t^* \to 0$ as $t \to \infty$ under

the standard assumption used for RHC in the literature, namely that the terminal cost function $W_0(x)$ satisfies a control Lyapunov function (CLF) condition as in [15], [33] and [50]. This first development leads to Proposition 4.1, which places UTC-RHC in the context of standard approaches to RHC. Next, it is shown in Theorem 4.1 that UTC-RHC in fact guarantees uniform exponential stability for any positive semidefinite terminal cost function W_0 and any horizon length $T \geq \delta$, and the UTC-RHC control actually converges to the infinite horizon optimal control law $u^*_\infty(t)$ which solves the infinite horizon optimal control problem $\mathscr{P}(x_t, t)$:

$$\min_{\substack{u_\tau \\ t \leq \tau < \infty}} \int_t^\infty r(x_\tau, u_\tau) d\tau \tag{4.15}$$

subject to system (4.1). It is well known that

$$u^*_\infty(t) = -R^{-1}B^T P_\infty x_t \triangleq L_\infty x_t, \tag{4.16}$$

where P_∞ is the maximal positive semidefinite solution of the associated algebraic Riccati equation (ARE)

$$0 = A^T P_\infty + P_\infty A + Q - P_\infty B R^{-1} B^T P_\infty, \tag{4.17}$$

and the optimal cost is $V^*_\infty(x) = x^T P_\infty x$.

Similar with "MPC value function" defined in [15], for $t \in [t_i, t_{i+1})$, we first define $V_{t_i}(t, x^*_t)$ as the value function for the optimal problem $\mathscr{P}(x^*_t, t, T - (t - t_i), W_i)$ or equivalently $\mathscr{P}(x^*_t, t, (i+1)T - (t - t_i), W_0)$. By the principle of optimality,

$$V_{t_i}(t, x^*_t) = V^*(x_{t_i}, t_i, (i+1)T, W_0) - \int_{t_i}^t r(x^*_\tau, u^*_\tau) d\tau, \tag{4.18}$$

where $u^*_\tau = \bar{u}^*(\tau; x_{t_i}, t_i, T, W_i)$ for $\tau \in [t_i, t]$. Then we define the "UTC-RHC value function" as $V^\delta(t, x^*_t) \triangleq V_\pi(t, x^*_t)$ where $\pi = \max_i\{t_i : t_i \leq t\}$.

The following sequence of Lemmas provide the machinery for Proposition 4.1.

Lemma 4.2 Pick an initial value $P_0 \geq 0$ such that

$$A^T P_0 + P_0 A + Q - P_0 B R^{-1} B^T P_0 \leq 0. \tag{4.19}$$

By applying $u_t = -R^{-1}B^T P_0 x_t$ to system (4.1) in the interval $[a, b]$ with $0 \leq a < b$,

$$\int_a^b r(x_t, u_t) dt + x_b^T P_0 x_b \leq x_a^T P_0 x_a \tag{4.20}$$

is satisfied. ∎

Proof:
With $K = -R^{-1}B^T P_0$, inequality (4.19) can be written as

$$(A + BK)^T P_0 + P_0(A + BK) \leq -(Q + K^T R K). \tag{4.21}$$

It follows that

$$\begin{aligned} & x_t^T[(A+BK)^T P_0 + P_0(A+BK)]x_t \\ \leq & -x_t^T(Q+K^T RK)x_t = -r(x_t,u_t), \ \ \forall\, x_t \in \mathbb{R}^n \text{ and } t \in [a,b], \end{aligned} \tag{4.22}$$

which implies

$$\frac{d(x_t^T P_0 x_t)}{dt} = \dot{x}_t^T P_0 x_t + x_t^T P_0 \dot{x}_t \leq -r(x_t,u_t). \tag{4.23}$$

Integrating (4.23) in the interval $[a,b]$ yields (4.20). □

Remark 4.2 Since $P(t)$ satisfies the RDE (4.6) and $P(0) = P_0$, inequality (4.19) is equivalent to $\dot{P}(0) \leq 0$. This ensures the monotonically non-increasing property of $P(t)$ (Theorem 10.11 in [8]), which is usually a key condition to ensure stability of instantaneous RHC [33]. Condition (4.19) is also the LTI equivalent of the conditions on terminal cost function required in [15] and [50]. ▮

Lemma 4.3 Pick $P(0) = P_0 \geq 0$ which satisfies the Riccati inequality (4.19), then

$$V_{t_{i+1}}(t_{i+1}, x^*_{t_{i+1}}) - V_{t_i}(t_i, x^*_{t_i}) \leq -\int_{t_i}^{t_{i+1}} M(x^*_\tau)d\tau, \ \ \forall i = 0,1,2,\cdots \tag{4.24}$$

where x_t^* denotes the UTC-RHC state trajectory. ▮

Proof:

$$V_{t_i}(t_i, x^*_{t_i}) = \int_{t_i}^{t_i+(i+1)T} r(\bar{x}^*_\tau, \bar{u}^*_\tau)d\tau + \bar{x}^{*T}_{t_i+(i+1)T} P_0 \bar{x}^*_{t_i+(i+1)T} \tag{4.25}$$

where $\bar{u}^*_\tau, \bar{x}^*_\tau$ is the open-loop trajectory pair solving the problem $\mathscr{P}(x^*_{t_i}, t_i, (i+1)T, W_0)$. Note $\bar{x}^*_{t_i} = x^*_{t_i}$ and the UTC-RHC state trajectory x_t^* coincides with $\bar{x}_t^*$ during the interval $[t_i, t_{i+1})$.

Extend $\bar{x}_t^*$, $\bar{u}_t^*$ to the interval $[t_i, t_{i+1} + (i+2)T]$ by concatenating to $\bar{u}_t^*$ the controller $\tilde{u}_t = K\tilde{x}_t = -R^{-1}B^T P_0 \tilde{x}_t$ for $t \in [t_i + (i+1)T, t_{i+1} + (i+2)T]$. Thus $\dot{\tilde{x}}_t = (A+BK)\tilde{x}_t$ for $t \in [t_i + (i+1)T, t_{i+1} + (i+2)T]$ and $\tilde{x}_{t_i+(i+1)T} = \bar{x}^*_{t_i+(i+1)T}$. Define

$$\begin{aligned} \tilde{V}_{t_{i+1}}(t_{i+1}, x^*_{t_{i+1}}) \triangleq & \int_{t_{i+1}}^{t_i+(i+1)T} r(\bar{x}^*_\tau, \bar{u}^*_\tau)d\tau + \int_{t_i+(i+1)T}^{t_{i+1}+(i+2)T} r(\tilde{x}_\tau, \tilde{u}_\tau)d\tau \\ & + \tilde{x}^T_{t_{i+1}+(i+2)T} P_0 \tilde{x}_{t_{i+1}+(i+2)T}. \end{aligned} \tag{4.26}$$

Then,

$$
\begin{aligned}
&V_{t_{i+1}}(t_{i+1}, x^*_{t_{i+1}}) - V_{t_i}(t_i, x^*_{t_i}) \\
\leq &\tilde{V}_{t_{i+1}}(t_{i+1}, x^*_{t_{i+1}}) - V_{t_i}(t_i, x^*_{t_i}) \\
= &\int_{t_{i+1}}^{t_i+(i+1)T} r(\bar{x}^*_\tau, \bar{u}^*_\tau)d\tau + \int_{t_i+(i+1)T}^{t_{i+1}+(i+2)T} r(\tilde{x}_\tau, \tilde{u}_\tau)d\tau + \tilde{x}^T_{t_{i+1}+(i+2)T} P_0 \tilde{x}_{t_{i+1}+(i+2)T} \\
&- \int_{t_i}^{t_{i+1}} r(x^*_\tau, u^*_\tau)d\tau - \int_{t_{i+1}}^{t_i+(i+1)T} r(\bar{x}^*_\tau, \bar{u}^*_\tau)d\tau - \bar{x}^{*T}_{t_i+(i+1)T} P_0 \bar{x}^*_{t_i+(i+1)T} \\
= &- \int_{t_i}^{t_{i+1}} r(x^*_\tau, u^*_\tau)d\tau + \{\tilde{x}^T_{t_{i+1}+(i+2)T} P_0 \tilde{x}_{t_{i+1}+(i+2)T} \\
&+ \int_{t_i+(i+1)T}^{t_{i+1}+(i+2)T} r(\tilde{x}_\tau, \tilde{u}_\tau)d\tau - \bar{x}^{*T}_{t_i+(i+1)T} P_0 \bar{x}^*_{t_i+(i+1)T}\} \\
\leq &- \int_{t_i}^{t_{i+1}} r(x^*_\tau, u^*_\tau)d\tau \\
\leq &- \int_{t_i}^{t_{i+1}} M(x^*_\tau)d\tau.
\end{aligned}
\tag{4.27}
$$

The first inequality in (4.27) is due to the optimality of $V_{t_{i+1}}(t_{i+1}, x^*_{t_{i+1}})$; the second inequality is derived straightforwardly from Lemma 4.2; the third inequality follows from $R > 0$. □

Lemma 4.4

$$
V^\delta(t, x^*_t) + \int_{t_0}^{t} M(x^*_\tau)d\tau \leq V^\delta(t_0, x_0), \quad \forall\, t \geq t_0 \tag{4.28}
$$

∎

Proof:
Lemma 4.3 implies

$$
V_{t_i}(t_i, x^*_{t_i}) - V_{t_0}(t_0, x_0) \leq - \int_{t_0}^{t_i} M(x^*_\tau)d\tau, \quad \forall\, t \geq t_0. \tag{4.29}
$$

Without loss of generality, we assume $t \in [t_i, t_{i+1})$, then

$$
\begin{aligned}
V^\delta(t, x^*_t) &= V_{t_i}(t_i, x^*_{t_i}) - \int_{t_i}^{t} r(x^*_\tau, u^*_\tau)d\tau \\
&\leq V_{t_i}(t_i, x^*_{t_i}) - \int_{t_i}^{t} M(x^*_\tau)d\tau \\
&\leq V_{t_0}(t_0, x_0) - \int_{t_0}^{t_i} M(x^*_\tau)d\tau - \int_{t_i}^{t} M(x^*_\tau)d\tau \\
&= V^\delta(t_0, x_0) - \int_{t_0}^{t} M(x^*_\tau)d\tau.
\end{aligned}
\tag{4.30}
$$

The first inequality in (4.30) follows from $R > 0$; the second inequality in (4.30) is obtained by considering (4.29). □

Based on these constructions, we are now ready to present the next result that shows UTC-RHC guarantees the convergence of the state under the same control Lyapunov function assumption used in standard RHC control. The proof of Proposition 4.1 is inspired by and adopts a technique similar to that used in [11] and [15], which is effective when the control trajectory is not continuous.

Proposition 4.1 Under assumptions A1-A3 (except for $Q > 0$), and suppose there exists an initial value $P_0 \geq 0$ with the property specified in inequality (4.19), then the state generated by the UTC-RHC closed-loop system (4.14) will converge to the origin, i.e., $x_t^* \to 0$ as $t \to \infty$. ∎

Proof:
Given an initial state x_0 at initial time t_0, it is clear that $0 \leq V^\delta(t_0, x_0) < \infty$. Since $V^\delta(t, x_t^*) \geq 0$ and $M(x) = x^T Q x$ is positive definite, Lemma 4.4 implies $0 \leq \int_{t_0}^{t} M(x_\tau^*) d\tau < \infty$ for any $t \geq t_0$. Since $\bar{x}_t^*$ is generated by solving $\mathscr{P}(x_{t_i}, t_i, (i+1)T, W_0)$, $\bar{x}^*(t)$ is continuous and bounded for $t \in [t_i, t_i + (i+1)T)$. Therefore the UTC-RHC state trajectory x_t^* is also continuous and bounded for all $t \geq t_0$, for x_t^* is concatenated by $\bar{x}_t^*$ for $t \in [t_i, t_{i+1})$ where $i = 0, 1, 2, \cdots$. This together with the boundedness of $\bar{u}_t^*$ for $t \in [t_i, t_i + (i+1)T)$ implies the boundedness of $\dot{\bar{x}}_t^*$ for $t \in [t_i, t_i + (i+1)T)$. Thus, $\dot{x}_t^*$ is bounded for all $t \geq t_0$, which means x_t^* is uniformly continuous for $t \geq t_0$. Considering the fact that $0 \leq \int_{t_0}^{t} M(x_\tau^*) d\tau < \infty$, the convergence of $x^*(t)$ follows straightforwardly from the Barbalet's lemma [28]. □

Remark 4.3 Condition (4.19) is a special case of condition (9) in [33], which is used to ensure monotonicity of $P(t)$, thence the stability of instantaneous RHC.

Similar stability conditions for RHC algorithms for nonlinear systems are given in [11] and [15]. Lemma 1 shows that by properly picking the initial condition P_0, the stability condition SC5 in [15] is automatically satisfied while the condition on δ (i.e., $0 < \delta \leq \varepsilon$) is relaxed to arbitrary $\delta > 0$ as long as we pick a long horizon T satisfying $\delta \leq T$. ∎

Proposition 4.1 is put here to relate our algorithm to the standard RHC. The next Theorem is our main result in this chapter. The proof technique is similar with that in [91]. It shows that UTC-RHC guarantees a stronger stability under milder conditions that possible for standard RHC methods, and our control gain will ultimately converge to the infinite horizon optimal control gain L_∞.

Theorem 4.1 Under assumptions A1-A3, for arbitrary $P_0 \geq 0$ and $T > 0$ with $T \geq \delta > 0$,

1. UTC-RHC closed-loop system (4.14) is uniformly exponentially stable in the sense that

$$\|x_t^*\| \leq \alpha e^{-\beta(t-t_0)} \|x_0\|, \quad t \geq t_0$$

   for some positive real constants α and β.

2. The UTC-RHC control gain converges to the control gain of the infinite horizon optimal control problem $\mathscr{P}(x_t, t)$, i.e., $L_t^{UR} \to L_\infty$ as $t \to \infty$.

∎

Proof:
1) Let

$$\begin{aligned} A_\infty =& A - BR^{-1}B^T P_\infty, \\ A_t =& BR^{-1}B^T P_\infty - BR^{-1}B^T P(t_i + (i+1)T - t). \end{aligned}$$

Then system (4.14) can be rewritten as

$$\dot{x}_t^* = (A_\infty + A_t)x_t^*, \quad x_{t_0}^* = x_0. \tag{4.31}$$

By LQR theory (Theorem 10.15 in [8]), A_∞ is stable, i.e., all the eigenvalues of A_∞ have negative real parts. Therefore, there exists a unique positive definite matrix P^* such that the Lyapunov equation

$$A_\infty^T P + P A_\infty = -I$$

is satisfied, where I is the identity matrix with appropriate dimensions. Then we have

$$\rho I = \lambda_{min}(P^*)I \leq P^* \leq \lambda_{max}(P^*)I = \eta I, \tag{4.32}$$

where ρ and η are positive real numbers.

Let $V(x) = x^T P^* x$ be a Lyapunov function candidate for the system (4.31). The derivative of $V(x)$ along the closed-loop trajectory x_t^* is

$$\begin{aligned} \dot{V}(x_t^*) &= \dot{x}_t^{*T} P^* x_t^* + x_t^{*T} P^* \dot{x}_t^* \\ &= (x_t^*)^T (A_\infty + A_t)^T P^* x_t^* + (x_t^*)^T P^* (A_\infty + A_t) x_t^* \\ &= x_t^{*T} (A_\infty^T P^* + A_t^T P^* + P^* A_\infty + P^* A_t) x_t^* \\ &= x_t^{*T} (-I + A_t^T P^* + P^* A_t) x_t^* \\ &= -(x_t^*)^T x_t^* + (x_t^*)^T (A_t^T P^* + P^* A_t) x_t^* \\ &\leq -\|x_t^*\|^2 + |(x_t^*)^T (A_t^T P^* + P^* A_t) x_t^*| \\ &\leq -\|x_t^*\|^2 + \|x_t^*\|^2 \cdot \|A_t^T P^* + P^* A_t\| \\ &\leq -\|x_t^*\|^2 (1 - \|A_t^T P^* + P^* A_t\|). \end{aligned} \tag{4.33}$$

Since $t \in [t_i, t_{i+1})$, $t_i = t_0 + i\delta$ and $0 < \delta \leq T$, it follows that

$$iT \leq t_i + (i+1)T - t \leq (i+1)T. \tag{4.34}$$

When $t \to \infty$, i also tends to ∞ (for $t_0 + i\delta \leq t \leq t_0 + (i+1)\delta$), hence $t_i + (i+1)T - t \to \infty$, considering (4.34).

By convergence property of the solution of RDE (4.6) (Theorem 10.10 in [8]), $P(t_i + (i+1)T - t) \to P_\infty$, thence $A_t \to 0$ as $t \to \infty$. This implies that $\|A_t^T P^* + P^* A_t\| \to 0$ as $t \to \infty$. Then, for any $0 < \varepsilon < 1$, there exists $T_0 > 0$ which only depends on A, B, Q, R, P_0 and ε, such that for all $t \geq T_0$,

$\left\|A_t^T P^* + P^* A_t\right\| < \varepsilon$ and $\dot{V}(x_t^*) \leq -(1-\varepsilon)\left\|x_t^*\right\|^2$, which means $\dot{V}(x_t^*)$ is negative definite when $t \geq T_0$. Using essentially the same proof as that of Theorem 7.4 in [70], it can be shown that

$$\left\|x_t^*\right\| \leq \sqrt{\frac{\eta}{\rho}}\, e^{-\frac{1-\varepsilon}{2\eta}(t-T_0)} \left\|x_{T_0}^*\right\|, \quad t \geq T_0$$

Let $\Phi(t, t_0)$ be the state transition matrix of system (4.14). Since (4.14) is linear, $\Phi(t, t_0)$ $(t_0 \leq t \leq T_0)$ is bounded. Denote $\Omega = \max_{t_0 \leq t \leq T_0} \Phi(t, t_0)$, then $\left\|x_t^*\right\| \leq \Omega \left\|x_0\right\|$ for $t_0 \leq t \leq T_0$. Since $\eta \geq \rho$, we have $\sqrt{\frac{\eta}{\rho}}\Omega \left\|x_0\right\| \geq max\{\sqrt{\frac{\eta}{\rho}} \left\|x_{T_0}^*\right\|, \Omega \left\|x_0\right\|\}$. It follows straightforwardly that

$$\left\|x_t^*\right\| \leq \alpha e^{-\beta(t-t_0)} \left\|x_0\right\|, \quad t \geq t_0$$

where $\alpha = \sqrt{\frac{\eta}{\rho}}\Omega e^{\frac{1-\varepsilon}{2\eta}(T_0 - t_0)} > 0$ and $\beta = \frac{1-\varepsilon}{2\eta} > 0$. Therefore, system (4.14) is uniformly exponentially stable.

