Repetitive Learning Control for Remote Control Systems

By

Long Sheng

(B.Eng. Beijing University of Technology)

A Thesis Submitted to

Saint Mary's University, Halifax, Nova Scotia

in Partial Fulfillment of the Requirements for

the Degree of Master of Science in Applied Science

July 2006

Approved: Dr.Ya-jun Pan (Supervisor)

Approved: Dr.John Young (Co-Supervisor)

Approved: Dr.Craig Lake (Examiner)

Approved: Dr.Dale Retallack (Examiner)

Founded 1802

Your file Votre référence
ISBN: 978-0-494-23860-8
Our file Notre référence
ISBN: 978-0-494-23860-8

Acknowledgments

First I'd like to thank my supervisor, Dr.Ya-jun Pan. Without her excellent guidance, great help and strong patience, I cannot imagine how I could have finished this work. She is my guide who shows me the way to the control kingdom which attracts me very much. Her help is not only useful for my work but also for my whole life.

Thanks should be given to Dr.David Richardson for his great help encouraging me to finish this work. Thanks also to Dr.John Young and Dr.Craig Lake, for helping me finish my study at SMU. Their comments and suggestions have improved the final version of the thesis considerably. I also want to thank Dr.Dale Retallack, who spends his time to check and approve my thesis.

Many people have contributed in one way or another to help me finish my work. I'd like to mention Dr.Ken Wilkie, who tried his best to help me finish my course and begin my advanced study at Dalhousie University. I want to thank Ms.Susan Dore and Dr.Kevin Vessey for helping me finish my work.

I would also like to thank my parents. Their endless love and support have brought me where I am today.

Finally, I would like to thank Xiaocan Zhang, her love, support and encouragement have given me the energy to finish this work.

Long Sheng

Contents

Abstract

Repetitive Learning Control for Remote Control Systems

By Long Sheng

In this thesis, a Repetitive Learning Control (RLC) approach is proposed for a class of remote control nonlinear systems satisfying the global Lipschitz condition. The proposed approach is to deal with the remote tracking control problem when the environment is periodic over the infinite time domain. Since there exists a time delay, tracking a desired trajectory through a remote controller is not an easy task. A predictor is designed on the controller side to predict the future state of the nonlinear system based on the delayed measurements from the sensor. The convergence of the estimation error of the predictor is ensured. The gain design of the predictor applies linear matrix inequality - LMI techniques. The repetitive learning control law is designed based on the feedback error from the predicted state. The proof of the stability is based on a constructed Lyapunov function. By incorporating the predictor and the RLC controller, the system state tracks the desired trajectory independently of the influence of time delays. A numerical simulation example is shown to illustrate the effectiveness of the proposed approach.

July 23, 2006.

List of Figures

Chapter 1

Introduction

1.1 Brief Background

Iterative learning control (ILC) is a relatively new technique for improving tracking response in systems that repeat a given task over and over again. A systematic design provided by ILC can improve tracking performance by iterations (each repetition sometimes being called a trial) in a fixed time interval. A diagram for ILC appears below in Fig.1.1.

As shown in Fig.1.1, the learning controller calculates the input value for the current trial based on information from the previous cycle. We call this process "a learning process".

Another way to improve the tracking performance from trial to trial is called repetitive learning control (RLC). RLC and ILC are similar in nature. However, the

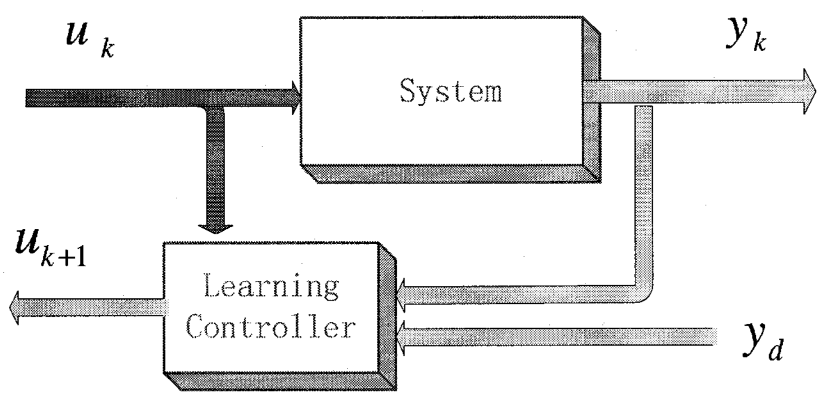

Figure 1.1. Block diagram of the ILC controlled system

difference is that ILC needs an initialization, i.e. the system should be started with the same initial condition at the beginning of each repetition, while for RLC the initial condition of current repetition is set to the terminal condition of the previous trial. Before going in to a more technical discussion of ILC, the background of ILC is provided including a brief history and an overview of possible connections with other areas in the control field.

In recent years, extensive research on Networked Control Systems (NCSs) has been under taken, due to the emergence of the field of communication. The basic definition of a network is that it comprises an interconnection of three or more communicating entities. A typical diagram of a network is shown in Fig.1.2

Compared with conventional systems, NCS has its own advantage, such as low cost, reduced weight, system wiring and power requirements, simple installation, simple system diagnosis and maintenance, and higher reliability (Zhang *et al.*, 2001). A diagram of an NCS is given in Fig.1.3.

The control loops in the NCS are closed through a real-time communication channel

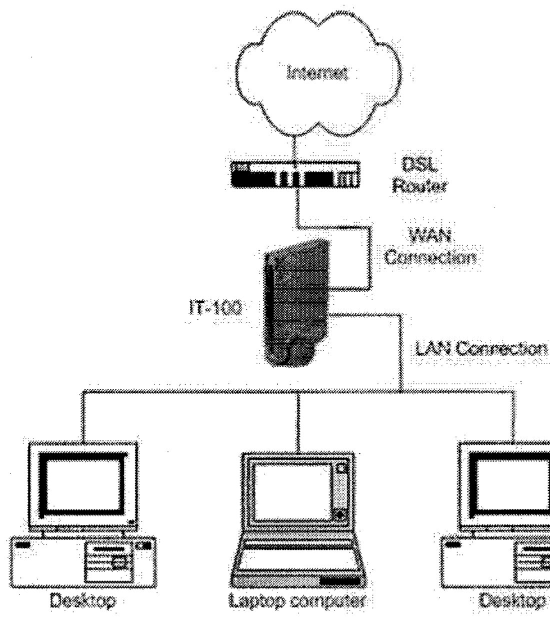

Figure 1.2. Diagram of the network

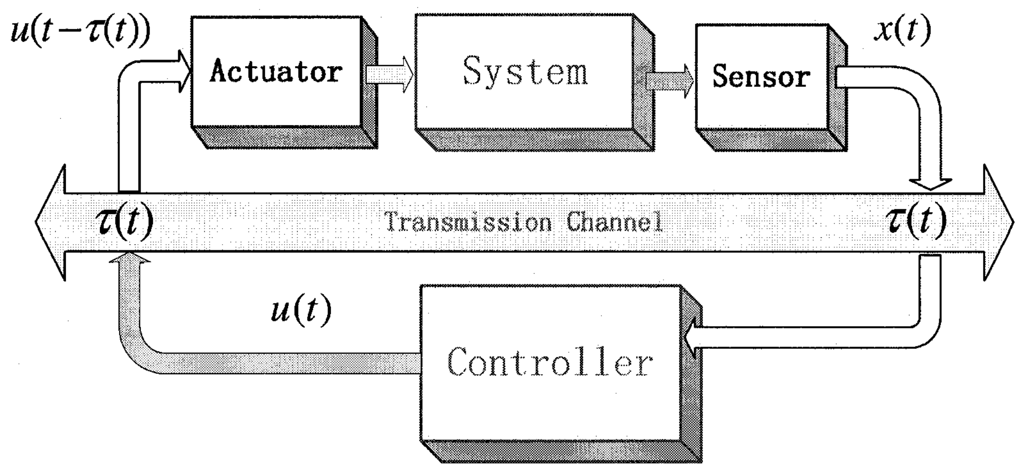

Figure 1.3. Block diagram of the NCS

which transmits signals from the sensors to the controller and from the controller to the actuator (Pan *et al.*, 2004). However, an important issue occurs in NCSs which can make the analysis and control design more complicated than for classical feedback loops. This is the network-induced delay, which is composed of sensor-to-controller delay and controller-to-actuator delay. The network-induced time delay normally occurs while exchanging data among devices connected to the shared medium. Such delays may degrade the performance or even destabilize a control system designed without considering the effects caused by the delay (Lozano *et al.*, 2004) (Wu *et al.*, 2004) (Yue *et al.*, 2005).

In this thesis, a Repetitive Learning Control (RLC) approach is proposed for a class of remote control nonlinear systems satisfying the global Lipschitz condition. The proposed approach is to deal with the remote tracking control problem when the environment is periodic or repeatable over an infinite time domain. Since there exists time delays in two data transmission channels called controller to actuator channel and sensor to controller channel, which could make the whole control system unstable, tracking a desired trajectory through a remote controller is not an easy task. A predictor is designed to solve the problem caused by the time delay. Simulation results indicate that good performance has been achieved.

1.2 Thesis Outline

The thesis is divided into seven Chapters; 1. Introduction, 2. Background of ILC and RLC, 3. Learning Control for Network Related Application, 4. Predictor Design, 5. Repetitive Learning Controller Design, 6. Simulation Results, 7. Conclusions and future work.

Chapter 1 gives general ideas about ILC, RLC, NCS and the main work of this thesis. Chapter 2 gives a general introduction to ILC and RLC including the history and an example showing the applicability of ILC. In Chapter 3, the problem formulation and control design objective of RLC are discussed. The foundational RLC algorithm also has been provided. Another important aspect of this chapter concerns Networks. The background information is presented, then the protocols of Networks are introduced. Network delay and packet lost are also discussed. In Chapter 4, the predictor is designed for the nonlinear system, using LMI techniques. Chapter 5 first presents the error dynamics, and then develops a repetitive-learning-based algorithm; a Lyapunov-based stability analysis is utilized to prove the globally asymptotic tracking result. In Chapter 6, simulation results demonstrate the effectiveness of the proposed repetitive learning algorithm for an example remote control nonlinear system.

1.3 Contributions

In this thesis, for a class of nonlinear systems controlled remotely, a repetitive learning control approach is proposed. This approach is intended to deal with control problems when the environment is periodic or repeatable. The finite-time tracking problem can be solved without having to reposition the system at the beginning of each cycle. The nonlinear system satisfies the global Lipschitz condition. Due to the existence of time delays in the signal transmissions of both channels, the conventional RLC without any delay compensation does not work for the tracking problem. A predictor is then designed to facilitate the RLC by predicting the future state of the nonlinear system based on the delayed measurements. Linear matrix inequality (LMI) techniques and the Lyapunov method (Pan *et al.*, 2006*b*) are used for the predictor design. In the presence of time delays, the system state tracks the desired trajectory asymptotically. The main contributions of this work fall in the following aspects: i) the proposed learning-based controller utilizes a simple modification of the standard repetitive update law to realize the tracking control tasks in the periodic environment; ii) a predictor is designed and well incorporated on the controller side, so that the effects of time-delays in both channels can be eliminated by predicting the future state of the system; iii) the Lyapunov Kravoskii functional approach and LMI techniques are utilized to ensure the convergence of the estimation error; iv) a Constructive Lyapunov Functional is applied to guarantee the convergence of the tracking error, and v) zero tracking error can be achieved asymptotically in the existence of communication delays.

Chapter 2

Iterative Learning Control and Repetitive Learning Control

2.1 History of Learning Control

The idea of using an iterative method to compensate for a repetitive error was suggested first in the late $70's$. Machines, such as robotic arms in product lines, were invented to do the same tasks repeatedly. Some researchers found that using knowledge from previous iterations of the same tasks could effectively reduce the error the next time the same task was performed. In the ILC community it is now widely accepted that (Uchiyama, 1978) first introduced the ILC concept. However, because this publication was written in Japanese only, non-Japanese researchers were not aware of this publication when ILC research initially started in the USA and Western Europe. It is remarkable as well that a US patent on "Learning control of actuators in control systems" was granted earlier (Garden, 1971) based on work done in 1967 and accepted in 1971. The main idea of his work is to store a "command signal" in a computer memory and iteratively update the command signal using the error between the actual response and the desired response of the actuator. This is clearly an implementation of ILC, although the actual ILC updating equation is not explicitly formulated in the patent.

From an academic perspective, ILC did not start to become an active research area until 1984. In 1984 (Arimoto *et al.*, 1984*a*), (Casalino and Bartolini, n.d.) and (Craig, 1984), respectively published papers to expound a method that could iteratively compensate for mode errors and disturbances by using the tracking error between the actual and desired system outputs. The name Iterative Learning Control was first introduced in (Arimoto *et al.*, 1984*b*).

The development of ILC stems originally from the robotics area, where repetitive motions show up naturally in many applications. Examples of contributions where ILC is applied in robotics are (Arimoto *et al.*, 1984*a*), (Casalino and Bartolini, n.d.), (Arimoto *et al.*, 1985), (Bondi *et al.*, 1988), (Poloni and Ulivi, 1991), (Horowitz *et al.*, 1991), (Horowitz, 1993), (Guglielmo and Sadegh, 1996), (Burdet *et al.*, 1997), (Jiang *et al.*, 1999) and (Lange and Hirzinger, 1999).

Examples of surveys on ILC are (Horowitz, 1993), (Moore, 1998), and (Bien and Xu, n.d.). (Moore, 1998) contains a very good overview of ILC research.

In the late 1990′s and at the beginning of the 2000′s, the focus for ILC research moved from being very focused on stability towards also considering design and performance. Examples in this direction are (Bien and Xu, n.d.), (Lee *et al.*, 2000), and (Longman, 2000).

The classic formulation of the ILC problem uses an iterative procedure to find the input for a given system such that the output follows a given desired trajectory as accurately as possible. It is clear that if a description of the system is available, the optimal solution is to invert the description and use this to calculate the input that produces the desired output. This is a one-step procedure, which can be considered as a feed-forward control scheme. If the system representation, describing the mapping from input to output, is not completely known, then it is obvious that the inverse dynamics approach will never achieve perfect tracking. If it is assumed that the structure of the system is known, but the exact value of one or more of the parameters are unknown, adaptive control, which is another well known technique, might be applied. The adaptive control approach is very good since it will, theoretically, provide good performance for all input signals, during all working conditions.

When a particular reference trajectory and a system are given, iterative learning control can be applied as an alternative to the inverse dynamics and the adaptive control approaches. The input signal can be calculated by an iterative procedure, such that the output follows the desired reference trajectory as well as possible. This can be seen as an iterative search procedure which obviously has to converge to give a successful result. Convergence, or stability as it will be referred to in this thesis, is an important research field for ILC. Recently, transient behavior and the design of ILC schemes that give a desired transient behavior have been focused on more and more. This means that practical aspects such as convergence speed and robust performance become more and more well-understood.

2.1.1 Fundamentals of Iterative Learning Control

Before we discuss the fundamentals of Iterative Learning Control (ILC), the classical feedback control will be reviewed. Also, the major differences between classical feedback control and ILC will be discussed. As a starting point in classical feedback control, a model which describes the dynamical behaviour of a given system is given as following:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), x(0) = x_0 \\ \dot{y}(t) &= Cx(t) + Du(t), \end{aligned} \tag{2.1}$$

where A,B,C and D are matrices of appropriate dimensions, $u(t)$ is the input variable, $x(t)$ is the state variable, x_0 is the initial state, $y(t)$ is the output variable. We assume that $D = 0$ because it is extremely rare in physical systems that the input u(t) would affect directly and instantaneously the output $y(t)$.

The design of a controller for the dynamical system (2.1) is typically divided into two different design problems. *i*) The first design problem is to find a control law that manipulates the input variable $u(t)$ so that the system automatically holds the output $y(t)$ at a constant value, even when unknown disturbances try to move $y(t)$ away from this constant set point. *ii)* The second design problem is a tracking problem; the objective is to make the output $y(t)$ follow a given reference signal $y_d(t)$ by manipulating the input variable $u(t)$. The designs for these two types of control systems have been accomplished successfully in both classical and modern control theory by using feedback control: the idea is to measure the output $y(t)$ of the system, and based on the difference between the reference signal $y_d(t)$ and the output signal $y(t)$, the control input $u(t)$ is changed according to some given rule so that the difference between $y_d(t)$ and $y(t)$ is reduced. The crucial point in the design is then to find a control algorithm that will keep the tracking error $e(t) = y_d(t) - y(t)$ as small as possible. In order to solve this design problem, a lot of work has been done. Nowadays there exist many different algorithms, such as PID-control, adaptive control and robust control. These design methods have been used with great success in practical applications, including oil refineries, jumbo jets and washing machines, which clearly demonstrates the importance of feedback control in a modern society.

