แนวทางการหาค่าเหมาะที่สุดเชิงคอนเวกซ์เพื่อออกแบบการควบคุมแบบเรียนรู้วนรอบคงทนสำหรับระบบเชิงเส้นที่มีความไม่แน่นอนเชิงพารามิเตอร์

นาย ดิง โฮ เหงียน

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2552

A CONVEX OPTIMIZATION APPROACH TO
ROBUST ITERATIVE LEARNING CONTROL DESIGN
FOR LINEAR SYSTEMS WITH PARAMETRIC UNCERTAINTIES

Mr. Dinh Hoa Nguyen

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2009

Thesis Title	A CONVEX OPTIMIZATION APPROACH TO ROBUST ITERATIVE LEARNING CONTROL DESIGN FOR LINEAR SYSTEMS WITH PARAMETRIC UNCERTAINTIES
By	Mr. Dinh Hoa Nguyen
Field of Study	Electrical Engineering
Thesis Advisor	Associate Professor David Banjerdpongchai, Ph.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.... Dean of the Faculty of Engineering
(Associate Professor Boonsom Lerdhirunwong, Dr. Ing.)

THESIS COMMITTEE

.... Chairman
(Assistant Professor Manop Wongsaisuwan, Ph.D.)

.... Thesis Advisor
(Associate Professor David Banjerdpongchai, Ph.D.)

.... Examiner
(Associate Professor Waree Kongprawechnon, Ph.D.)

.... Examiner
(Naebboon Hoonchareon, Ph.D.)

ดิง โฮ เหงียน (A CONVEX OPTIMIZATION APPROACH TO ROBUST ITERATIVE LEARNING CONTROL DESIGN FOR LINEAR SYSTEMS WITH PARAMETRIC UNCERTAINTIES), อ. ที่ปรึกษาวิทยานิพนธ์หลัก: รศ.ดร. เดวิด บรรเจิดพงศ์ชัย, 104 หน้า

การควบคุมแบบเรียนรู้วนรอบเป็นกลยุทธ์ค่อนข้างใหม่ที่ขั้นตอนวิธีการควบคุมถูกออกแบบสำหรับระบบการทำงานแบบซ้ำ ๆ เพื่อปรับปรุงสมรรถนะของระบบหลังจากการวนรอบ การควบคุมแบบเรียนรู้วนรอบผลักดันให้สัญญาณขาออกของระบบสามารถติดตาม แนววิถีที่ต้องการ ณ ทุกตัวอย่างในช่วงเวลาจำกัดและคงตัว อีกทั้ง สามารถกำจัดสัญญาณรบกวนแบบซ้ำ ๆ และลดผลกระทบเนื่องจากความไม่แน่นอนเชิงพารามิเตอร์

ในวิทยานิพนธ์นี้ เรานำเสนอการออกแบบการควบคุมแบบเรียนรู้วนรอบคงทนสำหรับระบบเชิงเส้นที่มีเมทริกซ์มาร์คอฟเป็นฟังก์ชันสัมพรรคของความไม่แน่นอน โดยเฉพาะอย่างยิ่ง เราพิจารณาความไม่แน่นอนเชิงพารามิเตอร์ 3 แบบ ได้แก่ แบบเวลายืนยง แบบแปรผันตามเวลา และแบบแปรผันตามการวนรอบ เรานำเสนอแนวทางการออกแบบขั้นตอนวิธีที่มีเอกภาพและเป็นระบบ เริ่มต้น การออกแบบการควบคุมแบบเรียนรู้วนรอบคงทนมีรูปแบบเป็นปัญหาค่าต่ำสุดค่าสูงสุดโดยใช้เกณฑ์สมรรถนะกำลังสองภายใต้เงื่อนไขบังคับของสัญญาณควบคุมขาเข้า ต่อมา เราผ่อนคลายปัญหาค่าต่ำสุดค่าสูงสุดให้เป็นปัญหาการหาค่าต่ำสุดโดยอาศัยขอบเขตบนของสมรรถนะกรณีเลวสุด การประยุกต์ใช้ภาวะคู่กันลากรานจ์กับปัญหาการหาค่าต่ำสุดนำไปสู่ปัญหาคู่กันที่สมมูลกับการหาค่าเหมาะที่สุดเชิงคอนเวกซ์บนอสมการเมทริกซ์เชิงเส้น เรานำเสนอขั้นตอนวิธีเพื่อออกแบบการควบคุมแบบเรียนรู้วนรอบคงทน อีกทั้ง พิสูจน์สมบัติการลู่เข้าของสัญญาณควบคุมขาเข้า และสัญญาณความคลาดเคลื่อนของระบบ สุดท้าย เราประยุกต์ใช้ขั้นตอนวิธีใหม่กับระบบเชื่อมโยงอ่อนตัวและระบบหอกลั่นภายใต้ความไม่แน่นอนเชิงพารามิเตอร์ การจำลองผลด้วยคอมพิวเตอร์แสดงให้เห็นประสิทธิผลของขั้นตอนวิธีที่นำเสนอ

ภาควิชา วิศวกรรมไฟฟ้า	**ลายมือชื่อนิสิต** ....
สาขาวิชา วิศวกรรมไฟฟ้า	**ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก** ....
ปีการศึกษา 2552	

##4970750421: MAJOR ELECTRICAL ENGINEERING

KEYWORDS: ITERATIVE LEARNING CONTROL / LINEAR SYSTEMS / MIN-MAX PROBLEM / PARAMETRIC UNCERTAINTIES / QUADRATIC PERFORMANCE / LINEAR MATRIX INEQUALITIES /

DINH HOA NGUYEN: A CONVEX OPTIMIZATION APPROACH TO ROBUST ITERATIVE LEARNING CONTROL DESIGN FOR LINEAR SYSTEMS WITH PARAMETRIC UNCERTAINTIES. THESIS ADVISOR: ASSOC. PROF. DAVID BANJERDPONGCHAI, Ph.D, 104 pp.

Iterative learning control (ILC) is a relatively new control strategy in which the control algorithms are designed for repetitively working systems so that the system performance is improved after iterations. ILC drives the system output to the target trajectory at all samples in a fixed, finite time interval. Moreover, it rejects the repeated disturbances and reduces the effects due to parametric uncertainties.

In this thesis, we propose robust ILC design for linear systems whose Markov matrices are affine functions of the uncertainties. In particular, three types of parametric uncertainties are considered in this thesis including time-invariant, time-varying, and iteration-varying. We develop a unified, systematic approach to design robust ILC algorithms for these types of uncertainties. First, the robust ILC design problem is formulated as a min-max problem using a quadratic performance criterion subject to constraints of the iterative control input update. Then, the min-max problem is relaxed to be a minimization problem by employing an upper-bound of the worst-case performance. Applying Lagrangian duality to this minimization problem leads to a dual problem which can be reformulated as a convex optimization problem over linear matrix inequalities. ILC algorithms are given afterward and the convergences of the control input as well as the system error are proved. Finally, we apply the proposed ILC algorithms to a flexible link system and distillation column in presence of parametric uncertainties. The computer simulation illustrates the effectiveness of the proposed algorithms.

Department: Electrical Engineering .... **Student's signature:** ....
Field of study: Electrical Engineering .... **Principal Advisor's signature:** ....
Academic year: 2009 ....

Acknowledgments

From bottom of my heart, I would like to send the most sincere thanks to my advisor, Assoc. Prof. Dr. David Banjerdpongchai, for his so kind and enthusiastic care during the two years of my graduate study as a Master student in Control Systems, Chulalongkorn University. With his remarkable patience and brilliant, careful guidance as well as his strong support, I have an absolute confidence to develop my ideas and make them be true results in this thesis. He always spend his time and his best ability to reply my inquiries in both research and ordinary life. I am also indebted to him for introducing me to the area of Iterative Learning Control which then becomes my research interest.

Next, I would like to gratefully thank all professors in the Control Systems Research Laboratory who organize and keep a friendly, well-equipped working environment but strictly ruled with an fully informative research database and modern experiment sets. It makes me comfortable while studying here and help me a lot in my research. Special thanks to the students in the laboratory who discuss with me about their troubles and mine in our research and our study. Those discussions are effective not only to them but also to me for a better understanding about different subjects.

My thesis would not have been completed without the useful and contributed comments of the committee members, Assist. Prof. Manop Wongsaisuwan, Doctor Neabboon Hoonchareon and Assoc. Prof. Waree Kongprawechnon. I greatly appreciate their conscientious reading and attention of my thesis in spite of their tight schedules. I also gratefully acknowledge the financial support from JICA project for AUN/SEED-Net through collaborative research program and from the Department of Electrical Engineering, Faculty of Engineering, Chulalongkorn University.

Finally, I would like to dedicate this work to my parents for their endless love and perpetual support. Words are never enough to express my gratitude to them.

Contents

List of Tables

List of Figures

Notations

Symbols

R	The set of real numbers
Z^+	The set of positive integers
I	The identity matrix
$\mathbf{S}^+$	The set of symmetric positive semi-definite matrices
dom	The domain of a function

Acronyms

ILC	Iterative Learning Control
ILC-TI	Iterative learning control algorithm for linear systems with time-invariant parametric uncertainties
ILC-TV	Iterative learning control algorithm for linear systems with time-varying parametric uncertainties
ILC-IV	Iterative learning control algorithm for linear systems with iteration-varying parametric uncertainties
LMI	Linear Matrix Inequality
MIMO	Multi Input Multi Ouput
SISO	Single Input Single Ouput

CHAPTER I

INTRODUCTION

1.1 Iterative learning control

In many industrial applications nowadays, the control command is executed and terminated after a finite time period, and then is repeated over and over. Since conventional control designs cannot always give the desired performance, a new control synthesis has been developed for this kind of repetitively working systems. This opens a new area in the control theory called Iterative Learning Control (ILC). Japanese scientists are the first ones who studied it when they considered robotic systems, however, the first papers about learning control were published in Japanese, hence, it was not known widely until 1984, Arimoto *et al.* presented the terminology Iterative Learning Control (ILC) and then the first ILC algorithm in their English paper [1]. After that, ILC was immediately gained a big consideration from researchers, and has been a very active research area until now because there are many systems not only robotics systems operate repetitively which can be controlled using ILC algorithms, such as rotary systems, chemical engineering, power electronics. In addition, ILC is proved to be a methodology that can achieve perfect tracking control, thus, it is a very effective control methodology for periodically working systems, especially when it is combined with traditional control methods. Such an example of an ILC system is provided in Figure 1.1.

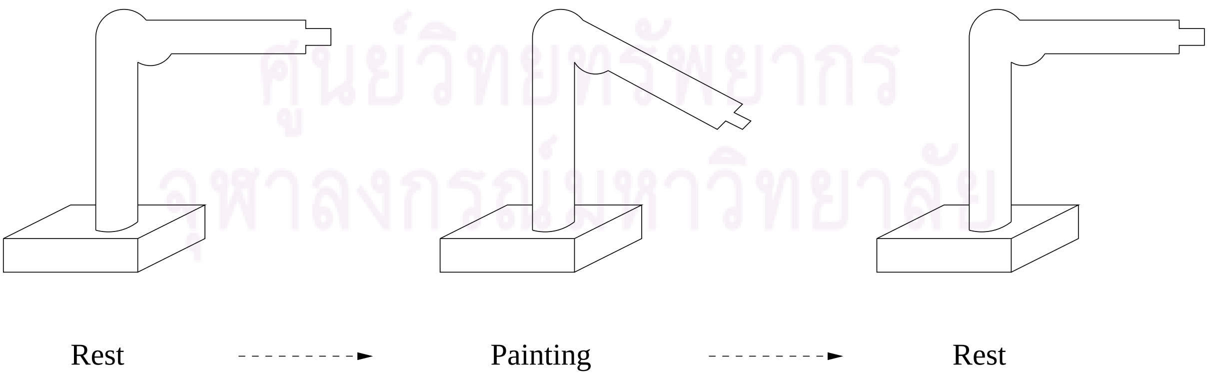

Figure 1.1: An example of a system working repetitively

In the literature, numerous of definitions of ILC have been given [2], [3], however, the common point of these definitions which is also the key idea of ILC is that the information of previous executions such as the control input and the error between the desired trajectory and the system output are utilized to make the system performance improved throughout next iterations. The structure of an ILC system is illustrated in Figure 1.2.