2) As shown in the above proof, $P(t_i + (i+1)T - t) \to P_\infty$ as $t \to \infty$. Thus $L_t^{UR} \to L_\infty$ as $t \to \infty$. □

Remark 4.4 The closed-loop system (4.31) is a hybrid switched system. The Lyapunov function used in the proof is effective for this system, and it is not the standard Lyapunov function used in the CT RHC literature. ∎

4.4 Simulation results

In this section, we compare the performances of standard sampled-data RHC scheme with UTC-RHC scheme. Consider an open-loop unstable system

$$\dot{x} = \begin{bmatrix} 2 & 1 \\ -0.64 & -0.16 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{4.35}$$

Let $Q = I \in \mathbb{R}^{2\times 2}$ be the identity matrix, $R = 1$, the terminal cost weighting matrix $P_0 = 0$ and the initial state $x_0 = [-3 \;\; 5]^T$. In this case, the solution of ARE (4.17) is $P_\infty = \begin{bmatrix} 35.6959 & 11.3680 \\ 11.3680 & 4.7146 \end{bmatrix}$ and the infinite horizon optimal control gain is $L_\infty = [-11.3680 \;\; -4.7146]$.

We select $T = 0.5$ and $\delta = 0.3$ in this example. Using standard RHC control algorithm, the state trajectory $x^* = \begin{bmatrix} x_{11}^* & x_{21}^* \end{bmatrix}^T$, the control gain $L = \begin{bmatrix} L_{11} & L_{12} \end{bmatrix}$ and the control trajectory u^* are shown in Fig.4.1. Using the UTC-RHC control scheme, the state trajectory x^*, the control gain L and the control trajectory u^* are shown in Fig.4.2.

The figures show that under the same T and δ, the state of the standard RHC closed-loop system diverges while that of the UTC-RHC closed-loop system converges to the origin.

The figures also show that the RHC control gain repeats its value during each sampling interval, which does not converge to the optimal value L_∞ at all; while the UTC-RHC control gain gets improved with time and converges to the optimal value L_∞ after a few seconds.

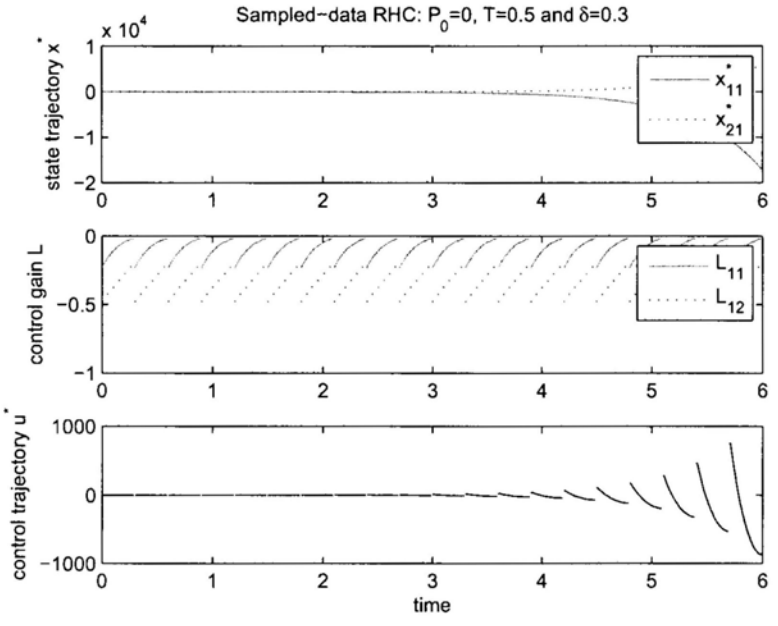

Figure 4.1: Profiles of x^*, L and u^* for the standard sampled-data RHC algorithm

Also, under UTC-RHC scheme, the control trajectory u gets smoother as time t increases, and the control trajectory under RHC scheme is discontinuous at each sampling instant unless the state converges to the origin.

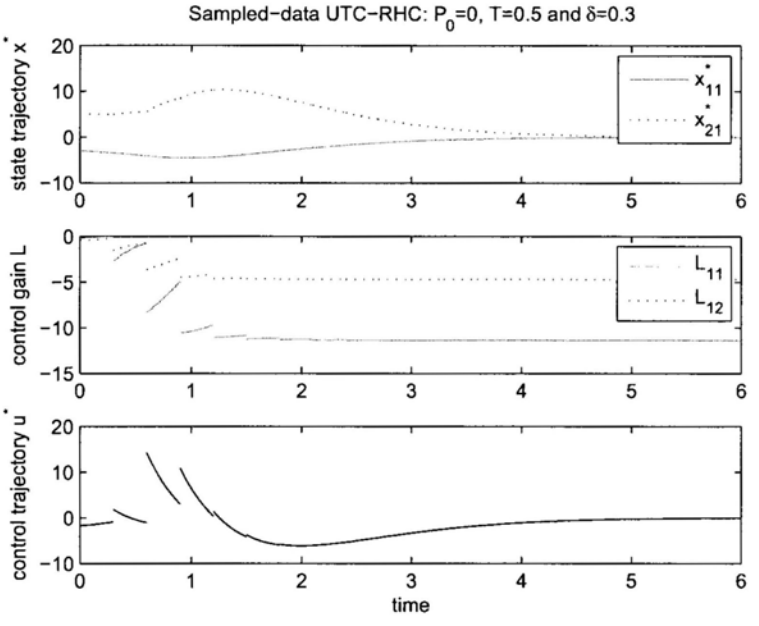

Figure 4.2: Profiles of x^*, L and u^* for the sampled-data UTC-RHC algorithm

4.5 Conclusion

In this chapter, we extend the discrete time UTC-RHC algorithm in Chapter 3 to continuous-time LTI systems under the sampled-data system framework. Parallel stability and convergence results are obtained.

□ **End of chapter.**

Chapter 5

Synchronization of networked nonlinear systems

5.1 Introduction

This chapter is devoted to the second topic of the dissertation, i.e. synchronization for higher-order systems or consensus of higher-order multi-agent systems with a leader.

Considerable effort has focused on two subjects of the networked systems, i.e. cooperative regulator problem and tracking problem. For cooperative regulator problem, controllers are designed to drive all the agents / nodes to the consensus equilibrium, which depends on the initial states values of those agents who have a directed path to all the other agents [67]. This is also known as leaderless consensus or synchronization problem in literature. As for the tracking problem, there is a leader node who only gives commands to a small portion of the other nodes. All the nodes are trying to tracking the trajectory generated by the leader node. It is called consensus with a leader, or synchronization to a leader, or pinning control in literature. Our research focuses on the tracking problem of higher-order nonlinear dynamics and is motivated by several points. First, most existing work on networked systems studies the first order or second order dynamics. Second, even for the first order or second order synchronization problems, dynamics are often chosen to be single integrators or double integrators (see [66] for a survey), while synchronization of multi-agent system with complicated nonlinear dynamics has not been fully investigated yet. Third, in literature, the dynamics of each agent of the multi-agent system is often assumed to be known exactly, but this is often not the case in practice. Many factors result in imprecision of the model, e.g. modeling the friction as a linear model for design purpose; system parameters drifting with time; and imperfect plant data. Finally, external disturbances are often neglected for the current research. However, disturbances exist almost in every practical application, such as white noise, gust to the aircraft.

Due to the universal approximation property of the neural network [21], it has been applied to control systems for more than two decades. Adaptive neural control of centralized (compared to networked sys-

tems) nonlinear systems has been well developed with Lyapunov based stability analysis. Nonlinearities of the system, either known or unknown, can be compensated by NNs and the NNs weights are tuned in an adaptive online fashion. Some representative works are, to name a few, [16], [40], [43], [53], [56], [57], [78], [96], etc.

Motivated by the above facts, we consider the higher-order synchronization problem of networked nonlinear systems with a time-varying active leader agent. Each agent is a higher-order system with a totally unknown nonlinear dynamics and an unknown external disturbance. The leader node is a higher-order non-autonomous nonlinear system, whose dynamics is unknown to all the other nodes. The leader node gives commands to only a small portion of the follower nodes. The communication graph studied in this chapter is a weighted digraph with fixed topology, which is general than undirected graphs. Using neural networks to approximate the unknown dynamics and a sliding mode control scheme to handle the higher-order model, we propose a robust adaptive controller for the multi-agent system. We derive distributed controllers and adaptive NN tuning algorithms to guarantee that all the nodes synchronize to the leader node. The NN is tuned on-line and no off-line tuning phase is needed. Moreover, the controller is totally distributed in the sense that each agent is controlled by its own controller, and these distributed controllers can only use their own information and information from their neighbors, which depends on the topology of the communication graph.

A notable recent work [22] also studies the consensus problem of multi-agent systems with unknown nonlinear dynamics and unknown external disturbances. Robust adaptive distributed NN controllers are proposed. The major distinctions between our work and their work are as follows. First, our communication graph is a directed graph, which is more general than the undirected graph studied by [22], since the communication between the nodes may not be mutual; Second, we study the tracking problem (or synchronization to a prescribed time-varying leader), while [22] studies the leaderless consensus or cooperative regulator problem, where the steady state consensus equilibrium depends on the initial conditions; Third, we deal with the higher-order dynamics with sliding mode control scheme, while [22] proves the case for the first order dynamics, and shows the method can be extended to higher-order systems using backstepping technique. As is well known, backstepping is a recursive design procedure and the complexity increases drastically with the order of the systems [78]. In this sense, our design is more elegant, especially when the order of the dynamics gets very high.

This chapter is organized as follows. First, background of graph theory is presented in Section 5.2; In Section 5.3, the higher-order consensus problem is formulated; Section 5.4 is the main part of this chapter, an robust adaptive synchronization controller for higher-order nonlinear systems is designed using Lyapunov technique, and rigorous proof is provided; Examples in Section 5.5 show the effectiveness of our algorithm; Section 5.6 ends this chapter with a conclusion.

5.2 Basic graph theory and notations

A graph is usually expressed by $\mathcal{G} = (V, E)$. V is a nonempty set of nodes/agents $V = \{v_1, v_2, \cdots, v_N\}$ and E is the set of edges/arcs $E \subseteq V \times V$, where $\times$ is the Cartesian product. $(v_i, v_j) \in E$ means there is an edge from node i to node j , i.e. node j can get information from node i, but not vice versa. An edge from node i to node j is represented by an arrow which starts from node i and ends at node j. If weights are associated with edges, then the graph is called a weighted graph. The topology of a weighted graph is often represented by the adjacency/connectivity matrix $A = [a_{ij}] \in \mathbb{R}^{N\times N}$ with a_{ij} the weights of edges, and $a_{ij} > 0$ if $(v_j, v_i) \in E$; otherwise $a_{ij} = 0$. Throughout this chapter, we assume there is no self loop, i.e. $a_{ii} = 0$. A digraph is a directed graph, whose adjacency matrix A is non-symmetric; while a symmetric adjacency matrix A implies an undirected graph. Define a nonnegative matrix as a matrix whose entries are all nonnegative. Then the adjacency matrix A is a nonnegative matrix. (One should distinguish nonnegative matrix from nonnegative definite matrix. The latter is a matrix with all its eigenvalues nonnegative). The i-th row sum of A , i.e. $d_i \triangleq \sum_{j=1}^{N} a_{ij}$ is the weighted in-degree of node i. Define the in-degree matrix as $D = diag\{d_i\} \in \mathbb{R}^{N\times N}$. The graph Laplacian matrix is $L = D - A$. Clearly, the row sum of matrix L is zero. Let $\underline{1} = [1, 1, \cdots, 1]$ be a vector of elements 1 with appropriate dimension. Then $L\underline{1} = 0$.

The set of neighbors of node i is denoted as $N_i = \{v_j | (v_j, v_i) \in E\}$. If node j is a neighbor of node i, then node i can get information from node j, not necessarily vice versa for directed graph. For undirected graph, neighborhood is a mutual relation.

For directed graph, a direct path from node i to node j is a sequence of successive edges in the form $\{(v_i, v_k), (v_k, v_l), \cdots, (v_m, v_j)\}$. A directed graph is said to be strongly connected, if for any pair of nodes (v_i, v_j) with $i \neq j$, there is a direct path from node i to node j. The following definition, fact and lemmas are standard in graph theory.

Definition 5.1 [62] A matrix $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ is said to be *reducible* if its indices can be divided into two disjoint nonempty sets $\{i_1, i_2, \cdots, i_p\}$ and $\{j_1, j_2, \cdots, j_q\}$ with $p + q = n$, such that $a_{i_\alpha i_\beta} = 0$ for $\alpha = 1, 2, \cdots, p$ and $\beta = 1, 2, \cdots, q$. Matrix A is called *irreducible* if it is not reducible. ∎

Another frequently used definition of irreducible is:

Definition 5.2 A matrix $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ is irreducible if it is not cogredient to a lower triangular matrix, i.e., there is no permutation matrix U such that

$$A = U \begin{bmatrix} * & 0 \\ * & * \end{bmatrix} U^T$$

∎

Lemma 5.1 [88] A graph $\mathcal{G}$ is strongly connected if and only if its adjacency matrix A (or Laplacian matrix L) is irreducible. ∎

Definition 5.3 (diagonally dominant)([62][77]) A matrix $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ is said to be *diagonally dominant*, if $a_{ii} \geq \sum_{j\neq i} |a_{ij}|$ for all $i = 1, 2, \cdots, n$. It is said to be *strictly diagonally dominant*, if the above inequality holds strictly for all $i = 1, 2, \cdots, n$. A is *irreducibly diagonally dominant* if A is irreducible and diagonally dominant, with strict inequality holding for at least one i. ∎

Lemma 5.2 [74] If A is irreducibly diagonally dominant, then A is nonsingular. ∎

Lemma 5.3 [62] Let the graph be strongly connected and at least one node can get information from the leader node, i.e. $b_i > 0$ for at least one i. Then $L + B$ is nonsingular. Moreover, define

$$q = [q_1, q_2, \cdots, q_N]^T = (L + B)^{-1}\underline{1}, \tag{5.1}$$

$$P = diag\{p_i\} = diag\{1/q_i\}, \tag{5.2}$$

$$Q = P(L + B) + (L + B)^T P, \tag{5.3}$$

then $P > 0$ and $Q > 0$. ∎

5.3 Higher-order synchronization: problem formulation

Consider a group of N ($N \geq 1$) agents with non-identical dynamics. The dynamics of the i-th ($i = 1, 2, \cdots, N$) is described by the following system of nonlinear differential equations,

$$\begin{aligned} \dot{x}_i^1 &= x_i^2 \\ \dot{x}_i^2 &= x_i^3 \\ &\vdots \\ \dot{x}_i^{M-1} &= x_i^M \\ \dot{x}_i^M &= f_i(x_i) + u_i + \zeta_i, \end{aligned} \tag{5.4}$$

where $x_i^m(t) \in \mathbb{R}$ ($m = 1, 2, \cdots, M$) is the m-th state of node i; $x_i = [x_i^1, x_i^2, \cdots, x_i^M]^T \in \mathbb{R}^M$ is the state vector of the i-th node; $f_i(x_i)$ is locally Lipschitz and it is assumed to be unknown; $u_i(t) \in \mathbb{R}$ is the control input/protocol; $\zeta_i(t) \in \mathbb{R}$ is an external disturbance, which is also unknown, but assumed to be bounded. Define $x^m = [x_1^m, x_2^m, \cdots, x_N^m]^T \in \mathbb{R}^N$ ($m = 1, 2, \cdots, M$) as the m-th global states vector. Specifically, x^1, x^2 and x^3 are the global position vector, global velocity vector and global acceleration vector, respectively. Define $f(x) = [f_1(x_1), f_2(x_2), \cdots, f_N(x_N)]^T \in \mathbb{R}^N$, $u = [u_1, u_2, \cdots, u_N]^T \in \mathbb{R}^N$ and $\zeta = [\zeta_1, \zeta_2, \cdots, \zeta_N]^T \in \mathbb{R}^N$. Then one has the global dynamics of the group of agents,

$$\begin{aligned} \dot{x}^1 &= x^2 \\ \dot{x}^2 &= x^3 \\ &\vdots \\ \dot{x}^{M-1} &= x^M \\ \dot{x}^M &= f(x) + u + \zeta. \end{aligned} \tag{5.5}$$

The dynamics of the leader/control node is given by

$$\begin{aligned} \dot{x}_0^1 &= x_0^2 \\ \dot{x}_0^2 &= x_0^3 \\ &\vdots \\ \dot{x}_0^{M-1} &= x_0^M \\ \dot{x}_0^M &= f_0(x_0, t), \end{aligned} \tag{5.6}$$

where $f_0(x_0, t)$ is piecewise continuous in t and locally Lipschitz in x_0 for all $t \geq 0$ and all $x_0 \in \mathbb{R}^M$, and it is unknown to all the nodes in graph $\mathcal{G}$; $x_0 = [x_0^1, x_0^2, \cdots, x_0^M]^T \in \mathbb{R}^M$ is the state vector of the leader node. Denote $\underline{x_0^m} = x_0^m \underline{1} = [x_0^m, x_0^m, \cdots, x_0^m]^T \in \mathbb{R}^N$ and $\underline{f_0} = f_0 \underline{1} = [f_0(x_0, t), f_0(x_0, t), \cdots, f_0(x_0, t)]^T \in \mathbb{R}^N$. It is assumed that the leader node can give commands to at least one node i ($i = 1, 2, \cdots, N$). Let b_i be the weight of the edge from the leader node to node i, then $b_i \geq 0$ for all i, and $b_i > 0$ for at least one i. When $b_i > 0$, the leader node 0 is considered as a neighbor of node i and the node i is said to be a controlled node. The control node dynamics (5.6) can be considered as an exosystem that generates a desired command trajectory. The problem confronted in this chapter is the following.

Definition 5.4 (Higher-order synchronization problem)
Design control protocols u_i for all the nodes in graph $\mathcal{G}$, such that $x_i^m \to x_0^m$, $\forall i = 1, 2, \cdots, N$ and $m = 1, 2, \cdots, M$. ∎

Define the m-th disagreement vector as $\delta^m = x^m - \underline{x_0^m}$, then all nodes are said to synchronize to the leader node if $\lim_{t\to\infty} \delta^m = 0$ for all $m = 1, 2, \cdots, M$.

Definition 5.5 [29] (Neighborhood synchronization error)
Consider the leader node 0, the neighborhood synchronization error for i-th node is defined as:

$$\begin{aligned} e_i^1 &= \sum_{j\in N_i} a_{ij}(x_j^1 - x_i^1) + b_i(x_0^1 - x_i^1) \\ e_i^2 &= \sum_{j\in N_i} a_{ij}(x_j^2 - x_i^2) + b_i(x_0^2 - x_i^2) \\ &\vdots \\ e_i^M &= \sum_{j\in N_i} a_{ij}(x_j^M - x_i^M) + b_i(x_0^M - x_i^M). \end{aligned} \tag{5.7}$$

∎

Note that although (5.7) looks similar with the state update law in [24], it is not the state law, but synchronization error. Note, defining the local synchronization error in this form is a key to analyzing the directed graph.