Iterative Learning Control (ILC) has its own special problem definition; the control system design of ILC is more flexible than feedback control systems. In ILC the dynamical model is exactly the same as in (2.1), but the system (2.1) is defined only over a finite time-interval $t \in [0, T]$. Furthermore, a desired trajectory $y_d(t)$ is given and the system (2.1) has to track this trajectory as accurately as possible, so that this problem can be treated as a tracking problem over a finite time-interval.

The differences between ILC and standard feedback control are these: when the system (2.1) has reached the final time $t = T$, the final state x(T) of the system in (2.1) is reset to the original x_0, after which the system is supposed to track the same reference trajectory signal $y_d(t)$.

An illustrative example which presents the ILC control problem is a welding robot arm in car manufacturing. The task for the robot manipulator involves following a given geometric trajectory and welding at specific points along the trajectory. After the robot has finished welding the first car, the robot is reset to the starting point of the trajectory and a new car of exactly of the same dimensions as the previous car is delivered for welding. The robot carries out the same trajectory tracking and welding task.

In the past, the control scheme for this kind of robot was set up once, at the every beginning. This meant that the control action $u(t)$ was once only, in the form of a fixed feedback control, which resulted in a control action $u(t) = u_{fix}(t)$. The problem, however, is that the controller will produce the same input $u_{fix}(t)$ during every iteration, so that if the corresponding output function $y_{fix}(t)$ is not equal to $y_d(t)$ for each $t \in [0, T]$, the resulting nonzero tracking error $e_{fix}(t)$ is repeated during each iteration. It was suggested in (Arimoto *et al.*, 1984*a*) that one could use the information from the previous iterations to come up with a new input function u_k, where k is the iteration number, so that the tracking error will go to zero as the number of iterations increased. In summary the experience from previous iterations or repetitions is used such that the ILC system will gradually learn the control action that will result in perfect tracking performance. Therefore in the robot example the robot manipulator would learn the control action by itself that gives perfect tracking performance, resulting in an autonomous system capable of manufacturing high quality products.

2.1.2 Fundamentals of Repetitive Learning Control

In Repetitive Learning Control (RLC), the starting point is also the plant model in (2.1), which is defined, as in standard feedback control, over the infinite-time interval $t \in [0, \infty)$. Furthermore, the system output $y(t)$ is supposed to track a T-periodic reference signal $y_d(t)$, i.e. $y_d(t) = y_d(t+T)$, other information is assumed to be not available for control algorithm design. There are a lot of important applications of RLC, which can be found in robotics (Kaneko and Horowitz, 1997), motors (Kobayashi *et al.*, 1999), hard-disc control (Smith *et al.*, 1999),rotating mechanisms (Fung *et al.*, 2000) and PWM converters (Zhou and Wang, 2001). Repetitive control has also been applied to active vibration and noise cancellation problems, which is a very active research topic in the control community.

Actually the RLC problem setting is very similar to the ILC case, the only difference being that the ILC needs an initialization , i.e., the system should be started with the same initial condition at the beginning of each repetition, while RLC is supposed to track the periodical reference trajectory, i.e., the initial condition of current repetition is set to the terminal condition of the previous repetition. As the reference signal or the desired trajectory is periodic, which means the reference signal is the same for each period, one can use information from previous periods to modify the input $u(t)$ so that eventually the system will learn the input signal that gives the desired periodic behavior.

2.2 A Formal Definition of Iterative Learning Control

In order to give a precise mathematical definition of the ILC problem, we first give the following standard continuous time-varying linear state-space model defined over a finite time domain $t \in [0, T]$:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), x(0) = x_0 \\ \dot{y}(t) &= Cx(t), \end{aligned} \tag{2.2}$$

where $x(t) \in R^n$ is the system state, $y(t) \in R^m$ is the system output, $u(t) \in R^m$ is the system input. The operators A, B and C are matrices of appropriate dimensions. In order to avoid technical difficulties in analysis, it is typically assumed that matrices are continuous with respect to time t. Furthermore, a reference signal $y_d(t)$ is given and the task is to construct a control law which can decide the input $u(t)$ so that the output $y(t)$ would track $y_d(t)$ as accurately as possible. The same as we discuss before, the system in (2.2) is supposed to follow the reference signal in a repetitive form, i.e. after the system has reached the final time point $t = T$, the state of the system is reset to the initial condition x_0 and the system is supposed to track the same reference signal $y_d(t)$ again. Assuming that $u_k(t)$ is the input applied at trial $k \in N$ and $e_k(t) = y_d(t) - y_k(t)$ is the resulting tracking error, a control law can be constructed as follows:

$$u_{k+1}(t) = f(e_{k+1}(\cdot), e_k(\cdot), \ldots, e_{k-s}(\cdot), u_k(\cdot), u_{k-1}(\cdot), \ldots, u_{k-r}(\cdot)), \tag{2.3}$$

so that $\lim_{k \longrightarrow \infty} u_k = \tilde{u}$ and $\lim_{k \longrightarrow \infty} e_k = 0$ in a suitable topology. In addition, it is required that $u_{k+1}(t)$ is a function of $e_{k+1}(s)$ for $s \leq t$. Note that in the problem definition it is assumed that there exists an input u^* which gives perfect tracking. If this is not the case, the problem can be modified in the following manner: the algorithm should converge to a fixed point u^* where u^* is the solution of the problem

$$u^* = arg \min_{u \in \mu} \|y_d - Gu\|^2. \tag{2.4}$$

where μ is set of possible inputs, G is the transform of the system model in the input-output form and $\| \cdot \|$ is a suitable norm.

Selecting a suitable norm space is important for the convergence analysis. Convergence is naturally the most important requirement for an ILC algorithm. However, additional requirements also have been suggested, the most common ones are *i)* Convergence should be achieved with a minimal amount of information about the plant; *i)* Convergence should be achieved even if there is uncertainty in the plant model. *iii)* Convergence should be achieved even if the resetting is not perfect. Note that the first additional requirement is not always sensible. This is due to the fact that for example in robotics either accurate models are available from the robot manufacturer, or they can be obtained rather easily by using modern identification techniques, and it would be unwise to discard this information about the plant model in the ILC algorithm design.

2.3 Linear ILC updating law

In this section some different approaches to updating the signal $u_k(t)$ in the linear ILC algorithms will be discussed. The class of linear ILC updating formulas can be categorized in two groups according to how the information from previous iterations is utilized. The two groups are: First order ILC and High order ILC algorithms.

First order ILC

An ILC updating formula that only uses measurements from the previous iteration is called a first order ILC. Several first order ILC algorithms have been suggested in literature. The most common of the suggested algorithms, e.g., (Arimoto *et al.*, 1984*a*), (Hara *et al.*, 1988), (Bien and Xu, n.d.), are given by

$$u_{k+1}(t) = Q(q)(u_k(t) + R(q)e_k(t)). \tag{2.5}$$

where Q(q) and R(q) are considered to be linear transfer operators or simply discrete filters. Usually the ILC is considered to be implemented in continuous time. The use of the Q-filter is suggested in (Hara *et al.*, 1988) and (Tomizuka *et al.*, 1989). In many of the references, the Q-filter is chosen as a constant equal to 1. An even more general form of the first order ILC updating formula, is given by the following equation:

$$u_{k+1} = Q_k(u_k + R_k e_k). \tag{2.6}$$

where the matrices Q_k and R_k can be realizations of iteration as well as time variable filters.

High order ILC

When the ILC updating formula uses measurements from more than the previous iteration it is called a high order ILC. Although most contributions on ILC have been on the first order case, the idea of utilizing the measurements from more than the previous iteration has been covered in some articles. In (Liang and Looze, 1993) two dimensional transforms are used to analyze the behavior of the system in both the time and the iteration directions. In the paper by (Arimoto, 1991) the errors from previous iterations are used in an indirect way. (Chen *et al.*, 1998) have also investigated the use of high order ILC.

2.4 Nonlinear ILC

Most of the work in the area of ILC has been done on linear ILC updating formulations. The linear ILC mapping can be a general mapping from the reference signal, the previous measurements, and the previous control signals. In this very general framework not so many results are available. There are, however, some results in the survey on ILC by (Moore, 1993) and in a recent book edited by (Bien and Xu, n.d.). Moore used a chapter in his paper to discuss the use of Artificial Neural Networks (ANN) in ILC. This can be seen as a kind of nonlinear black-box identification approach, in this approach not only the control signal changes over the iterations but the ILC algorithm changes as well. Another possible approach that leads to an overall nonlinear ILC combines a system identification and model based design procedure for the ILC algorithm. This is discussed in (Norrlof, 2000).

2.5 A Formal Definition of Repetitive Learning Control

Before we discuss the formal definition of RLC, as offered previously for the ILC, a linear time-invariant continuous-time model is given

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), x(0) = x_0 \\ y(t) &= Cx(t) \end{aligned} \tag{2.7}$$

In this equation the state $x(\cdot) \in R^n$, output $y(\cdot) \in R^m$, input $u(\cdot) \in R^m$. Both system are defined over an infinite-time interval $t \in [0, \infty)$. A, B and C are matrices of appropriate dimensions. Nonlinear models could be also considered, however, linear model can keep the analysis fairly simple, here linear model has been used for the reason above.

Here the control design problem of RLC is to design a feedback controller so that output of the system in (2.7) would track a T-periodic reference signal $y_d(t)$, i.e.,

$y_d(t) = y_d(t+T)$, so that

$$\lim_{t \to \infty} e(t) = 0. \tag{2.8}$$

where $e(t) = y_d(t) - y(t)$. In addition, in the RLC control law it is possible to use the information from previous periods, i.e., the RLC algorithm is given in the following form:

$$u(t) = f(u(t-T), u(t-2T), \ldots, u(t-mT), e(t), e(t-T), \ldots, e(t-nT)) \tag{2.9}$$

Additional requirements could be that: *i)* Convergence should be achieved even if there is uncertainty in T. *ii)* Convergence should be achieved in the presence of model uncertainty in (2.7).

It is noted that sometimes, especially in servo systems, it is very common that the reference signal (or called desired trajectory) is not periodic with respect to time t, but, rather, with respect to the angular position of the servo system. e.g., it is shown in (Mahawan and Luo, 2000), under suitable assumptions this problem also can be solved with modified RLC, where the independent variable is no longer time t but the angular position of the servo system.

2.6 Convergence Analysis for Iterative Process

As we discussed in the previous section, the basic idea of ILC and RLC is to use the repetitive nature of the problem definition to make the system learn the input function that results in perfect tracking. During this process, a new axis is introduced: namely the iteration axis k. This results in two-dimensional system, where the independent variables are the finite time axis $t \in [0, T]$ and the infinite iteration axis $k \in N$. As a first observation towards convergence and stability analysis, note that due to the finite nature of the time axis, the output of a finite-dimensional linear time-varying system can not become unbounded in finite time. Hence, compared with the classical feedback control, the properties of the ILC system along the time-axis do not play a major role in convergence analysis. On the other hand the iteration axis is infinite. Therefore different with the case in the finite time axis, the output of a finite-dimensional linear time-varying system can either converge or diverge, depending on the chosen learning mechanism. In order to show how can the convergence or stability be approached mathematically, the following example are given:

Consider the following ILC algorithm

$$u_{k+1}(t) = u_k(t) + Ke_k(t), \tag{2.10}$$

where K is a learning gain, $t \in [0, T]$ and the input-output plant model is given as follows:

$$y_{k+1}(t) = Gu_{k+1}(t) + z_0(t). \tag{2.11}$$

In order to analyze the convergence properties of the algorithm, it is necessary to find how the tracking error $e_k(t) = y_d(t) - y_k(t)$ evolves as a function of the iteration round k. In order to find this evolution equation, substitute the control algorithm (2.10) into the plant model (2.11), we have:

$$y_{k+1}(t) = Gu_{k+1}(t) + z_0(t) = Gu_k(t) + z_0(t) + GKe_k(t). \tag{2.12}$$

Then we multiply (2.12) with -1 and after that add $y_d(t)$ on both sides of the equation. This results in

$$y_d(t) - Gu_{k+1}(t) - z_0(t) = y_d(t) - Gu_k(t) - z_0(t) - GKe_k(t). \tag{2.13}$$

Using the process model (2.11) and the definition of the tracking error $e_k(t)$ this equation can be written equivalently as:

$$e_{k+1}(t) = (I - GK)e_k(t). \tag{2.14}$$

or in more compact form: $e_{k+1}(t) = Le_k(t)$, where $L = (I - GK)$. Hence L is the operator that maps $e_k(\cdot)$ to $e_{k+1}(\cdot)$, and thus it is assumed that its mathematical properties somehow define whether or not the algorithm converges. If the operator is designed so that $\|I - GK\| < 1$.

Many of the learning control schemes in the literature require a condition of this from to achieve convergence with zero error. The solution will make the norm of previous error is smaller than that of the final error. In this example, L is the learning operator that maps the tracking error from the previous trial to the current trial. In fact most of the existing ILC algorithms in the literature result in the general error evolution equation:

$$e_{k+1}(t) = Le_k(t), k = 0, 1, \ldots \tag{2.15}$$

It is important to analyze the conditions under which this kind of iterative process converges. Two different conditions are given for convergence in following sections. The first condition is a norm or a contraction mapping condition for the learning operator L, which guarantees that the tracking error converges to zero in the infinite time domain. Furthermore, if this norm condition is met, the tracking error asymptotically decreases as the number of iterations increases. This is sometimes very important in practical applications. The second one is called Lyapunov-based analysis. Lyapunov direct method has been used to analysis the stability of the system. It involves two steps: find a suitable scalar function, called a Lyapunov function, and then evaluate the property of its first-order time derivative along the trajectory of the system. The basic approach is to choose an energy-like function, which is mathematically defined as a positive-difinite function, such that the defined energy keeps dissipating which is mathematically reformulated as the negative property on the time derivative of the energy-like function. This reasoning is intuitively straightforward, and the method is applicable to all linear and nonlinear systems, known or uncertain.

2.6.1 Contraction mapping condition

Let x_0 be an arbitrary element of a norm space χ with a metric d and let T be an operator T. Consider now the iteration

$$x_{k+1} = Tx_k, k = 0, 1, 2, \ldots \tag{2.16}$$

The sequence x_k will converge in the norm to a unique fixed point in χ if the two following conditions hold: *i*) The norm space χ is complete. *ii*) The operator T is a contraction mapping, i.e. there exists $0 < \alpha < 1$ so that

$$d(Tx, Ty) \leq \alpha d(x, y), \forall x, y \in \chi \tag{2.17}$$

The related proof process is standard and can be found from (Pugh 2002). Note that the result is exactly the same for the modified iteration

$$x_{k+1} = Tx_k + b. \tag{2.18}$$

In other words the convergence depends purely on T and the completeness of χ. However, the fixed point where the iteration converges to is different and is given by the equation

$$x_\infty = (I - T)^{-1}b. \tag{2.19}$$

due the uniqueness of $lim_{k \to \infty} x_k = x_\infty$. Because this condition is only a sufficient condition, a violation of these conditions does not necessarily imply that the iteration would diverge.

Now it is assumed that the metric space χ is in fact a complete norm space, i.e. χ is complete and the space is equipped with a norm $||\cdot|| : \chi \rightarrow R_+$ where R_+ is defined to be the set of non-negative real numbers. In this case the metric $d(x, y)$ becomes $d(x, y) = ||x-y||$ for an arbitrary $x, y \in \chi$. In addition the operator T is assumed to be a linear and bounded space, i.e. $T(\alpha x) = \alpha T(x)$ and $T(x_1+x_2) = T(x_1)+T(x_2)$ for linearity, and there exists $M \in R$, $M > 0$ so that for an arbitrary $x \in \chi$ it holds that $||Tx|| \leq M||x||$ for boundedness. In this case we have

$$d(Tx, Ty) = ||Tx - Ty|| = ||T(x - y)|| \leq \alpha||x - y||. \tag{2.20}$$

Furthermore, it is standard result for bounded linear operator that $||Tx|| \leq ||T||||x||$ where $||T||$ is the operator norm. Hence the following estimate holds

$$||Tx - Ty|| = ||T(x - y)|| \leq ||T||||x - y||. \tag{2.21}$$

and comparing this estimate with (2.17) it is clear that if

$$||T|| < 1. \tag{2.22}$$

then the sequence $x_{k+1} = Tx_k$ converges. In the ILC literature it was assumed that the sequence of tracking errors satisfies $e_{k+1} = Le_k$ where L is a again the learning operator. If L is now a contraction mapping, then

$$||e_{k+1}|| = ||Le_k|| \leq ||L||||e_k|| < ||e_k||. \tag{2.23}$$

if $e_k \neq 0$. Based on this estimate a learning operator that is a contraction mapping results in a sequence of tracking errors where the norm of each tracking error is smaller than the norm of the tracking error from the previous iteration. From a more mathematical point of view, it is said that the algorithm results in monotonic convergence, and this is a very desirable property for an ILC algorithm.