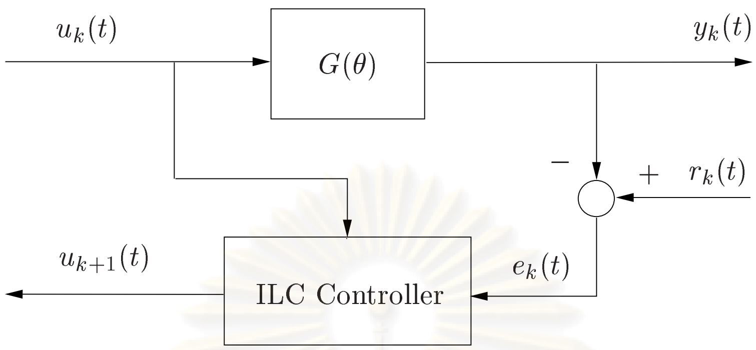

Figure 1.2: The structure of ILC system.

Some advantages of ILC controllers are a simple structure and the ability to track a reference input without having an exact model of the system. ILC controllers work on a fixed, finite time interval and drive the system output to the desired trajectory for all time samples as the iteration increases while conventional controllers try to converge to the desired trajectory as time goes to infinity. Figure 1.3 shows a conventional output feedback controller.

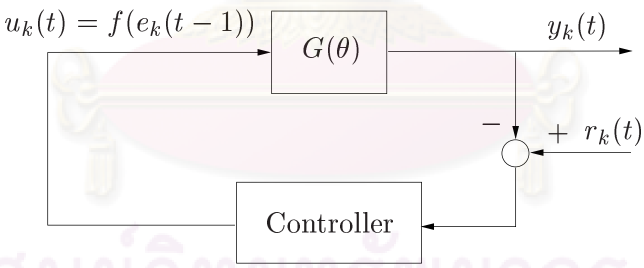

Figure 1.3: Output feedback control.

Looking at Figure 1.2 and Figure 1.3, we can see obviously the difference between an ILC system and an output feedback system. An ILC controller regulates the system output for all samples in each iteration by employing the system errors and the control inputs at all samples of previous iterations whereas the conventional output feedback controller only uses the system error at one sample to modify the system output for the consecutive sample in the time interval. In addition, when there are parametric uncertainties or repeated disturbances, conventional controllers cannot always achieve a desired performance whereas ILC can obtain perfect tracking. Thus, ILC has great advantages over conventional controllers when dealing with repetitive operations.

1.2 Literature Review

We recommend the survey articles by Bristow *et al.* [4] and Ahn *et al.* [5] for the thorough overview of ILC terminologies and definitions, ILC problems and solving techniques as well as the up-to-date development of ILC from the starting milestone. Among a wide range of ILC research areas, optimal ILC [6], [7–10] and robustness of ILC algorithms are the main issues. Such a well-known optimal ILC method namely norm-optimal ILC proposed by Amann *et al.* in [8] that minimizes a quadratic cost function consisting of N samples ahead current sample. This leads to a prediction part in the update formula for control input. Moreover, there are more and more optimization-based methods to design ILC rules, for instance, gradient-based, Newton-based, genetic algorithm [11–13]. On the other hand, robust ILC is also a well-established area. Numerous types of uncertainties, disturbances, noises have been studied in the literature and many ILC algorithms have been proposed to rule out the undesired effects of the exogenous signals or uncertainties in the system. From the view point of this thesis, we only investigate the parametric uncertainties in linear systems. In addition, we categorize the uncertainties into three types: time-invariant, time-varying and iteration-varying.

1.2.1 Time-invariant parametric uncertainties

Some techniques in robust control theory such as μ-synthesis [14], H_∞ approach [15–17], feedback-based approach [18–20] have been applied to design ILC algorithms for linear systems with time-invariant uncertainties. Nevertheless, these robust ILC problems are formulated in continuous-time and frequency domain, and they implicitly use the assumption that the time domain is infinite whilst in practice, the systems controlled by an ILC algorithm work on a finite time interval and the algorithm is implemented in discrete-time domain.

1.2.2 Time-varying parametric uncertainties

Otherwise, with time-varying uncertainties, many ILC designs have been developed for nonlinear time-varying systems, for instance [20–22], whereas only some other ILC designs have been applied to linear time-varying (LTV) systems, such as [23–25]. In [23], Arif *et al.* have used the error prediction method to construct the ILC law for LTV systems with unknown but bounded disturbances. Thereto, Tan *et. al* [24] have proposed an ILC strategy for LTV systems with state disturbances and measurement noise at the output. The ILC algorithms in [23] and [24] are applicable to systems perturbed by disturbances, but parametric uncertainties have not been considered in the dynamic system. In [25], Hladowski *et al.* have developed an ILC law for linear systems with time-varying uncertainties by solving a generalized eigenvalue problem (GEVP). Their approach applies the linear repetitive process stability theory to design iterative input updates and the control law is specified by a linear combination of the state change and the output error. Moreover, an upper bound of the time-varying uncertainties can be determined by the solution of the GEVP.

1.2.3 Iteration-varying parametric uncertainties

While numerous articles on the robust ILC design study systems containing time-invariant uncertainties and several others investigate the systems with time-varying uncertainties, a few papers examine the systems in the presence of iteration-varying uncertainties. To the best of our knowledge, the robust ILC designs with iteration-varying uncertainties have been explored by Ahn, Moore and Chen in a series of articles [26–28] which also appear in a research monograph [2]. They formulate the plant models with iteration-varying uncertainty in a super-vector framework, then convert the Markov matrix of the plant into a form of bounded additive model uncertainty. As a result, the Markov matrix belongs to an interval set and all the ILC updates are designed for this class of uncertain interval plants.

1.3 Scope of Thesis

1. Studying Iterative Learning Control, a relatively new control area.
2. Designing efficient and applicable robust Iterative Learning Control algorithms for linear systems with various types of parametric uncertainties.
3. Applying the designed algorithms to some physical models of robotics and industrial processes.

1.4 Contributions

In this thesis, we expect the following research outcome

1. A review of robust Iterative Learning Control for both researchers and engineers who have some or even no exprience in designing Iterative Learning Control algorithms.
2. A unified and systematic approach to design robust Iterative Learning Control algorithms for a certain class of uncertain linear systems in the presence of various types of parametric uncertainties including time-invariant, time-varying, iteration-varying uncertainties using convex optimization and linear matrix inequalities.
3. Application of developed robust ILC algorithms to some physical models in robotics, industrial processes such as flexible link, distillation column.

1.5 Conclusion

In Chapter 1, we present a brief overview of ILC and the difference between ILC and the conventional control methods. Then, we give a literature review of robust iterative learning control for linear systems with different types of uncertainties. Lastly, the scope of this thesis is introduced and our contributions are listed.

CHAPTER II

MATHEMATICS PRELIMINARY

This chapter presents the basic definitions and properties of the mathematic tools used in this thesis. Section 2.1 introduces the most basic terminologies and their properties in convex optimization problems. The Lagrangian duality is presented in Section 2.2. Finally, Section 2.3 provides a brief review of linear matrix inequalities. The results presented is taken from [29].

2.1 Convex optimization problems

Definition 1. *(Affine set)*
A set $\boldsymbol{C} \in \boldsymbol{R}^n$ is affine if for any points $x_1, \ldots, x_k \in \boldsymbol{C}$, and $\alpha_1, \ldots, \alpha_k \in \boldsymbol{R}$ such that $\alpha_1 + \cdots + \alpha_k = 1$, the point $\alpha_1 x_1 + \cdots + \alpha_k x_k$ is also belongs to $\boldsymbol{C}$. Moreover, the point $\alpha_1 x_1 + \cdots + \alpha_k x_k$ is called an affine combination of $x_1, \ldots, x_k$.

Definition 2. *(Convex set)*
A set $\boldsymbol{C} \in \boldsymbol{R}^n$ is convex if for any points $x_1, x_2 \in \boldsymbol{C}$, and $\alpha \in \boldsymbol{R}$ such that $0 \leq \alpha \leq 1$, we have $\alpha x_1 + (1-\alpha)x_2 \in \boldsymbol{C}$.

Like in the case of affine set, the point $\alpha_1 x_1 + \cdots + \alpha_k x_k$ with $0 \leq \alpha_1, \ldots, \alpha_k \leq 1$, $\alpha_1 + \cdots + \alpha_k = 1$ is refered as a convex combination of $x_1, \ldots, x_k$. Then, the convex hull of the set $\mathbf{C}$ is the set of all convex combination of $\mathbf{C}$.

Definition 3. *(Convex function)*
*A function $f : \boldsymbol{R}^n \rightarrow \boldsymbol{R}$ is convex if **dom** f is a convex set and for all $x, y \in$ **dom** f, $0 \leq \alpha \leq 1$, we have*

$$f\left(\alpha x + (1-\alpha)y\right) \leq \alpha f(x) + (1-\alpha) f(y) \tag{2.1}$$

When the inequality (2.1) becomes strict with $x \neq y$ and $0 < \alpha < 1$, the function f is strictly convex.

Theorem 1. *(First-order condition)*
*Suppose that $f : \boldsymbol{R}^n \rightarrow \boldsymbol{R}$ is diffentiable, then f is convex if and only if **dom** f is convex and*

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \tag{2.2}$$

*holds for all $x, y \in$ **dom** f.*

In addition, we have the following theorem to verify whether a function is convex or not.

Theorem 2. *(Second-order condition)*
*Suppose that $f : \boldsymbol{R}^n \to \boldsymbol{R}$ is twice diffentiable, then f is convex if and only if **dom** f is convex and its Hessian matrix is positive semidefinite, i.e,*

$$\nabla^2 f(x) \geq 0 \tag{2.3}$$

*for all $x \in$ **dom** f.*

After considering the definitions of convex sets, convex functions, we now come up with convex optimization problems. An optimization problem is denoted by the following form.

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0,\ i = 1, \dots, m \\ & g_i(x) = 0,\ i = 1, \dots, p \end{aligned} \tag{2.4}$$

where $x \in \mathbf{R}^n$ is the optimization variable,

$f_0(x)$ is the cost function or objective function,

$f_i(x) \leq 0$ are the inequality constraints and $f_i(x)$ are the inequality constraint functions,

$g_i(x) = 0$ are the equality constraints and $g_i(x)$ are the equality constraint functions.

The set of points for which the objective and all constraint functions are defined,

$$\mathbf{D} = \bigcap_{i=0}^{m} \mathbf{dom}\ f_i \ \cap \ \bigcap_{i=1}^{p} \mathbf{dom}\ g_i \tag{2.5}$$

is called the domain of the optimization problem (2.4). A point $x \in \mathbf{D}$ is feasible if it satisfies all the constraints $f_i(x) \leq 0,\ i = 1, \dots, m,\ g_i(x) = 0,\ i = 1, \dots, p$, and the set of all feasible points is called feasible set or constraint set. Moreover, the optimization problem (2.4) is said to be feasible if there exists at least one feasible point and infeasible otherwise.

The optimal value of (2.4) is defined as

$$\gamma^* = \inf \{ f_0(x) |\ f_i(x) \leq 0,\ i = 1, \dots, m,\ g_i(x) = 0,\ i = 1, \dots, p \} \tag{2.6}$$

Then, if (2.4) is infeasible, $\gamma^* = \infty$, on the other hand, if $\gamma^* = -\infty$, the optimization problem (2.4) is unbounded below.