Define $e^m = [e_1^m, e_2^m, \cdots, e_N^m]^T$. Specifically, e^1, e^2 and e^3 can be regarded as the global position error vector, global velocity error vector and global acceleration error vector, respectively. Considering $L\underline{1} = 0$, a straightforward computation gives $e^m = -(L+B)(x^m - \underline{x}_0^m)$. Derivation is shown below

$$
\begin{aligned}
e^m &= \begin{bmatrix} e_1^m \\ e_2^m \\ \vdots \\ e_N^m \end{bmatrix} = \begin{bmatrix} \sum_{j\in N_1} a_{1j}(x_j^m - x_1^m) + b_1(x_0^m - x_1^m) \\ \sum_{j\in N_2} a_{2j}(x_j^m - x_2^m) + b_2(x_0^m - x_2^m) \\ \vdots \\ \sum_{j\in N_N} a_{Nj}(x_j^m - x_N^m) + b_N(x_0^m - x_N^m) \end{bmatrix} \\
&= \begin{bmatrix} \sum_{j\in N_1} a_{1j}x_j^m \\ \sum_{j\in N_2} a_{2j}x_j^m \\ \vdots \\ \sum_{j\in N_N} a_{Nj}x_j^m \end{bmatrix} - \begin{bmatrix} \sum_{j\in N_1} a_{1j}x_1^m \\ \sum_{j\in N_2} a_{2j}x_1^m \\ \vdots \\ \sum_{j\in N_N} a_{Nj}x_1^m \end{bmatrix} + \begin{bmatrix} b_1 & 0 & 0 & 0 \\ 0 & b_2 & 0 & 0 \\ 0 & 0 & b_3 & 0 \\ 0 & 0 & 0 & b_4 \end{bmatrix} (\underline{x}_0^m - x^m) \\
&= \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1^m \\ x_2^m \\ \vdots \\ x_N^m \end{bmatrix} - \begin{bmatrix} \sum_{j\in N_1} a_{1j} & 0 & \cdots & 0 \\ 0 & \sum_{j\in N_2} a_{2j} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{j\in N_N} a_{Nj} \end{bmatrix} \begin{bmatrix} x_1^m \\ x_2^m \\ \vdots \\ x_N^m \end{bmatrix} \\
&\quad + B(\underline{x}_0^m - x^m) \\
&= Ax^m - Dx^m + B(\underline{x}_0^m - x^m) = -Lx^m + B(\underline{x}_0^m - x^m) + L\underline{x}_0^m \\
&= -(L+B)x^m + (L+B)\underline{x}_0^m = -(L+B)(x^m - \underline{x}_0^m)
\end{aligned}
$$

Then we have the global error dynamics in a vector form as

$$
\begin{aligned}
\dot{e}^1 &= e^2 \\
\dot{e}^2 &= e^3 \\
&\vdots \\
\dot{e}^M &= -(L+B)(\dot{x}^M - \underline{\dot{x}_0}^M) = -(L+B)(f(x) + u + \zeta - \underline{f_0}),
\end{aligned} \tag{5.8}
$$

where $B = diag\{b_j\} \in \mathbb{R}^{N\times N}$.

Remark 5.1 In this chapter, we do not use disagreement vector δ^m for the distributed controller design, because it is global information that can not be accessed locally at each node, in contrast to the neighborhood synchronization error in (5.7). This will be clearly shown in the proof of Theorem 5.1. ∎

Lemma 5.4 Let the graph $\mathcal{G}$ be strongly connected and $B \neq 0$. Then

$$
\|\delta^m\| \leq \|e^m\| / \underline{\sigma}(L+B), m = 1, 2, \cdots, M, \tag{5.9}
$$

where $\underline{\sigma}(L+B)$ is the minimum singular value of matrix $(L+B)$. ∎

Proof.
Since the graph $\mathcal{G}$ is strongly connected and $B \neq 0$, $L+B$ is nonsingular. Noticing $e^m = -(L+B)\delta^m$, it follows that $\delta^m = -(L+B)^{-1}e^m$, therefore $\|\delta^m\| = \left\|(L+B)^{-1}e^m\right\| \leq \bar{\sigma}((L+B)^{-1})\|e^m\| = \|e^m\|/\underline{\sigma}(L+B)$. □

Remark 5.2 Lemma 5.4 implies that as long as $\|e^m\|$ is bounded, $\|\delta^m\|$ is bounded. Moreover, $e^m \to 0$ implies $\delta^m \to 0$, i.e. $x_i^m \to x_0^m$, and consensus to the leader is reached. ∎

5.4 Robust adaptive synchronization: Lyapunov design

In this section, using the control Lyapunov function technique, we show how to design the distributed controllers for each node, such that the higher-order synchronization problem is solved.

5.4.1 Sliding mode error

To deal with the higher-order synchronization problem, we introduce the sliding mode error r_i for each node i,

$$r_i = \lambda_1 e_i^1 + \lambda_2 e_i^2 + \cdots + \lambda_{M-1} e_i^{M-1} + e_i^M, i = 1, 2, \cdots, N. \tag{5.10}$$

The design parameter λ_i is chosen such that the polynomial $s^{M-1} + \lambda_{M-1}s^{M-2} + \cdots + \lambda_1$ is Hurwitz. Then on the sliding surface $r_i = 0$, $e_i \to 0$ exponentially. To ease the design and analysis, we obtain such λ_i by writing $s^{M-1} + \lambda_{M-1}s^{M-2} + \cdots + \lambda_1 = (s-\alpha_1)(s-\alpha_2)\cdots(s-\alpha_{M-1})$ and choosing positive real numbers α_i. Define the global sliding mode error as $r = [r_1, r_2, \cdots, r_N]^T$, then $r = \lambda_1 e^1 + \lambda_2 e^2 + \cdots + \lambda_{M-1}e^{M-1} + e^M$.

Define $E^1 = [e^1, e^2, \cdots, e^{M-1}] \in \mathbb{R}^{N\times(M-1)}$, $E^2 = [e^2, e^3, \cdots, e^M] \in \mathbb{R}^{N\times(M-1)}$,

$l = [0, 0, \cdots, 0, 1]^T \in \mathbb{R}^{M-1}$ and $\Lambda = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -\lambda_1 & -\lambda_2 & -\lambda_3 & \cdots & -\lambda_{M-1} \end{bmatrix} \in \mathbb{R}^{(M-1)\times(M-1)}$.

Then we have

$$E^2 = E^1\Lambda^T + rl^T \tag{5.11}$$

Since matrix Λ is Hurwitz, given any positive real number β, there exists a positive definite matrix P_1, such that the following Lyapunov equation holds,

$$\Lambda^T P_1 + P_1\Lambda = -\beta I, \tag{5.12}$$

where $I \in \mathbb{R}^{(M-1)\times(M-1)}$ is an identity matrix.

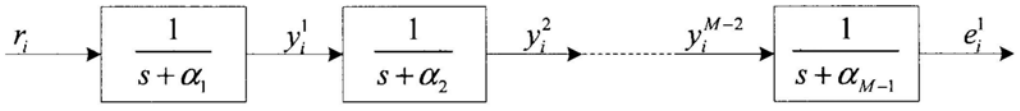

Figure 5.1: Block diagram showing relation between r_i and e_i^1

The dynamics of the sliding mode error r is

$$\begin{aligned}\dot{r} &= \lambda_1 \dot{e}^1 + \lambda_2 \dot{e}^2 + \cdots + \lambda_{M-1}\dot{e}^{M-1} + \dot{e}^M \\ &= \lambda_1 e^2 + \lambda_2 e^3 + \cdots + \lambda_{M-1} e^M - (L+B)(f(x) + u + \zeta - \underline{f_0}) \\ &= \rho - (L+B)(f(x) + u + \zeta - \underline{f_0}),\end{aligned} \tag{5.13}$$

where

$$\rho = \lambda_1 e^2 + \lambda_2 e^3 + \cdots + \lambda_{M-1} e^M = E^2 \bar{\lambda} \tag{5.14}$$

with $\bar{\lambda} = [\lambda_1, \lambda_2, \cdots, \lambda_{M-1}]^T$.

Next lemma shows that if r is bounded, then e^m is bounded for every $m = 1, 2, \cdots, M$.

Lemma 5.5 If r is bounded by ψ, i.e., $\forall t > 0$, $\|r\| \leq \psi$, or in other words, each r_i is bounded by χ_i with $\psi = \sqrt{\chi_1^2 + \chi_2^2 + \cdots + \chi_N^2}$, then each e^m $(m = 1, 2, \cdots, M)$ is ultimately bounded by $\frac{2^{m-1}\psi}{\Pi_{j=0}^{M-m}\alpha_j}$ with $\alpha_0 = 1$. ∎

Proof. The proof adopts essentially the same technique used in Section 7.1 in [71]. To be self contained, details are shown as follows.

First, we derive the bound for e_i^1. For i-th node, we have

$$\begin{aligned}r_i &= \lambda_1 e_i^1 + \lambda_2 e_i^2 + \cdots + \lambda_{M-1} e_i^{M-1} + e_i^M \\ &= \lambda_1 e_i^1 + \lambda_2 \dot{e}_i^1 + \cdots + \lambda_{M-1} e_i^{1(M-2)} + e_i^{1(M-1)},\end{aligned} \tag{5.15}$$

where $e_i^{1(m)}$ denote the m-th derivative of e_i^1. Consider the zero initial condition (i.e. $e_i = 0$), applying the Laplace transform to (5.15) gives

$$r_i(s) = (\lambda_1 + \lambda_2 s + \cdots + \lambda_{M-1} s^{M-2} + s^{M-1}) e_i^1(s), \tag{5.16}$$

where s is the Laplace operator. (5.16) can be written as

$$e_i^1(s) = \frac{r_i(s)}{\lambda_1 + \lambda_2 s + \cdots + \lambda_{M-1} s^{M-2} + s^{M-1}} = \frac{r_i(s)}{(s+\alpha_1)(s+\alpha_2)\cdots(s+\alpha_{M-1})}, \tag{5.17}$$

which means the local position error e_i^1 is obtained from r_i through a series of low-pass filters, see Fig.5.1.

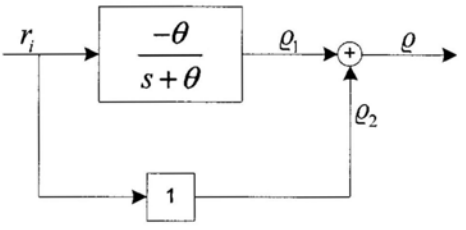

Figure 5.2: Block diagram showing relation between r_i and ϱ

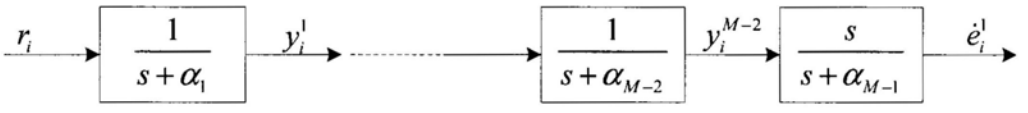

Figure 5.3: Block diagram showing relation between r_i and $\dot{e}_i^1$

Let y_i^1 be the output of the first filter, then

$$y_i^1 = \int_0^t e^{-\alpha_1(t-\tau)} r_i(\tau) d\tau. \tag{5.18}$$

Then

$$|y_i^1| \leq \chi_i \int_0^t e^{-\alpha_1(t-\tau)} d\tau = \frac{\chi_i}{\alpha_i}(1 - e^{-\alpha_1 t}) \leq \frac{\chi_i}{\alpha_i} \tag{5.19}$$

Similarly, applying the same reasoning all the way to e_i^1, we have $|e_i^1| \leq \frac{\chi_i}{\alpha_1 \alpha_2 \cdots \alpha_{M-1}}$.

Next, we derive the bound for e_i^m when $m > 1$. Before proceeding, we need to show a fact that if $\varrho(s) = \frac{s}{s+\theta} r_i(s) = \left(1 - \frac{\theta}{s+\theta}\right) r_i(s)$, then $|\varrho(t)| \leq 2\chi_i$.

As show by Figure 5.2, $\varrho = \varrho_1 + \varrho_2$. By previous derivation, one has $\varrho_1 = \int_0^t e^{-\theta(t-\tau)}(-\theta r_i(\tau)) d\tau$. Then $|\varrho_1| \leq \theta \chi_i \int_0^t e^{-\theta(t-\tau)} d\tau = \chi_i(1 - e^{-\theta t}) \leq \chi_i$. Thus $|\varrho| = |\varrho_1 + \varrho_2| \leq |\varrho_1| + |\varrho_2| \leq 2\chi_i$.

Note that $e_i^2 = \dot{e}_i^1$ can be expressed by Figure 5.3. Then $|y_i^{M-2}| \leq \frac{\chi_i}{\alpha_1 \alpha_2 \cdots \alpha_{M-2}}$ and thus $|e_i^2| = |\dot{e}_i^1| \leq \frac{2\chi_i}{\alpha_1 \alpha_2 \cdots \alpha_{M-2}}$. By induction, $|e_i^m| = |\dot{e}_i^{1(m-1)}| \leq \frac{2^{m-1}\chi_i}{\Pi_{j=0}^{M-m} \alpha_j}$, where $\alpha_0 = 1$.

The bound of global error e^m is given by

$$\|e^m\| = \sqrt{|e_1^m|^2 + |e_2^m|^2 + \cdots + |e_N^m|^2} \leq \left(\frac{2^{m-1}}{\Pi_{j=0}^{M-m} \alpha_j}\right) \sqrt{\sum_{j=1}^{N} (\chi_i)^2} = \frac{2^{m-1}\psi}{\Pi_{j=0}^{M-m} \alpha_j}.$$

Note that for the nonzero initial condition cases, the bounds can still be obtained, but asymptotically, because the polynomial $\lambda_1 + \lambda_2 s + \cdots + \lambda_{M-1} s^{M-2} + s^{M-1}$ is Hurwitz. □

Remark 5.3 Lemma 5.5 implies that for a given bound of r, larger poles α_j can make a smaller bound for e^m. ∎

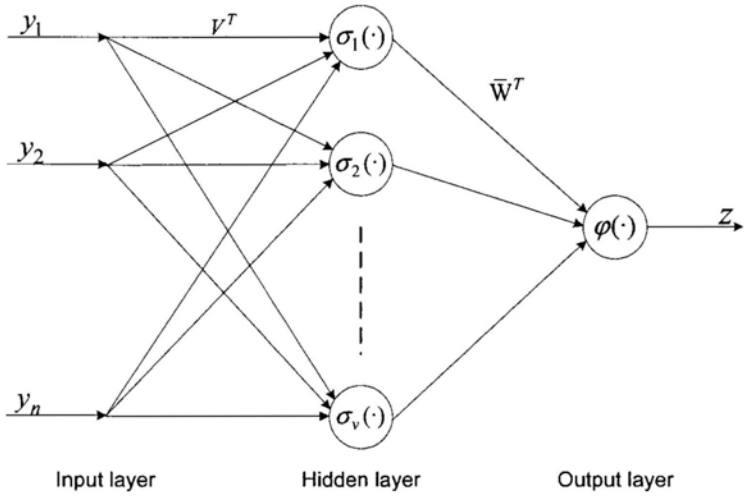

Figure 5.4: A two layer neural network

5.4.2 Linear in parameter neural network

Function approximation property makes neural network a useful tool for solving the control problem [41]. It is well known that two layer neural network is a universal approximator [21]. A two layer neural network with v neurons in the hidden layer is depicted in Fig.5.4 with the mathematic expression be

$$z = \varphi(\bar{W}^T \sigma(V^T y)), \tag{5.20}$$

where $y = [y_1, y_2, \cdots, y_n]^T$ is the n inputs vector, $\sigma = [\sigma_1, \sigma_2, \cdots, \sigma_v]^T$ is the activation function vector for the hidden layer; $V \in \mathbb{R}^{n \times v}$ is the input layer weights matrix; $\bar{W} \in \mathbb{R}^v$ is the output layer weights vector; φ is the activation function for the output layer. According to the NN approximation literature [21], for different applications, a variety of NN activation functions can be selected such as sigmoids, Gaussians/radial basis function (RBF), etc. A list of frequently used activation functions can be found in Fig. 1.1.3 in [41].

To avoid the distraction from the main issues being introduced, in this chapter, we assume a linear-in-parameter (LIP) NN, i.e. the hidden layer activation functions $\sigma_i(\cdot)$ and the input layer weights matrix V are fixed, the output layer activation function is simply a summation function, only the output weights $\bar{W}$ are tuned. The following development can be extended to a two-layer NN as in [41].

Assume the unknown nonlinearities $f_i(x_i)$ in (5.4) can be expressed on a compact set $\Omega \subset \mathbb{R}^M$ by

$$f_i(x_i) = W_i^T \phi_i(x_i) + \epsilon_i \tag{5.21}$$

where $\phi_i(x_i) = \sigma_i(V^T x_i) \in \mathbb{R}^{v_i}$ with $\sigma_i = [\sigma_{i1}, \sigma_{i2}, \cdots, \sigma_{iv_i}]^T \in \mathbb{R}^{v_i}$ be a suitable set of v_i activation functions; $W_i \in \mathbb{R}^{v_i}$ is the idea neural network (NN) output weights vector; and ϵ_i is the approximation error.

Remark 5.4 By Stone-Weierstrass approximation theorem [72], in the compact set $\Omega \subset \mathbb{R}^M$, $\forall \varepsilon_i > 0$, there exist a large enough positive integer v_i^*, ideal weights W_i and suitable basis set $\phi_i(\cdot)$ such that for $v_i \geq v_i^*$, $max_{x_i \in \Omega} \|\epsilon_i\| \leq \varepsilon_i$. Then there exist positive numbers $W_{iM} > 0$ and $\phi_{iM} > 0$, such that $\|W_i\| \leq W_{iM}$ and $max_{x_i \in \Omega} \|\phi_i\| \leq \phi_{iM}$. As is often done in literature, this prescribed compact set Ω is assumed to be as large as necessary. ∎

We also assume the ideal weights vector W_i in (5.21) to be unknown. Each node maintains a neural network locally to keep track of the current estimates for its unknown nonlinearities. Since the NN are maintained locally at each node, they need only to approximate the local nonlinearities in the dynamics of that node. Therefore, the number of neurons needed at each node is reduced compared to centralized neural adaptive control.

Define the approximation of the nonlinearity $f_i(x_i)$ as

$$\hat{f}_i(x_i) = \hat{W}_i^T \phi_i(x_i), \tag{5.22}$$

where $\hat{W}_i \in \mathbb{R}^{v_i}$ is the current estimate of the NN weights for i-th node. $\hat{f}_i(x_i)$ is the actual output of the LIP NN. The NN tuning law is shown in Theorem 5.1 using only the local information available to node i.