2.6.2 Lyapunov-based method

The Lyapunov based method often can be used to analyze the convergence properties of linear and nonlinear systems. A Lyapunov function candidate is called a Lyapunov function for a given system if the time derivative of the candidate along the trajectory of the system has a certain type of dissipative property. The use of energy-related function approaches in ILC, such as Lyapunov functions, is often exploited in the literature. Different stability results can be established, depending on the properties of the time derivative of the Lyapunov function candidate. Here, we state the following fundamental theorem on Lyapunov stability and asymptotic stability.

Theorem 2.1. *Let V be a Lyapunov function candidate as defined in appendix definition (.3) in some neighborhood of the origin denoted by $\Omega \subset R^n$. Suppose the time derivative of V has the property that, for all $(x, t) \in \Omega$,*

$$\dot{v}(x(t), t) \leq -\gamma_3(\|x(t)\|), \tag{2.24}$$

where γ_3 is continuous and nonnegative with $\gamma_0(0) = 0$. Then, the system has the following stability property: i) either globally or locally uniformly Lyapunov stable if $\gamma_3(\|x(t)\|$ is positive semidefinite, ii) either globally or locally uniformly *asymptotically stable if* $\gamma_3(\|x(t)\|$ *is positive definite,* *iii)* *either globally or locally exponentially stable if* $\gamma_3(\|x(t)\| \geq \lambda V(x,t)$ *for some constant* $\lambda > 0$ *or if* $\gamma_i(\|x(t)\| = \lambda_i \|x\|^2 fori = 1,2,3$ *and for positive constants* λ_i, *iv)* *either globally or locally exponentially stable with finite convergence time if* $\gamma_3(\|x(t)\| \geq \lambda V^p(x,t)$ *for constants* $\lambda > 0$ *and* $0 < p < 1$.

The definition of the Lyapunov function candidate and the above fundamental theorem reveal the basics of the Lyapunov direct method. Indeed, looking into the recent advances in control theories and applications, most progress was made in state space with the Lyapunov direct method. It would be very meaningful to look into these control methods, henceforth derive the energy function based ILC (EF-based ILC).By incorporating EF-based ILC, it may be possible to prove the asymptotic eliminate of the tracking error.

The use of Lyapunov based approaches in analyzing dynamic stability has been discussed on many occasions in the literatures. In (Xu and Qu, 1998), the authors utilize a Lyapunov-based approach to illustrate how an ILC can be combined with a variable structure controller to handle a broad class of nonlinear systems. In (Ham *et al.*, 2000), Lyapunov-based techniques are utilized to develop an ILC that is combined with a robust control design to achieve global uniformly ultimately bounded link position tracking for robot manipulators. The applicability of this design was extended to a broader class of nonlinear systems by (Ham *et al.*, 2001). In (Dixon *et al.*, 2002), a learning-based feedforward term is generated from a straightforward Lyapunov-like stability analysis, the control designer can utilize other Lyapunov-based design techniques to develop combined control schemes that utilize learning-based feedforward terms to compensate for periodic dynamics and other Lyapunov-based approaches to compensate for nonperiodic dynamics.

In (Jiang *et al.*, 1995), the authors presents a repetitive learning control scheme and an adaptive repetitive control scheme for a class of nonlinear uncertain systems. The Lyapunov direct method is used to construct a sliding mode and a stabilizing feedback controller for nonlinear uncertain systems where the upper bound of the uncertainties is known. The repetitive controller is designed using the idea of driving the state on to a sliding manifold. Asymptotic stability of an uncertain system under mild assumptions is guaranteed with the proposed repetitive learning control. When the upper bound of the uncertainty is uncertain, an adaptively adjusted gain in the feedback controller ensures uniform boundedness of the system. The performance of this system is enhanced by learning control incorporating a forgetting factor. It is shown that the overall system is uniformly ultimately bounded without the knowledge of the size of modelling uncertainties and input disturbances.

In (Sun and Ge, 2004), the authors consider adaptive RLC for trajectory tracking of servo mechanisms, a special case of robotic manipulators. Lyapunov-like function has been used, through the introduction of this novel Lyapunov-like function, the proposed adaptive learning control only requires the system to start from where it stopped at the last cycle, and avoids the strict requirement for initial repositioning for each new cycles. Good performance of the system was attained, and the iterative trajectories were proven to follow the entire profile of the desired trajectory.

Chapter 3

Learning Control for a Network Related Application

3.1 Problem definition

As was explained in the introduction, as a starting point in continuous-time Repetitive learning control (RLC) it is assumed that a SISO model of the plant exists with $x(0) = x_0$, $t \in [0, \infty)$,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{3.1}$$

Furthermore, A, B, C and D are finite-dimensional matrices of appropriate dimensions. From now on it is assumed that $D = 0$, because in practice it is very rare to find a system where the input function $u(t)$ has an immediate effect on the output variable $y(t)$. Furthermore, a reference signal $y_d(t)$ is given, and it is known that $y_d(t) = y_d(t+T)$ for a given T. The control design objective is to find a feedback controller that makes the system in (3.1) to track the reference signal as accurately as possible, i.e., $\lim_{t\to\infty} e(t) = 0$, $e(t) = y_d(t) - y(t)$, under the assumption that the reference signal $y_d(t)$ is T-periodic. Note that the only difference between RLC and ILC problem focus on resetting: in ILC the state of the system is reset at the end of each period (iteration), whereas in RLC the state at the end of the previous period is the initial condition for the next period. In order to start the analysis of RLC systems, note that in the ILC framework a necessary condition for asymptotic convergence is that a controller

$$Mu(t) = Ne(t), \tag{3.2}$$

where M and N are suitable operators, has to have an initial model or the reference signal inside the operator M. Because $y_d(T)$ is T-periodic in RLC, in (Yamamoto, 1993) it was suggested that one possible RLC algorithm could be

$$u(t) = u(t-T) + e(t). \tag{3.3}$$

3.2 Networks

Communication networks were introduced in digital control systems in the $1970's$. At that time the driving force was the car industry. The motives for introducing communication networks were reduced cost for cabling, modularization of systems, and flexibility in system setup. Since then, several types of communication networks have been developed. Communication protocols can be divided into fieldbuses, e.g., FIP and PROFIBUS, automotive buses, e.g., CAN, machine buses, e.g., 1553B and the IEC train communication network, general purpose networks, e.g.,IEEE LAN's and ATM-LAN and a number of research protocol, e.g., TTP. Fieldbuses are intended for real-time control applications, but in some applications other networks may have to be used for control. For instance, if another network already is used for other functions it could be cost effective to use this network for control too. The fieldbuses are usually only made for connection of low-level devices. If high-level function, for instance, a work station, is to be connected, other networks may be more suitable. There is vast number of communication protocols and fieldbuses.

Foundation Fieldbus

The Foundation Fieldbus was developed by the organization Feildbus Foundation, a not for profit organization with over 100 member companies, including several major international automation companies. Foundation Fieldbus is released for two speeds, 31.25$kbit/s$, and 1$Mbit/s$. A faster bus with bus speed 2.5$Mbit/s$, is announced. The low speed bus, 31.25$Kbit/s$, is intended for replacement of traditional $4-20mA$ analog signals, without chaning the wiring. Each bus can be built. Using a hierarchical network structure more devices can be connected.

Access to the bus is controlled by a centralized bus scheduler called the Link Active Scheduler (LAS). During configuration of the fieldbus all devices on the bus will inform the LAS which data it needs, and at which times the data is needed. During runtime the LAS will tell the devices to broadcast data simultaneously. Spare time is reserved in the schedule for unscheduled messages. A system global clock is also distributed on the fieldbus. The distributed clock will allow connected devices to know the time within 1 ms.

Factory Instrumentation Protocol (FIP)

FIP was developed by a group of French, German and Italian companies. FIP uses a twisted pair conductor and the transmission speeds are from $31.25Kbit/s$ up to $2.5Mbit/s$, depending on the spatial dimension of the bus. For a transmission speed of $1Mbit/s$ the maximum length of the bus is 500 m. The maximum number of nodes in a FIP network is 256.

In a FIP-network one node acts as bus arbitrator. The bus arbitrator cyclically polls all nodes in the network to broadcast its data on the network. The inactive nodes listen to the communication and recognize when data of interest to the node is sent. The FIP-network can be seen as a distributed database, where the database is updated periodically.

Process fieldbus (PROFIBUS)

PROFIBUS was developed by a group of German companies and is now a German standard. A screened twisted pair is used as conductor. The transfer speed can be from $9.6Kbit/s$ to $500Kbit/s$. The maximum length of the bus is 1200 m. Up to 127 stations can be connected to the network. PROFIBUS messages can be up to 256 bytes long. PROFIBUS is a token-passing network. The nodes are divided into active and passive nodes. The node which holds the token has the permission to send data on the network. The token is passed around in the network between the active nodes. Active nodes can transmit when they hold the token. Pasive nodes need to be addressed by an active node to be allowed to send data on the network.

Controller area network (CAN)

CAN was developed by the German company Bosch for the automation industry. CAN was one of the first fieldbuses and is now in use in cars from several manufactures. CAN is defined in the ISO standards 11898 and $11519 - 1$. The transfer speed on the bus can be programmed. The transfer speed can be $1Mbit/s$ if the bus is no longer than 50 m, and $500Kbit/s$ if the bus is longer than 50 m. If the cable quality is low, as it can be in mass produced cars, the maximum transfer speed may be lower. There is no limit on the number of nodes. A node can start transmitting at any time if the bus is silent. If several nodes are trying to transmit, then an arbitration starts. The node trying to send the message with highest priority gets the right to use the bus. CAN-controllers can usually be programmed to cause an interrupt when a message is sent. This feature makes back-propagation of the size of the delay from controller to the actuator.

Ethernet

Ethernet is one of the most used local area network (LAN) technologies. It transmits data with the speeds $10Mbit/s$ or $100Mbit/s$. Ethernet is not intended for real-time communications. However, the large number of installed Ethernets will make it attractive for use in real-time control systems. There is no central bus controller, instead Ethernet uses a bus access method called $CSMA/CD$, which means Carrier Sence Multiple Access with Collision Detection. This means that before sending to the network the station listens to the channel, and when the channel appears to be idle, then transmission starts. If several stations start sending to the bus, the collision is detected, and the colliding stations back off, and try a retransmission after a random wait. An almost unlimited number of stations is limited by the six bytes address. The first three bytes are used as a vendor ID, and the last three bytes are defined by the vendor, so every Ethernet interface has a unique address. An Ethernet frame, or packet, is between 64 and roughly 1500 bytes in length.

3.3 Network Induced Delay and Packet Loss

Network delays have different characteristics depending on the network hardware and software. The simplest mode of the network delay is to model it as being constant for all transfers in the communication network. This can be a good model even if the network has varying delays. Continuous-time network control systems (NCS) with time-delays are infinite dimensional systems. A finite dimensional description of the control loop can be formulated by sampling of the continuous-time process. Let the control system model be

$$\dot{x}(t) = Ax(t) + Bu(t) + v(t), \tag{3.4}$$

where $x(t) \in R^n$, $u(t) \in R^m$ and $v(t) \in R^n$. A and B are matrices of appropriate sizes, $u(t)$ is the controlled input and $v(t)$ is disturbance with zero mean and incremental covariance R_v. The introduction of communication networks makes the analysis and control design more complicated than classical feedback loops. Two main issues occur in NCS. The first is the network-induced delays, called sensor-to-controller delay and controller-to-actuator delay, that occur while exchanging data among devices connected to the shared medium. Such delays, either constant or time varying, may destabilize the system, or degrade the performance of control systems designed without considering the delays. The second is that some packets not only suffer transmission delay but, even worse, can be lost in the transmission channel.

Network Induced Delay

Before we analysis the effect caused by the time delays and design a control law which can be used to reduce the effect and achieve the control objective, the general measure of the time delays should be known first. Fig.3.1 presents the distribution of the time delays when using UDP protocol in transmission. It can be seen that τ takes several values in the interval $[0.15, 0.17]$. Usually the distributions can be explained in two directions: *i*) **Sensor to Controller Delay** When the message is to be sent the bus can be idle or a message can be under transmission.

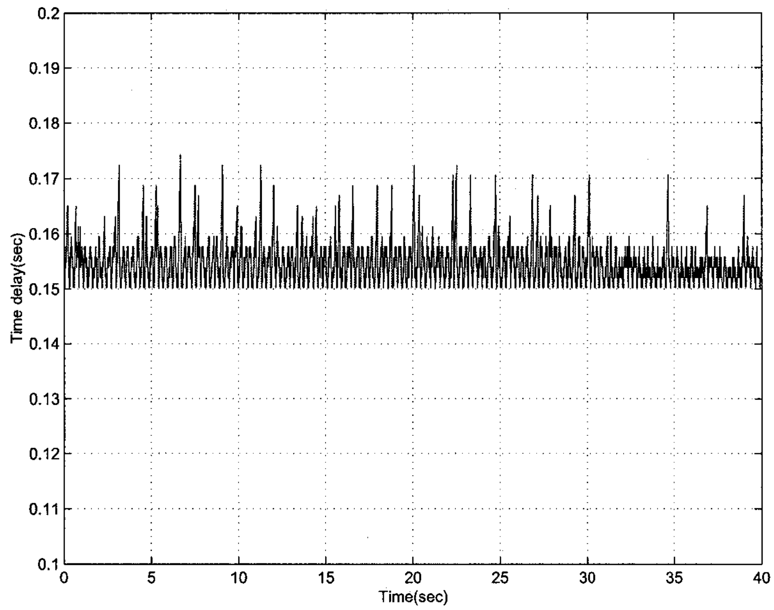

Figure 3.1. The measured time delays when using UDP protocol in transmission

The probability for bus idle depends on the period of the load process. If the bus is busy we will get a nonzero τ. The delay τ will be uniformly distributed from zero to the time it takes to send a message. *ii*) **Controller to Actuator Delay** The delay from controller to actuator can only take two values when we have one load process. The reason for this is that if there were a message waiting when the message was sent from the sensor, the transmission of the waiting message starts before the message to the actuator is ready for transmission. In this case, the delay until the transmission starts will be the time to transmit the load message. If there is no waiting message the message to the actuator will be sent immediately after some computation time in the controller node. For the case when the network induced delay τ is time-invariant and known, i.e., $\tau = \tau_0$ where τ_0 is a constant, the controller design problem of NCSs has been investigated in (Park *et al.*, 2002) (Kim *et al.*, 2003). In our work the delay τ is assumed to be a constant.

Effects of Packets Loss

Packet loss occurs when the network is busy and under heavy load. We take the case of Sensor to Controller Delay as an example to discuss the effect of packets loss. Assuming an iterative learning control system is performing via a network and there is one packet lost at time t. In this case, the actuator side will not receive this packet. On the actuator side, since it is event driven, the packet sent at the time before t from the controller side will continue to be applied in the system until the next packet arrives. This can cause a system distortion because the actuator still use the input signal which actually comes from the previous time.

3.4 Objective

When a system performs a given task repeatedly, iterative learning control (ILC) offers a systematic design that can improve tracking performance by iterations in a fixed time interval. The literature regarding ILC has been reviewed many times by researchers, and the idea of ILC is clearly applicable to the task of improving control performance from run to run. Some surveys can be found in (Moore *et al.*, 1992) (Horowitz, 1993) (Moore, 1998). Another way to improve the tracking performance of periodic systems is called repetitive learning control (RLC). It should be pointed out that the repetitive learning control, (Sadegh *et al.*, 1990) for example, and ILC are similar in nature. However, the difference is that the ILC needs an initialization , i.e., the system should be started with the same initial condition at the beginning of each repetition, while RLC is supposed to track the periodical reference trajectory, i.e., the initial condition of current repetition is set to the terminal condition of the previous repetition (Dixon *et al.*, 2002) (Wang *et al.*, 2005). One of the advantages of RLC is that the system is not required to have the exact same initial condition after each learning trial; we have only the less restrictive requirement that the desired trajectory of the system be periodic. Some of the learning control research for nonlinear systems with time delays were performed in (Chen *et al.*, 1998) (Song *et al.*, 2001), but they did not consider compensating the effects caused by time delays. For nonlinear systems with input delay, (Pan *et al.*, 2006*a*) proposed a sampled-data pervious cycle based learning control approach to deal with control problems when the environment is periodic over iterations in a finite interval. For the application of the repetitive learning control in nonlinear NCSs, to achieve tracking control tasks, no results have been available in the literature yet, which also motivates the proposed study of this thesis.