Now, we consider a special class of (2.4), namely, convex optimization problems in which, the cost function $f_0(x)$ must be convex, the inequality constraint functions $f_i(x),\ i = 1, \dots, m$ must be convex, and the equality constraint functions $g_i(x),\ i = 1, \dots, p$ must be affine, i.e, $g_i(x) = a_i^T x - b_i,\ i = 1, \dots, p$. Therefore, the convex optimization problems have the following standard form.

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0,\ i = 1, \dots, m \\ & a_i^T x = b_i,\ i = 1, \dots, p \end{aligned} \tag{2.7}$$

The advantage of convex optimization problem is that if x^* is a local minimizer of the function f_0, then it is also the global minimizer of f_0. Since the inequality and equality constraint functions are convex, the set of inequality constraints and the set of equality constraints are also convex, hence, the feasible set of convex optimization problem (2.7) is convex. That means, in a convex optimization problem, we minimize a convex function over a convex set, and therefore, according to Theorem 1, if we can find a minimizer in the feasible set, it will be the global solution of (2.7). Moreover, in comparision with other nonlinear optimization problems, the convex optimization problems are much easier to solve with efficient algorithms. In the next parts, the involved techniques to solve the convex optimization problems will be presented [29].

2.2 Lagrangian duality

Consider the optimization problem (2.4) in which the domain of the problem is non-empty and (2.4) need not to be convex. Since (2.4) is a nonlinear optimization problem and it has some constraints, so, in general, it is very hard to solve (2.4) directly. Therefore, we should take find a way to eliminate the constraints and make the optimization problem easier to solve. Lagrangian duality is such this way. The most strong point of Lagrangian duality is the convexity of the dual problem regardless of the convexity of the primal problem. Hence, usually, the dual problem can be solve more favourably.

We define a Lagrangian $L : \mathbf{R}^n \times \mathbf{R}^m \times \mathbf{R}^p \to \mathbf{R}$ associated with (2.4) as follows.

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i + \sum_{i=1}^{p} \nu_i g_i \tag{2.8}$$

where λ_i's are Lagrange multipliers associated with the inequality constraints $f_i(x) \leq 0,\ i = 1, \ldots, m$,

ν_i's are Lagrange multipliers associated with the equality constraints $g_i(x) = 0,\ i = 1, \ldots, p$.

The vectors $\lambda = [\lambda_1, \ldots, \lambda_m]^T,\ \nu = [\nu_1, \ldots, \nu_p]^T$ are called the dual variables or Lagrange multiplier vectors of the optimization problem (2.4).

Next, the Lagrange dual function $h : \mathbf{R}^m \times \mathbf{R}^p \to \mathbf{R}$ is defined as the minimum of $L(x, \lambda, \nu)$ over $x \in \mathbf{R}^n$ with $\lambda \in \mathbf{R}^m, \nu \in \mathbf{R}^p$.

$$h(\lambda, \nu) = \inf_{x \in \mathbf{D}} L(x, \lambda, \nu) = \inf_{x \in \mathbf{D}} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i + \sum_{i=1}^{p} \nu_i g_i \right) \tag{2.9}$$

Since the dual function is the pointwise infimum of a family of affine functions of (λ, ν), it is always concave whether the optimization problem (2.4) is convex or not.

The dual function give us a lower bound of the optimal value γ^* of (2.4), i.e., for any $\lambda \geq 0$ and any ν, we have,

$$h(\lambda, \nu) \leq \gamma^* \tag{2.10}$$

Therefore, to find the best lower bound or even exact value of γ^*, we need to come up with the following optimization problem,

$$\begin{aligned} \max \quad & h(\lambda, \nu) \\ \text{s.t.} \quad & \lambda \geq 0 \end{aligned} \tag{2.11}$$

This problem is called the Lagrange dual problem of the optimization problem (2.4). Note that, $g(\lambda, \nu)$ is concave and the constraint $\lambda \geq 0$ is convex, so, the dual problem (2.12) is always convex. This is a very important property of Lagrangian duality as mentioned before.

Let ξ^* be the optimal value of problem (2.12), then the following inequality always holds.

$$\xi^* \leq \gamma^* \tag{2.12}$$

The property (2.12) is called weak duality and the difference $\gamma^* - \xi^*$ is referred to as optimal duality gap correspondingly. As a nature, once Lagrangian duality is applied to an optimization problem, we expect that there is no gap between the optimal value of the primal problem and the optimal value of the dual problem, i.e., the strong duality holds or $\gamma^* = \xi^*$. Nevertheless, the strong duality does not hold for every optimization problem, hence, under what conditions the strong duality holds is a very considerable question. There are some constraints qualifications to establish the conditions for strong duality, but we will use one simple condition, named Slater's condition. The Slater's condition states that the there exists a feasible point in the interior of **domD** such that the equality constraints in (2.4) satisfied and the inequality constraints in which the constraint functions are non-affine become strict. It leads to the following theorem.

Theorem 3. *(Slater's theorem)*
Consider the optimization problem (2.4) and its dual problem (2.12). Then, the strong duality holds whenever the Slater's condition holds.

In the next part, we provide the optimality condition for the primal problem and the dual problem through the Karush-Kuhn-Tucker (KKT) conditions.

Theorem 4. *(KKT conditions) Suppose that x^* and λ^*, ν^* are primal and dual optimal points with zero duality gap, then, they satisfy following conditions.*

$$\begin{aligned} & f_i(x^*) \leq 0, \; i = 1, \ldots, m \\ & g_i(x^*) = 0, \; i = 1, \ldots, p \\ & \lambda_i^* \geq 0, \; i = 1, \ldots, m \\ & \lambda_i^* f_i(x^*) = 0, \; i = 1, \ldots, m \\ & \nabla f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) + \sum_{i=1}^{p} \nu_i^* \nabla g_i(x^*) = 0. \end{aligned} \tag{2.13}$$

Moreover, when the optimization problem (2.4) is convex, the KKT conditions are also sufficient.

2.3 Linear matrix inequalities

An LMI has the form [29–32]

$$F(x) = F_0 + \sum_{i=1}^{m} x_i F_i > 0 \tag{2.14}$$

where $x \in \mathbf{R}^m$ is the variable and the symmetric matrices $F_0, F_1, \ldots, F_m \in \mathbf{R}^{n\times n}$ are given. In fact, the variable in the LMI (2.14) can be a matrix instead of a vector since many control problems lead to functions of matrices.

Theorem 5. *(Non-strict Schur complement [32])*
Let Q and R be symmetric matrices, then the following statements are equivalent.

- $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0,$
- $R \geq 0,\ Q - SR^{\dagger}S^T \geq 0,\ S(I - RR^{\dagger}) = 0,$

where $R^{\dagger}$ is the pseudo-inverse of matrix R.

Theorem 6. *(Strict Schur complement [29, 30])*
Let the matrix $F(x)$ in LMI (2.14) be partitioned as follows.

$$F(x) = \begin{bmatrix} Q(x) & S(x) \\ S(x)^T & R(x) \end{bmatrix} \tag{2.15}$$

where $Q(x) = Q(x)^T,\ R(x) = R(x)^T$ and $S(x)$ depends affinely on x. Then, (2.14) holds if and only if

$$R(x) > 0,\ Q(x) - S(x)R(x)^{-1}S(x)^T > 0. \tag{2.16}$$

In addition, (2.14) holds if and only if

$$Q(x) > 0,\ R(x) - S(x)^T Q(x)^{-1} S(x) > 0. \tag{2.17}$$

Since $F(x)$ is an affine function of variable x, we can easily verify that (2.14) defines a convex set. Therefore, in an optimization problem, if the cost function is convex and the constraints can be reformulated as LMIs, we get a convex optimization problem. Subsequently, using the tools of convex optimization as mentioned above, the optimization problem now is much easier to solve.

2.4 Robust ILC for linear systems with parametric uncertainties

2.4.1 System description and modeling

Consider an uncertain discrete-time linear system described by the following state-space model

$$\begin{cases} x(t+1) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) \end{cases} \tag{2.18}$$

where $x \in \mathbf{R}^r$, $u \in \mathbf{R}^q$, $y \in \mathbf{R}^p$ are state vector, control input, and output of system (2.18),
A, B, C are system matrices with appropriate dimensions,
$t \in [0, N]$ is the time sample, N is the number of samples.

When the control task is repeated over and over, the system now works not only on time domain but also on iteration domain. Consequently, the system model (2.18) becomes

$$\begin{cases} x_k(t+1) &= Ax_k(t) + Bu_k(t), \\ y_k(t) &= Cx_k(t) \end{cases} \tag{2.19}$$

where k is the iteration index.

Given a reference input $r(t)$, our control objective is to design the control input $u_k(t)$ so that the system output $y_k(t)$ tracks the reference input after some iterations with a specified error. The designed ILC algorithm should discard the effect of the parametric uncertainties in the system, and guarantee the expected system performance. Among many control strategies, lifting technique is an effective method that has been widely utilized to design ILC algorithm [2], [12], [13], [6], [33]. All the sample times are taken into account in a vector makes it the strong point of this technique since an ILC system works in a finite time interval and the designed ILC algorithm should drive the system output to the target trajectory at all sample times. Thus, in the following part, we will use the lifting technique to reformulate the system in a super-vector framework and design our ILC algorithm bases on this formulation.

Define

$$\begin{aligned} \mathbf{y}_k &= \left[y_k(1)^T \quad y_k(2)^T \quad \dots \quad y_k(N)^T\right]^T \\ \mathbf{u}_k &= \left[u_k(0)^T \quad u_k(1)^T \quad \dots \quad u_k(N-1)^T\right]^T \\ \mathbf{x}_k &= \left[x_k(1)^T \quad x_k(2)^T \quad \dots \quad x_k(N)^T\right]^T \end{aligned} \tag{2.20}$$

where $\mathbf{y}_k, \mathbf{u}_k, \mathbf{x}_k$ are the corresponding super vectors including of the system output, control input and state vector at all sample times in the time interval $[0, N]$, in the kth iteration.

The system (2.19) now can be reformulated in the super-vector framework as follows.

$$\mathbf{y}_k = G\mathbf{u}_k \tag{2.21}$$

where G is a Markov matrix.

Let us denote $h(t)$ as the impulse responses of the system (2.19), namely,

$$h(t) \quad = CA^{t-1}B, \quad t = 1, 2, \dots, N. \tag{2.22}$$

Without loss of generality, assume that the relative degree of system (2.19) equals to 1. Then, G can be described as follows.

$$G = \begin{bmatrix} h(1) & 0 & 0 & \cdots & 0 \\ h(2) & h(1) & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ h(N) & h(N-1) & \cdots & \cdots & h(1) \end{bmatrix} \tag{2.23}$$

Suppose that in the system (2.19), there are parametric uncertainties which can be denoted by $\theta_1, \theta_2, \ldots, \theta_m$, or simply denoted by a vector uncertainty θ. These parametric uncertainties can be in any system matrices A, B, or C, so, we will use the notation $A_\theta, B_\theta, C_\theta$ to describe their effect to the system matrices. Define a sequence of the impulse responses in the presence of uncertainties as follows.

$$h(t, \theta) = C_\theta A_\theta^{t-1} B_\theta,\ t = 1, 2, \ldots, N \tag{2.24}$$

As a result, the Markov matrix G is a function of these uncertainties. Therefore, from now on, we use the notation $G(\theta)$ in stead of G.

In our approach, we are interested in the systems satisfying the following conditions.

A1. $\theta \in \Theta$ where Θ is a set of bounded parametric uncertainties. Without loss of generality,

$$\Theta = \{\theta : \|\theta\|_\infty \leq 1\}.$$

A2. The Markov matrix $G(\theta)$ of system (2.19) is an affine function of θ.