Denote

$$W = \begin{bmatrix} W_1 & 0 & \cdots & 0 \\ 0 & W_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_N \end{bmatrix}, \quad \hat{W} = \begin{bmatrix} \hat{W}_1 & 0 & \cdots & 0 \\ 0 & \hat{W}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \hat{W}_N \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_N \end{bmatrix}, \quad \phi(x) = \begin{bmatrix} \phi_1(x_1) \\ \phi_2(x_2) \\ \vdots \\ \phi_N(x_N) \end{bmatrix},$$

then the global nonlinearity $f(x)$ can be written as

$$f(x) = W^T \phi(x) + \epsilon, \tag{5.23}$$

and the approximation is

$$\hat{f}(x) = \hat{W}^T \phi(x). \tag{5.24}$$

The error of the NN weights is defined by

$$\tilde{W} = W - \hat{W}. \tag{5.25}$$

It is important to note that matrices W, $\hat{W}$ and $\tilde{W}$ are all block diagonal, a fact that is used in the proof of Theorem 5.1.

Remark 5.5 By Remark 5.4 and the definitions of W, ϕ and ϵ, there exist positive numbers W_M, Φ_M and ϵ_M, such that $\|W\|_F \leq W_M$, $\|\phi\| \leq \Phi_M$ and $\|\epsilon\| \leq \epsilon_M$. Later in the proof of Theorem 5.1, we will see that bounds W_M and ϵ_M are actually not used in the controller design, thus they do not have to be known. They only appear in the ultimate error bounds in the proof of Theorem 5.1 and so affect the

performance of the controller. The bound ϕ_M is needed to choose the design parameter c as in (5.31). It can be expressed explicitly if we choose the squashing functions as the activation functions, such as the sigmoids, Gaussians, hyperbolic tangents. For example, if we choose sigmoid functions $\frac{1}{1+e^{-x}}$ as the activation functions and v_i neurons are used for the NN of node i, then $\|\phi_i\| \leq \sqrt{v_i}$ and $\|\phi\| \leq \sqrt{Nv_i}$.

∎

5.4.3 Distributed controller design: Lyapunov approach

This section presents the main result. We show how to design the distributed control law u_i and the distributed NN weights tuning law, such that all nodes synchronize to the leader node, i.e. $x_i \to x_0, \forall i$.

The following definition extends the standard concept of *uniformly ultimately bounded* (UUB) ([28], [40]) to cooperative control systems.

Definition 5.6 For any $m = 1, 2, \cdots, M$, the tracking errors $\delta^m = x^m - \underline{x}_0^m$ are said to be cooperative uniformly ultimately bounded (CUUB) if there exist compact sets $\Omega^m \in \mathbb{R}$ which contain the origin, so that for any $\delta_i^m(t_0) \in \Omega^m$ $(\forall i = 1, 2, \cdots, N)$, there exist bounds B^m and time $T^m(B^m, \delta^m(t_0)$ such that $\|\delta^m(t)\| \leq B^m, \forall t \geq t_0 + T^m$. ∎

Before proceeding, we make the following assumptions,

Assumption 5.1

a) The trajectory of the leader node is bounded, i.e., there exists a positive number $X_M > 0$ such that $\|x_0(t)\| \leq X_M, \forall t \geq 0$.

b) There exists a continuous function $g(x) : \mathbb{R}^M \to \mathbb{R}$, such that $|f_0(x, t)| \leq |g(x)|, \forall x \in \mathbb{R}^M$ and $\forall t \geq 0$.

c) For each node i, the disturbance ζ_i is unknown, but bounded. Thus the overall disturbance vector ζ is also bounded by $\|\zeta\| \leq \zeta_M$ with ζ_M a known constant.

∎

Since node 0 acts as the command generator, Assumption 5.1 (a) is reasonable.

The main result of this chapter is given by the following theorem, which shows how to design the distributed controller u_i and the NN tuning law such that the tracking errors δ^m are cooperative uniformly ultimately bounded, thereby showing synchronization and cooperative stability for the whole graph $\mathcal{G}$.

Theorem 5.1 *Consider the distributed system (5.4) and the leader node (5.6). Design the distributed control law for each node i as*

$$u_i = \frac{1}{d_i + b_i}(\lambda_1 e_i^2 + \lambda_2 e_i^3 + \cdots + \lambda_{M-1} e_i^M) - \hat{f}_i(x_i) + cr_i, \tag{5.26}$$

where c is the control gain; $\hat{f}_i(x_i) = \hat{W}_i^T \phi_i(x_i)$ *with* $\hat{W}_i$ *the current estimate of the NN weights for i-th node, and* ϕ_i *the suitable basis set as in Remark 5.4. With* ρ *defined in (5.14), controller (5.26) can be written collectively as*

$$u = (D + B)^{-1}\rho - \hat{f} + cr. \tag{5.27}$$

Let the NN adaptive tuning law for each node i be

$$\dot{\hat{W}}_i = -F_i \phi_i r_i p_i (d_i + b_i) - \kappa F_i \hat{W}_i, \tag{5.28}$$

or collectively

$$\dot{\hat{W}} = -F\phi r P(D + B) - \kappa F \hat{W}, \tag{5.29}$$

where the design parameters $F_i = F_i^T \in \mathbb{R}^{v_i \times v_i}$ *can be arbitrary positive definite matrices,* $F = diag\{F_1, F_2, \cdots, F_N\} \in \mathbb{R}^{\sum_{i=1}^N v_i \times \sum_{i=1}^N v_i}$ *, and* v_i *is the suitable number of the neurons defined in Remark 5.4;* $\kappa > 0$ *is a scalar tuning gain; P is defined in Lemma 5.3 and depends on the topology of the graph.*

Choose the NN tuning gain κ *and the control gain c such that*

$$\kappa > 0 \tag{5.30}$$

and

$$c > \frac{2}{\underline{\sigma}(Q)} \left(\frac{\gamma^2}{\kappa} + \frac{2}{\beta} g^2 + h \right) \tag{5.31}$$

with $\gamma = -\frac{1}{2}\Phi_M \bar{\sigma}(P)\bar{\sigma}(A)$, $h = \frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)} \left\| \bar{\lambda} \right\|$ *and* $g = -\frac{1}{2}\left(\frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)} \left\| \Lambda \right\|_F \left\| \bar{\lambda} \right\| + \bar{\sigma}(P_1) \right)$*, where* P_1 *is defined in (5.12) for any* $\beta > 0$*, Q is defined as in Lemma 5.3 and* Φ_M *is defined in Remark 5.5.*

Then under Assumption 5.1, we have the following results

1) *the sliding mode error r is ultimately practically bounded according to (5.48), and the NN approximation error matrix* $\tilde{W}$ *is bounded according to (5.50). Thence, the disagreement vectors* δ^m, $\forall m = 1, 2, \cdots, M$ *are cooperative uniformly ultimately bounded, which implies all nodes in graph* $\mathcal{G}$ *synchronize to the leader node 0.*

2) *the states* $x_i(t)$ *are bounded* $\forall i$ *and* $\forall t > 0$.

Proof.

(1) Consider the Lyapunov function candidate

$$V = V_1 + V_2 + V_3 \tag{5.32}$$

with $V_1 = \frac{1}{2} r^T P r$, $V_2 = \frac{1}{2} tr\{\tilde{W}^T F^{-1} \tilde{W}\}$ and $V_3 = \frac{1}{2} tr\{E^1 P_1 (E^1)^T\}$.

First we compute the derivative of V_1 along the closed loop state trajectory.

$$\dot{V}_1 = r^T P \dot{r} = r^T P[\rho - (L + B)(f(x) + u + \zeta - \underline{f_0})] \tag{5.33}$$

To simplify the notation, we denote $f(x)$ as f, $\hat{f}(x)$ as $\hat{f}$ and $\phi(x)$ as ϕ in the sequel. Substituting (5.27) into (5.33), and considering (5.23), (5.24) , and $L = D - A$, we have

$$\begin{aligned}
\dot{V}_1 &= r^T P[\rho - (L+B)(f + \zeta - \underline{f_0} + (D+B)^{-1}\rho - \hat{f} + cr)] \\
&= r^T P[-(L+B)(f + \zeta - \underline{f_0} - \hat{f} + cr) + A(D+B)^{-1}\rho] \\
&= r^T P[-(L+B)(\tilde{W}\phi + \epsilon + \zeta - \underline{f_0} + cr) + A(D+B)^{-1}\rho] \\
&= -r^T P(L+B)\tilde{W}\phi - r^T P(L+B)(\epsilon + \zeta - \underline{f_0}) - cr^T P(L+B)r + r^T PA(D+B)^{-1}\rho \\
&= -r^T P(L+B)(\epsilon + \zeta - \underline{f_0}) - cr^T P(L+B)r - r^T P(D+B)\tilde{W}\phi \\
&\quad + r^T PA\tilde{W}^T\phi + r^T PA(D+B)^{-1}\rho.
\end{aligned} \tag{5.34}$$

Note the fact that $x^T y = tr\{yx^T\}$ if $x, y \in \mathbb{R}^N$, and consider (5.3),

$$\begin{aligned}
\dot{V}_1 = &-r^T P(L+B)(\epsilon + \zeta - \underline{f_0}) - \frac{1}{2}cr^T Qr - tr\{\tilde{W}\phi r^T P(D+B)\} \\
&+ tr\{\tilde{W}^T\phi r^T PA\} + r^T PA(D+B)^{-1}\rho.
\end{aligned} \tag{5.35}$$

Since $\dot{\tilde{W}} = \dot{W} - \dot{\hat{W}} = -\dot{\hat{W}}$,

$$\begin{aligned}
\dot{V}_1 + \dot{V}_2 = &-r^T P(L+B)(\epsilon + \zeta - \underline{f_0}) - \frac{1}{2}cr^T Qr - tr\{\tilde{W}\phi r^T P(D+B)\} \\
&+ tr\{\tilde{W}^T\phi r^T PA\} + r^T PA(D+B)^{-1}\rho - tr\{\tilde{W}^T F^{-1}\dot{\hat{W}}\}.
\end{aligned} \tag{5.36}$$

Substitute (5.29) into (5.36) yields

$$\begin{aligned}
\dot{V}_1 + \dot{V}_2 = &-\frac{1}{2}cr^T Qr - r^T P(L+B)(\epsilon + \zeta - \underline{f_0}) \\
&+ \kappa tr\{\tilde{W}^T\hat{W}\} + tr\{\tilde{W}^T\phi r^T PA\} + r^T PA(D+B)^{-1}\rho.
\end{aligned} \tag{5.37}$$

Considering (5.14) and (5.11), (5.37) is

$$\begin{aligned}
\dot{V}_1 + \dot{V}_2 = &-\frac{1}{2}cr^T Qr - r^T P(L+B)(\epsilon + \zeta - \underline{f_0}) + \kappa tr\{\tilde{W}^T W\} - \kappa \left\|\tilde{W}\right\|_F^2 \\
&+ tr\{\tilde{W}^T\phi r^T PA\} + r^T PA(D+B)^{-1}E^1\Lambda^T\bar{\lambda} + r^T PA(D+B)^{-1}rl^T\bar{\lambda}.
\end{aligned} \tag{5.38}$$

Denote $\Omega_0 = \{x_0 \in \mathbb{R}^M : \|x_0(t)\| \leq X_M\}$, then by Assumption 5.1 (a), $x_0 \in \Omega_0$ for all $t \geq 0$. Assumption 5.1 (b) implies the boundedness of $|f_0(x_0, t)|$. Suppose $|f_0(x_0, t)|$ is bounded by f_{0M}, then $\left\|\underline{f_0}\right\| \leq \sqrt{N} f_{0M}$. Denote $\sqrt{N} f_{0M} = F_M$. Let $T_M = \epsilon_M + \zeta_M + F_M$, then

$$\begin{aligned}
\dot{V}_1 + \dot{V}_2 \leq &-\frac{1}{2}c\underline{\sigma}(Q)\|r\|^2 + \bar{\sigma}(P)\bar{\sigma}(L+B)T_M\|r\| + \kappa W_M\left\|\tilde{W}\right\|_F + \Phi_M\bar{\sigma}(P)\bar{\sigma}(A)\left\|\tilde{W}\right\|_F\|r\| \\
&- \kappa\left\|\tilde{W}\right\|_F^2 + \|r\|\frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)}\left\|E^1\right\|_F\|\Lambda\|_F\left\|\bar{\lambda}\right\| + \frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)}\|r\|^2\|l\|\left\|\bar{\lambda}\right\| \\
= &-\left(\frac{1}{2}c\underline{\sigma}(Q) - \frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)}\left\|\bar{\lambda}\right\|\right)\|r\|^2 - \kappa\left\|\tilde{W}\right\|_F^2 + \Phi_M\bar{\sigma}(P)\bar{\sigma}(A)\left\|\tilde{W}\right\|_F\|r\| \\
&+ \kappa W_M\left\|\tilde{W}\right\|_F + \bar{\sigma}(P)\bar{\sigma}(L+B)T_M\|r\| + \frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)}\|\Lambda\|_F\left\|\bar{\lambda}\right\|\left\|E^1\right\|_F\|r\|.
\end{aligned} \tag{5.39}$$

Next we compute the derivative of V_3. Noting that $tr\{A + A^T\} = 2tr\{A\}$ for any square matrix A, we have

$$\dot{V}_3 = \frac{1}{2}\{\dot{E}^1 P_1 (E^1)^T + E^1 P_1 (\dot{E}^1)^T\} = tr\{E^2 P_1 (E^1)^T\}. \tag{5.40}$$

Substituting (5.11) into (5.40) and considering (5.12) gives

$$\begin{aligned} \dot{V}_3 &= tr\{E^1 \Lambda^T P_1 (E^1)^T\} + tr\{r l^T P_1 (E^1)^T\} \\ &= \frac{1}{2}\{E^1 (\Lambda^T P_1 + P_1 \Lambda)(E^1)^T\} + tr\{r l^T P_1 (E^1)^T\} \\ &= -\frac{\beta}{2} tr\{E^1 (E^1)^T\} + tr\{r l^T P_1 (E^1)^T\}. \end{aligned} \tag{5.41}$$

Thus,

$$\begin{aligned} \dot{V}_3 &\leq -\frac{\beta}{2} \left\|E^1\right\|_F^2 + \bar{\sigma}(P_1) \|l\| \|r\| \left\|E^1\right\|_F \\ &= -\frac{\beta}{2} \left\|E^1\right\|_F^2 + \bar{\sigma}(P_1) \|r\| \left\|E^1\right\|_F. \end{aligned} \tag{5.42}$$

Therefore,

$$\begin{aligned} \dot{V} \leq &-\left(\frac{1}{2} c\underline{\sigma}(Q) - \frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)} \|\bar{\lambda}\|\right) \|r\|^2 - \kappa \left\|\tilde{W}\right\|_F^2 - \frac{\beta}{2} \left\|E^1\right\|_F^2 + \Phi_M \bar{\sigma}(P)\bar{\sigma}(A) \left\|\tilde{W}\right\|_F \|r\| \\ &\left(\frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)} \|\Lambda\|_F \|\bar{\lambda}\| + \bar{\sigma}(P_1)\right) \|r\| \left\|E^1\right\|_F + \bar{\sigma}(P)\bar{\sigma}(L+B) T_M \|r\| + \kappa W_M \left\|\tilde{W}\right\|_F. \end{aligned} \tag{5.43}$$

To simplify the notations, let

$$\gamma = -\frac{1}{2}\Phi_M \bar{\sigma}(P)\bar{\sigma}(A),$$

$$h = \frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)} \|\bar{\lambda}\|,$$

and

$$g = -\frac{1}{2}\left(\frac{\bar{\sigma}(P)\bar{\sigma}(A)}{\underline{\sigma}(D+B)} \|\Lambda\|_F \|\bar{\lambda}\| + \bar{\sigma}(P_1)\right),$$

then (5.43) is written as

$$\dot{V} \leq -z^T K z + \omega^T z, \tag{5.44}$$

where $z = \left[\left\|E^1\right\|_F, \left\|\tilde{W}\right\|_F, \|r\|\right]^T$, $K = \begin{bmatrix} \frac{\beta}{2} & 0 & g \\ 0 & \kappa & \gamma \\ g & \gamma & \mu \end{bmatrix}$, $\mu = \frac{1}{2} c\underline{\sigma}(Q) - h$ and $\omega = [0, \kappa W_M, \bar{\sigma}(P)\bar{\sigma}(L+B)T_M]^T$.

Then $\dot{V} < 0$ if the following two conditions C1 and C2 hold,

C1) K is positive definite.

C2) $\|z\| > \frac{\|\omega\|}{\underline{\sigma}(K)}$, where $\underline{\sigma}(K) > 0$ is the smallest singular value of matrix K.

According to Sylvester's criterion, K is positive definite if all its leading principle minors are positive, i.e.,

$$\beta > 0 \tag{5.45}$$

$$\beta\kappa > 0 \tag{5.46}$$

$$\kappa(\beta\mu - 2g^2) - \beta\gamma^2 > 0 \tag{5.47}$$

Solving the above system of equations gives conditions (5.30) and (5.31). Consider a property of vector norm, i.e. $\|\omega\|_1 \geq \|\omega\|_2 \geq \cdots \geq \|\omega\|_\infty$, condition C2 holds if either

$$\|r\| > \frac{\|\omega\|_1}{\underline{\sigma}(K)} = \frac{\bar{\sigma}(P)\bar{\sigma}(L+B)T_M + \kappa W_M}{\underline{\sigma}(K)}, \tag{5.48}$$

or

$$\left\|E^1\right\|_F > \frac{\|\omega\|_1}{\underline{\sigma}(K)} = \frac{\bar{\sigma}(P)\bar{\sigma}(L+B)T_M + \kappa W_M}{\underline{\sigma}(K)}, \tag{5.49}$$

or

$$\left\|\tilde{W}\right\|_F > \frac{\|\omega\|_1}{\underline{\sigma}(K)} = \frac{\bar{\sigma}(P)\bar{\sigma}(L+B)T_M + \kappa W_M}{\underline{\sigma}(K)}. \tag{5.50}$$

Therefore, as long as the design parameter κ and c satisfy (5.30) and (5.31), then sliding mode error r and the NN weights approximation error $\tilde{W}$ are ultimately practically bounded by $Bd = \frac{\bar{\sigma}(P)\bar{\sigma}(L+B)T_M+\kappa W_M}{\underline{\sigma}(K)}$. It then follows from Lemma 5.4 and Lemma 5.5, that the synchronization error $\delta^m(t)$ is cooperative uniformly ultimately bounded and all nodes in graph $\mathcal{G}$ synchronize to the trajectory of the leader node $x_0(t)$.