In summary, the objective of this project is to apply repetitive learning control to nonlinear NCSs to make the real system output track the periodic desired trajectory as closely as possible. The effect caused by the network-induced delays will be reduced so as to be as small as possible.

3.5 Problem Formulation

In an NCS with a continuous and nonlinear plant will be studied in this thesis. The plant contains two parts, which are called the "linear part" and "nonlinear part". A repetitive learning control approach is proposed for the remote control nonlinear system satisfying the global Lipschitz condition. The proposed approach deals with the remote tracking control problem when the environment is periodic or repeatable over the infinite time domain. The network induced delay τ is assumed to be a constant, and is used in the predictor design.

Consider a class of nonlinear systems with input time delay,

$$\dot{x}_p(t) = \Sigma_{j=1}^{n}(a_{pj}x_j(t)) + f_p(\mathbf{x}, t) + b_p u_p(t-\tau), \qquad p = 1, \cdots, n, \tag{3.5}$$

where x_p is the state variable, $f_p(\cdot)$ is a known nonlinear function, a_{pj} and b_p are known constants, and u_p is the control input variable, $\forall p, j = 1, \cdots, n$. The nonlinear system in (3.5) can be rewritten in the state space form as follows:

$$\begin{bmatrix} \dot{x}_1(t) \\ \vdots \\ \dot{x}_n(t) \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix} + \begin{bmatrix} f_1(\mathbf{x}, t) \\ \vdots \\ f_n(\mathbf{x}, t) \end{bmatrix} + \begin{bmatrix} b_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & b_n \end{bmatrix} \begin{bmatrix} u_1(t) \\ \vdots \\ u_n(t) \end{bmatrix} \tag{3.6}$$

The equation in (3.6) can be written as:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{f}(\mathbf{x}, t) + B\mathbf{u}(t-\tau), \tag{3.7}$$

where $\mathbf{x}(t) = [x_1, \cdots, x_n]^T \in \Re^n$ is the state vector, $\mathbf{u}(t-\tau) \in \Re^n$ is the input vector and τ is the constant time delay from the controller to the actuator channel. $t \in [iT, (i+1)T]$ is the finite time for the i^{th} periodic operation of the system and T is the known period. i denotes the ith repetitive operation of the nonlinear system. $\mathbf{f}(\mathbf{x}, t)$ is a known function which is piecewise continuous in t. $A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$ and $B = diag(b_1, \cdots, b_n)$ are known matrices. The constant time delay τ is assumed to be known throughout this paper. In the following part, all discussions are based on the system dynamics in (3.7).

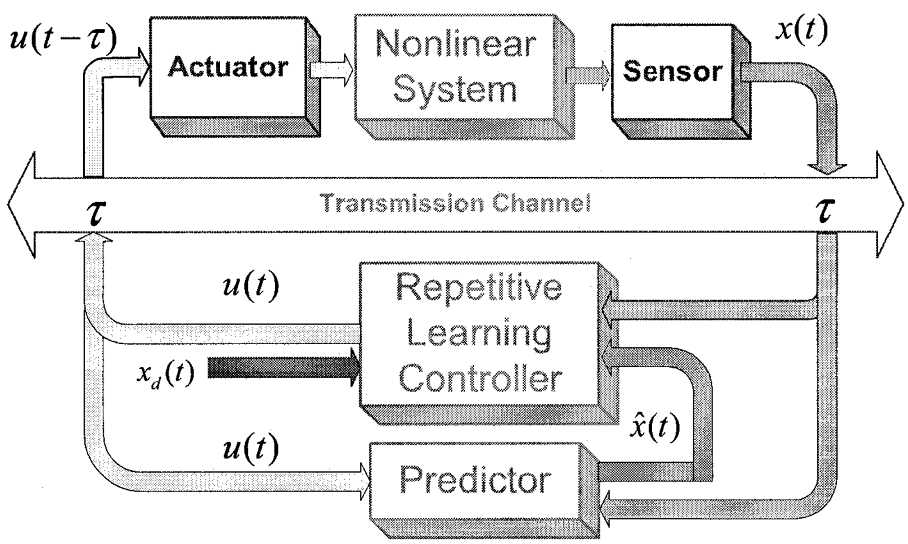

Figure 3.2. Block diagram of the controlled system

The block diagram of the control system in (3.7) is illustrated as in Fig.3.2. The sensor, actuator and the nonlinear system are remotely controlled by repetitive learning controller that interchanges measured output and control signals through a communication channel. Because network induced delays exist at both transmission channels, in order to reduce the effect caused by the delays, a predictor is designed on the controller side to provide the controller with the measured state information which can be treated as the estimation of the nonlinear system state. The controller will calculate the input value based on the information from both the predictor and the sensor.

The objective of the controlled system is to track the desired trajectory $\mathbf{x}_d(t)$. The desired trajectory can be realized by the following dynamics of the form:

$$\dot{\mathbf{x}}_d(t) = A\mathbf{x}_d + \mathbf{f}(\mathbf{x}_d, t) + B\mathbf{u}_d(t-\tau), \tag{3.8}$$

where A, B and $\mathbf{f}$ are as same as in (3.7). It means that $\mathbf{x}_d(t)$ is realizable with a unique input bounded as $\|\mathbf{u}_d(\cdot)\| \leq \beta_{ud}$, where β_{ud} is a positive constant. Throughout this paper, the following assumptions hold.

Assumption 3.1. *The system* (3.7) *is causal. Furthermore, for a given bounded desired output* $\mathbf{x}_d$*, there exists a unique bounded input* $\mathbf{u}_d$*, such that when* $\mathbf{u}(t) = \mathbf{u}_d(t)$*, the system has a unique bounded state* $\mathbf{x_d(t)}$*,* $t \in [iT, (i+1)T]$.

Assumption 3.2. *The function* $\mathbf{f}(\mathbf{x}, t)$ *is globally uniformly Lipschitz on the finite period [iT,(i+1)T] as,*

$$\| \mathbf{f}(\mathbf{x}_1, t) - \mathbf{f}(\mathbf{x}_2, t) \| \leq l_f \| \mathbf{x}_1 - \mathbf{x}_2 \|, \tag{3.9}$$

where l_f *is a known constant.*

Assumption 3.3. *The known function* $\mathbf{f}(\cdot)$ *has the following property,*

$$\|\mathbf{f}(\mathbf{x}(t-t_0), t-t_0) - \mathbf{f}(\mathbf{x}(t), t)\| \leq \; c_f\|\mathbf{x}(t-t_0) - \mathbf{x}(t)\|, \tag{3.10}$$

where c_f *is a known constant.*

Assumption 3.4. *The elements b_i of the B matrix, $i = 1, \cdots, n$, are nonzero constants.*

Lemma 3.1. *Jensen Inequality (Gu* et al.*, 2003) For any constant matrix $E \in \mathcal{R}^{n \times n}$, $E = E^T > 0$, vector function $\boldsymbol{\omega} : [0, \tau] \rightarrow \mathcal{R}^n$ such that the integrations concerned are well defined, then,*

$$\tau \int_0^\tau \boldsymbol{\omega}^T(s) E \boldsymbol{\omega}(s) ds \geq \left[\int_0^\tau \boldsymbol{\omega}(s) ds \right]^T E \left[\int_0^\tau \boldsymbol{\omega}(s) ds \right]. \tag{3.11}$$

In the following chapter, predictor design is first addressed since it plays an important role.

3.6 Application Limitations

Similarly to the work done by other researchers, the application in this project has its own limitations. These limitations are as follows. *i)* The application of repetitive learning control in this project can be only applied to periodic systems. *ii)* The network induced delay was assumed to be constant and known, while in real networks the time delays may be time-varying or distributed in a stable interval. *iii)* The nonlinear system studied has limited forms, e.g., the known matrix B is diagonal and the learning gain K is designed to be diagonal too. The elimination or reduction of these limitations will be the main tasks of the future works.

Chapter 4

Predictor Design

As discussed in Chapter 3, the controller design needs the instantly measured state information, we proposed a predictor in order to facilitate the controller design based on the delayed signal from the sensor. Then predictor-based repetitive controller is designed to eliminate the effects caused by time delays at both transmission channels.

4.1 Predictor Design

Based on the delayed state signal available at the controller side, e.g. $\mathbf{x}(t-\tau)$, the predictor is designed to predict the state signal $\mathbf{x}(t+\tau)$. We presents the predictor algorithm as follows:

$$\dot{\hat{\mathbf{x}}}(t) = A\hat{\mathbf{x}} + \mathbf{f}(\hat{\mathbf{x}}, t) + B\mathbf{u}(t) + L[\hat{\mathbf{x}}(t-2\tau) - \mathbf{x}(t-\tau)], \tag{4.1}$$

where $\hat{\mathbf{x}}(t) \in \Re^n$ is the predictor state vector, $\mathbf{f}(\cdot), A$, and B are the same as in (3.7). L is the predictor gain to be designed. $\mathbf{x}(t-\tau)$ is the measuered state of the nonlinear system transmitted from the sensor side. From (4.1), by the translation of τ, we have

$$\dot{\hat{\mathbf{x}}}(t-\tau) = A\hat{\mathbf{x}}(t-\tau) + \mathbf{f}(\hat{\mathbf{x}}(t-\tau), t-\tau) + B\mathbf{u}(t-\tau) + L[\hat{\mathbf{x}}(t-3\tau) - \mathbf{x}(t-2\tau)], \quad (4.2)$$

Denote $\tilde{\mathbf{x}}(t) = \hat{\mathbf{x}}(t-\tau) - \mathbf{x}(t)$. Comparing (4.2) with (3.7), the error dynamics is as

$$\dot{\tilde{\mathbf{x}}}(t) = A\tilde{\mathbf{x}}(t) + \tilde{\mathbf{f}}(t) + L\tilde{\mathbf{x}}(t-2\tau), \tag{4.3}$$

where $\tilde{\mathbf{f}}(t) \triangleq \mathbf{f}(\hat{\mathbf{x}}(t-\tau), t-\tau) - \mathbf{f}(\mathbf{x}, t)$. In the following theorem, L is designed according to the linear matrix inequality derived based on the Lyapunov Kravoskii method.

4.2 Convergence Analysis

Theorem 4.1. *Consider the estimation error dynamics* (4.3), *for a given time delay* τ, *if there exist symmetric positive definite matrices* $S > 0, P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix} > 0$, $Q = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix} > 0$, $R = \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} \end{bmatrix} > 0$, *matrices* $L, M_i, N_i, i = 1, ..., 5$, *with appropriate dimensions and a scalar* $\varepsilon > 0$ *such that the following inequality* *holds*

$$\begin{bmatrix} H & M \\ M^T & -\varepsilon I \end{bmatrix} < 0, \tag{4.4}$$

where $M = [M_1^T, M_2^T, M_3^T, M_4^T, M_5^T]^T$, *and*

$$H = \begin{bmatrix} H_{11} & * & * & * & * \\ H_{21} & H_{22} & * & * & * \\ H_{31} & H_{32} & H_{33} & * & * \\ H_{41} & H_{42} & H_{43} & H_{44} & * \\ H_{51} & H_{52} & H_{53} & H_{54} & H_{55} \end{bmatrix}, \tag{4.5}$$

with

$$\begin{aligned}
H_{11} &= Q_{11} + P_{12} + P_{12}^T + \tau R_{11} + \tau R_{11}^T + N_1 + N_1^T - M_1 A - A^T M_1^T + \varepsilon C_f^2 \\
H_{21} &= -P_{12}^T + N_2 - M_2 A - N_1^T - L^T M_1^T, \\
H_{22} &= S - M_2 L - (M_2 L)^T - N_2 - N_2^T - Q_{11}, \\
H_{31} &= P_{11} + N_3 - M_3 A + M_1^T, \\
H_{32} &= -N_3 + M_2^T - M_3 L, \\
H_{33} &= Q_{22} + M_3 + M_3^T + 2\tau R_{22}, \\
H_{41} &= P_{22} + N_4 - M_4 A, \\
H_{42} &= -P_{22} - N_4 - M_4 L, \\
H_{43} &= M_4 + P_{12}^T, \\
H_{44} &= -\frac{R_{11}}{2\tau}, \\
H_{51} &= N_5 - N_1^T - M_5 A,
\end{aligned}$$

$$\begin{aligned}
H_{52} &= -M_5L - N_5 - N_2^T, \\
H_{53} &= M_5 - N_3^T, \\
H_{54} &= -N_4^T, \\
H_{55} &= -\frac{R_{22}}{2\tau} - N_5 - N_5^T,
\end{aligned}$$

then the system (4.3) *is asymptotically stable, e.g.* $\tilde{\mathbf{x}}(t)$ *tends to zero asymptotically.*

Proof. Consider the following Lyapunov Krasovskii functional candidate:

$$\begin{aligned}
V &= \tilde{\mathbf{x}}^T(t)P_{11}\tilde{\mathbf{x}}(t) + 2\tilde{\mathbf{x}}^T(t)P_{12}[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds] + [\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds]^T P_{22}[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds] \\
&\quad + \int_{t-2\tau}^{t} [\tilde{\mathbf{x}}^T(s) + \dot{\tilde{\mathbf{x}}}^T(s)]^T\ Q \begin{bmatrix} \tilde{\mathbf{x}}(s) \\ \dot{\tilde{\mathbf{x}}}(s) \end{bmatrix} ds \\
&\quad + \int_{-2\tau}^{0}\int_{t+\theta}^{t} [\tilde{\mathbf{x}}^T(s)\dot{\tilde{\mathbf{x}}}^T(s)]^T\ R \begin{bmatrix} \tilde{\mathbf{x}}(s) \\ \dot{\tilde{\mathbf{x}}}(s) \end{bmatrix} dsd\theta,
\end{aligned} \tag{4.6}$$

where

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix} > 0 \quad Q = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix} > 0 \quad R = \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} \end{bmatrix} > 0. \tag{4.7}$$

With appropriate dimensions, the following two zero equations hold:

$$\begin{aligned}
\Phi_1 &= 2\{\tilde{\mathbf{x}}^T(t)N_1 + \tilde{\mathbf{x}}^T(t-2\tau)N_2 + \dot{\tilde{\mathbf{x}}}^T(t)N_3 \\
&\quad +[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds]^T N_4 + [\int_{t-2\tau}^{t}\dot{\tilde{\mathbf{x}}}(s)ds]^T N_5\} \cdot [\tilde{\mathbf{x}}(t) \\
&\quad - \int_{t-2\tau}^{t}\dot{\tilde{\mathbf{x}}}(s)ds - \tilde{\mathbf{x}}(t-2\tau)] = 0,
\end{aligned} \tag{4.8}$$

$$\Phi_2 = 2\{\tilde{\mathbf{x}}^T(t)M_1 + \tilde{\mathbf{x}}^T(t-2\tau)M_2 + \dot{\tilde{\mathbf{x}}}^T(t)M_3 +$$

$$[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds]^T M_4 + [\int_{t-2\tau}^{t} \dot{\tilde{\mathbf{x}}}(s)ds]^T M_5\}$$
$$\cdot\{\dot{\tilde{\mathbf{x}}}(t) - A\tilde{\mathbf{x}}(t) - [\hat{f} - f] - L\tilde{\mathbf{x}}(t-2\tau)\} = 0. \tag{4.9}$$