Let e be the error between the output y and the reference input r defined as $e(t) = r(t) - y(t)$. Thus, we have

$$\mathbf{e}_k \quad = \quad \mathbf{r} - \mathbf{y}_k$$

where

$$\begin{aligned} \mathbf{r} &= \begin{bmatrix} r(1)^T & r(2)^T & \ldots & r(N)^T \end{bmatrix}^T \\ \mathbf{e}_k &= \begin{bmatrix} e_k(1)^T & e_k(2)^T & \ldots & e_k(N)^T \end{bmatrix}^T \end{aligned}$$

Note that the reference input is invariant with respect to iterations, so the iteration index k is dropped out from the super-vector $\mathbf{r}$. Hence, the error update model of the system (2.21) is

$$\mathbf{e}_{k+1} = \mathbf{e}_k - G(\theta)\Delta\mathbf{u}_{k+1} \tag{2.25}$$

where $\Delta\mathbf{u}_{k+1} = \mathbf{u}_{k+1} - \mathbf{u}_k$ is the difference of the control input between iterations which also is defined as the control input update of the system.

Since the elements of the Markov matrix are the impulse responses of the discrete-time system (2.19) at different samples, we can experimentally analyse these impulse responses and model them as an affine function of the uncertainties. Hence, the Markov matrix will be an affine function of the uncertainties. Let G_0 is the nominal Markov matrix of the system (2.19), $G_1, G_2, \ldots, G_m$ are the uncertainties of Markov matrix corresponding to the parametric uncertainties $\theta_1, \theta_2, \ldots, \theta_m$. We propose the following procedure to determine the matrices $G_0, G_1, \ldots, G_m$.

Step 1. Determine the nominal impulse responses $h(t,0)$.

Set $\theta = 0$ and calculate a sequence of impulse responses

$$h(t,0) = C_0 A_0^{t-1} B_0,\ t = 1, 2, \ldots, N.$$

Step 2. Determine the nominal Markov matrix G_0.

G_0 is computed as follows.

$$G_0 = \begin{bmatrix} h(1,0) & 0 & 0 & \cdots & 0 \\ h(2,0) & h(1,0) & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ h(N,0) & h(N-1,0) & \cdots & \cdots & h(1,0) \end{bmatrix} \quad (2.26)$$

Step 3. Determine the uncertainties of impulse responses.

For each $i = 1, \ldots, m$, we discretize the uncertainty interval $[-1, 1]$ into $\Theta_i = \{-1, -1+d, -1+2d, \ldots, 1\}$ where $d = 2/n$ and $n+1$ is the number of discretized points. Let $\theta_i \in \Theta_i$ whereas $\theta_j = 0, j \neq i, j = 1, 2, \ldots, m$. Afterward, we calculate the impulse responses $h(t,\theta)$ for all $\theta_i \in \Theta, t = 1, 2, \ldots, N$, and define

$$h_{\mathrm{l}}(t,i) = \min_{\theta_i \in \Theta_i} h(t,\theta),\ t = 1, 2, \ldots, N \quad (2.27)$$

$$h_{\mathrm{u}}(t,i) = \max_{\theta_i \in \Theta_i} h(t,\theta),\ t = 1, 2, \ldots, N \quad (2.28)$$

Now, the uncertainties of impulse responses are computed as follows.

$$h_{\Delta}(t,i) = \frac{1}{2}[h_{\mathrm{u}}(t,i) - h_{\mathrm{l}}(t,i)],\ t = 1, 2, \ldots, N$$

Step 4. Determine the uncertainties of Markov matrices $G_1, \ldots, G_m$.

For each $i = 1, \ldots, m$, G_i can be expressed in terms of $h_{\Delta}(t,i)$ as follows.

$$G_i = \begin{bmatrix} h_{\Delta}(1,i) & 0 & 0 & \cdots & 0 \\ h_{\Delta}(2,i) & h_{\Delta}(1,i) & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ h_{\Delta}(N,i) & h_{\Delta}(N-1,i) & \cdots & \cdots & h_{\Delta}(1,i) \end{bmatrix}. \quad (2.29)$$

2.4.2 The robust ILC design problem

In real applications, there are some restrictions on the control inputs which can be described by the following constraints.

C1. Bounded magnitude: $\mathbf{u}_{\mathrm{l}} \leq \mathbf{u}_{k+1} \leq \mathbf{u}_{\mathrm{h}}$.

C2. Bounded rate w.r.t. time index: $\delta\mathbf{u}_{\mathrm{l}} \leq \delta\mathbf{u}_{k+1} \leq \delta\mathbf{u}_{\mathrm{h}}$.

C3. Bounded rate w.r.t. iteration index: $\Delta\mathbf{u}_{\mathrm{l}} \leq \Delta\mathbf{u}_{k+1} \leq \Delta\mathbf{u}_{\mathrm{h}}$, $\Delta\mathbf{u}_{\mathrm{l}} \leq 0, \Delta\mathbf{u}_{\mathrm{h}} > 0$.

where $\delta\mathbf{u}_{k+1} = \begin{bmatrix} u_{k+1}(0) \\ u_{k+1}(1) - u_{k+1}(0) \\ \vdots \\ u_{k+1}(N-1) - u_{k+1}(N-2) \end{bmatrix} = J\mathbf{u}_{k+1}$ with $J = \begin{bmatrix} I & 0 & \cdots & 0 & 0 \\ -I & I & \cdots & 0 & 0 \\ 0 & -I & \ddots & \vdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -I & I \end{bmatrix}$.

Rewrite the constraints C1–C3 as

$$\Pi \Delta \mathbf{u}_{k+1} \leq \phi \tag{2.30}$$

where $\Pi = \begin{bmatrix} -I \\ I \\ -J \\ J \end{bmatrix}$, $\phi = \begin{bmatrix} -\underline{\Delta}\mathbf{u}_{\mathrm{l}} \\ \underline{\Delta}\mathbf{u}_{\mathrm{h}} \\ -\delta\mathbf{u}_{\mathrm{l}} + J\mathbf{u}_k \\ \delta\mathbf{u}_{\mathrm{h}} - J\mathbf{u}_k \end{bmatrix}$, $\underline{\Delta}\mathbf{u}_{\mathrm{l}} = \max\{\mathbf{u}_{\mathrm{l}} - \mathbf{u}_k, \Delta\mathbf{u}_{\mathrm{l}}\}$, $\underline{\Delta}\mathbf{u}_{\mathrm{h}} = \min\{\mathbf{u}_{\mathrm{h}} - \mathbf{u}_k, \Delta\mathbf{u}_{\mathrm{h}}\}$.

Note that (2.30) is an affine inequality of $\Delta\mathbf{u}_{k+1}$. In the controller design procedure, normally, a linearized model of a non-linear dynamic system is utilized and the system is considered to work around an operating point. Therefore, the constraints on the control input $\mathbf{u}_{k+1}$ and its change with time $\delta\mathbf{u}_{k+1}$ should be employed to force the system work in a linearized region around the operating point. In addition, the constraint on the control input's change with iterations $\Delta\mathbf{u}_{k+1}$ guarantee the smooth running of the system throughout executions. It is very important that the sudden changes should be avoided to ensure the safety in the system.

To design the robust ILC algorithm, we use the quadratic performance criterion

$$J_{k+1} = \mathbf{e}_{k+1}^T Q \mathbf{e}_{k+1} + \Delta\mathbf{u}_{k+1}^T R \Delta\mathbf{u}_{k+1} \tag{2.31}$$

where Q, R are symmetric, positive definite matrices. Then, the design of control input is formulated as a min-max problem

$$\min_{\Delta\mathbf{u}_{k+1} \in U_{k+1}} \max_{\theta \in \Theta} J_{k+1} \tag{2.32}$$

where U_{k+1} is a convex set defined by (2.30).
Now, substitute (2.25) into J_{k+1}, we have

$$J_{k+1} = \Delta\mathbf{u}_{k+1}^T \left(R + G(\theta)^T Q G(\theta)\right) \Delta\mathbf{u}_{k+1} - 2\mathbf{e}_k^T Q G(\theta) \Delta\mathbf{u}_{k+1} + \mathbf{e}_k^T Q \mathbf{e}_k$$

Since $\mathbf{e}_k$ is known from previous iteration, in the design, we can consider the cost function without $\mathbf{e}_k^T Q \mathbf{e}_k$, namely,

$$J_{k+1} = \Delta\mathbf{u}_{k+1}^T \left(R + G(\theta)^T Q G(\theta)\right) \Delta\mathbf{u}_{k+1} - 2\mathbf{e}_k^T Q G(\theta) \Delta\mathbf{u}_{k+1} \tag{2.33}$$

2.4.3 Methodology

In this thesis, we employ a unified approach as described in the flowchart of Figure 2.1 to design robust ILC algorithms for uncertain linear systems. The initial min-max problem (2.32) is hard to solve directly. First of all, we find an upper bound of the maximization problem, then combine the problem of finding the least-upper bound of the maximization

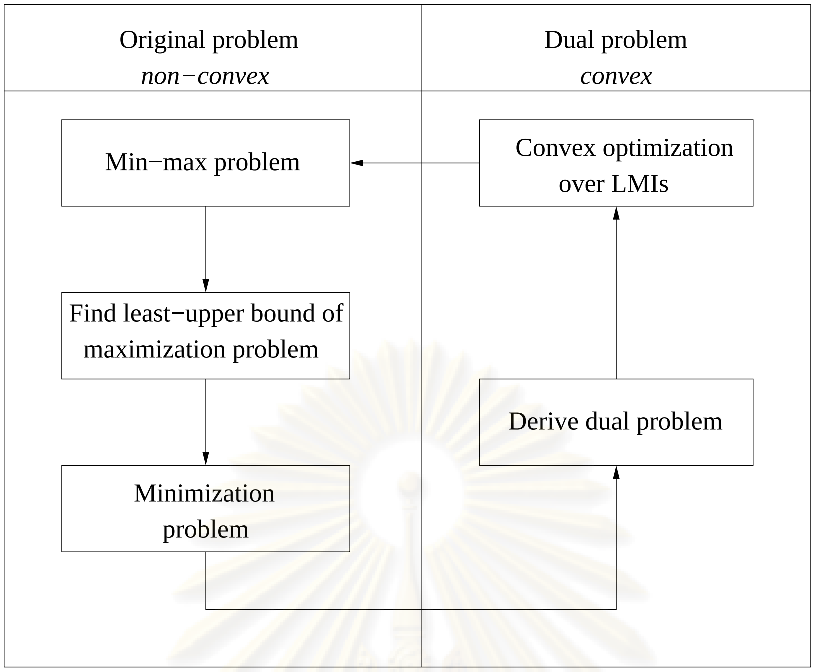

Figure 2.1: The methodology to design robust ILC algorithms for uncertain linear systems.

problem and the initial minimization problem. As a result, the min-max problem (2.32) becomes a minimization problem. The minimization problem is not a convex optimization, therefore, its dual problem is considered instead. Using Schur complement [29], the dual problem is reformulated as a convex optimization problem over linear matrix inequalities whose solution is obtained utilizing available convex optimization solvers. Applying Lagrange duality, the iterative input update is achieved as minimizing the Lagrangian. Hence, at each iteration, the control input update is based on the solution of the dual problem. This makes our ILC controller having a dynamic structure, not a fixed one as usually seen in many articles.

2.5 Conclusion

In this chapter, the mathematical tools employed in the thesis are briefly introduced. We present the convex optimization problem and linear matrix inequality with their definitions and main properties utilized in the thesis. Next, the system description are given. We consider discrete-time linear systems with parametric uncertainties controlled by an ILC controller. Then, the system is reformulated in a super-vector framework which incorporates the system information at all samples in the time interval. Afterward, a procedure is proposed to experimentally analyse the impulse responses of the system in the presence of parametric uncertainties to obtain an affine Markov model of the system. Consequently, the robust ILC design problem is formulated as a min-max optimization problem. And finally, we propose a methodology to derive the ILC controller as a sub-optimal solution of the min-max problem.