(2) Similar to standard neural adaptive control (e.g. not for networked systems) results, trajectories of all nodes $x_i(t)$ are bounded, $\forall t > 0$. The proof technique is adopted from [16]. Thus details are omitted and only some flavors are shown as below. Straightforward computation of (5.32) implies

$$\underline{\sigma}(\Gamma)\|z\|^2 \leq V(t) \leq \bar{\sigma}(T)\|z\|^2, \tag{5.51}$$

where $\Gamma = diag\left\{\frac{\underline{\sigma}(P_1)}{2}, \frac{1}{2\bar{\sigma}(F)}, \frac{\underline{\sigma}(P_1)}{2}\right\} \in \mathbb{R}^{3\times 3}$ and $T = diag\left\{\frac{\bar{\sigma}(P_1)}{2}, \frac{1}{2\underline{\sigma}(F)}, \frac{\bar{\sigma}(P_1)}{2}\right\} \in \mathbb{R}^{3\times 3}$. (5.44) implies

$$\dot{V} \leq -\underline{\sigma}(K)\|z\|^2 + \|\omega\|\|z\|. \tag{5.52}$$

Then the combination of (5.51) and (5.52) gives

$$\frac{d}{dt}(\sqrt{V}) \leq -\frac{\underline{\sigma}(K)}{2\bar{\sigma}(T)}\sqrt{V} + \frac{\|\omega\|}{2\sqrt{\underline{\sigma}(\Gamma)}}. \tag{5.53}$$

Thus $V(t)$ is bounded by Corollary 1.1 in [16]. Since (5.32) implies $\|r\|^2 \leq \frac{2V(t)}{\underline{\sigma}(P)}$, it then follows the boundedness of $r(t)$. By Lemma 5.5 and Lemma 5.2, $\delta^m(t)$ is also bounded. Since $\delta^m = x^m - \underline{x}_0^m$ and considering Assumption 5.1 (a), we can see $x^m(t)$, $\forall m = 1, 2, \cdots, M$, are bounded all the time, i.e. $x_i^m(t)$ are bounded. The bounds depend on the initial conditions $V(0)$. Bigger initial condition results in bigger bounds for $x_i^m(t)$ [16]. Therefore given any initial condition $V(0)$, our result holds as long as

the compact set Ω, on which the NN approximation property holds, is chosen to be large enough to make sure the states $x_i(t) \in \Omega$, $\forall i, m$ and $\forall t > 0$. In this sense, our stability result is semiglobal. Similarly $\tilde{W}$ can also be shown bounded. □

Remark 5.6 Note that the gains $p_i > 0$ in NN tuning law (5.28) are computed using Lemma 5.3, which requires global information of the graph for computing $(L + B)^{-1}$. Therefore, they are not known locally and the controller u_i can not be implemented in a distributed fashion. However, NN tuning gain matrices $F_i > 0$ are arbitrary, which implies $p_i F_i$ are arbitrary. This finally implies that we can pick an arbitrary $p_i > 0$ for the NN tuning law. This is demonstrated by the example in Section 5.5. Therefore, the proposed controller u_i is thoroughly distributed, as one can see from (5.26) and (5.28). For design purpose, the control gain c can be smaller than in the condition (5.31), for (5.31) provides an conservative upper bound of the control gain c. ∎

From the proof of Theorem 5.1, it is obvious that the bound Bd is conservative, i.e. it is larger than the actual bound for the sliding mode error r. We now show how to choose the design parameters c and κ, such that the bound Bd is minimized. Note that once the topology of the graph and the design parameters c and κ are fixed, the actual ultimate bound on the sliding mode error is also fixed. Also note that β is NOT a controller design parameter, it is only used to show boundedness. Choosing different β does not affect the actual bound, nor the guide of how to choose the design parameter c and κ. So to ease the following analysis, we let $\beta = 2$. Then $K = \begin{bmatrix} 1 & 0 & g \\ 0 & \kappa & \gamma \\ g & \gamma & \mu \end{bmatrix}$.

Define $\delta_1 > 0$, such that

$$\frac{1}{2} c \underline{\sigma}(Q) = \frac{\gamma^2}{\kappa} + \frac{2}{\beta} g^2 + h + \delta_1. \tag{5.54}$$

Note that for symmetric positive definite matrix K, its smallest singular value $\underline{\sigma}(K)$ is equal to its smallest eigenvalue. One of the lower bound of the smallest eigenvalue [48] of the positive definite matrix K is given by

$$\begin{aligned} \underline{\lambda_m} &= \frac{\mu + \eta_1}{2} - \sqrt{\frac{(\mu + \eta_1)^2}{4} - (\mu - g^2 - \frac{\gamma^2}{\kappa})\eta_1} \\ &= \frac{1}{2}\left((\mu + \eta_1) - \sqrt{(\mu + \eta_1)^2 - 4\delta_1 \eta_1} \right), \end{aligned} \tag{5.55}$$

where η_1 is the smallest eigenvalue of the submatrix $\begin{bmatrix} 1 & 0 \\ 0 & \kappa \end{bmatrix}$, i.e. $\eta_1 = min\{1, \kappa\}$. Using the lower bound of the singular value $\underline{\sigma}(K)$, we can have a conservative expression for bound Bd as follows.Two cases are considered with respect to the value of κ.

Case 1: $1 \leq \kappa$

Then $\eta_1 = 1$ and the bound is

$$Bd = \frac{2\bar{\sigma}(P)\bar{\sigma}(L + B)T_M + 2\kappa W_M}{\left(\frac{\gamma^2}{\kappa} + g^2 + \delta_1 + 1\right) - \sqrt{\left(\frac{\gamma^2}{\kappa} + g^2 + \delta_1 + 1\right)^2 - 4\delta_1}}. \tag{5.56}$$

Case 2: $0 < \kappa < 1$

Then $\eta_1 = \kappa$ and the bound is

$$Bd = \frac{2\bar{\sigma}(P)\bar{\sigma}(L+B)T_M + 2\kappa W_M}{\left(\frac{\gamma^2}{\kappa} + g^2 + \delta_1 + \kappa\right) - \sqrt{\left(\frac{\gamma^2}{\kappa} + g^2 + \delta_1 + \kappa\right)^2 - 4\kappa\delta_1}}. \tag{5.57}$$

For both cases, when κ is fixed, increase δ_1 (or equivalently increase the control gain c) can decrease the bound Bd; But when δ_1 (or c) is fixed, to decrease the bound Bd, κ should be designed properly case by case (i.e. it depends on all the other parameters in the expression of Bd). Loosely speaking, smaller κ can decrease the effect of W_M, thus decrease Bd to some extent.

Remark 5.7 Another interesting thing is that the controller designed in Theorem 5.1 can even control a networked systems with switching graph topology, as long as the switched topology is also strongly connected and the leader node can still give commands to at least one node. This can be intuitively justified from the forms of controllers (5.26) and adaptive NN tuning laws (5.28), since they consider the information of the topology of the graph. When the topology changes, control laws (5.26) and NN tuning laws (5.28) changes accordingly. An example is given in Section 5.5 to demonstrate this ability.

∎

5.5 Simulation results

Two examples are presented to demonstrate that the designed controller works not only for communication graph with a fixed topology, but also for communication graph with switching topology.

5.5.1 Example 1: case for graph with fixed topology

Consider a 4-node strongly connected digraph $\mathcal{G}$ given in Fig.5.5. Each node i ($i = 1, 2, 3, 4$) is modeled by a third order nonlinear dynamics. The leader node, labeled 0, only gives commands to node 1 with the weighted edge $b_1 = 5$. Then the adjacency matrix is

$$A = \begin{bmatrix} 0 & 0 & 2 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 4 & 0 & 1 \\ 3 & 0 & 0 & 0 \end{bmatrix}, \text{ and } B = \begin{bmatrix} 5 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Let the dynamics of the leader node be

$$\begin{aligned} \dot{x}_0^1 &= x_0^2 \\ \dot{x}_0^2 &= x_0^3 \\ \dot{x}_0^3 &= -x_0^2 - 2x_0^3 + 1 + 3sin(2t) + 6cos(2t) - \frac{1}{3}(x_0^1 + x_0^2 - 1)^2(x_0^1 + 4x_0^2 + 3x_0^3 - 1), \end{aligned} \tag{5.58}$$

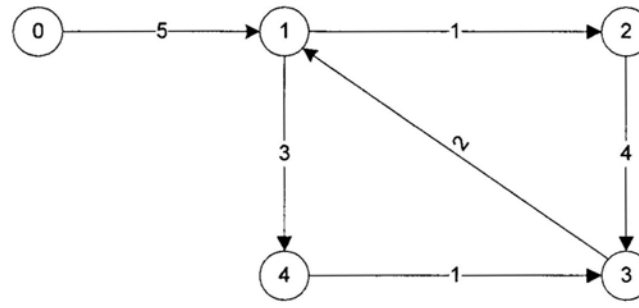

Figure 5.5: Topology of the communication graph $\mathcal{G}$

which is obtained from FitzHugh-Nagumo model [69] by applying coordinate transformation. System 5.58 exhibits a chaotic behavior as shown in Fig.5.6-5.7 with initial condition $x_0 = [3, 4, 1]^T$. The trajectory is bounded, thus satisfying Assumption 5.1 (a).

Nodes i ($i = 1, 2, 3, 4$) are described by third order nonlinear systems (5.4) with $M = 3$ and

$$\begin{aligned}
\dot{x}_1^3 &= x_1^2 sin(x_1^1) + cos((x_1^3)^2) + u_1 + \zeta_1 \\
\dot{x}_2^3 &= x_2^1 + cos(x_2^2) + (x_2^3)^2 + u_2 + \zeta_2 \\
\dot{x}_3^3 &= x_3^2 + sin(x_3^3) + u_3 + \zeta_3 \\
\dot{x}_4^3 &= -x_4^2 - x_4^3 + 0.5sin(2t) + cos(2t) - 3(x_4^1 + x_4^2 - 1)^2(x_4^1 + x_4^2 + x_4^3 - 1) + u_4 + \zeta_4,
\end{aligned} \tag{5.59}$$

where disturbances ζ_i are taking bounded but randomly as $\zeta_i = 0.2sin(randn)$ (*randn* is a MATLAB function, which generates normally distributed random numbers with mean be 0 and variance be 1). Note that open-loop system of node 3 (without disturbance) has a finite escape time. Open-loop system of node 4 is also a transformed FitzHugh-Nagumo model with different parameters from the leader node and it also has a chaotic behavior. See Fig.5.8 and Fig.5.9 when initial condition be $x_4(0) = [1, 1, 0]^T$.

The initial values of the states for each node i, $\forall i = 1, 2, 3, 4$, are randomly chosen as long as $x_i(0) \in \Omega$, which is as large as desired. In simulation, we take $x_1(0) = [4, 1.6, 0.9]^T$, $x_2(0) = [2, -3, 2.8]^T$, $x_3(0) = [0.4, 0, -1]^T$ and $x_4(0) = [1, 1, 0]^T$. During simulation, we find NNs with only a small amount of neurons can give good performance. In this example, only 6 neurons are used for each NN. Initialize the NN weights be zero, i.e. $\hat{W}_i(0) = [0, 0, 0, 0, 0, 0]^T$, $\forall i$. To qualify ϕ_i as a basis set, sigmoid basis functions are used and the input signal of each NN is preprocessed by a random matrix $v = randn(3, 6)$ ($randn$(3,3) is a MATLAB function, which generates a 3×3 matrix with each entry be a random number generated by MATLAB command $randn$), then $\hat{f}_i(x_i) = \hat{W}_i^T \phi_i(v^T x_i)$.

Choosing design parameter as $\alpha_1 = \alpha_2 = 10$, then $\lambda_1 = \alpha_1\alpha_2 = 100$ and $\lambda_2 = \alpha_1 + \alpha_2 = 20$; $c = 600$; $\kappa = 0.01$; $F_i = 2000I$ with I be the identity matrix with appropriate dimensions; p_i in the NN adaptive tuning law (5.28) is chosen arbitrarily as $p_i = 50(1 + sin(randn))$. Note, for the design purpose, p_i can be chosen arbitrarily, as long as it is positive.

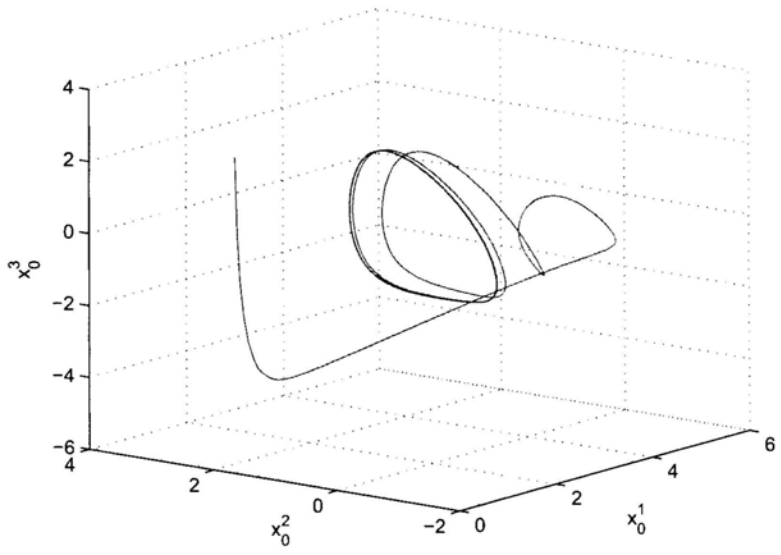

Figure 5.6: Phase plot of the leader node 0

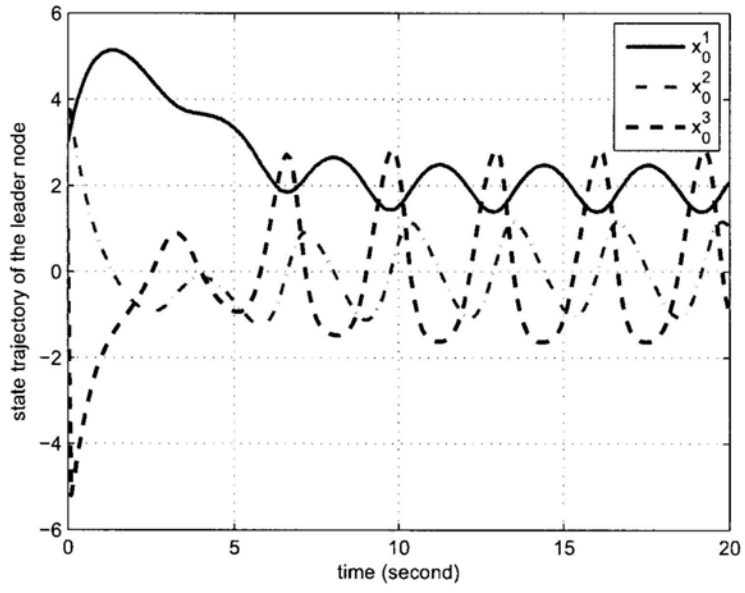

Figure 5.7: State trajectory of the leader node 0

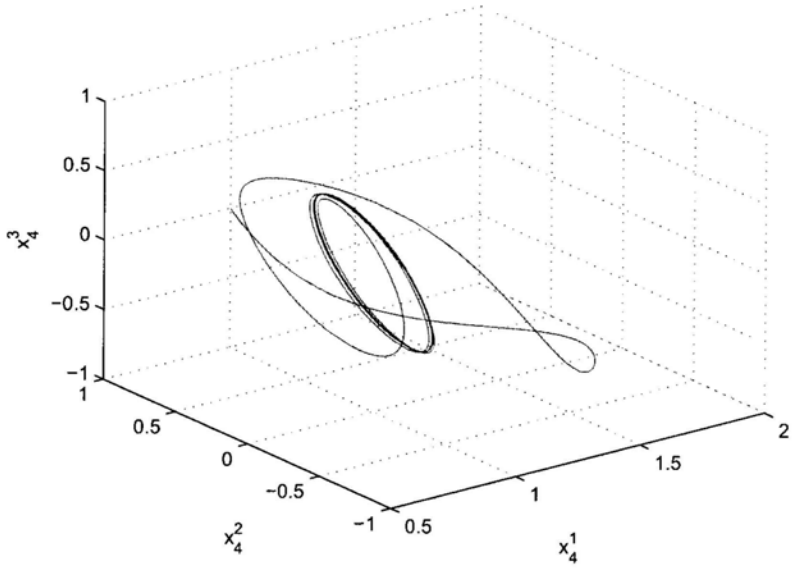

Figure 5.8: Phase plot of the open-loop system of node 4

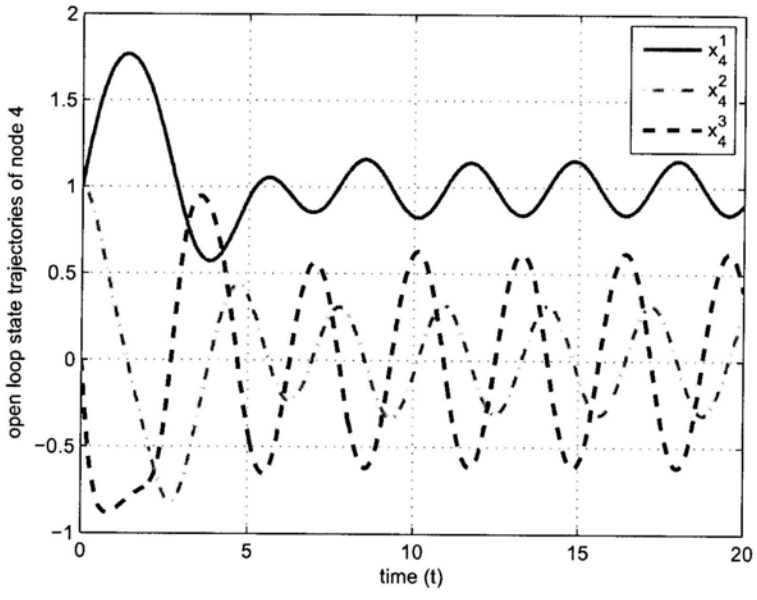

Figure 5.9: State trajectory of the open-loop system of node 4

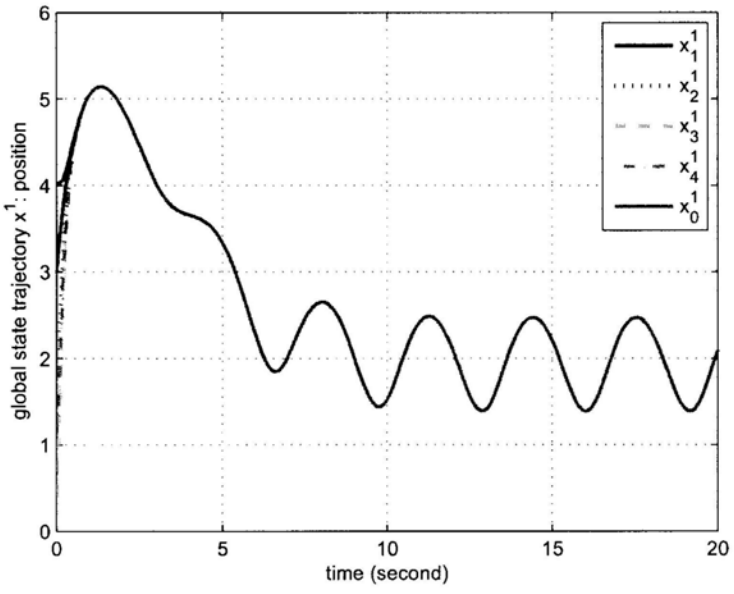

Figure 5.10: Profiles of the global position vector x^1

Applying controller (5.26) to the networked system considered, the profiles of the state trajectory for each node is shown in Fig.5.10–Fig.5.15. Profiles of the disagreement vector δ^m are shown in Fig.5.16–Fig.5.21, where Fig.5.17, Fig.5.19 and Fig.5.21 show that the disagreement vectors δ^m do not converge to zero, but are ultimately bounded by small residual errors. For example, $|\delta_i^1| \leq 6 \times 10^{-5}$. These figures demonstrate the fast tracking performance of our algorithm.