Then the derivative of the Lyapunov function candidate is as follows,

$$\begin{aligned}
\dot{V} \;=\; & \dot{V} + \Phi_1 + \Phi_2 = \tilde{\mathbf{x}}^T(t) P_{11}\dot{\tilde{\mathbf{x}}}(t) + \dot{\tilde{\mathbf{x}}}^T(t) P_{11}\tilde{\mathbf{x}}(t) + 2\dot{\tilde{\mathbf{x}}}^T(t) P_{12}[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds] \\
& +2\tilde{\mathbf{x}}^T(t) P_{12}[\tilde{\mathbf{x}}(t) - \tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}(t) - \tilde{\mathbf{x}}(t-2\tau)]^T P_{22} \int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds \\
& +[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds]^T P_{22}[\tilde{\mathbf{x}}(t) - \tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}^T(t)\ \dot{\tilde{\mathbf{x}}}^T(t)]^T Q \begin{bmatrix} \tilde{\mathbf{x}}(t) \\ \dot{\tilde{\mathbf{x}}}(t) \end{bmatrix} \\
& -[\ \tilde{\mathbf{x}}^T(t-2\tau)\ \ \dot{\tilde{\mathbf{x}}}^T(t-2\tau)\ ]^T Q \begin{bmatrix} \tilde{\mathbf{x}}(t-2\tau) \\ \dot{\tilde{\mathbf{x}}}(t-2\tau) \end{bmatrix} \\
& +2\tau[\ \tilde{\mathbf{x}}^T(t)\ \dot{\tilde{\mathbf{x}}}^T(t)\ ]^T R \begin{bmatrix} \tilde{\mathbf{x}}(t) \\ \dot{\tilde{\mathbf{x}}}(t) \end{bmatrix} - \int_{t-2\tau}^{t} [\ \tilde{\mathbf{x}}^T(s)\ \ \dot{\tilde{\mathbf{x}}}^T(s)\ ]^T R \begin{bmatrix} \tilde{\mathbf{x}}(s) \\ \dot{\tilde{\mathbf{x}}}(s) \end{bmatrix} ds \\
& +2\mathbf{z}^T N[\tilde{\mathbf{x}}(t) - \int_{t-2\tau}^{t} \dot{\tilde{\mathbf{x}}}(s)ds - \tilde{\mathbf{x}}(t-2\tau)] + 2\mathbf{z}^T M[\dot{\tilde{\mathbf{x}}}(t) - A\tilde{\mathbf{x}}(t) - \tilde{\mathbf{f}}(t) \\
& -L\tilde{\mathbf{x}}(t-2\tau)],
\end{aligned} \tag{4.10}$$

where

$$\mathbf{z} = [\tilde{\mathbf{x}}^T(t)\ \ \tilde{\mathbf{x}}^T(t-2\tau)\ \ \dot{\tilde{\mathbf{x}}}^T(t)\ \ [\textstyle\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds]^T\ \ [\int_{t-2\tau}^{t} \dot{\tilde{\mathbf{x}}}(s)ds]^T]^T$$
$$N = [N_1^T\ N_2^T\ N_3^T\ N_4^T\ N_5^T]^T, \quad M = [M_1^T\ M_2^T\ M_3^T\ M_4^T\ M_5^T]^T.$$

Furthermore, we have

$$-2\mathbf{z}^T M\tilde{\mathbf{f}} \le \varepsilon^{-1}(\mathbf{z}^T M)(M^T\mathbf{z}) + \varepsilon\tilde{\mathbf{f}}^T\tilde{\mathbf{f}} \le \varepsilon^{-1}(\mathbf{z}^T M M^T \mathbf{z}) + \varepsilon C_f^2 \tilde{\mathbf{x}}^T(t)\tilde{\mathbf{x}}(t). \tag{4.11}$$

Using (4.10), (4.11) and the Jensen inequality in (3.11),

$$\begin{aligned}
\dot{V} \leq\; & \tilde{\mathbf{x}}^T(t)P_{11}\dot{\tilde{\mathbf{x}}}(t) + \dot{\tilde{\mathbf{x}}}^T(t)P_{11}\tilde{\mathbf{x}}(t) + 2\dot{\tilde{\mathbf{x}}}^T(t)P_{12}[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds] \\
& +2\tilde{\mathbf{x}}^T(t)P_{12}[\tilde{\mathbf{x}}(t)-\tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}(t)-\tilde{\mathbf{x}}(t-2\tau)]^T P_{22}\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds \\
& +[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds]^T P_{22}[\tilde{\mathbf{x}}(t)-\tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}^T(t)\;\; \dot{\tilde{\mathbf{x}}}^T(t)]^T\, Q \begin{bmatrix}\tilde{\mathbf{x}}(t)\\ \dot{\tilde{\mathbf{x}}}(t)\end{bmatrix} \\
& -[\;\tilde{\mathbf{x}}^T(t-2\tau)\;\; \dot{\tilde{\mathbf{x}}}^T(t-2\tau)\;]^T Q \begin{bmatrix}\tilde{\mathbf{x}}(t-2\tau)\\ \dot{\tilde{\mathbf{x}}}(t-2\tau)\end{bmatrix} \\
& +2\tau[\;\tilde{\mathbf{x}}^T(t)\;\; \dot{\tilde{\mathbf{x}}}^T(t)\;]^T R \begin{bmatrix}\tilde{\mathbf{x}}(t)\\ \dot{\tilde{\mathbf{x}}}(t)\end{bmatrix} - \int_{t-2\tau}^{t}[\;\tilde{\mathbf{x}}^T(s)\;\; \dot{\tilde{\mathbf{x}}}^T(s)\;]^T R \begin{bmatrix}\tilde{\mathbf{x}}(s)\\ \dot{\tilde{\mathbf{x}}}(s)\end{bmatrix} ds \\
& +2Z^T N[\tilde{\mathbf{x}}(t) - \int_{t-2\tau}^{t}\dot{\tilde{\mathbf{x}}}(s)ds - \tilde{\mathbf{x}}(t-2\tau)] + 2\mathbf{z}^T M[\dot{\tilde{\mathbf{x}}}(t) - A\tilde{\mathbf{x}}(t) + L\tilde{\mathbf{x}}(t-2\tau)] \\
& +\varepsilon^{-1}(\mathbf{z}^T M M^T \mathbf{z}) + \varepsilon C_f^2 \tilde{\mathbf{x}}^T(t)\tilde{\mathbf{x}}(t) \\
=\; & \tilde{\mathbf{x}}^T(t)P_{11}\dot{\tilde{\mathbf{x}}}(t) + \dot{\tilde{\mathbf{x}}}^T(t)P_{11}\tilde{\mathbf{x}}(t) + 2\dot{\tilde{\mathbf{x}}}^T(t)P_{12}[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds] \\
& +2\tilde{\mathbf{x}}^T(t)P_{12}[\tilde{\mathbf{x}}(t)-\tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}(t)-\tilde{\mathbf{x}}(t-2\tau)]^T P_{22}\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds \\
& +[\int_{t-2\tau}^{t}\tilde{\mathbf{x}}(s)ds]^T P_{22}[\tilde{\mathbf{x}}(t)-\tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}^T(t)\;\; \dot{\tilde{\mathbf{x}}}^T(t)]^T\, Q \begin{bmatrix}\tilde{\mathbf{x}}(t)\\ \dot{\tilde{\mathbf{x}}}(t)\end{bmatrix} \\
& -[\;\tilde{\mathbf{x}}^T(t-2\tau)\;\; \dot{\tilde{\mathbf{x}}}^T(t-2\tau)\;]^T Q \begin{bmatrix}\tilde{\mathbf{x}}(t-2\tau)\\ \dot{\tilde{\mathbf{x}}}(t-2\tau)\end{bmatrix} \\
& +2\tau[\;\tilde{\mathbf{x}}^T(t)\;\; \dot{\tilde{\mathbf{x}}}^T(t)\;]^T R \begin{bmatrix}\tilde{\mathbf{x}}(t)\\ \dot{\tilde{\mathbf{x}}}(t)\end{bmatrix} - \int_{t-2\tau}^{t}[\;\tilde{\mathbf{x}}^T(s)\;\; \dot{\tilde{\mathbf{x}}}^T(s)\;]^T R \begin{bmatrix}\tilde{\mathbf{x}}(s)\\ \dot{\tilde{\mathbf{x}}}(s)\end{bmatrix} ds \\
& +2Z^T N[\tilde{\mathbf{x}}(t) - \int_{t-2\tau}^{t}\dot{\tilde{\mathbf{x}}}(s)ds - \tilde{\mathbf{x}}(t-2\tau)] + 2\mathbf{z}^T M[\dot{\tilde{\mathbf{x}}}(t) - A\tilde{\mathbf{x}}(t) - L\tilde{\mathbf{x}}(t-2\tau)] \\
& +\varepsilon^{-1}(\mathbf{z}^T M M^T \mathbf{z}) + \varepsilon C_f^2 \tilde{\mathbf{x}}^T(t)\tilde{\mathbf{x}}(t) + \tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau)
\end{aligned}$$

$$
\begin{aligned}
& -\tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau) \\
\leq\ & \tilde{\mathbf{x}}^T(t)P_{11}\dot{\tilde{\mathbf{x}}}(t) + \dot{\tilde{\mathbf{x}}}^T(t)P_{11}\tilde{\mathbf{x}}(t) + 2\dot{\tilde{\mathbf{x}}}^T(t)P_{12}[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds] \\
& +2\tilde{\mathbf{x}}^T(t)P_{12}[\tilde{\mathbf{x}}(t) - \tilde{\mathbf{x}}(t-2\tau)] + [\tilde{\mathbf{x}}(t) - \tilde{\mathbf{x}}(t-2\tau)]^T P_{22} \int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds \\
& +[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds]^T P_{22}[\tilde{\mathbf{x}}(t) - \tilde{\mathbf{x}}(t-2\tau)] + \ [\tilde{\mathbf{x}}^T(t)\ \dot{\tilde{\mathbf{x}}}^T(t)]^T\ Q \begin{bmatrix} \tilde{\mathbf{x}}(t) \\ \dot{\tilde{\mathbf{x}}}(t) \end{bmatrix} \\
& -\tilde{\mathbf{x}}^T(t-2\tau)Q_{11}\tilde{\mathbf{x}}(t-2\tau) + 2\tau\ [\tilde{\mathbf{x}}^T(t)\ \ \dot{\tilde{\mathbf{x}}}^T(t)]^T\ R \begin{bmatrix} \tilde{\mathbf{x}}(t) \\ \dot{\tilde{\mathbf{x}}}(t) \end{bmatrix} \\
& -\frac{1}{2\tau}\left[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds\right]^T R \left[\int_{t-2\tau}^{t} \tilde{\mathbf{x}}(s)ds\right] \\
& +2\mathbf{z}^T N[\tilde{\mathbf{x}}(t) - \int_{t-2\tau}^{t} \dot{\tilde{\mathbf{x}}}(s)ds - \tilde{\mathbf{x}}(t-2\tau)] + 2\mathbf{z}^T M[\dot{\tilde{\mathbf{x}}}(t) - A\tilde{\mathbf{x}}(t) - L\tilde{\mathbf{x}}(t-2\tau)] \\
& +\varepsilon^{-1}(\mathbf{z}^T M M^T \mathbf{z}) + \varepsilon C_f^2 \tilde{\mathbf{x}}^T(t)\tilde{\mathbf{x}}(t) + \tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau) \\
& -\tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau) \\
& \leq -\tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau) + \mathbf{z}^T H\mathbf{z} + \varepsilon^{-1}(\mathbf{z}^T M M^T \mathbf{z}),
\end{aligned} \tag{4.12}
$$

where S is a symmetric definite matrix and H is as shown in (4.5). The inequality (4.12) is equivalent to

$$
\dot{V} \leq -\tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau) + \mathbf{z}^T \begin{bmatrix} H & M \\ M^T & -\varepsilon I \end{bmatrix} \mathbf{z}. \tag{4.13}
$$

If there exist symmetric positive definite matrices $S > 0, P > 0, Q > 0, R > 0$, matrices $L, M_i, N_i, i = 1, ...5$, with appropriate dimensions and a scalar $\varepsilon > 0$ such that the inequality (4.4) holds, then from (4.13) we have

$$
\dot{V} \leq -\tilde{\mathbf{x}}^T(t-2\tau)S\tilde{\mathbf{x}}(t-2\tau) < 0. \tag{4.14}
$$

Since S is a positive symmetric definite matrix, from the Lyapunov stability theory, the system (4.3) is asymptotically stable. □

In the steady state, the predictor can estimate the future state of the nonlinear system, e.g. $\tilde{\mathbf{x}}(t) = 0 \Rightarrow \hat{\mathbf{x}}(t-\tau) = \mathbf{x}(t)$. Hence the output of the predictor is as $\hat{\mathbf{x}}(t) = \mathbf{x}(t+\tau)$, which can be used to design a repetitive learning controller to realize the control objective. However, if we design a predictor to achieve $\hat{\mathbf{x}}(t) = \mathbf{x}(t)$ of the the current time stamp instead of the future information, then the controller will not be able to compensate the influence of the time delay τ in the channel from the controller to the actuator.

The LMI condition in (4.5) is non-convex and hence the following theorem is proposed to be the equivalent sufficient condition as in Theorem 1.

Theorem 4.2. *For given scalars θ_i, $i = 1, \cdots, 5$, and a given time delay constant τ, if there exist symmetric positive definite matrices $\bar{S}$, $\bar{P} = \begin{bmatrix} \bar{P}_{11} & \bar{P}_{12} \\ \bar{P}_{12}^T & \bar{P}_{22} \end{bmatrix} > 0$, $\bar{Q} = \begin{bmatrix} \bar{Q}_{11} & 0 \\ 0 & \bar{Q}_{22} \end{bmatrix} > 0$, $\bar{R} = \begin{bmatrix} \bar{R}_{11} & 0 \\ 0 & \bar{R}_{22} \end{bmatrix} > 0$, matrices Y, $\bar{N}_i$, $i = 1, \cdots, 5$, nonsingular matrix X with appropriate dimensions and constant $\varepsilon > 0$ such that the following inequality holds,*

$$\begin{bmatrix} \bar{H}_{11} & * & * & * & * & * & * \\ \bar{H}_{21} & \bar{H}_{22} & * & * & * & * & * \\ \bar{H}_{31} & \bar{H}_{32} & \bar{H}_{33} & * & * & * & * \\ \bar{H}_{41} & \bar{H}_{42} & \bar{H}_{43} & \bar{H}_{44} & * & * & * \\ \bar{H}_{51} & \bar{H}_{52} & \bar{H}_{53} & \bar{H}_{54} & \bar{H}_{55} & * & * \\ \theta_1 I & \theta_2 I & \theta_3 I & \theta_4 I & \theta_5 I & -\varepsilon I & 0 \\ \varepsilon c_f X^T & 0 & 0 & 0 & 0 & 0 & -\varepsilon I \end{bmatrix} < 0, \tag{4.15}$$

where

$$
\begin{aligned}
\bar{H}_{11} &= \bar{Q}_{11} + \bar{P}_{12} + \bar{P}_{12}^T + \tau\bar{R}_{11} + \tau\bar{R}_{11}^T + \bar{N}_1 + \bar{N}_1^T - \theta_1 A X^T - \theta_1 X A^T \\
\bar{H}_{21} &= -\bar{P}_{12}^T + \bar{N}_2 - \theta_2 A X^T - \bar{N}_1^T - \theta_1 Y^T, \\
\bar{H}_{22} &= S - \theta_2 Y - \theta_2 Y^T - \bar{N}_2 - \bar{N}_2^T - \bar{Q}_{11}, \\
\bar{H}_{31} &= \bar{P}_{11} + \bar{N}_3 - \theta_3 A X^T + \theta_1 X, \\
\bar{H}_{32} &= -\bar{N}_3 + \theta_2 X - \theta_3 Y, \\
\bar{H}_{33} &= \bar{Q}_{22} + \theta_3 X + \theta_3 X^T + 2\tau\bar{R}_{22}, \\
\bar{H}_{41} &= \bar{P}_{22} + \bar{N}_4 - \theta_4 A X^T, \\
\bar{H}_{42} &= -\bar{P}_{22} - \bar{N}_4 - \theta_4 Y, \\
\bar{H}_{43} &= \theta_4 X^T + \bar{P}_{12}^T, \\
\bar{H}_{44} &= -\frac{\bar{R}_{11}}{2\tau}, \\
\bar{H}_{51} &= \bar{N}_5 - \bar{N}_1^T - \theta_5 A X^T, \\
\bar{H}_{52} &= -\theta_5 Y - \bar{N}_5 - \bar{N}_2^T, \\
\bar{H}_{53} &= \theta_5 X^T - \bar{N}_3^T, \\
\bar{H}_{54} &= -\bar{N}_4^T, \\
\bar{H}_{55} &= -\frac{\bar{R}_{22}}{2\tau} - \bar{N}_5 - \bar{N}_5^T,
\end{aligned}
$$

then matrices L and S in Theorem 1 is obtained as

$$L = Y X^{-T}, \qquad S = X^{-1}\bar{S}X^{-T}. \tag{4.16}$$

As a result, the error dynamics (4.3) *is asymptotically stable, e.g.* $\tilde{\mathbf{x}}(t)$ *tends to zero asymptotically.*

Proof: In order to transform the nonconvex LMI in (4.5) into a solvable LMI, (4.5) could be represented as the following form by schur complement,

$$\begin{bmatrix} H_{11}-\varepsilon c_f^2 I & * & * & * & * & * & * \\ H_{21} & H_{22} & * & * & * & * & * \\ H_{31} & H_{32} & H_{33} & * & * & * & * \\ H_{41} & H_{42} & H_{43} & H_{44} & * & * & * \\ H_{51} & H_{52} & H_{53} & H_{54} & H_{55} & * & * \\ M_1 & M_2 & M_3 & M_4 & M_5 & -\varepsilon I & 0 \\ \varepsilon c_f X^T & 0 & 0 & 0 & 0 & 0 & -\varepsilon I \end{bmatrix} < 0, \tag{4.17}$$

we assume that we have some relations in M_i's, $i = 1, \cdots, 5$. One possibility is that $M_i = \theta_i M_0$ where M_0 is nonsingular and θ_i is known and given. Define $X = M_0^{-1}$, $W = diag(X, X, X, X, X, I, I)$ and $Y = LX^T$. Then by pre-multiplying the inequality in (4.17) by W and post-multiplying by W^T, we can obtain the inequality (4.15). Note that the inequality in (4.15) is only a sufficient condition for the solvability of (4.5) based on the derivation.