CHAPTER III

TIME-INVARIANT PARAMETRIC UNCERTAINTIES

Robustness is one of the critical issues in control systems including ILC systems. Consequently, it is requisite to develop robust algorithms for ILC systems. Some techniques in robust control such as μ-synthesis [14], H_∞ approach [15], [16], feedback-based approach [18–20] were applied to design ILC algorithms for uncertain linear systems. Nevertheless, they usually formulate the robust ILC design problems in either infinite continuous time domain or frequency domain. In addition, they implicitly assume that the time domain is infinite, whilst, in practice, the systems controlled by an ILC algorithm work on a finite time interval and the algorithm is implemented in discrete-time domain.

This chapter aims to design a novel ILC algorithm for linear systems with time-invariant parametric uncertainties in a finite discrete-time domain. Our algorithm is built in a super-vector framework and the robust ILC design is formulated as a min-max problem using a quadratic performance. This quadratic performance criterion has been considered in a number of robust ILC designs [33–35]. In [33], Lee *et al.* have considered a robust ILC design problem for a class of constrained linear systems with parametric uncertainty but they have not presented an explicit ILC algorithm. Recently, we have proposed an approach to the robust ILC design for linear systems with time-invariant parametric uncertainty [35]. However, the approach in [35] works for linear systems containing a single uncertainty and at each iteration, two convex optimization problems need to be solved. In this paper, we generalize the results in [35] to be applicable for linear systems with multiple uncertainties and reduce the number of optimization problems in the design algorithm. An explicit formula for iterative input update is given and at each iteration, only one convex optimization problem is solved. To solve the min-max problem, we first determine an upper bound of the worst-case performance, then consider the dual problem which can be transformed into convex optimization over linear matrix inequalities (LMIs).

In particular, we investigate linear systems with time-invariant parametric uncertainties:

$$G(\theta) = G_0 + G_1\theta_1 + G_2\theta_2 + \cdots + G_m\theta_m \tag{3.1}$$

where G_0 is the nominal matrix, $G_1, \ldots, G_m$ are dynamic matrices, and θ_i, $i = 1, \ldots, m$ are parametric uncertainties.

3.1 The worst-case performance analysis

Using the system model (3.1), we get

$$\begin{aligned} G(\theta)^T Q G(\theta) &= \left(G_0^T + \sum_{i=1}^{m} G_i^T \theta_i \right) Q \left(G_0 + \sum_{j=1}^{m} G_j \theta_j \right) \\ &= \sum_{i=1}^{m} \sum_{j=1}^{m} G_i^T Q G_j \theta_i \theta_j + \sum_{i=1}^{m} \left(G_i^T Q G_0 + G_0^T Q G_i \right) \theta_i + G_0^T Q G_0 \end{aligned} \quad (3.2)$$

Moreover,

$$\begin{aligned} \mathbf{e}_k^T Q G(\theta) \Delta \mathbf{u}_{k+1} &= \mathbf{e}_k^T Q \left(G_0 + \sum_{i=1}^{m} G_i \theta_i \right) \Delta \mathbf{u}_{k+1} \\ &= \sum_{i=1}^{m} \mathbf{e}_k^T Q G_i \Delta \mathbf{u}_{k+1} \theta_i + \mathbf{e}_k^T Q G_0 \Delta \mathbf{u}_{k+1} \end{aligned} \quad (3.3)$$

Substituting (3.2) and (3.3) into (2.33) results in

$$J_{k+1} = \theta^T P \theta + 2\theta^T q + r \quad (3.4)$$

where
$P = (P_{ij})_{i,j=\overline{1,m}}$ with $P_{ij} = \Delta \mathbf{u}_{k+1}^T G_i^T Q G_j \Delta \mathbf{u}_{k+1}$,
$q = (q_i)_{i=\overline{1,m}}$ with $q_i = \Delta \mathbf{u}_{k+1}^T \left(\frac{G_i^T Q G_0 + G_0^T Q G_i}{2} \right) \Delta \mathbf{u}_{k+1} - \mathbf{e}_k^T Q G_i \Delta \mathbf{u}_{k+1}$,
$r = \Delta \mathbf{u}_{k+1}^T \left(G_0^T Q G_0 + R \right) \Delta \mathbf{u}_{k+1} - 2\mathbf{e}_k^T Q G_0 \Delta \mathbf{u}_{k+1}$.

Rewrite the maximization problem in (2.32) as

$$\max_{z \in Z} z^T H z \quad (3.5)$$

where $z = \begin{bmatrix} \theta \\ 1 \end{bmatrix}$, $H = \begin{bmatrix} P & q \\ q & r \end{bmatrix}$, $Z = \left\{ z = \begin{bmatrix} \theta \\ 1 \end{bmatrix} \middle| \theta \in \Theta \right\}$.
If there exists a diagonal matrix T such that $T \geq H$, then

$$z^T H z \leq z^T T z = \sum_{i=1}^{m} t_i z_i^2 + t_{m+1} \leq \sum_{i=1}^{m+1} t_i = \text{trace } T$$

where t_i's, $i = 1, 2, \ldots, m+1$ are elements on the diagonal of matrix T.
Consequently, the least upper bound of (3.5) can be found by solving the following minimization problem

$$\begin{aligned} \min \quad & \text{trace } T \\ \text{s.t.} \quad & T \geq H \\ & T \text{ is diagonal} \end{aligned} \quad (3.6)$$

Remark 1. When the system (5.1) contains a single uncertainty, an analytical solution of (3.6) has been derived, then the minimization problem in (2.32) is solved to give the iterative input update [35,36]. Nevertheless, for system (5.1) with multiple uncertainties, it is difficult to obtain an analytical solution of (3.6) using the same technique. Therefore, a new ILC approach is proposed in the next section to deal with multiple uncertainties [37].

3.2 LMI-based ILC algorithm

Replacing (3.6) into (2.32) and combining two minimization problems, the iterative input update $\Delta\mathbf{u}_{k+1}$ can be calculated by solving the following minimization problem.

$$\begin{aligned} \min \quad & \text{trace } T \\ \text{s.t.} \quad & T \geq H \\ & T \text{ is diagonal} \\ & \Delta\mathbf{u}_{k+1} \in U_{k+1} \end{aligned} \tag{3.7}$$

This is a minimization problem with variables T and $\Delta\mathbf{u}_{k+1}$. To solve (3.7), we will solve its dual problem. First, let us reformulate (3.7) as follows. Rewrite T as

$$T = \sum_{i=1}^{m+1} t_i F_i$$

where F_i is a matrix with the same dimension as T having the ith diagonal element equal to 1 and the other elements equal to 0, $i = \overline{1, m+1}$. Then, (3.7) becomes

$$\begin{aligned} \min \quad & \mathbf{1}^T t \\ \text{s.t.} \quad & H - \sum_{i=1}^{m+1} t_i F_i \leq 0 \\ & \Pi \Delta\mathbf{u}_{k+1} \leq \phi \end{aligned} \tag{3.8}$$

where $t = [t_1, t_2, \ldots, t_{m+1}]^T$, $\mathbf{1}$ is the $(m+1) \times 1$ vector whose elements are all equal to 1.

Define a Lagrangian

$$L(t, \Delta\mathbf{u}_{k+1}, W, \nu) = \mathbf{1}^T t + \text{tr}\left(\left(H - \sum_{i=1}^{m+1} t_i F_i\right) W\right) + \nu^T(\Pi\Delta\mathbf{u}_{k+1} - \phi) \tag{3.9}$$

where $W \in \mathbf{S}_+^{m+1}$, in which, $\mathbf{S}_+^{m+1}$ is the set of symmetric semi-definite positive matrices with dimension $(m+1) \times (m+1)$, ν is a vector.

Next, to obtain a dual function, we find the minimum of the Lagrangian with respect to t and $\Delta\mathbf{u}_{k+1}$. It is straightforward to see that

$$\begin{aligned}\inf_{t,\Delta\mathbf{u}_{k+1}} L(t,\Delta\mathbf{u}_{k+1},W,\nu) &= \inf_t\left\{\mathbf{1}^T t - \mathrm{tr}\left(\sum_{i=1}^{m+1} t_i F_i W\right) - \nu^T\phi\right\} + \inf_{\Delta\mathbf{u}_{k+1}}\left\{\mathrm{tr}\,(HW) + \nu^T\Pi\Delta\mathbf{u}_{k+1}\right\}\\ &= -\nu^T\phi + \inf_t\left\{\sum_{i=1}^{m+1} t_i(1-\mathrm{tr}(F_iW))\right\} + \inf_{\Delta\mathbf{u}_{k+1}}\left\{\mathrm{tr}\,(HW) + \nu^T\Pi\Delta\mathbf{u}_{k+1}\right\}\\ &= -\nu^T\phi + \inf_{\Delta\mathbf{u}_{k+1}}\left\{\mathrm{tr}\,(HW) + \nu^T\Pi\Delta\mathbf{u}_{k+1}\right\} \qquad (3.10)\end{aligned}$$

Note that (3.10) is obtained when

$$\mathrm{tr}\,(F_iW) = 1\ \forall\ i = \overline{1,m+1} \Leftrightarrow w_{ii} = 1\ \forall\ i = \overline{1,m+1}$$

with w_{ii}'s, $i = \overline{1,m+1}$ are elements on the diagonal of matrix W.
Moreover,

$$\begin{aligned}\mathrm{tr}\,(HW) &= \sum_{i=1}^{m+1}\sum_{j=1}^{m+1} h_{ij}w_{ij}\\ &= \sum_{i=1}^{m}\sum_{j=1}^{m} h_{ij}w_{ij} + 2\sum_{i=1}^{m} h_{i,m+1}w_{i,m+1}\\ &\quad + h_{m+1,m+1}w_{m+1,m+1}\end{aligned}$$

where $h_{ij}, w_{ij}, i,j = 1,2,\ldots,m+1$ are elements of corresponding matrices H, W.

$$\sum_{i=1}^{m}\sum_{j=1}^{m} h_{ij}w_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{m} \Delta\mathbf{u}_{k+1}^T G_i^T Q G_j \Delta\mathbf{u}_{k+1} w_{ij}$$

$$\begin{aligned}&\sum_{i=1}^{m} h_{i,m+1}w_{i,m+1}\\ &= \sum_{i=1}^{m}\Delta\mathbf{u}_{k+1}^T\left(\frac{G_i^TQG_0 + G_0^TQG_i}{2}\right)\Delta\mathbf{u}_{k+1}w_{i,m+1}\\ &\quad - \sum_{i=1}^{m}\mathbf{e}_k^TQG_i\Delta\mathbf{u}_{k+1}w_{i,m+1}\end{aligned}$$

$$\begin{aligned}&h_{m+1,m+1}w_{m+1,m+1}\\ &= \Delta\mathbf{u}_{k+1}^T\left(G_0^TQG_0 + R\right)\Delta\mathbf{u}_{k+1} - 2\mathbf{e}_k^TQG_0\Delta\mathbf{u}_{k+1}\end{aligned}$$

Accordingly,

$$\begin{aligned}\mathrm{tr}\,(HW) &= \Delta\mathbf{u}_{k+1}^T\widehat{G}\Delta\mathbf{u}_{k+1}\\ &\quad -2\left(\sum_{i=1}^{m}\mathbf{e}_k^TQG_iw_{i,m+1} + \mathbf{e}_k^TQG_0\right)\Delta\mathbf{u}_{k+1}\end{aligned}$$

where

$$\widehat{G} = \sum_{i=1}^{m}\sum_{j=1}^{m} G_i^T Q G_j w_{ij} + \sum_{i=1}^{m} (G_i^T Q G_0 + G_0^T Q G_i) w_{i,m+1} + G_0^T Q G_0 + R \tag{3.11}$$