5.5.2 Example 2: case for graph with switching topology

Let the dynamics of the considered networked system be the same as in Example 5.5.1. For the first 10 seconds, the topology of the communication graph is the same as in Example 5.5.1 as shown in Fig.5.5; during the time $t = 10s$ to $t = 20s$, the topology of the communication graph $\mathcal{G}$ switches to Fig.5.22, which is also strongly connected. Let the number of the neurons for each NN be 6, and the other design parameter be the same as in Example 5.5.1.

Then the profiles of the state trajectory for each node is shown in Fig.5.23–Fig.5.28. As we expected, fast tracking is obtained and affects of the switching topology can not even be observed in these figures.

Profiles of the disagreement vector δ^m are shown in Fig.5.29–Fig.5.34, which also show the fast tracking ability and small tracking residual errors. It can be seen from Fig.5.30, Fig.5.32 and Fig.5.34 that at the switching instant, i.e. $t = 10s$, the disagreement variables δ_i^m changes instantly, but are also bounded with different bounds from that during $t = [0, 10]$. The different ultimate bounds is caused by

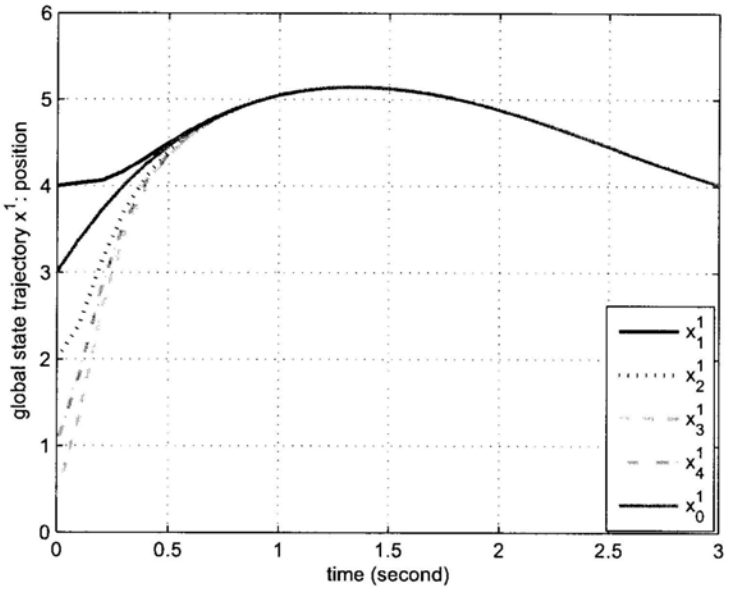

Figure 5.11: Profiles of the global position vector x^1 for $t = [0, 3]$

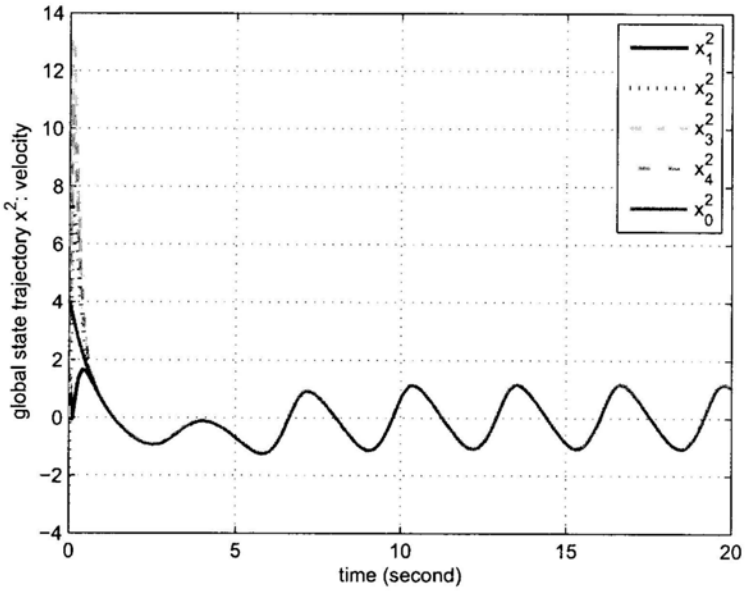

Figure 5.12: Profiles of the global velocity vector x^2

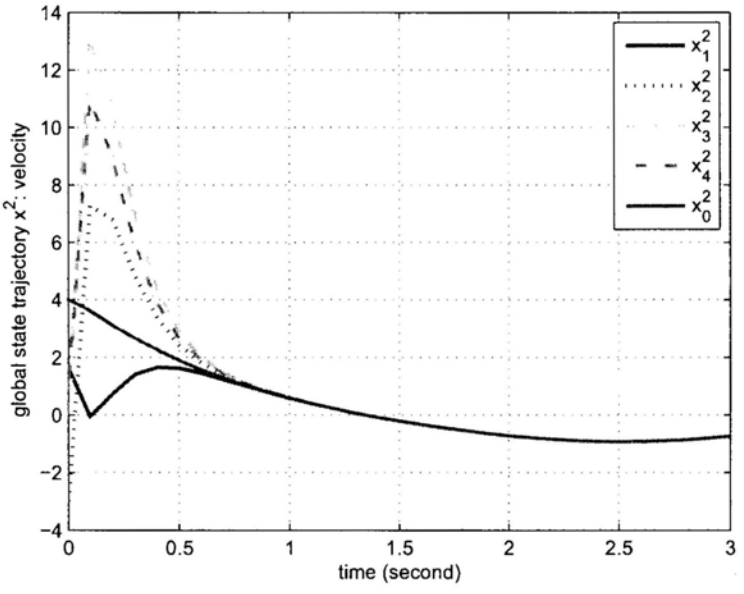

Figure 5.13: Profiles of the global velocity vector x^2 for $t = [0, 3]$

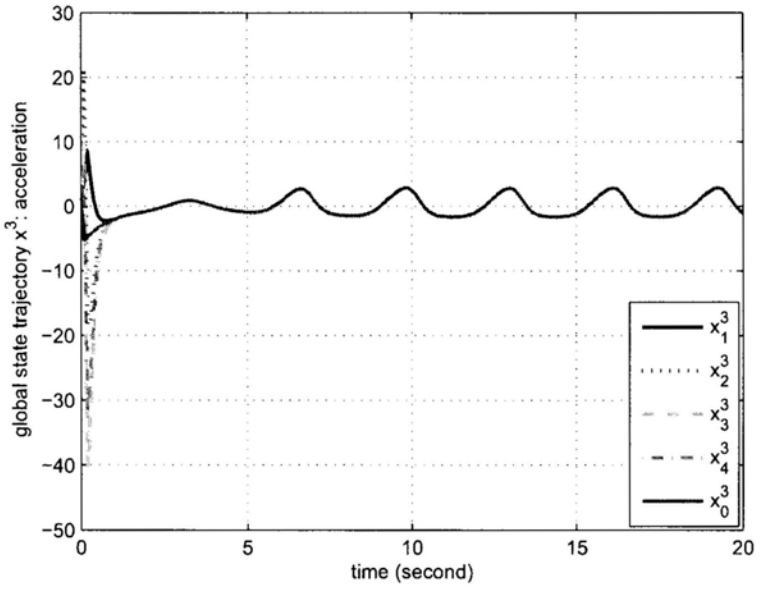

Figure 5.14: Profiles of the global acceleration vector x^3

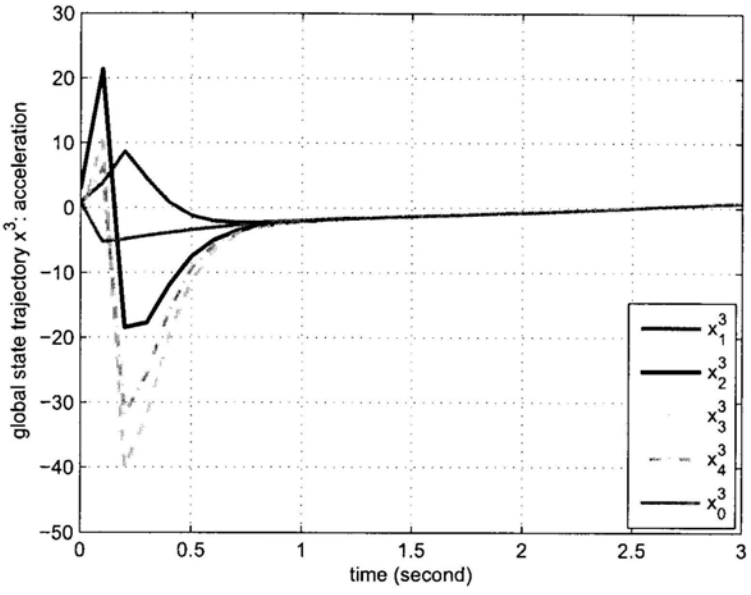

Figure 5.15: Profiles of the global acceleration vector x^3 for $t = [0, 3]$

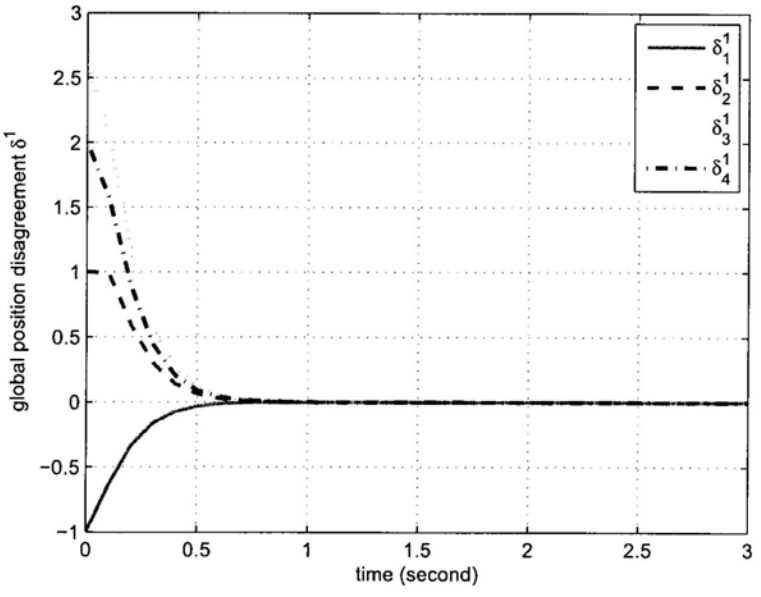

Figure 5.16: Profiles of the global position disagreement vector δ^1 for $t = [0, 3]$

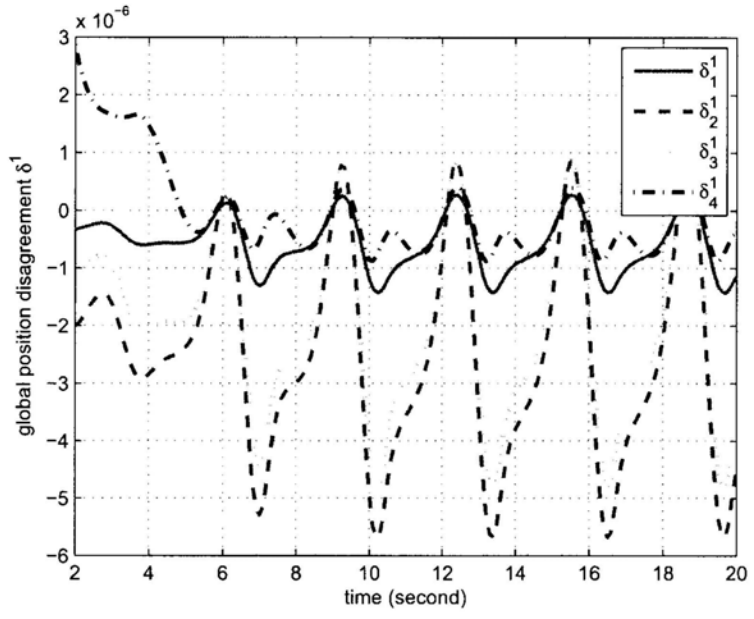

Figure 5.17: Profiles of the global position disagreement vector δ^1 for $t = [2, 20]$

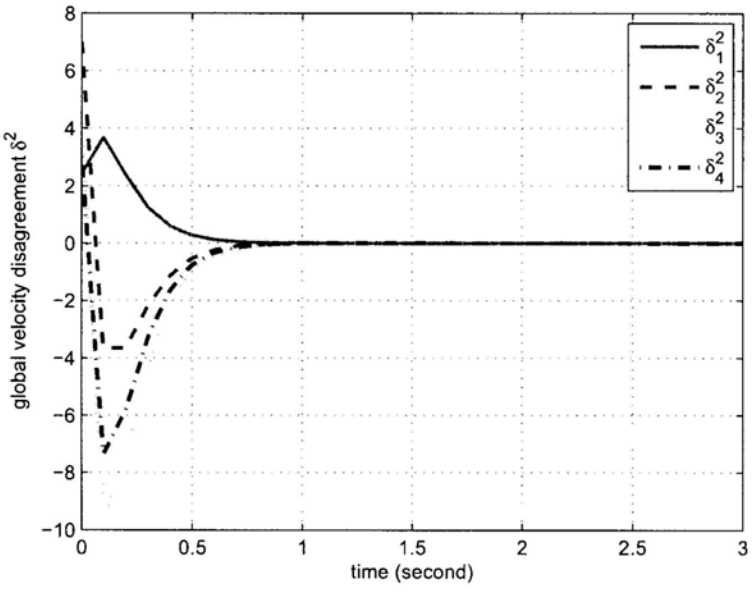

Figure 5.18: Profiles of the global velocity disagreement vector δ^2 for $t = [0, 3]$

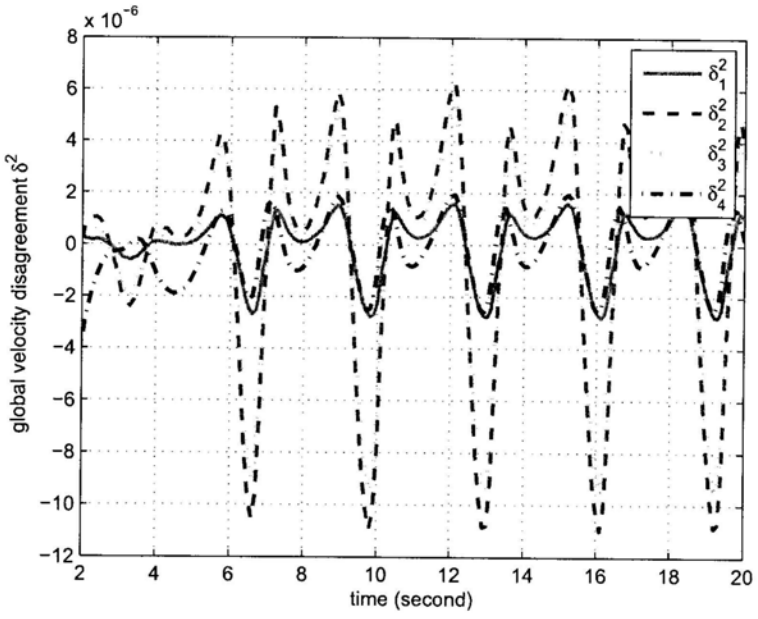

Figure 5.19: Profiles of the global velocity disagreement vector δ^2 for $t = [2, 20]$

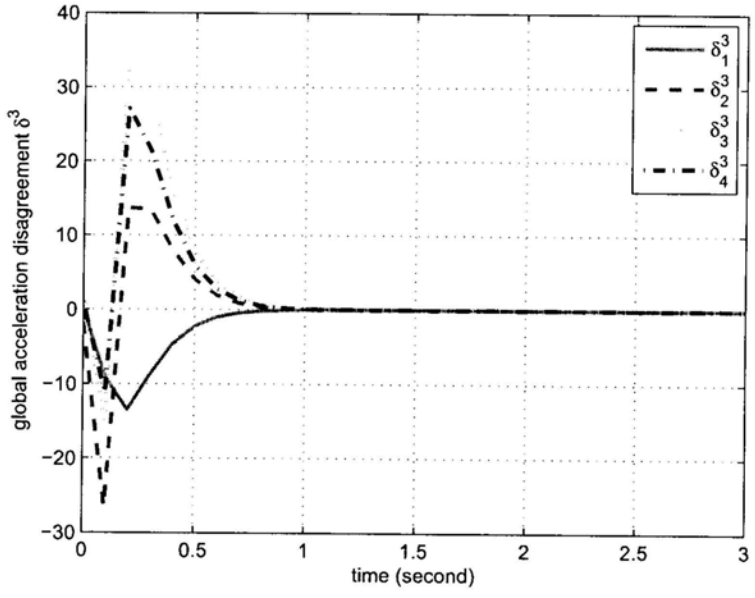

Figure 5.20: Profiles of the global acceleration disagreement vector δ^3 for $t = [0, 3]$

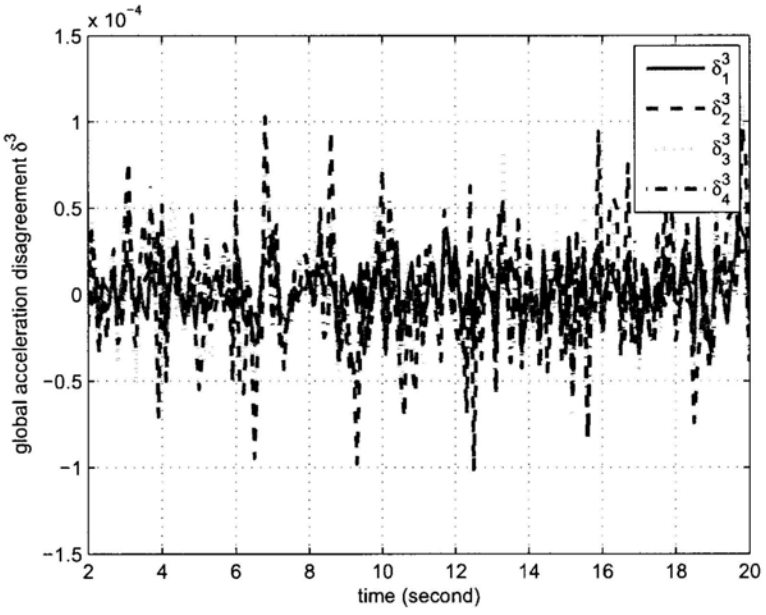

Figure 5.21: Profiles of the global acceleration disagreement vector δ^3 for $t = [2, 20]$

different topologies. These figures demonstrate that our algorithm may apply to networked systems with switching topology. The rigorous proof is under investigation.