In the next Chapter, the problem of the repetitive learning controller design will be discussed, the stability analysis using Lyapunov direct method will also be presented.

Chapter 5

Repetitive Learning Controller Design

In this chapter, a repetitive learning controller is designed to achieve the control objective, the stability has been analyzed using Lyapunov direct method.

Notation: $\|\mathbf{x}\|$ is the norm defined as $\|\cdot\| = \sqrt{\mathbf{x}^T\mathbf{x}}$, where $\mathbf{x}$ is a vector.

5.1 Repetitive Learning Controller Design

The repetitive learning controller is designed for the periodic control task of the nonlinear system (3.7) as follows:

$$\mathbf{u}(t) = \mathbf{u}(t-T) + K\hat{\mathbf{e}}(t), \tag{5.1}$$

where $\mathbf{u}(t)$ is the current new control input, $\mathbf{u}(t-T)$ is the known control input from the previous cycle, K is the repetitive learning gain, and $\hat{\mathbf{e}}(t) = \mathbf{x}_d(t+\tau) - \hat{\mathbf{x}}(t)$ is the current cycle error, obtained by comparing the desired output and the predicted state signal. Here the current tracking error is used for the repetitive learning scheme. $\mathbf{u}(t-T)$ then can be written as:

$$\mathbf{u}(t-\tau) = \mathbf{u}(t-T-\tau) + K\hat{\mathbf{e}}(t-\tau). \tag{5.2}$$

If the error $\hat{\mathbf{e}}(t-\tau)$ tends to zero when t tends to infinite, the system state $\mathbf{x}(t)$ will track the desired output trajectory $\mathbf{x}_d(t)$ perfectly. The convergence property of the closed-loop learning system is analyzed in the following theorem.

5.2 Stability Analysis

Theorem 5.1. *Consider the system* (3.7) *with the predictor* (4.2) *and under the repetitive learning control law in* (5.1), *it satisfies that*

$$\lim_{t\to\infty} \hat{\mathbf{e}}(t-\tau) = 0. \tag{5.3}$$

Since $\lim_{t\to\infty} \tilde{\mathbf{x}}(t) = 0$, *and* $\mathbf{e}(t) = \mathbf{x}_d(t) - \mathbf{x}(t) = \hat{\mathbf{e}}(t-\tau) + \tilde{\mathbf{x}}(t) \longrightarrow 0$, *we have* $\lim_{t\to\infty} \mathbf{e}(t) = 0$.

Proof. Translating (3.8) by time delay τ,

$$\dot{\mathbf{x}}_d(t+\tau) = A\mathbf{x}_d(t+\tau) + \mathbf{f}(\mathbf{x}_d(t+\tau), t+\tau) + B\mathbf{u}_d(t), \tag{5.4}$$

The current tracking error of the estimated state $\hat{\mathbf{e}}(t)$ is represented as

$$\hat{\mathbf{e}}(t) = \mathbf{x}_d(t+\tau) - \hat{\mathbf{x}}(t). \tag{5.5}$$

Consider the predictor in (4.1), the error dynamics can be represented as

$$\begin{aligned}
\dot{\hat{\mathbf{e}}}(t) &= \dot{\mathbf{x}}_d(t+\tau) - \dot{\hat{\mathbf{x}}}(t) \\
&= A\mathbf{x}_d(t+\tau) + \mathbf{f}(\mathbf{x}_d(t+\tau), t+\tau) + B\mathbf{u}_d(t) - \{A\hat{\mathbf{x}}(t) + \mathbf{f}(\hat{\mathbf{x}}, t) + B\mathbf{u}(t) \\
&\quad + L[\hat{\mathbf{x}}(t-2\tau) - \mathbf{x}(t-\tau)]\} \\
&= A\mathbf{x}_d(t+\tau) + \mathbf{f}(\mathbf{x}_d(t+\tau), t+\tau) + B\mathbf{u}_d(t) - A\hat{\mathbf{x}}(t) - \mathbf{f}(\hat{\mathbf{x}}, t) - B\mathbf{u}(t) \\
&\quad - L[\hat{\mathbf{x}}(t-2\tau) - \mathbf{x}(t-\tau)] \\
&= A[\mathbf{x}_d(t+\tau) - \hat{\mathbf{x}}(t)] + \mathbf{f}(\mathbf{x}_d(t+\tau), t+\tau) - \mathbf{f}(\hat{\mathbf{x}}(t), t) + B[\mathbf{u}_d(t) - \mathbf{u}(t)] \\
&\quad - L[\hat{\mathbf{x}}(t-2\tau) - \mathbf{x}(t-\tau)] \\
&= A\hat{\mathbf{e}}(t) + \mathbf{f}(\mathbf{x}_d(t+\tau), t+\tau) - \mathbf{f}(\hat{\mathbf{x}}(t), t) + B[\mathbf{u}_d(t) - \mathbf{u}(t)] \\
&\quad - L\tilde{\mathbf{x}}(t-\tau).
\end{aligned} \tag{5.6}$$

From the error dynamics in (5.6) and the repetitive learning control law in (5.1),

$$\begin{aligned}
\mathbf{u}(t-T) - \mathbf{u}_d(t) &= [B^T B]^{-1} B^T [A\hat{\mathbf{e}}(t) + \mathbf{f}(\mathbf{x}_d(t+\tau), t+\tau) - \mathbf{f}(\hat{\mathbf{x}}(t), t) \\
&\quad - BK\hat{\mathbf{e}}(t) - \dot{\hat{\mathbf{e}}}(t) - L\tilde{\mathbf{x}}(t-\tau)].
\end{aligned} \tag{5.7}$$

Shifting (5.7) with the time delay τ, the equation in (5.7) can be represented as

$$\begin{aligned}
\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) &= [B^T B]^{-1} B^T [A\hat{\mathbf{e}}(t-\tau) + f(\mathbf{x}_d(t), t) \\
&\quad - f(\hat{\mathbf{x}}(t-\tau), t-\tau) - BK\hat{\mathbf{e}}(t-\tau) \\
&\quad - \dot{\hat{\mathbf{e}}}(t-\tau) - L\tilde{\mathbf{x}}(t-2\tau)].
\end{aligned} \tag{5.8}$$

Define the Lyapunov function candidate as follows:

$$J = \int_{t-T}^{t} \|\mathbf{u}(s-\tau) - \mathbf{u}_d(s-\tau)\|^2 ds + V. \tag{5.9}$$

Differentiating J with respect to time t, using equations (5.2), (5.8) and $\mathbf{u}_d(t-T-\tau) = \mathbf{u}_d(t-\tau)$,we have

$$\begin{aligned}
\dot{J} &= \|\mathbf{u}(t-\tau) - \mathbf{u}_d(t-\tau)\|^2 - \|\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-T-\tau)\|^2 + \dot{V} \\
&= [\mathbf{u}(t-\tau) - \mathbf{u}_d(t-\tau)]^T[\mathbf{u}(t-\tau) - \mathbf{u}_d(t-\tau)] \\
&\quad -[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-T-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-T-\tau)] + \dot{V} \\
&= [\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) + K\hat{\mathbf{e}}(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) \\
&\quad +K\hat{\mathbf{e}}(t-\tau)] - [\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-T-\tau)]^T[\mathbf{u}(t-T-\tau) \\
&\quad -\mathbf{u}_d(t-T-\tau)] + \dot{V} \\
&= [\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) + K\hat{\mathbf{e}}(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) \\
&\quad +K\hat{\mathbf{e}}(t-\tau)] - [\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau)]^T[\mathbf{u}(t-T-\tau) \\
&\quad -\mathbf{u}_d(t-\tau)] + \dot{V} \\
&= [\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) + K\hat{\mathbf{e}}(t-\tau)] \\
&\quad +[K\hat{\mathbf{e}}(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) + K\hat{\mathbf{e}}(t-\tau)] \\
&\quad -[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau)] + \dot{V}.
\end{aligned} \tag{5.10}$$

From (5.10) we have

$$\begin{aligned}
\dot{J} &= [\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) + K\hat{\mathbf{e}}(t-\tau) \\
&\quad -\mathbf{u}(t-T-\tau) + \mathbf{u}_d(t-\tau)] + [K\hat{\mathbf{e}}(t-\tau)]^T[\mathbf{u}(t-T-\tau) - \mathbf{u}_d(t-\tau) \\
&\quad +K\hat{\mathbf{e}}(t-\tau)] + \dot{V}
\end{aligned}$$

$$
\begin{aligned}
&= [\mathbf{u}(t-T-\tau)-\mathbf{u}_d(t-\tau)]^T[K\hat{\mathbf{e}}(t-\tau)]+[K\hat{\mathbf{e}}(t-\tau)]^T[\mathbf{u}(t-T-\tau) \\
&\quad -\mathbf{u}_d(t-\tau)+K\hat{\mathbf{e}}(t-\tau)]+\dot{V} \\
&= [K\hat{\mathbf{e}}(t-\tau)]^T\{[\mathbf{u}(t-T-\tau)-\mathbf{u}_d(t-\tau)]+[\mathbf{u}(t-T-\tau)-\mathbf{u}_d(t-\tau) \\
&\quad +K\hat{\mathbf{e}}(t-\tau)]\}+\dot{V} \\
&= 2\hat{\mathbf{e}}^T(t-\tau)K^T[\mathbf{u}(t-T-\tau)-\mathbf{u}_d(t-\tau)]+\hat{\mathbf{e}}^T(t-\tau)K^TK\hat{\mathbf{e}}(t-\tau)+\dot{V} \\
&= 2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^T[A\hat{\mathbf{e}}(t-\tau)+f(\mathbf{x}_d(t),t)-f(\hat{\mathbf{x}}(t-\tau),t-\tau) \\
&\quad -BK\hat{\mathbf{e}}(t-\tau)-\dot{\hat{\mathbf{e}}}(t-\tau)-L\tilde{\mathbf{x}}(t-2\tau)]+\hat{\mathbf{e}}^T(t-\tau)K^TK\hat{\mathbf{e}}(t-\tau)+\dot{V} \\
&= 2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^TA\hat{\mathbf{e}}(t-\tau)+2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^T \\
&\quad [f(\mathbf{x}_d(t),t)-f(\hat{\mathbf{x}}(t-\tau),t-\tau)]-\hat{\mathbf{e}}^T(t-\tau)K^TK\hat{\mathbf{e}}(t-\tau) \\
&\quad -2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^T\dot{\hat{\mathbf{e}}}(t-\tau)-2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^TL\tilde{\mathbf{x}}(t-2\tau) \\
&\quad +\dot{V}. \qquad (5.11)
\end{aligned}
$$

Consider Assumption 3, (5.11) becomes the following inequality

$$
\begin{aligned}
\dot{J} &\leq 2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^TA\hat{\mathbf{e}}(t-\tau)+2c_f\|K^T[B^TB]^{-1}B^T\|\|\hat{\mathbf{e}}(t-\tau)\|^2 \\
&\quad -\hat{\mathbf{e}}^T(t-\tau)K^TK\hat{\mathbf{e}}(t-\tau)-2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^T\dot{\hat{\mathbf{e}}}(t-\tau) \\
&\quad -2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^TL\tilde{\mathbf{x}}(t-2\tau)+\dot{V} \\
&= -\hat{\mathbf{e}}^T(t-\tau)\{K^TK-2K^T[B^TB]^{-1}B^TA-2c_f\|K^T[B^TB]^{-1}B^T\|I\}\hat{\mathbf{e}}(t-\tau) \\
&\quad -2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^T\dot{\hat{\mathbf{e}}}(t-\tau)-2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^TL\tilde{\mathbf{x}}(t-2\tau) \\
&\quad +\dot{V}. \qquad (5.12)
\end{aligned}
$$

Note that the following equation holds,

$$2\hat{\mathbf{e}}^T(t-\tau)K^T[B^TB]^{-1}B^TL\tilde{\mathbf{x}}(t-2\tau)$$

$$
\begin{aligned}
&= [\tilde{\mathbf{x}}(t-2\tau)+\hat{\mathbf{e}}(t-\tau)]^T K^T [B^T B]^{-1} B^T L [\tilde{\mathbf{x}}(t-2\tau)+\hat{\mathbf{e}}(t-\tau)] \\
&\quad -\tilde{\mathbf{x}}^T(t-2\tau) K^T [B^T B]^{-1} B^T L \tilde{\mathbf{x}}(t-2\tau) \\
&\quad -\hat{\mathbf{e}}^T(t-\tau) K^T [B^T B]^{-1} B^T L \hat{\mathbf{e}}(t-\tau).
\end{aligned} \tag{5.13}
$$

Substitute (4.14) and (5.13) into (5.12), we have

$$
\begin{aligned}
\dot{J} \;\le\;& -\hat{\mathbf{e}}^T(t-\tau)\{K^T K - 2K^T[B^T B]^{-1} B^T A - 2c_f \|K^T [B^T B]^{-1} B^T\| I \\
& -K^T [B^T B]^{-1} B^T L\}\hat{\mathbf{e}}(t-\tau) + \tilde{\mathbf{x}}^T(t-2\tau) K^T [B^T B]^{-1} B^T L \tilde{\mathbf{x}}(t-2\tau) \\
& -[\tilde{\mathbf{x}}(t-2\tau)+\hat{\mathbf{e}}(t-\tau)]^T K^T [B^T B]^{-1} B^T L [\tilde{\mathbf{x}}(t-2\tau)+\hat{\mathbf{e}}(t-\tau)] \\
& -2\hat{\mathbf{e}}^T(t-\tau) K^T [B^T B]^{-1} B^T \dot{\hat{\mathbf{e}}}(t-\tau) + \dot{V} \\
\le\;& -\hat{\mathbf{e}}^T(t-\tau)\{K^T K - 2K^T[B^T B]^{-1} B^T A - 2c_f \|K^T [B^T B]^{-1} B^T\| I \\
& -K^T [B^T B]^{-1} B^T L\}\hat{\mathbf{e}}(t-\tau) - \tilde{\mathbf{x}}^T(t-2\tau)[S - K^T [B^T B]^{-1} B^T L] \tilde{\mathbf{x}}(t-2\tau) \\
& -[\tilde{\mathbf{x}}(t-2\tau)+\hat{\mathbf{e}}(t-\tau)]^T K^T [B^T B]^{-1} B^T L [\tilde{\mathbf{x}}(t-2\tau)+\hat{\mathbf{e}}(t-\tau)] \\
& -2\hat{\mathbf{e}}^T(t-\tau) K^T [B^T B]^{-1} B^T \dot{\hat{\mathbf{e}}}(t-\tau).
\end{aligned} \tag{5.14}
$$

If we select K such that

$$
\begin{aligned}
D_1 &= K^T K - 2K^T[B^T B]^{-1} B^T A - 2c_f \|K^T [B^T B]^{-1} B^T\| I - K^T [B^T B]^{-1} B^T L > 0, \\
D_2 &= K^T [B^T B]^{-1} B^T L > 0 \\
D_3 &= S - K^T [B^T B]^{-1} B^T L > 0 \\
D_4 &= K^T [B^T B]^{-1} B^T > 0,
\end{aligned}
$$

and D_4 is positive symmetric definite. Then (5.14) becomes

$$
\dot{J} \le -\hat{\mathbf{e}}^T(t-\tau) D_1 \hat{\mathbf{e}}(t-\tau) - 2\hat{\mathbf{e}}^T(t-\tau) D_4 \dot{\hat{\mathbf{e}}}(t-\tau). \tag{5.15}
$$

Because the following equation holds

$$\begin{aligned} 2\hat{\mathbf{e}}^T(t-\tau)D_4\dot{\hat{\mathbf{e}}}(t-\tau) &= \hat{\mathbf{e}}^T(t-\tau)D_4\dot{\hat{\mathbf{e}}}(t-\tau)+\dot{\hat{\mathbf{e}}}(t-\tau)^T D_4\hat{\mathbf{e}}(t-\tau) \\ &= d[\hat{\mathbf{e}}^T(s-\tau)D_4\hat{\mathbf{e}}(s-\tau)]/ds. \end{aligned} \tag{5.16}$$

Then substituting (5.16) into (5.15), integrating on both sides of (5.15) yields

$$\textstyle\int_T^t \dot{J}ds \leq -\int_T^t \hat{\mathbf{e}}^T(s-\tau)D_1\hat{\mathbf{e}}(s-\tau)ds - \int_T^t 1\cdot d[\hat{\mathbf{e}}^T(s-\tau)D_4\hat{\mathbf{e}}(s-\tau)]. \tag{5.17}$$

Then (5.17) becomes

$$J(t)-J(T) \leq -\textstyle\int_T^t \hat{\mathbf{e}}^T(s-\tau)D_1\hat{\mathbf{e}}(s-\tau)ds - \hat{\mathbf{e}}^T(s-\tau)D_4\hat{\mathbf{e}}(s-\tau)\mid_T^t . \tag{5.18}$$

From (5.18) we have

$$\begin{aligned} \min(\lambda(D_1))\int_T^t \|\hat{\mathbf{e}}(s-\tau)\|^2 ds &\leq \int_T^t \hat{\mathbf{e}}^T(s-\tau)D_1\hat{\mathbf{e}}(s-\tau)ds \\ &\leq J(T)+\hat{\mathbf{e}}^T(T-\tau)D_4\hat{\mathbf{e}}(T-\tau). \end{aligned} \tag{5.19}$$

Based on Barbalat's Lemma (Narendra and Annaswamy, 1989), we have

$$\lim_{t\to\infty}\hat{\mathbf{e}}(t-\tau)=0. \tag{5.20}$$

□

Remark 5.1. *As shown in the proof of Theorem 5.1, the learning gain K is designed based on the following condition:*

$D_1 = K^TK - 2K^T[B^TB]^{-1}B^TA - 2c_f\|K^T[B^TB]^{-1}B^T\|I - K^T[B^TB]^{-1}B^TL > 0,$

$D_2 = K^T[B^TB]^{-1}B^TL > 0,$

$D_3 = S - K^T[B^TB]^{-1}B^TL > 0,$

$D_4 = K^T[B^TB]^{-1}B^T > 0,$

and D_4 is a symmetric matrix.