Therefore,

$$\operatorname{tr}(HW) + \nu^T \Pi \Delta\mathbf{u}_{k+1} = \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + \beta^T \Delta\mathbf{u}_{k+1} \tag{3.12}$$

where

$$\beta = \Pi^T \nu - 2\sum_{i=1}^{m} G_i^T Q \mathbf{e}_k w_{i,m+1} - 2G_0^T Q \mathbf{e}_k \tag{3.13}$$

Hence,

$$\inf_{\Delta\mathbf{u}_{k+1}} \left\{ \operatorname{tr}(HW) + \nu^T \Pi \Delta\mathbf{u}_{k+1} \right\} = -\frac{1}{4}\beta^T \widehat{G}^{-1}\beta \tag{3.14}$$

with the optimal value of $\Delta\mathbf{u}_{k+1}$ is

$$\Delta\mathbf{u}_{k+1}^* = -\frac{1}{2}\widehat{G}^{-1}\beta \tag{3.15}$$

Thus,

$$\inf_{t,\Delta\mathbf{u}_{k+1}} L(t, \Delta\mathbf{u}_{k+1}, W, \nu) = -\nu^T\phi - \frac{1}{4}\beta^T \widehat{G}^{-1}\beta \tag{3.16}$$

Consequently, the dual problem of (3.7) is

$$\begin{aligned} \min \quad & 4\nu^T\phi + \beta^T \widehat{G}^{-1}\beta \\ \text{s.t.} \quad & \widehat{G} \geq 0 \\ & W \in \mathbf{S}_+^{m+1} \\ & w_{ii} = 1,\ \nu \geq 0 \end{aligned} \tag{3.17}$$

which is equivalent to the following optimization problem

$$\begin{aligned} \min \quad & \rho \\ \text{s.t.} \quad & 4\nu^T\phi + \beta^T \widehat{G}^{-1}\beta \leq \rho \\ & \widehat{G} \geq 0 \\ & W \in \mathbf{S}_+^{m+1} \\ & w_{ii} = 1,\ \nu \geq 0 \end{aligned} \tag{3.18}$$

Using Schur complement [29], we can rewrite the dual problem (3.18) as the following LMI problem

$$\begin{aligned} \min \quad & \rho \\ \text{s.t.} \quad & \begin{bmatrix} \widehat{G} & \beta \\ \beta^T & \rho - 4\nu^T\phi \end{bmatrix} \geq 0 \\ & W \in \mathbf{S}_+^{m+1} \\ & w_{ii} = 1,\ \nu \geq 0 \end{aligned} \tag{3.19}$$

This problem can be solved using available software such as `cvx` [38]. The stopping criteria for iterative solution procedure are as follows.

$$\|\mathbf{e}_k\| \leq \epsilon \tag{3.20}$$

$$k = \texttt{iter\_max} \tag{3.21}$$

where ϵ is a tolerance chosen by the designer and `iter_max` is the maximum number of iterations.

Finally, we propose the following algorithm for the robust ILC design.

Algorithm 1. *An LMI algorithm for linear systems with time-invariant parametric uncertainties (ILC-TI)*

1. *Set* $k := 0$, $\boldsymbol{u}_k := 0$, *and measure* $\boldsymbol{e}_k$.
2. *Solve the LMI problem according to (3.19).*
3. *Calculate* $\Delta\boldsymbol{u}_{k+1}$ *according to (3.15).*
4. *Apply* $\boldsymbol{u}_{k+1}$ *to the system and measure* $\boldsymbol{e}_{k+1}$.
5. *If (3.20) or (3.21) is true, then stop the iteration, else, set* $k := k+1$, *return to step 2.*

Remark 2. There might be a conservatism in the proposed robust ILC design since an upper bound of the worst-case performance is used in the maximization problem (3.5). However, the algorithm appears to work well as demonstrated in the numerical example.

3.3 Convergence properties

Theorem 7. *Under assumptions A1-A2 and constraints C1-C3, the control input* $\boldsymbol{u}_k$ *of system (3.1) converges.*

Proof. Let

$$V(\mathbf{e}_k) = \min_{\Delta\mathbf{u}_{k+1}\in U_{k+1}} \max_{\theta\in\Theta} J_{k+1} \tag{3.22}$$

with J_{k+1} is in (2.33). Then, $V(\mathbf{e}_k) \geq 0 \;\forall\; k$ since $J_{k+1} \geq 0 \;\forall\; k$.

We have,

$$V(\mathbf{e}_k) \leq J_{k+1}|_{\Delta\mathbf{u}_{k+1}=0} = \mathbf{e}_k^T Q \mathbf{e}_k$$

Suppose that θ_k^* is the optimizer of the maximization problem at the kth iteration, hence,

$$\mathbf{e}_k^T Q \mathbf{e}_k \leq \mathbf{e}_k^T(\theta_k^*) Q \mathbf{e}_k(\theta_k^*) = V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k$$

Therefore,

$$V(\mathbf{e}_k) \;\;\leq\;\; V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k \tag{3.23}$$

Inequality (3.23) leads to

$$V(\mathbf{e}_k) + \sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \quad \leq \quad V(\mathbf{e}_0) \tag{3.24}$$

Since $V(\mathbf{e}_k) \geq 0$, we get

$$\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \quad \leq \quad V(\mathbf{e}_0) < \infty \tag{3.25}$$

Moreover, because R is positive definite, $\Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \geq 0 \,\forall\, i$ and the sequence $\left\{\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i\right\}$ is non-decreasing. Combine with (3.25), it deduces that $\left\{\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i\right\}$ converges. Accordingly,

$$\begin{aligned}
\lim_{k\to\infty} \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k &= \lim_{k\to\infty} \left(\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i - \sum_{i=1}^{k-1} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \right) \\
&= \lim_{k\to\infty} \sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i - \lim_{k\to\infty} \sum_{i=1}^{k-1} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \\
&= 0.
\end{aligned}$$

It implies that $\Delta\mathbf{u}_k \to 0$ as $k \to \infty$. Thus, $\{\mathbf{u}_k\}$ converges. □

Theorem 8. *Under assumptions A1-A2 and constraints C1-C3, the error $\mathbf{e}_k$ of system (3.1) converges.*

Proof. We have,

$$\begin{aligned}
\|G(\theta)\| &= \|G_0 + G_1\theta_1 + G_2\theta_2 + \cdots + G_m\theta_m\| \\
&\leq \|G_0\| + \|G_1\theta_1\| + \cdots + \|G_m\theta_m\| \\
&\leq \|G_0\| + \|G_1\| + \cdots + \|G_m\|
\end{aligned}$$

Hence, $\|G(\theta)\|$ is bounded. It leads to

$$G(\theta)\Delta\mathbf{u}_{k+1} \to 0 \text{ as } k \to \infty.$$

Equivalently,

$$(\mathbf{e}_k - \mathbf{e}_{k+1}) \to 0 \text{ as } k \to \infty.$$

This results in the convergence of $\{\mathbf{e}_k\}$. □

3.4 Numerical example

Consider the following system with transfer function

$$G(s) = \frac{1}{15s^2 + 8s + 1} + \theta_1 \frac{0.8e^{-s}}{5s + 1} + \theta_2 \frac{0.5e^{-s}}{2s + 1} \tag{3.26}$$

where θ_1, θ_2 are the uncertain parameters, $\theta_1, \theta_2 \in [-1, 1]$. Since the design of our ILC algorithm is built in discrete time domain, in particular, the super-vector framework, the system model (3.26) should be discretized and rewritten in the form of a Markov matrix as in equation (2.21). In this example, the sampling time is chosen to be 1 second and the number of samples is 41. The target reference trajectory is

$$r(t) = \begin{cases} 0, & t \in [0,5] \cup [36,40] \\ 0.1(t-5), & t \in [6,15] \\ 1, & t \in [16,25] \\ 1 - 0.1(t-25), & t \in [26,35] \end{cases} \tag{3.27}$$

The constraints of control inputs are specified by

$$\mathbf{u}_l = -2, \mathbf{u}_h = 2, \delta\mathbf{u}_l = -3, \delta\mathbf{u}_h = 3, \Delta\mathbf{u}_l = -2, \Delta\mathbf{u}_h = 2. \tag{3.28}$$

The design parameters are chosen as follows: $Q = I_1, R = 0.02I_2$ where I_1, I_2 are identity matrices with appropriate dimension. For the stopping criteria, we choose $\epsilon = 0.01$ and `iter_max` $= 20$. In the simulation, the parameters θ_1, θ_2 are randomly selected from the uncertainty interval and they are time-invariant. In our work, we use the software `cvx` [38] to solve the LMI problem (3.19).

The design results are shown in the Figures 3.1– 3.5. Figures 3.1, 3.2, and 3.3 show that the control input satisfies all input constraints (3.28). Moreover, Figures 3.2 exhibits that $\delta u_k(t)$ tends to converge whereas $\Delta u_k(t)$ tends to decay as illustrated in Figure 3.3. Hence, the control input converges. Figure 3.4 displays that the output of system (3.26) converges to the desired reference trajectory but there are still small errors of the system output at the 39th–40th samples after executing ILC for 20 iterations. The system error converges as demonstrated in Figure 3.5.

3.5 Conclusion

Chapter 3 presents the details to design an ILC controller for linear systems in the presence of time-invariant parametric uncertainties. The design procedure follows the methodology described in Chapter 2. The first step is the worst-case performance analysis in which, the affine Markov model of the system is introduced and a least upper bound of the maximization problem is achieved. Accordingly, the min-max problem is relaxed to a minimization one. Then, in the second step, we consider the dual problem of the minimization problem derived in the first step, and reformulate it as a convex optimization problem with LMI constraints. An LMI-based ILC algorithm is given as a result. Next, the convergence of the control input and the system error is proved. Lastly, a generic example is provided in section 3.3 to illustrate the effectiveness of the proposed ILC algorithm.

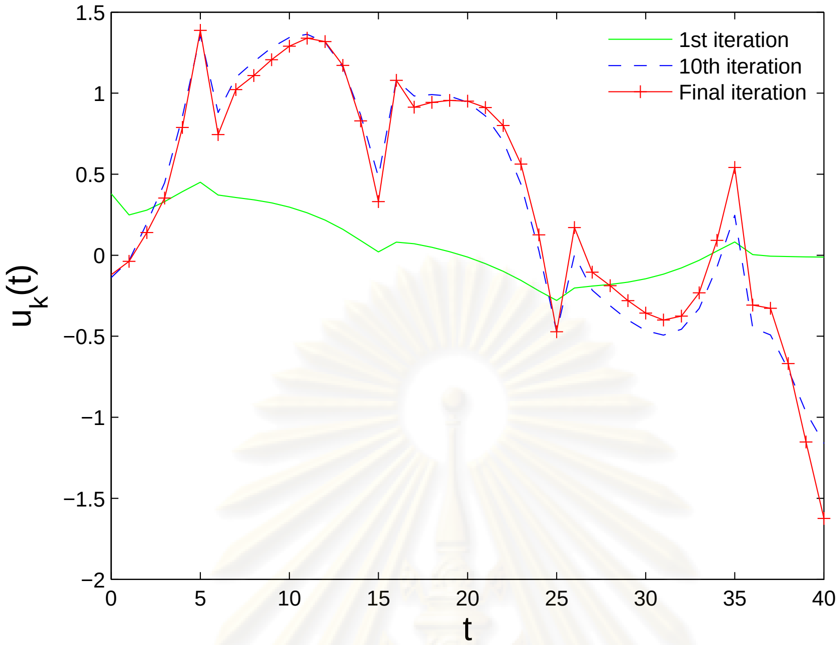

Figure 3.1: Control input of system with parametric time-invariant uncertainties.