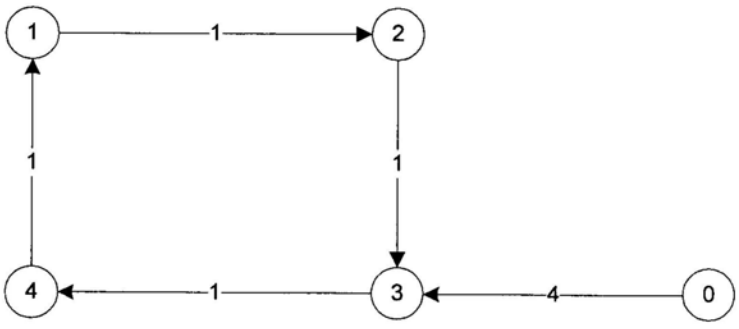

Figure 5.22: Topology of graph $\mathcal{G}$ for $t = [10, 20]$

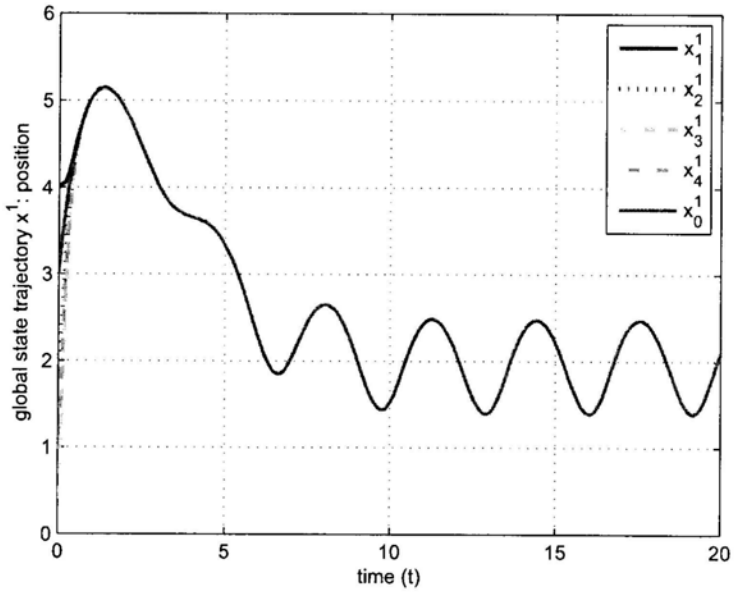

Figure 5.23: Profiles of the global position vector x^1

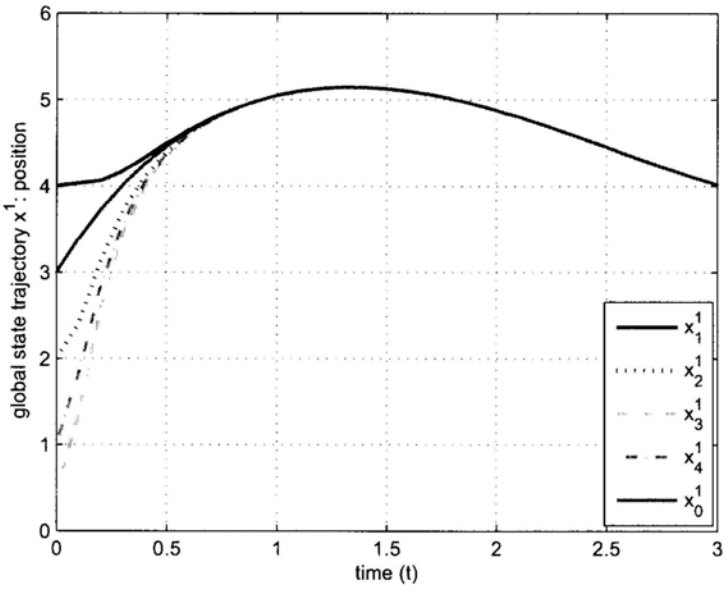

Figure 5.24: Profiles of the global position vector x^1 for $t = [0, 3]$

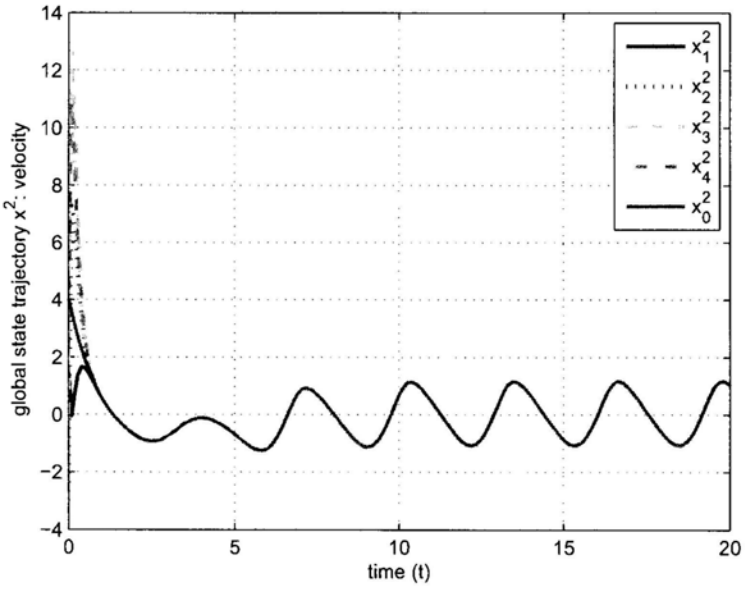

Figure 5.25: Profiles of the global velocity vector x^2

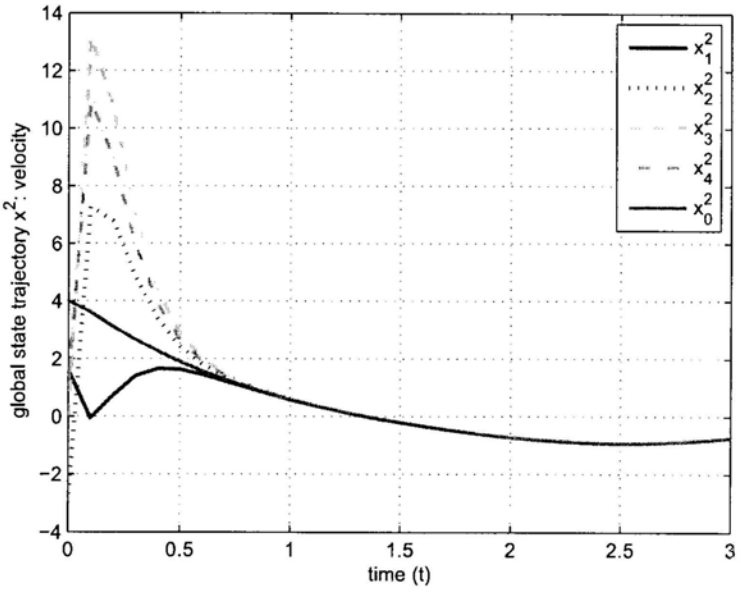

Figure 5.26: Profiles of the global velocity vector x^2 for $t = [0, 3]$

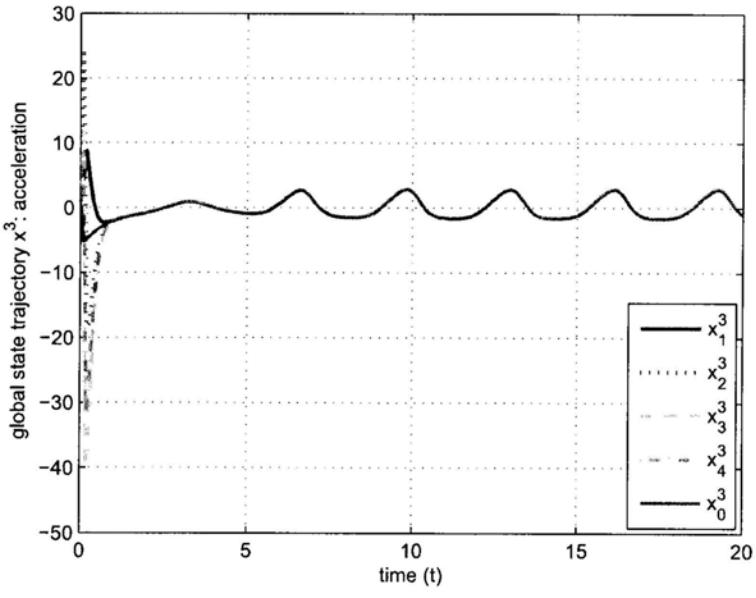

Figure 5.27: Profiles of the global acceleration vector x^3

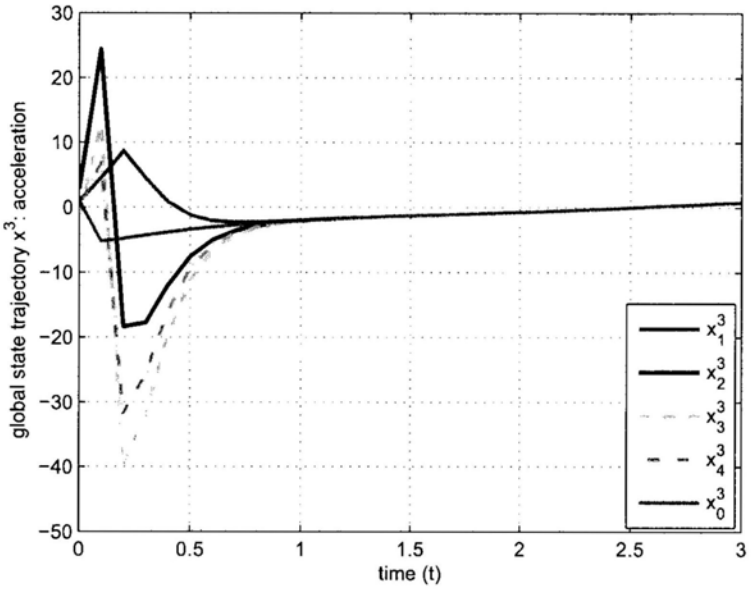

Figure 5.28: Profiles of the global acceleration vector x^3 for $t = [0, 3]$

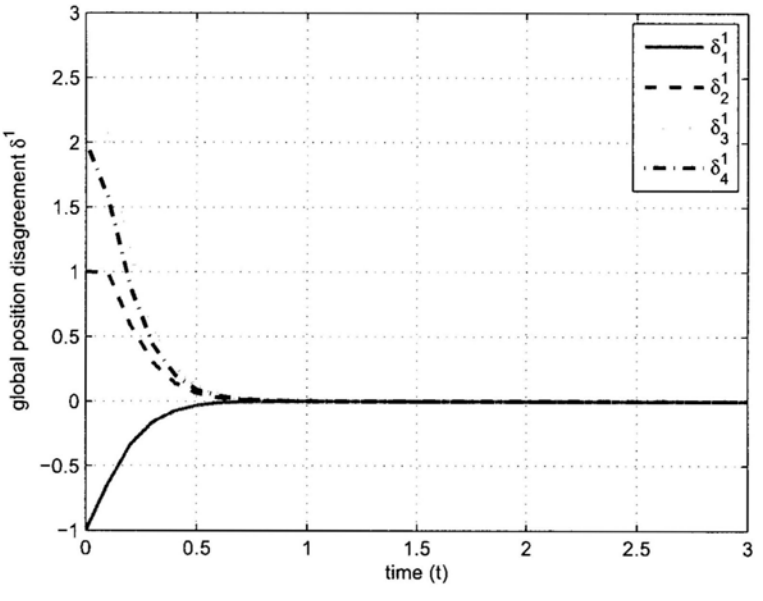

Figure 5.29: Profiles of the global position disagreement vector δ^1 for $t = [0, 3]$

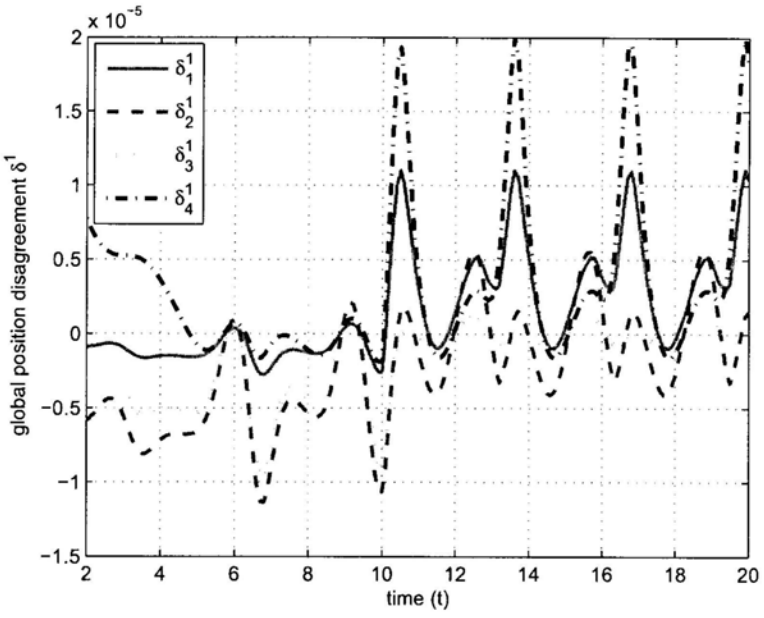

Figure 5.30: Profiles of the global position disagreement vector δ^1 for $t = [2, 20]$

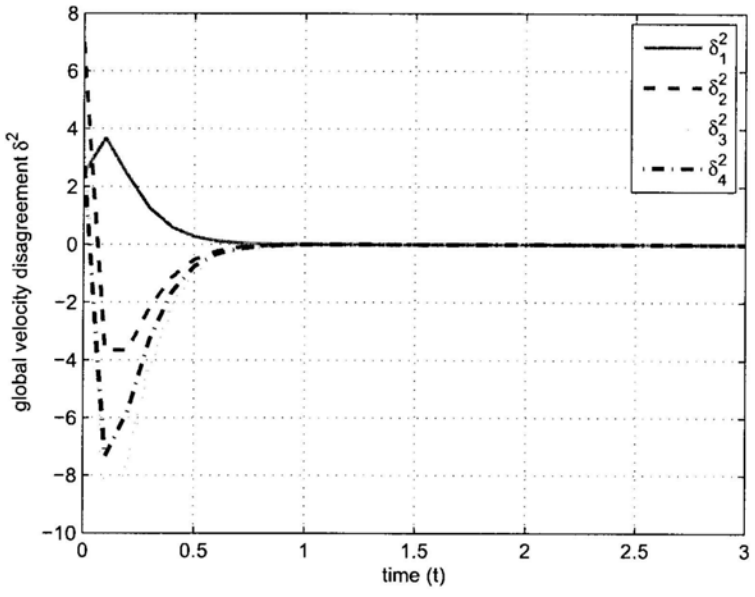

Figure 5.31: Profiles of the global velocity disagreement vector δ^2 for $t = [0, 3]$

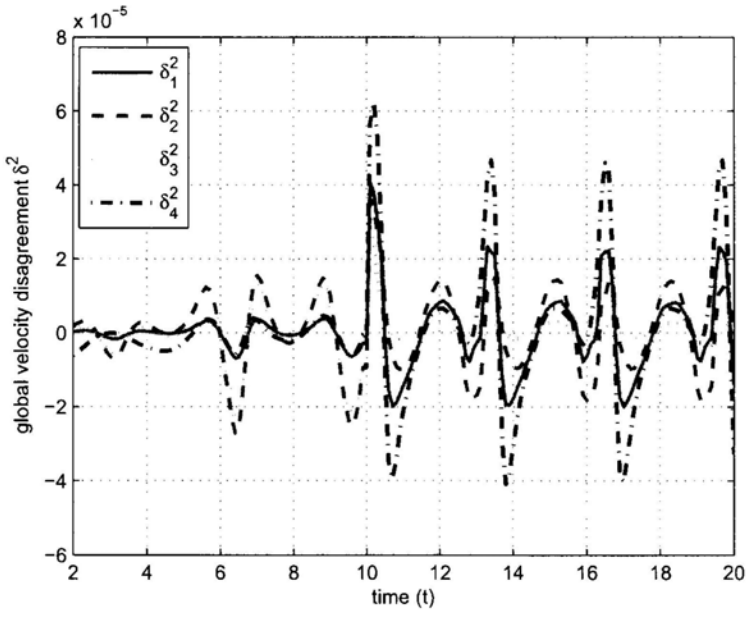

Figure 5.32: Profiles of the global velocity disagreement vector δ^2 for $t = [2, 20]$

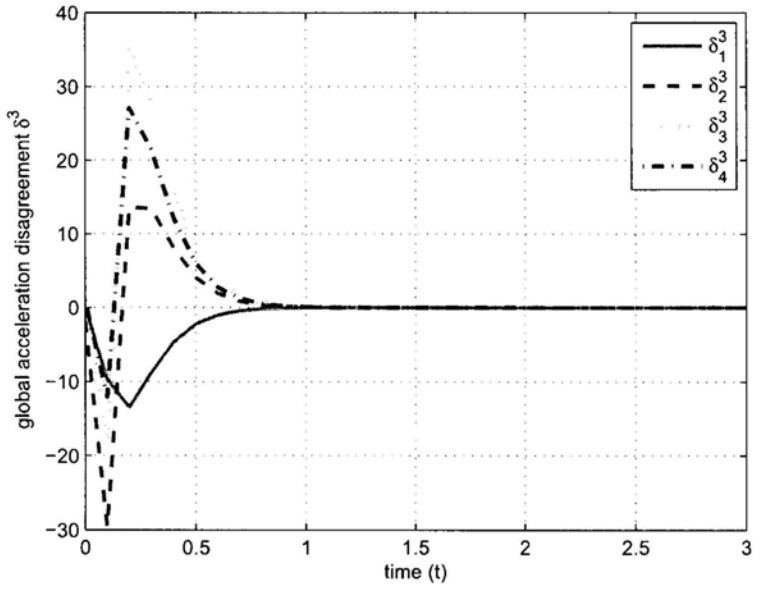

Figure 5.33: Profiles of the global acceleration disagreement vector δ^3 for $t = [0, 3]$

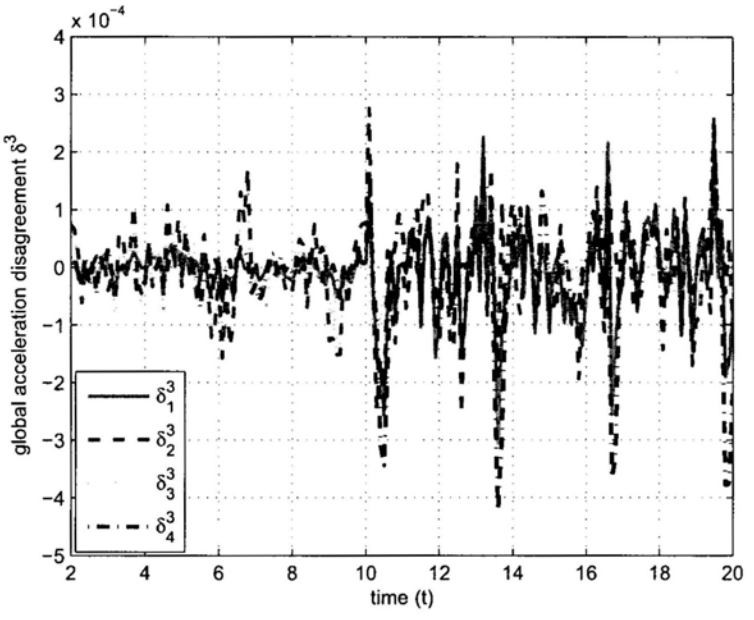

Figure 5.34: Profiles of the global acceleration disagreement vector δ^3 for $t = [2, 20]$

5.6 Conclusion

In this chapter, we consider a higher-order (the order is greater than 2) synchronization problem with unknown nonlinear dynamics and unknown disturbances, which is not investigated in the literature, to the best of the author's knowledge. A robust adaptive control law is proposed and it is totally distributed. The communication graph we investigated has a fixed topology. The future work is to extend our result to a graph with time varying topology, which will be more practical in the sense that the links between nodes may fail or created as they moving, considering the distance constraint of the transmitters and receivers.