In the next Chapter, simulation results will be shown to demonstrate the effectiveness of the proposed approach.

Chapter 6

Simulation Results

Consider the following nonlinear system which is controlled through some network:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -6 & 3 \\ 3 & -9 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ -\sin(x_1(t)) \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}(t-\tau), \quad (6.1)$$

with $\mathbf{x}(0) = [0.5, 0.5]^T$. The desired trajectory is given as

$$\begin{bmatrix} x_{d1}(t) \\ x_{d2}(t) \end{bmatrix} = \begin{bmatrix} 2\sin\frac{\pi}{2}t \\ 3\cos\frac{\pi}{2}t \end{bmatrix}, \tag{6.2}$$

The predictor is designed as follows

$$\begin{bmatrix} \dot{\hat{x}}_1(t) \\ \dot{\hat{x}}_2(t) \end{bmatrix} = \begin{bmatrix} -6 & 3 \\ 3 & -9 \end{bmatrix} \begin{bmatrix} \hat{x}_1(t) \\ \hat{x}_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ -\sin(\hat{x}_1(t)) \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u} \tag{6.3}$$

$$+L\tilde{\mathbf{x}}(t-\tau),$$

where

$$L = YX^{-T} = \begin{bmatrix} 1.5234 & 0.0062 \\ 0.0062 & 1.5172 \end{bmatrix} \begin{bmatrix} 0.4206 & 0.1450 \\ 0.145 & 0.2756 \end{bmatrix}^{-T} = \begin{bmatrix} 4.4143 & -2.2988 \\ -2.2988 & 6.7173 \end{bmatrix},$$

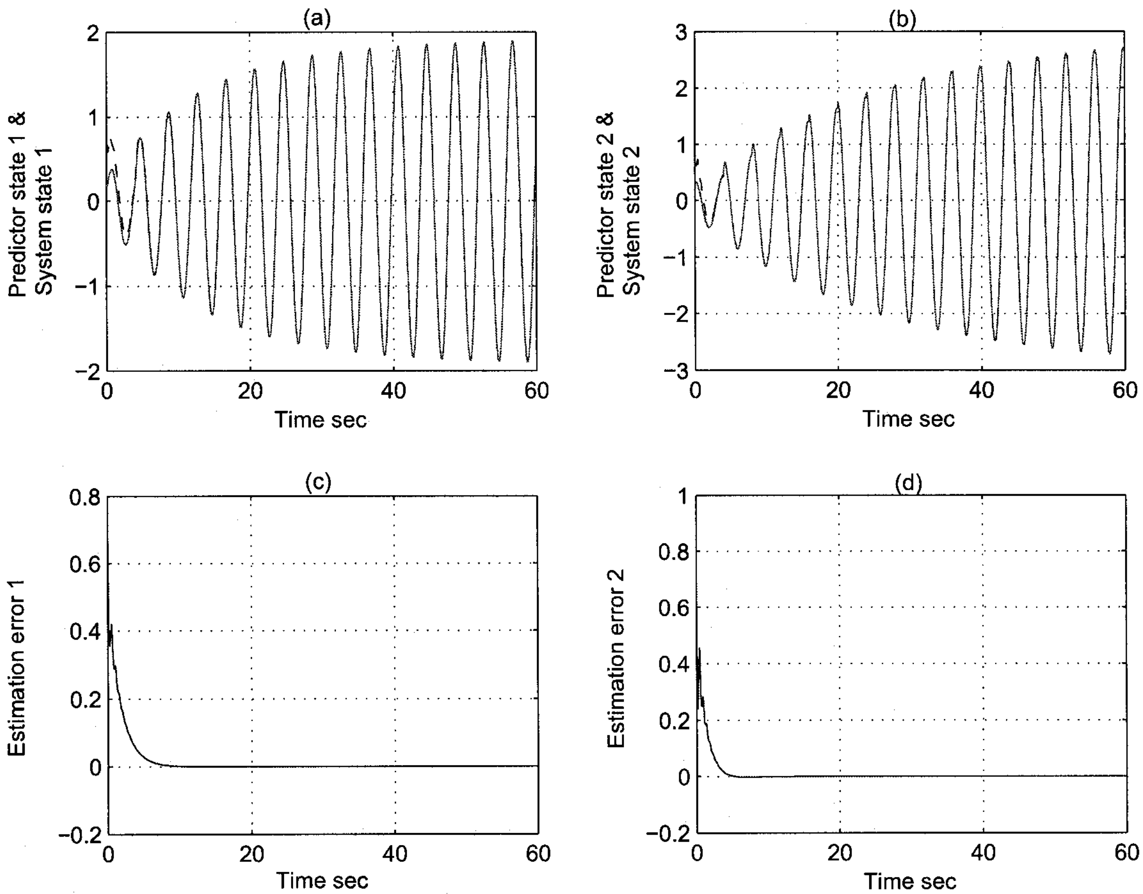

Figure 6.1. The profiles of the maximum estimation error versus time: (a) $\hat{x}_1(t-\tau)$ vs $x_1(t)$; (b) $\hat{x}_2(t-\tau)$ vs $x_2(t)$; (c) $\tilde{e}_1(t) = \hat{x}_1(t-\tau) - x_1(t)$; (d) $\tilde{e}_2(t) = \hat{x}_2(t-\tau) - x_2(t)$

is the solution from (4.16) in Theorem 4.2, $\tilde{\mathbf{x}}(t-\tau) = \hat{\mathbf{x}}(t-2\tau) - \mathbf{x}(t-\tau)$ and $\hat{\mathbf{x}}(0) = [1, 1]^T$. X and Y are matrices defined in Theorem 4.2, the values of X and Y are obtained by using the LMI toolbox of MATLAB.

In this simulation, the input time delay τ is 0.2 seconds, and the sampling time T_s is 0.005 seconds. The repetitive learning controller can be designed as in (5.1); it

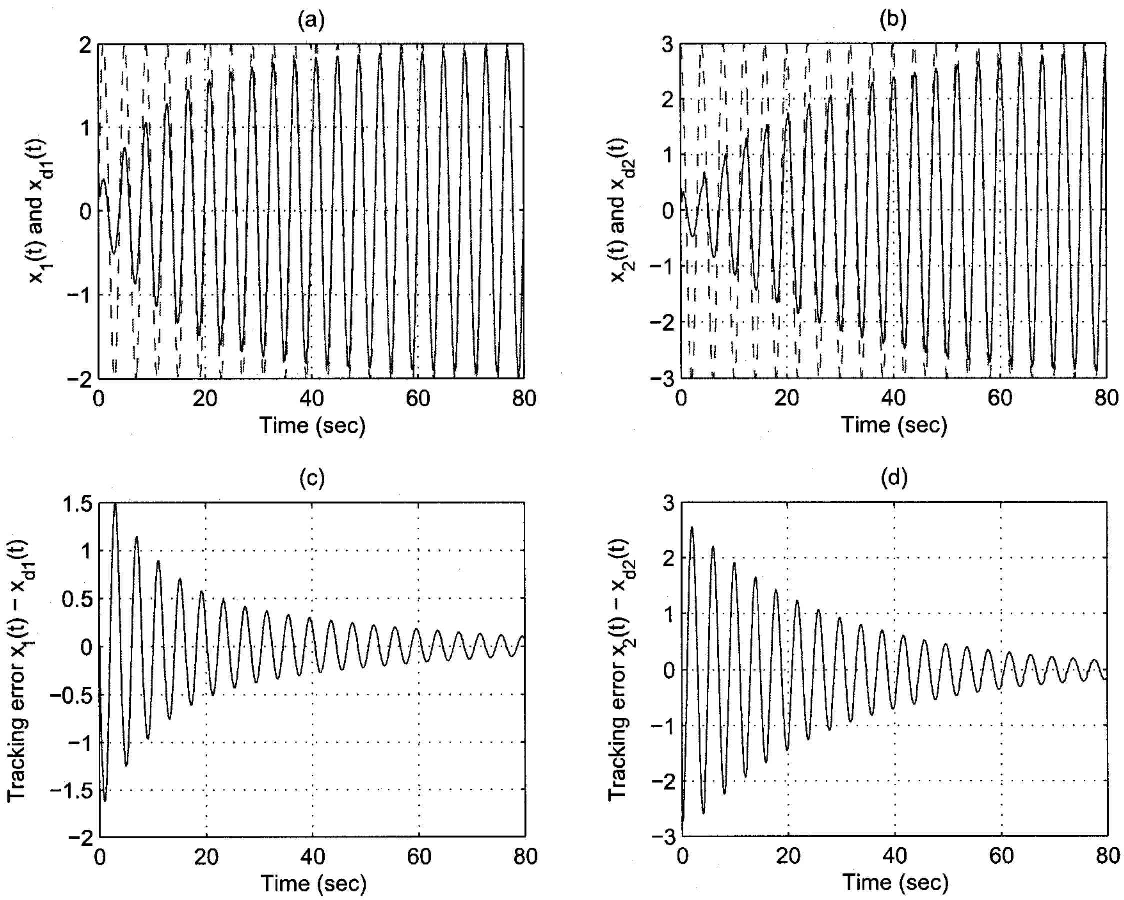

Figure 6.2. The evolution of $x_d(t)$ and $x(t)$: (a) $x_{d1}(t)$ vs $x_1(t)$; (b) $x_{d2}(t)$ vs $x_2(t)$; (c) $x_{d1}(t) - x_1(t)$; (d) $x_{d2}(t) - x_2(t)$

is as follows

$$\mathbf{u}(t) = \mathbf{u}(t-T) + K \begin{bmatrix} x_{d1}(t+\tau) - \hat{x}_1(t) \\ x_{d2}(t+\tau) - \hat{x}_2(t) \end{bmatrix}, \tag{6.4}$$

where the control gain K is designed to satisfy the condition in Theorem 5.1. In this simulation, the control gain is $K = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}$ which satisfies the conditions in Remark 1. Note that $S = \begin{bmatrix} 20.2006 & -2.3767 \\ -39.3153 & 41.0466 \end{bmatrix}$.

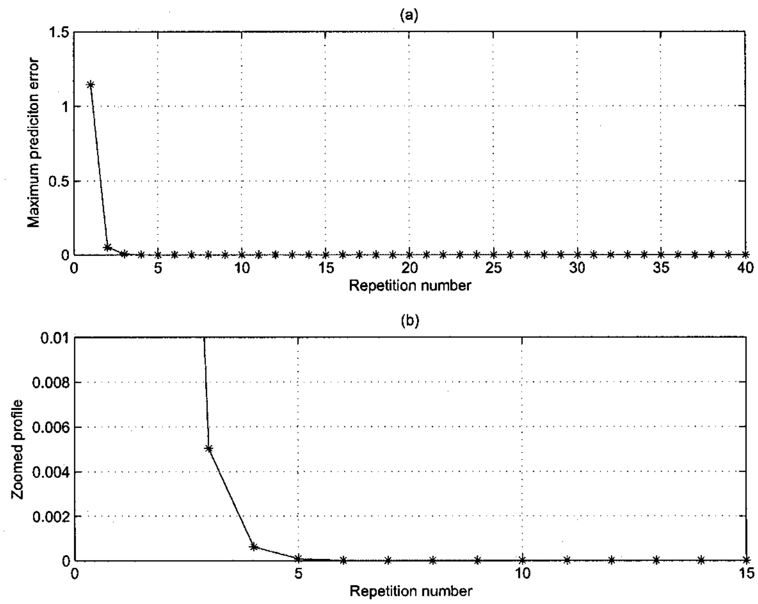

Figure 6.3. The evolution of the maximum prediction error $\tilde{\mathbf{x}}(t)$ versus repetition number

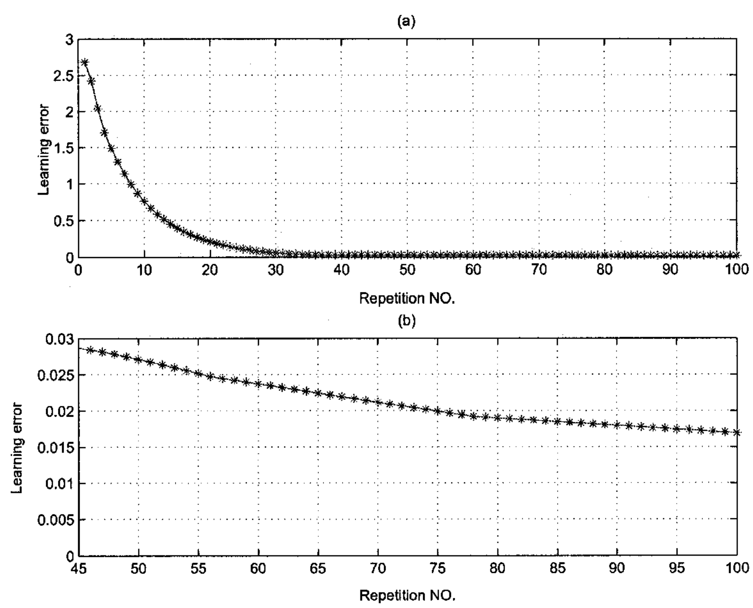

Figure 6.4. The evolution of the maximum learning error $\hat{\mathbf{e}}(t)$ versus repetition number

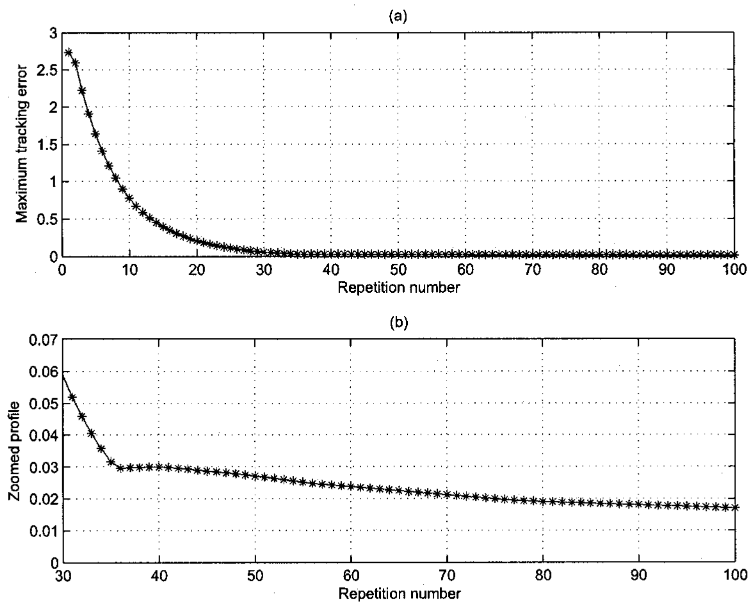

Figure 6.5. The evolution of the maximum tracking error $\mathbf{e}(t)$ versus repetition number

The profiles of the estimation error in Fig.6.1 show that the predictor's output converges to the system state. Fig.6.1.(a) and Fig.6.1.(b) show the profiles of the predicted state and the real nonlinear system state. It is straightforward to observe that the states $\hat{\mathbf{x}}(t)$ and $\mathbf{x}(t+\tau)$ are identical after a very short time which means that the predictor can predict accurately for the nonlinear system. As shown in Fig.6.1.(c) and Fig.6.1.(d), the difference between $\hat{\mathbf{x}}(t)$ and $\mathbf{x}(t+\tau)$ starts becoming zero in less than one second. This good performance is due to an appropriate design of L.