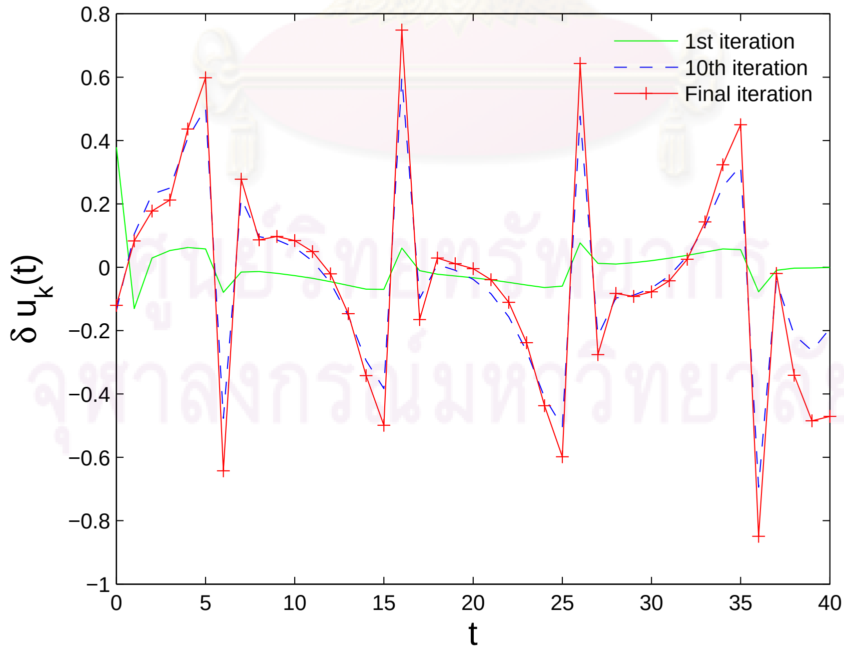

Figure 3.2: The difference of control input of system with parametric time-invariant uncertainties w.r.t. time index.

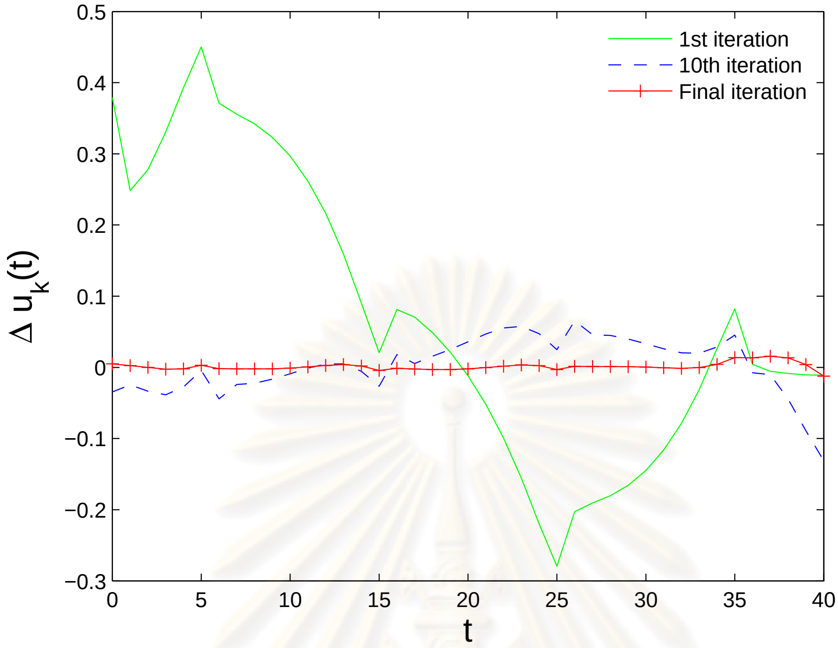

Figure 3.3: The difference of control input of system with parametric time-invariant uncertainties w.r.t. iteration index.

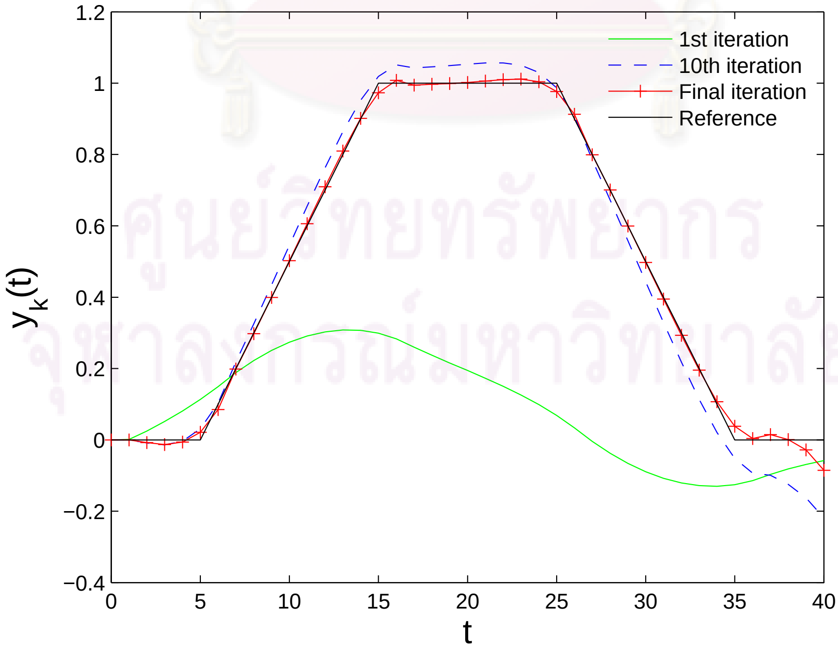

Figure 3.4: Output response of system with parametric time-invariant uncertainties.

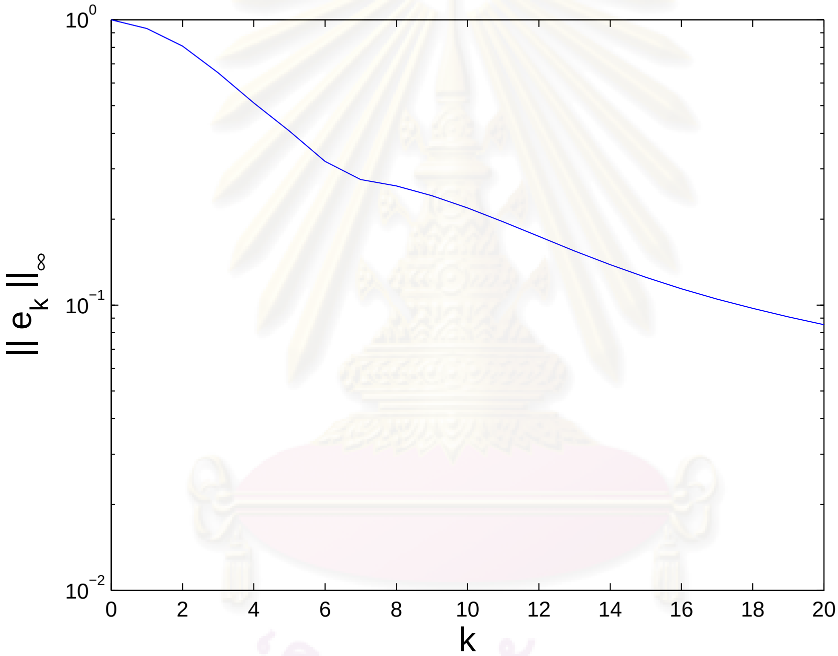

Figure 3.5: Infinity-norm of error of system with parametric time-invariant uncertainties.

CHAPTER IV

TIME-VARYING PARAMETRIC UNCERTAINTIES

There have been several results under ILC framework for systems in the presence of uncertainties. Many ILC designs have been developed for nonlinear time-varying systems, for instance [20–22], whereas only some other ILC designs have been applied to linear time-varying (LTV) systems, such as [23–25]. In [23], Arif *et al.* have used the error prediction method to construct the ILC law for LTV systems with unknown but bounded disturbances. Thereto, Tan *et. al* [24] have proposed an ILC strategy for LTV systems with state disturbances and measurement noise at the output. The ILC algorithms in [23] and [24] are applicable to systems perturbed by disturbances, but parametric uncertainties have not been considered in the dynamic system. In [25], Hladowski *et al.* have developed an ILC law for linear systems with time-varying uncertainties by solving a generalized eigenvalue problem (GEVP). Their approach applies the linear repetitive process stability theory to design iterative input updates and the control law is specified by a linear combination of the state change and the output error. Moreover, an upper bound of the time-varying uncertainties can be determined by the solution of the GEVP.

On the other hand, in this thesis, we develop another systematic approach to design robust ILC algorithms for linear systems subject to parametric time-varying uncertainties [39]. Our algorithm utilize the same quadratic performance index as in [35,36]. The control design problem is formulated as a min-max problem subjects to the constraints of the control input. An upper bound of the maximization problem is obtained first to make the initial min-max problem become a minimization problem. Then, applying Lagrange duality, we achieve a dual problem of the minimization problem which can be reformulated as a convex optimization problem over LMIs.

Based on the assumptions A1-A2, the input-output matrix of linear systems containing additive time-varying parametric uncertainties has the following form.

$$G(\theta) = G_0 + \Omega_1 G_1 + \Omega_2 G_2 + \cdots + \Omega_m G_m \tag{4.1}$$

where G_0 is the nominal matrix, $G_1, \dots, G_m$ are dynamic matrices of uncertainties, and

$$\Omega_i = \begin{bmatrix} \theta_i(1) & & & \\ & \theta_i(2) & & \\ & & \ddots & \\ & & & \theta_i(N) \end{bmatrix} \forall\ i = \overline{1,m}$$

where $\theta_i(j),\ i = \overline{1,m}$ are parametric uncertainties contained in θ at time j, $|\theta_i(j)| \le 1\ \forall\ i = \overline{1,m},\ \forall\ j = \overline{1,N}$.

4.1 The worst-case performance analysis

We express $G(\theta)^T QG(\theta)$ as

$$\left(G_0^T + \sum_{i=1}^{m} G_i^T \Omega_i\right) Q \left(G_0 + \sum_{j=1}^{m} \Omega_j G_j\right) = G_0^T QG_0 + \sum_{i=1}^{m}\sum_{j=1}^{m} G_i^T \Omega_i Q \Omega_j G_j + \sum_{i=1}^{m} G_i^T \Omega_i Q G_0 + \sum_{i=1}^{m} G_0^T Q \Omega_i G_i. \quad (4.2)$$

Moreover, $\mathbf{e}_k^T QG(\theta)\Delta\mathbf{u}_{k+1}$ is expressed as

$$\mathbf{e}_k^T Q \left(G_0 + \sum_{i=1}^{m} \Omega_i G_i\right) \Delta\mathbf{u}_{k+1} = \sum_{i=1}^{m} \mathbf{e}_k^T Q \Omega_i G_i \Delta\mathbf{u}_{k+1} + \mathbf{e}_k^T QG_0 \Delta\mathbf{u}_{k+1}. \quad (4.3)$$

Substituting (4.2) and (4.3) into (2.33) results in

$$J_{k+1} = z^T Pz + r \quad (4.4)$$

where

$$z = \begin{bmatrix} G_1\Delta\mathbf{u}_{k+1} \\ \vdots \\ G_m\Delta\mathbf{u}_{k+1} \\ QG_0\Delta\mathbf{u}_{k+1} - Q\mathbf{e}_k \end{bmatrix}, \; P = \begin{bmatrix} \Omega_1 Q\Omega_1 & \dots & \Omega_1 Q\Omega_m & \Omega_1 \\ \Omega_2 Q\Omega_1 & \dots & \Omega_2 Q\Omega_m & \Omega_2 \\ \vdots & & \vdots & \vdots \\ \Omega_m Q\Omega_1 & \dots & \Omega_m Q\Omega_m & \Omega_m \\ \Omega_1 & \dots & \Omega_m & 0 \end{bmatrix},$$

$r = \Delta\mathbf{u}_{k+1}^T \left(G_0^T QG_0 + R\right) \Delta\mathbf{u}_{k+1} - 2\mathbf{e}_k^T QG_0\Delta\mathbf{u}_{k+1}.$

Let $P_1 = \begin{bmatrix} \Omega_1 Q\Omega_1 & \dots & \Omega_1 Q\Omega_m \\ \Omega_2 Q\Omega_1 & \dots & \Omega_2 Q\Omega_m \\ \vdots & & \vdots \\ \Omega_m Q\Omega_1 & \dots & \Omega_m Q\Omega_m \end{bmatrix}$, $P_2 = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \vdots \\ \Omega_m \end{bmatrix}$, then $P = \begin{bmatrix} P_1 & P_2 \\ P_2^T & 0 \end{bmatrix}$ and $P_1 = P_2 Q P_2^T$.