□ **End of chapter.**

Chapter 6

Conclusions

In this dissertation, we have investigated two topics of the modern control, i.e. receding horizon control, and cooperative control of networked systems.

For the first topic, inspired by the reinforcement learning scheme, we incorporate a learning feature into the standard receding horizon control algorithm and propose a new receding horizon control algorithm, named updated terminal cost receding horizon control (UTC-RHC). In particularly, we propose UTC-RHC in the framework of discrete-time linear time invariant systems and sampled-data UTC-RHC in the framework of continuous-time linear systems. We show under both cases, the stability conditions required for standard receding horizon control are relaxed; the yielding closed-loop systems are uniformly exponentially stable; and the UTC-RHC control gains ultimately converge to their optimal values corresponding to the infinite horizon optimal control problems.

Since a merit of RHC is to handle the input and / or state constraints, our future work on UTC-RHC will be extending it to nonlinear and / or constrained systems.

For the second topic, we solves a tracking problem of networked higher-order nonlinear systems with an active leader. Each agent is a higher-order nonlinear systems with unknown nonlinear dynamics, and is perturbed by an unknown external disturbance. The leader node itself is also a higher-order non-autonomous nonlinear systems. We apply neural adaptive control techniques to eliminate the effect of the unknown nonlinear dynamics of each agent. Sliding mode control scheme is applied to handle the higher-order of the agents dynamics. Using the control Lyapunov function technique, we design distributed robust neural adaptive controllers such that all nodes synchronize to the leader node with small residual errors.

Our work focuses on the communication graph with fixed topology. Although we show by simulation that our algorithm may work for graph with switching topologies, rigorous proof is need. Our future work will consider the graph with time-varying topologies. Moreover, it will be desired to find a more tighter bound of the synchronization errors.

□ **End of chapter.**

Bibliography

[1] A. Al-Tamimi, F.L. Lewis, and M. Abu-Khalaf. Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, 38(4):943–949, 2008.

[2] S.N. Balakrishnan, J. Ding, and F.L. Lewis. Issues on stability of ADP feedback controller for dynamical systems. *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, 38(4):1–5, 2008.

[3] R.W. Beard, T.W. McLain, M.A. Goodrich, and E.P. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Transactions on Robotics and Automation*, 18(6):911–922, 2002.

[4] D.S. Bernstein. *Matrix mathematics: theory, facts, and formulas with application to linear systems theory*. Princeton University Press, NJ, 2005.

[5] D.P. Bertsekas. Dynamic programming and suboptimal control: a survey from ADP to MPC. Tech Rep. 2632, Lab. for Information and Decision Systems, MIT, 2005.

[6] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

[7] R.R. Bitmead, M. Gever, and V. Wertz. *Adaptive Optimal Control : the Thinking Man'S GPC*. Prentice Hall, NY, 1990.

[8] R.R. Bitmead and M. Gevers. Riccati difference and differential equations: convergence, monotonicity and stability. In S. Bittani, A.J. Laub, and J.C. Willems, editors, *The Riccati Equation*, pages 263–292. Springer-Verlag, NY, 1991.

[9] R.R. Bitmead, M. Gevers, I.R. Petersen, and R.J. Kaye. Monotonicity and stabilizability properties of solutions of the Riccati difference equation: propositions, lemmas, theorems, fallacious conjectures and counterexamples. *Systems and Control Letters*, 5(5):309–315, 1985.

[10] C.T. Chen. *Linear System Theory And Design*. Holt, Rinehart and Winston, NY, 1984.

[11] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[12] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[13] J.A. Fax and R.M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

[14] R. Findeisen, T. Raff, and F. Allgöwer. Sampled-data nonlinear model predictive control for constrained continuous time systems. In G. Garcia S. Tarbouriech and A. H. Glattfelder, editors, *Advanced strategies in control systems with input and output constraints*, volume 346 of *Lecture Notes in Control and Information Sciences*, pages 207–235. Springer-Verlag, Berlin, 2007.

[15] F. Fontes. A general framework to design stabilizing nonlinear model predictive controllers. *Systems and Control Letters*, 42(2):127–143, 2001.

[16] S.S. Ge and C. Wang. Adaptive neural control of uncertain MIMO nonlinear systems. *IEEE Transactions on Neural Networks*, 15(3):674–692, 2004.

[17] G. Grimm, M.J. Messina, S.E. Tuna, and A.R. Teel. Model predictive control: for want of a local control Lyapunov function, all is not lost. *IEEE Transactions on Automatic Control*, 50(5):546–558, 2005.

[18] J. Han, M. Li, and L. Guo. Soft control on collective behavior of a group of autonomous agents by a shill agent. *Journal of Systems Science and Complexity*, 19(1):54–62, 2006.

[19] G.A. Hewer. An iterative technique for the computation of the steady state gains for the discrete optimal regulator. *IEEE Transactions on Automatic Control*, 16(4):382–384, 1971.

[20] R.A. Horn and C.R. Johnson. *Matrix analysis*. Cambridge University Press, NY, 1990.

[21] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[22] Z. Hou, L. Cheng, and M. Tan. Decentralized robust adaptive control for the multiagent system consensus problem using neural networks. *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, 39(3):636–647, 2009.

[23] A. Jadbabaie and J. Hauser. On the stability of receding horizon control with a general terminal cost. *IEEE Transactions on Automatic Control*, 50(5):674–678, 2005.

[24] A. Jadbabaie, J. Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[25] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2001.

[26] F. Jiang, L. Wang, and Y. Jia. Consensus in leaderless networks of high-order-integrator agents. In *Proc. of the American Control Conference*, pages 4458–4463, 2009.

[27] S.S. Keerthi and E.G. Gilbert. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations. *Journal of Optimization Theory and Applications*, 57(2):265–293, 1988.

[28] H.K. Khalil. *Nonlinear Systems*. Prentice Hall, NJ, 3rd edition, 2002.

[29] S. Khoo, L. Xie, and Z. Man. Robust finite-time consensus tracking algorithm for multirobot systems. *IEEE/ASME Transactions on Mechatronics*, 14(2):219–228, 2009.

[30] M. Krstic, I. Kanellakopoulos, and P. Kokotovic. *Nonlinear and Adaptive Control Design*. Wiley, NY, 1995.

[31] W.H. Kwon and S. Han. *Receding Horizon Control*. Springer-Verlag, London, 2005.

[32] W.H. Kwon and K.B. Kim. On stabilizing receding horizon controls for linear continuous time-invariant systems. *IEEE Transactions on Automatic Control*, 45(7):1329–1334, 2000.

[33] W.H. Kwon and K.B. Kim. On stabilizing receding horizon controls for linear continuous time-invariant systems. *IEEE Transactions on Automatic Control*, 45(7):1329–1334, 2000.

[34] W.H. Kwon and A.E. Pearson. A modified quadratic cost problem and feedback stabilization of a linear system. *IEEE Transactions on Automatic Control*, 22(5):838–842, 1977.

[35] W.H. Kwon and A.E. Pearson. On feedback stabilization of time-varying discrete linear systems. *IEEE Transactions on Automatic Control*, 23(3):479–481, 1978.

[36] P. Lancaster and L. Rodman. *Algebraic Riccati Equations*. Oxford University Press, USA, 1995.

[37] J.R. Lawton and R.W. Beard. Synchronized multiple spacecraft rotations. *Automatica*, 38(8):1359–1364, 2002.

[38] J.W. Lee, W.H. Kwon, and J.H. Choi. On stability of constrained receding horizon control with finite terminal weighting matrix. *Automatica*, 34(12):1607–1612, 1998.

[39] G.G. Lendaris. Higher level application of ADP: a next phase for the control field? *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, 38(4):901–912, 2008.

[40] F. L. Lewis, A. Yesildirek, and K. Liu. Multilayer neural net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Networks*, 7(2):388–399, 1996.

[41] F.L. Lewis, S. Jagannathan, and A. Yeşildirek. *Neural Network Control of Robot Manipulators and Nonlinear Systems*. Taylor and Francis, 1998.

[42] F.L. Lewis, D Liu, and G.G. Lendaris. Special issue on adaptive dynamic programming and reinforcement learning in feedback control. *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, 38(4):896–897, 2008.

[43] F.L. Lewis, K. Liu, and A. Yesildirek. Neural net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Networks*, 6(3):703–715, 1995.

[44] F.L. Lewis and V.L. Syrmos. *Optimal Control*. Wiley, NY, 2nd edition, 1995.

[45] X. Li, X. Wang, and G. Chen. Pinning a complex dynamical network to its equilibrium. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 51(10):2074–2087, 2004.

[46] Z. Li, Z. Duan, G. Chen, and L. Huang. Consensus of multiagent systems and synchronization of Complex networks: a unified viewpoint. *IEEE Transactions On Circuits and Systems I-Regular Papers*, 57(1):213–224, 2010.

[47] J. Lu and G. Chen. A time-varying complex dynamical network model and its controlled synchronization criteria. *IEEE Transactions on Automatic Control*, 50(6):841–846, 2005.

[48] E.M. Ma and C.J. Zarowski. On lower bounds for the smallest eigenvalue of a Hermitian positive-definite matrix. *IEEE Transactions on Information Theory*, 41(2):539–540, 1995.

[49] D.Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, 1990.

[50] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: stability and optimality. *Automatica*, 36(6):789–814, 2000.

[51] H. Michalska and D.Q. Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993.

[52] K.L. Moore and D. Lucarelli. Forced and constrained consensus among cooperating agents. In *Proc. of the IEEE Networking, Sensing and Control*, pages 449–454, 2005.

[53] K.S. Narendra. Adaptive control using neural networks. In W. T. Miller, R. S. Sutton, and P. J. Werbos, editors, *Neural Networks for Control*, pages 115–142. Cambridge: MIT Press, 1990.

[54] R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[55] R. Olfati-Saber and R.M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

[56] M.M. Polycarpou. Stable adaptive neural control scheme for nonlinear systems. *IEEE Transactions on Automatic Control*, 41(3):447–451, 1996.

[57] M.M. Polycarpou and P.A. Ioannou. Identification and control of nonlinear systems using neural network models: Design and stability analysis. *Tech Rep. 91-09-01, University of Southern California*, 1991.

[58] W.B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality.* Wiley-Interscience, NJ, 2007.

[59] J.A. Primbs and V. Nevistić. Feasibility and stability of constrained finite receding horizon control. *Automatica*, 36(7):965–971, 2000.

[60] J.A. Primbs, V. Nevistić, and J. Doyle. Nonlinear optimal control: a control Lyapunov function and receding horizon perspective. *Asian Journal of Control*, 1(1):14–24, 1999.

[61] S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003.

[62] Z. Qu. *Cooperative Control of Dynamical Systems: Applications to Autonomous Vehicles.* Springer, London, 2009.

[63] Z. Quan, S. Han, and W.H. Kwon. Stability-guaranteed horizon size for receding horizon control. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E90-A(2):523–525, 2007.

[64] W. Ren and R.W. Beard. Trajectory tracking for unmanned air vehicles with velocity and heading rate constraints. *IEEE Transactions on Control Systems Technology*, 12(5):706–716, 2004.

[65] W. Ren and R.W. Beard. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, 50(5):655–661, 2005.

[66] W. Ren, R.W. Beard, and E.M. Atkins. A survey of consensus problems in multi-agent coordination. In *Proc. of the American Control Conference*, pages 1859–1864, 2005.

[67] W. Ren, R.W. Beard, and E.M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, 2007.

[68] W. Ren, K.L. Moore, and Y. Chen. High-order and model reference consensus algorithms in cooperative control of multivehicle systems. *Journal of Dynamic Systems, Measurement, and Control*, 129(5):678–688, 2007.

[69] J. Rinzel. A formal classification of bursting mechanisms in excitable systems. In Teramoto E. and Yamaguti M., editors, *Mathematical Topics in Population Biology, Morphogenesis and Neurosciences*, pages 267–281. Springer, 1987.

[70] W.J. Rugh. *Linear System Theory*. Prentice Hall, NJ, 2nd edition, 1996.

[71] J.J.E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, NJ, 1991.

[72] M.H. Stone. The generalized Weierstrass approximation theorem. *Mathematics Magazine*, 21(4/5):167–184 / 237–254, 1948.

[73] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[74] O. Taussky. A recurring theorem on determinants. *The American Mathematical Monthly*, 56(10):672–676, 1949.

[75] J.N. Tsitsiklis. *Problems in decentralized decision making and computation*. PhD thesis, M. I. T., Dept. of Electrical Engineering and Computer Science, 1984.

[76] J.N. Tsitsiklis, D.P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

[77] R.S. Varga. *Matrix Iterative Analysis*. Springer, NY, 2000.

[78] D. Wang and J. Huang. Neural network-based adaptive dynamic surface control for a class of uncertain nonlinear systems in strict-feedback form. *IEEE Transactions on Neural Networks*, (1):195–202, 2005.

[79] F.Y. Wang, H. Zhang, and D. Liu. Adaptive dynamic programming: an introduction. *IEEE Computational Intelligence Magazine*, 4(2):39–47, 2009.

[80] J. Wang and D. Cheng. Consensus of multi-agent systems with higher order dynamics. In *Proc. of the 26th Chinese Control Conference*, pages 761–765, 2007.

[81] X. Wang and G. Chen. Pinning control of scale-free dynamical networks. *Physica A: Statistical Mechanics and its Applications*, 310(3-4):521–531, 2002.

[82] C. Watkins. *Learning from delayed rewards*. PhD thesis, Cambridge University, Cambridge, England, 1989.

[83] P.J. Werbos. A menu of designs for reinforcement learning over time. In W.T. Miller, R.S. Sutton, and P.J. Werbos, editors, *Neural Networks for Control*, pages 67–95. MIT Press, Cambridge, MA, 1990.

[84] P.J. Werbos. Approximate dynamic programming for real-time control and neural modeling. In D.A. White and D.A. Sofge, editors, *Handbook of Intelligent Control*, pages 493–525. Van Nostrand Reinhold, NY, 1992.

[85] P.J Werbos. Foreword ADP: the key direction for future research in intelligent control and understanding brain intelligence. *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, 38(4):898–900, 2008.

[86] P. Wieland, J.S. Kim, H. Scheu, and F. Allgöwer. On consensus in multi-agent systems with linear high-order agents. In *Proc. of the 17th IFAC world congress*, pages 1541–1546, 2008.

[87] R.L. Williams and D.A. Lawrence. *Linear State-Space Control Systems*. Wiley, NJ, 2007.

[88] C.W. Wu. *Synchronization in Complex Networks of Nonlinear Dynamical Systems*. World Scientific, NJ, 2007.

[89] H. Yamaguchi, T. Arai, and G. Beni. A distributed control scheme for multiple robotic vehicles to make group formations. *Robotics and Autonomous systems*, 36(4):125–147, 2001.

[90] W. Yu, G. Chen, M. Cao, and J. Kurths. Second order consensus for multi-agent systems with directed topologies and nonlinear dynamics. *IEEE Transactions on Systems Man and Cybernetics, Part B-Cybernetics*, to appear.

[91] H. Zhang, J. Huang, and F.L. Lewis. Algorithm and stability of ATC receding horizon control. In *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 28–35, 2009.

[92] H. Zhang, J. Huang, and F.L. Lewis. An improved method in receding horizon control with updating of terminal cost function. In Kimon P. Valavanis, editor, *Applications of Intelligent Control to Engineering Systems*, pages 365–393. Springer, 2009.

[93] H. Zhang, J. Huang, and F.L. Lewis. Updated terminal cost RHC for continuous-time systems. In *Proc. of the IEEE Conference on Decision and Control (CDC)*, pages 4056–4061, Shanghai, China, 2009.

[94] H. Zhang and F.L. Lewis. Synchronization of networked higher-order nonlinear systems with unknown dynamics. In *IEEE Conference on Decision and Control, submitted*, 2010.

[95] H. Zhang and F.L. Lewis. Synchronization of nonlinear cooperative systems with unknown dynamics using neural adaptive control. *IEEE Transactions on Neural Networks, submitted*, 2010.

[96] T. Zhang, S.S. Ge, and C.C. Hang. Adaptive neural network control for strict-feedback nonlinear systems using backstepping design. *Automatica*, 36(12):1835–1846, 2000.

Biography

Hongwei Zhang was born in 1979 in He'nan Province, China. He received his Bachelor degree of Engineering (B.E) and Master degree of Engineering (M.E) in 2003 and 2006 respectively, both from the Department of Automation, Tianjin University, China. He has been working towards his Doctor of Philosophy degree in the Department of Mechanical and Automation Engineering, the Chinese University of Hong Kong, since August 2006. His research interests are receding horizon control, approximate dynamic programming, cooperative control, and neural network for control. The work in this dissertation is based on the following publications.

[1] H. Zhang, J. Huang, and F.L. Lewis. Algorithm and stability of ATC receding horizon control. In *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 28-35, 2009.

[2] H. Zhang, J. Huang, and F.L. Lewis. An improved method in receding horizon control with updating of terminal cost function. In Kimon P. Valavanis, editor, *Applications of Intelligent Control to Engineering Systems*, pages 365-393. Springer, 2009.

[3] H. Zhang, J. Huang, and F.L. Lewis. Updated terminal cost RHC for continuous-time systems. In *Proc. of the 48th IEEE Conference on Decision and Control*, pages 4056-4061, Shanghai, China, 2009.

[4] H. Zhang and F.L. Lewis. Synchronization of networked higher-order nonlinear systems with unknown dynamics. *the 49th IEEE Conference on Decision and Control*, submitted, 2010.

[5] H. Zhang and F.L. Lewis. Synchronization of nonlinear cooperative systems with unknown dynamics using neural adaptive control. *IEEE Transactions on Neural Networks*, submitted, 2010.