The evolutions of the desired trajectory $\mathbf{x}_d(t)$ and the system state $\mathbf{x}(t)$ are shown in Fig.6.2. It is obvious that the system output converges to the desired trajectory. The system output starts tracking the desired system output perfectly in a short time. From the results, the original control objective has been achieved. The good performance shown by the system is due to the proper design of the learning gain K.

The evolution of the maximum estimation error $\tilde{\mathbf{x}}(t)$ is shown in Fig.6.3. From the figure, it is clear that the trend of $\tilde{\mathbf{x}}(t)$ converges to zero asymptotically as the number of repetitions increases, which further illustrates the results in Theorem 4.1.

The evolution of the maximum learning error $\hat{\mathbf{e}}(t)$ is shown in Fig.6.4. From the figure, it is clear that the trend of $\hat{\mathbf{e}}(t)$ converges to zero asymptotically as the number of repetitions increases, which further illustrates the results in Theorem 5.1.

Fig.6.5 shows the tracking error between the nonlinear system and the desired trajectory. As shown in the figure, the trend of the tracking error converges to zero asymptotically as the number of repetitions increases. This result demonstrates the efficiencies of the controller and the predictor designed in this paper to achieve the control objective.

Chapter 7

Conclusions and Future Work

This thesis mainly dealt with periodic tracking control problems for nonlinear remote control systems while there are transmission delays in the two communication channels: from the controller to the actuator and from the sensor to the controller. Since there exist time delays, the effect of the delays could cause the system to be unstable. In order to solve the problem caused by time delays, in Chapter 4 a predictor is designed on the controller side, to predict the future state of the nonlinear system based on the delayed measurements from the sensor. The convergence of the estimation error of the predictor is ensured. The gain design of the predictor applies linear matrix inequality - LMI techniques developed by the Lyapunov Kravoskii method for time delay systems. In Chapter 5 the repetitive learning control law is designed, based on the feedback error from the predicted state. The proof of the stability is based on a constructed Lyapunov function related to the Lyapunov Kravoskii functional used for the proof of the predictor's convergence.

The techniques are applied to a simulated example, in which the tracking error converges to zero asymptotically due to the proper design of the predictor and the controller. Note that the Lyapunov method and LMI techniques play important roles in ensuring the convergence and performance of the resultant closed-loop system.

Future work will be focusing on the exploration of the application of the learning control theories to a more general networked control environment.

This work carried out in this thesis has generated several problems for future work. *i*) More complicated situations with respect to the time delay τ will be considered in the application of learning control theories to networked control environments, in order to consider more realistically properties of real network induced delays. The effect caused by the packet loss will also be considered. *ii*) Other common models of system plants will be studied. New learning control algorithms will be required to make the systems get good performance. *iii*) A real hardware-based remote control environment will be set up. The theorems proved in our research will be applied to the real environment, so that the results show us whether the theorems can provide us with good performance in practice.

Bibliography

Arimoto, S. (1991). Passivity of robot dynamics implies capability of motor program learning. In: *Proceedings of the International Workshop on Nonlinear and Adaptive Control : Issues in Robotics.* pp. 49–68.

Arimoto, S., S. Kawamura and F. Miyazaki (1984*a*). Bettering operation of robots by learning. *Journal of Robotic Systems* **1**(2), 123–140.

Arimoto, S., S. Kawamura and F. Miyazaki (1984*b*). Iterative learning control for robot systems. In: *Proceedings of IECON, Tokyo, Japan.*

Arimoto, S., S. Kawamura, F. Miyazaki and S. Tamakie (1985). Learning control theory for dynamic systems. In: *Proceedings of of the 24th IEEE Conf. on Decision and Control.* pp. 1375–1380.

Bien, Z. and J. X. Xu (n.d.). *Iterative Learning Control: Analysis, Design, Integration and Application.*

Bondi, P, G Casalino and L. Gambardella (1988). On the iterative learning control theory for robotic manipulators. *IEEE Journal of Robotics and Automation*

4, 14–22.

Burdet, E., L. Rey and A. Codourey (1997). A trivial method of learning control. In: *Preprints of the 5th IFAC symposium on robot control, France.* Vol. 2.

Casalino, G. and G. Bartolini (n.d.). A learning procedure for the control of movements of robotic manipulators. *IASTED Symposium on Robotics and Automation.*

Chen, Y. Q., Z. M Gong and C. Y. Wen (1998). Analysis of a high-order iterative learning control algorithm for uncertain nonlinear system with state delays. *Automatica* **34**(3), 345–353.

Craig, J. J. (1984). Adaptive control of manipulators through repeated trials. In: *Proceedings of ACC,San Diego, CA.*

Dixon, W. E., E. Zergeroglu, D. M. Dawson and B. T. Costic (2002). Repetitive learning control: A lyapunov-based approach. *IEEE Transactions on Systems, Man and Cybernetics* **32**(4), 538–545.

Fung, R., J. Huang, J. Chien and Y Wang (2000). Design and application of continous time controller for rotation mechanisms. *International Journal of Mechanical Sciences* **42**, 1805–1819.

Garden, M (1971). *Learning control of actuators in control systems.* US Patent, US03555252, Jan 1971. Leeds and Northrup Company, Philadelphia, USA.

Gu, K., V. L. Kharitonov and J. Chen (2003). *Stability of Time-Delay Systems.* Birkhauser: Boston.

Guglielmo, K. and N. Sadegh (1996). Theory and implementation of a repetetive robot controller with cartesian trajectory description. *Journal of Dynamic Systems, Measurement, and Control.* **118**, 15–21.

Ham, C., Z. Qu and J. Kaloust (2001). Nonlinear learning control for a class of nonlinear systems. *Automatica* **37**(3), 419–428.

Ham, C., Z. Qu and R. Johnson (2000). A nonlinear iterative learning control for robot manipulators in the presence of actuator dynamics. *International Journal of Robotics and Automation* **15**(3), 107–119.

Hara, S., Y. Yamamoto, T. Omata and M. Nakano (1988). Repetitive control system: A new type servo system for periodic exogenous signals. *IEEE Transactions on Automatic Control* **33**(7), 659–668.

Horowitz, R. (1993). Learning control of robot manipulators. *Transactions of the ASME Journal, Dynamics System, Measurement, and Control* **115**, 402–411.

Horowitz, R., W. Messner and J. B. Moore (1991). Exponential convergence of a learning controller for robot manipulators. *IEEE Transactions on Automatic Control* **36**(7), 890–894.

Jiang, P., H. Chen and Y. Wang (1999). Iterative learning control for glass cutting robot. In: *Proceedings of the 14th IFAC Triennial World Congress.* Vol. 2. pp. 263–268.

Jiang, Y. A., D. J. Clements and T. Hesketh (1995). Adaptive repetitive control of nonlinear system. In: *Proceedings of the 34th Conference on Decision and Control.*

Kaneko, K. and R. Horowitz (1997). Repetitive and adaptive control of robot manipulators with velocity estimation. *IEEE Transactions on Robotics and Automation* **13**, 204–217.

Kim, D. S., Y. S. Lee, W. H. Kwon and H. S. Park (2003). Maximum allowable delay bounds of networked control systems. *Control Engineering Practice,* **11**, 1301–1313.

Kobayashi, Y., T. Kimuara and S. Yanabe (1999). Robust speed control of ultrasonic motor based on h_∞ control with repetitive compensator. *JSME International Journal Series Control* **42**, 884–890.

Lange, F. and G. Hirzinger (1999). Learning accurate path control of industrial robots with joint elasticity. In: *Proceedings IEEE Conference on Robotics and Automation.* pp. 2084–2089.

Lee, J. H., K. S. Lee and W. C. Kim (2000). Model-based iterative learning control with a quadratic criterion for time-varying linear systems. *Automatica* **36**(5), 641–657.

Liang, Y. J. and D. P. Looze (1993). Performance and robustness issues in iterative learning control. In: *Proceedings of the 32nd Conf. on Decision and Control.* pp. 1990–1995.

Longman, R. W. (2000). Iterative learning control and repetitive control for engineering practice. *International Journal of Control* **73**(10), 930–954.

Lozano, R., P. Castillo, P. Garcia and A. Dzul (2004). Robust prediction based control for unstable delay systems: Application to the yaw control of a mini-helicopter. *Automatica* **40**, 603–612.

Mahawan, B. and Z. Luo (2000). Reprtitive control of tracking systems with time-varying periodic references. *International Journal of Control* **73**(1), 1–10.

Moore, K. L. (1993). Iterative learning control for deterministic systems. In: *Industrial Control. Springer-Verlag.*

Moore, K. L. (1998). Iterative learning control-an expository survey. *Applied and Computational Controls, Signal Processing, and Circuits* **1**, 151–214.

Moore, K. L., M. Dahleh and S. P. Bhattacharyya (1992). Iterative learning control: A survey and new results. *Journal of Robotic Systems* **9**(5), 563–594.

Narendra, K.S. and A.M. Annaswamy (1989). *Stable Adaptive Systems.* Prentice-Hall Englewood Cliffs. New Jersey, USA.

Nilsson, J. (1998). *Real time control systems with dealys.* Ph.D. dissertation. Lund Institute of Technology.

Norrlof, M. (2000). *Adaptive iterative learning control algorithms for disturbance rejection.*Technical Report LiTH-ISY-R-2244. Linkopings University.

Norrlof, M. (2000). *Iterative learning control analysis, design, and experiments.* Ph.D. dissertation. Linkopings University.

Pan, Y. J., C. Canudas-de Wit and O. Sename (2004). Predictive controller design for bilateral teleoperation systems with time varying delays. In: *Proceedings of the 43rd IEEE Conference on Decision and Control,*.

Pan, Y. J., H. J. Marquez and T. Chen (2006*a*). Sampled-data iterative learning control for a class of nonlinear networked control systems. In: *Proceedings of the 2006 American Control Conference.* pp. 3494–3499.

Pan, Y. J., H. J. Marquez and T. Chen (2006*b*). Stabilization of remote control systems with unknown time varying delays by LMI techniques. *International Journal of Control* **79**(7), 752–763.

Pan, Y. J. and L. Sheng (2006). Predictor-based repetitive learning control for a class of remote control nonlinear systems (submitted) . *Submitted to International Journal of Robust and Nonlinear Control*

Park, H. S., Y. H. Kim, D. S. Kim and W. H. Kwon (2002). A scheduling method for network based control systems. *IEEE Transactions on Control Systems Technology* **10**, 318–330.

Poloni, M. and G. Ulivi (1991). Iterative learning control of a one-link flexible manipulator. In: *IFAC Robot Control,Austria.* pp. 393–398.

Sadegh, N., Horwitz R., W. W. Kao and M. Tomizuka (1990). A unified approach to the design of adaptive and repetitive controllers for robotic manipulators.

Transactions of the ASME Journal, Dynamics System, Measurement, and Control **112**, 618–629.

Smith, C., K. Takeuchi and M. Tomizuja (1999). Cost effective repetitive controllers for data storage devices. In: *Proceedings of the 14th IFAC World Congress.*

Song, Y., X. Yu, J. X. Xu and G. Chen (2001). Time-delayed chaos control with repetitive learning. In: *Proceedings of the 40th IEEE Conference on Decision and Control.* pp. 4915–4919.

Sun, M. and S. S. Ge (2004). Adaptive repetitive learning control of servo mechanisms. In: *Proceedings of the 5th World Congress on Intelligent Control and Automation.*

Tomizuka, M., T. C. Tsao and K. K. Chew (1989). Discrete-time domain analysis and sythesis of repetetive controllers. *Journal of Dynamic Systems, Measurement, and Control.* **111**, 353–358.

Uchiyama, M (1978). Formulation of high-speed motion pattern of a mechanical arm by trial. *Soc. Instrum. Contr. Eng.* **14**(6), 706–712.

Wang, Y., D. Wang, B. Zhang and Y. Ye (2005). From iterative learning control to robust repetitive learning control. In: *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics Monterey.* pp. 969–974.

Wu, M., Y. He, J. H. She and G. P. Liu (2004). Delay-dependent criteria for robust stability of time-varying delay systems. *Automatica* **40**, 1435–1439.

Xu, J. X. and Z. Qu (1998). Robust iterative learning control for a class of nonlinear systems. *Automatica* **34**(8), 983–988.

Yue, D., Q. L. Han and J. Lam (2005). Network-based robust h_∞ control of systems with uncertainty. *Automatica* **41**, 999–1007.

Zhang, W., M. Branicky and S. Phillips (2001). Stability of networked control systems. *IEEE Control Systems Magzines* **21**(1), 84–99.

Zhou, K. and D. Wang (2001). Digital repetitive learning controller for three-phase cvcf pwm inverter. *IEEE Transactions on Industrial Electronics* **48**(4), 820–830.

Appendix Definitions

Definition .1. *A continuous function* γ: $R^+ \rightarrow R^+$ *is a class* K *function if* $\gamma(0) = 0$ *and if it is strictly increasing. It is said to belong to class* K_∞ *if* $\gamma(p) \rightarrow \infty$ *as* $p \rightarrow \infty$.

Definition .2. *A function* V: $R^n \times R \rightarrow R^+$ *is called locally positive definite if there exists a class* K *function* γ_1: $R^+ \rightarrow R^+$ *such that, for some neighborhood of the origin* $\Omega \subset R^n$,$\gamma_1(\|x(t)\|) \leq V(x(t),t)$, $\forall(x,t) \in \Omega \times R^+$. *Function* V *is said to be locally decrescent if there exists a class* K *function* γ_2: $R^+ \rightarrow R^+$ *such that, for some neighborhood of the origin* $\Omega \subset R^n$, $V(x(t),t) \leq \gamma_2(\|x(t)\|)$, $\forall(x,t) \in \Omega \times R^+$. *The word "locally" is replaced by global if* $\Omega = R^n$. *Function* V *is radially unbounded if* γ_1 *is a class* K_∞ *function.*

Definition .3. *A function* $V : R^n \times R \rightarrow R^+$ *is a Lyapunov function candidate if it is continuously differentiable and if* *i*) *For concluding stability,* $V(x,t)$ *is positive definite.* *ii*) *For concluding uniform asymptotic stability or exponential stability or uniform boundedness or uniform ultimate boundedness, V(x,t) is positive definite and decrescent.* *iii*) *For concluding global stability,* $V(x,t)$ *is globally positive definite and radially unbounded.* *iv*) *For concluding global and uniform asymptotic stability or global exponential stability or global uniform boundedness or global uniform ultimate boundedness,* $V(x,t)$ *is globally positive definite, globally decrescent, radially unbounded.*

Definitions about norm Consider a matrix mapping $L : R^n \rightarrow R^n$. The space of R^n can be equipped with several different norms. The most frequently used norms are $l_1 - norm$, $l_2 - norm$ and $l_\infty - norm$ defined, respectively, by the equations

$$\begin{aligned} \|v\|_1 &= \textstyle\sum_{i=1}^n |v_i| \\ \|v\|_2 &= \sqrt{\textstyle\sum_{i=1}^n |v_i|^2} \\ \|v\|_\infty &= \max_{i\in I} |v_i|. \end{aligned} \tag{.1}$$

where I is the index-set $I = 1, 2, \ldots, n$, $v \in R^n$ and arbitrary, and $v = [v_1, v_2, \ldots, n]^T$. The corresponding operator norms for L become

$$\begin{aligned} \|L\|_1 &= \max_{j\in I} \textstyle\sum_{i=1}^n |L_{ij}| \\ \|L\|_2 &= \sigma_{\max}(L) \\ \|L\|_\infty &= \max_{i\in I} \textstyle\sum_{j=1}^n |L_{ij}|. \end{aligned} \tag{.2}$$

where $\sigma_{\max}(L)$ is the largest singular value of L. Note that frequently $\|L\|_\infty \neq \|L\|_2 \neq \|L\|_1$, and consequently it has to be always made clear which particular norm is being used to measure the convergence properties of the algorithm.