Consider the maximization problem in (2.32)

$$\max_{\theta\in\Theta} J_{k+1} \quad (4.5)$$

To solve (2.32), we first derive the least upper bound γ of the maximization problem (4.5), then solve the minimization problem. Suppose that there exists a block-diagonal matrix $T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix}$ such that

$$P \le T \Leftrightarrow \begin{bmatrix} T_1 - P_1 & -P_2 \\ -P_2^T & T_2 \end{bmatrix} \ge 0. \quad (4.6)$$

Hence, J_{k+1} is bounded by

$$J_{k+1} = z^T Pz + r \le z^T Tz + r. \quad (4.7)$$

If $T_2 > 0$, then, (4.6) is equivalent to

$$T_1 - P_1 - P_2 T_2^{-1} P_2^T \ge 0. \quad (4.8)$$

Choose $T_2 = Q^{-1}$, then $P_2 T_2^{-1} P_2^T = P_2 Q P_2^T = P_1$. Thus, (4.8) becomes

$$T_1 - 2P_1 \geq 0. \tag{4.9}$$

Let $\Omega = \begin{bmatrix} \Omega_1 & & & \\ & \Omega_2 & & \\ & & \ddots & \\ & & & \Omega_m \end{bmatrix}$, $\widetilde{Q} = \begin{bmatrix} Q & Q & \dots & Q \\ Q & Q & \dots & Q \\ \vdots & \vdots & & \vdots \\ Q & Q & \dots & Q \end{bmatrix}$, then

$$P_1 = \Omega \widetilde{Q} \Omega. \tag{4.10}$$

Therefore, (4.9) can be rewritten as

$$T_1 - 2\Omega \widetilde{Q} \Omega \geq 0. \tag{4.11}$$

Since T_2 is chosen to be constant, the problem of finding the least upper bound of J_{k+1} is equivalent to finding the "smallest" matrix T_1 satisfying (4.11) for all uncertainties in the uncertainty set. Thus, the worst-case performance with respect to the uncertainties can be formulated as

$$\begin{aligned} \max \quad & \text{trace } (\Omega \widetilde{Q} \Omega) \\ \text{s.t.} \quad & -I \leq \Omega \leq I. \end{aligned} \tag{4.12}$$

On the other hand, it is noted that

$$\text{trace } (\Omega \widetilde{Q} \Omega) = \text{trace } (\widetilde{Q} \Omega^2). \tag{4.13}$$

Since $Q > 0$, all the elements in the diagonal of Q are positive. Hence,

$$\max_{-I \leq \Omega \leq I} \text{trace } (\widetilde{Q} \Omega^2) = \text{trace } (\widetilde{Q}). \tag{4.14}$$

The condition for (4.14) is $\Omega^2 = I$, that means the worst-case performance happens when the uncertainty reaches the boundary of the uncertain interval at each sample time. Accordingly, the "smallest" matrix T_1 is the solution of the following minimization problem.

$$\begin{aligned} \min \quad & \text{trace } T_1 \\ \text{s.t.} \quad & T_1 \text{ is diagonal} \\ & T_1 - 2\Omega \widetilde{Q} \Omega \geq 0 \\ & \Omega^2 = I. \end{aligned} \tag{4.15}$$

Note that the congruence transformation preserves the positive definiteness of a matrix. Therefore,

$$\begin{aligned} T_1 - 2\Omega \widetilde{Q} \Omega \geq 0 \quad & \Leftrightarrow \quad \Omega (T_1 - 2\Omega \widetilde{Q} \Omega) \Omega \geq 0 \\ & \Leftrightarrow \quad \Omega T_1 \Omega - 2\widetilde{Q} \geq 0. \end{aligned} \tag{4.16}$$

It can be deduced that (4.16) is equivalent to

$$T_1 - 2\widetilde{Q} \geq 0 \tag{4.17}$$

since $\Omega^2 = I$. Therefore, (4.15) becomes

$$\begin{array}{ll} \min & \text{trace } T_1 \\ \text{s.t.} & T_1 \text{ is diagonal} \\ & T_1 \geq 2\widetilde{Q}. \end{array} \tag{4.18}$$

This problem can be numerically solved using available software that support convex optimization. Moreover, we need to calculate (4.18) for only once and it can be done off-line, *i.e.*, not in the iteration process. Now, the least upper bound of (4.5) is

$$\begin{aligned} \gamma &= z^T T z + r \\ &= \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + \beta^T \Delta\mathbf{u}_{k+1} + \mathbf{e}_k^T Q \mathbf{e}_k \end{aligned} \tag{4.19}$$

where

$$\begin{aligned} \widehat{G} &= \begin{bmatrix} G_1 \\ \vdots \\ G_m \end{bmatrix}^T T_1 \begin{bmatrix} G_1 \\ \vdots \\ G_m \end{bmatrix} + 2G_0^T Q G_0 + R \\ \beta &= -4G_0^T Q \mathbf{e}_k. \end{aligned}$$

4.2 LMI-based ILC algorithm

After obtaining the least upper bound of (4.5), the iterative input update can be found by solving the following problem

$$\min_{\Delta\mathbf{u}_{k+1} \in U_{k+1}} \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + \beta^T \Delta\mathbf{u}_{k+1} \tag{4.20}$$

which is equivalent to

$$\begin{array}{ll} \min & \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + \beta^T \Delta\mathbf{u}_{k+1} \\ \text{s.t.} & \Pi \Delta\mathbf{u}_{k+1} \leq \phi. \end{array} \tag{4.21}$$

Define a Lagrangian

$$\begin{aligned} L(\Delta\mathbf{u}_{k+1}, \nu) &= \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + \beta^T \Delta\mathbf{u}_{k+1} + \nu^T (\Pi \Delta\mathbf{u}_{k+1} - \phi) \\ &= \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + (\beta + \Pi^T \nu)^T \Delta\mathbf{u}_{k+1} - \nu^T \phi \end{aligned} \tag{4.22}$$

where $\nu \in \Re^{4N}$ is the Lagrange multiplier. It is straightforward to obtain the optimal solution as

$$\Delta\mathbf{u}_{k+1}^* = -\frac{1}{2} \widehat{G}^{-1} (\beta + \Pi^T \nu). \tag{4.23}$$

Then, we achieve a Lagrange dual function

$$\begin{aligned} f(\nu) &= \inf_{\Delta \mathbf{u}_{k+1}} L(\Delta \mathbf{u}_{k+1}, \nu) \\ &= -\frac{1}{4}\left\{(\beta + \Pi^T \nu)^T \widehat{G}^{-1}(\beta + \Pi^T \nu) + 4\nu^T \phi\right\}. \end{aligned}$$

Therefore, the dual problem of (4.21) is

$$\begin{array}{rl} \max & 4f(\nu) \\ \text{s.t.} & \nu \geq 0 \end{array}$$

which is equivalent to

$$\begin{array}{rl} \min & \rho \\ \text{s.t.} & -4f(\nu) \leq \rho \\ & \nu \geq 0. \end{array} \tag{4.24}$$

Using Schur complement [29], we can rewrite (4.24) as a convex optimization problem over LMIs.

$$\begin{array}{rl} \min & \rho \\ \text{s.t.} & \begin{bmatrix} \widehat{G} & \beta + \Pi^T \nu \\ (\beta + \Pi^T \nu)^T & \rho - 4\nu^T \phi \end{bmatrix} \geq 0 \\ & \nu \geq 0. \end{array} \tag{4.25}$$

This problem can be solved using available software such as `cvx` [38]. The stopping criteria for iterative solution procedure are as follows.

$$\|\mathbf{e}_k\| \leq \epsilon \tag{4.26}$$

$$k = \texttt{iter\_max} \tag{4.27}$$

where ϵ is a tolerance chosen by the designer and `iter_max` is the maximum number of iterations.

Here, we summarize the proposed algorithm.

Algorithm 2. *An LMI algorithm for linear systems with time-varying parametric uncertainties (ILC-TV)*

1. *Set* $k := 0$, $\boldsymbol{u}_k := 0$. *Measure* $\boldsymbol{e}_k$.
2. *Solve the LMI problem according to (4.25).*
3. *Calculate* $\Delta \boldsymbol{u}_{k+1}$ *according to (4.23).*
4. *Apply* $\boldsymbol{u}_{k+1}$ *to the system and measure* $\boldsymbol{e}_{k+1}$.
5. *If (4.26) or (4.27) is true, then, stop the iteration, else set* $k := k+1$, *return to step 2.*

Remark 3. It is noted that the minimization problem (4.21) has a quadratic cost function and an affine constraint. When we consider the dual problem, the strong duality always holds [29]. As a result, the solutions of (4.21) and (4.25) are equal.

Remark 4. There might be a conservatism in our approach since an upper bound of the worst-case performance is used in the maximization problem (4.5). Nevertheless, the algorithm appears to work well as we will demonstrate in the numerical example.

4.3 Convergence properties

Theorem 9. *Under assumptions A1-A2 and constraints C1-C3, the control input* $\mathbf{u}_k$ *of system (4.1) converges.*

Proof. Let

$$V(\mathbf{e}_k) = \min_{\Delta\mathbf{u}_{k+1}\in U_{k+1}} \max_{\theta\in\Theta} J_{k+1}$$

with J_{k+1} is in (2.33). Then, $V(\mathbf{e}_k) \geq 0 \;\forall\; k$ since $J_{k+1} \geq 0 \;\forall\; k$. We have

$$V(\mathbf{e}_k) \leq J_{k+1}|_{\Delta\mathbf{u}_{k+1}=0} = \mathbf{e}_k^T Q \mathbf{e}_k.$$

Suppose that θ_k^* is the optimizer of the maximization problem at the kth iteration. Hence,

$$\mathbf{e}_k^T Q \mathbf{e}_k \leq \mathbf{e}_k^T(\theta_k^*) Q \mathbf{e}_k(\theta_k^*) = V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k.$$

Therefore,

$$V(\mathbf{e}_k) \quad \leq \quad V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k. \tag{4.28}$$

Inequality (4.28) leads to

$$V(\mathbf{e}_k) + \sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \quad \leq \quad V(\mathbf{e}_0).$$

Since $V(\mathbf{e}_k) \geq 0$, we get

$$\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \quad \leq \quad V(\mathbf{e}_0) < \infty. \tag{4.29}$$

Moreover, because R is positive definite, $\Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \geq 0 \;\forall\; i$ and the sequence $\left\{\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i\right\}$ is non-decreasing. Combine with (4.29), it deduces that $\left\{\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i\right\}$ converges. Accordingly,

$$\begin{aligned}
\lim_{k\to\infty} \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k &= \lim_{k\to\infty} \left(\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i - \sum_{i=1}^{k-1} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \right) \\
&= \lim_{k\to\infty} \sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i - \lim_{k\to\infty} \sum_{i=1}^{k-1} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \\
&= 0.
\end{aligned}$$

It implies that $\Delta\mathbf{u}_k \to 0$ as $k \to \infty$. Thus, $\{\mathbf{u}_k\}$ converges. $\square$

Theorem 10. *Under assumptions A1-A2 and constraints C1-C3, the error* $\mathbf{e}_k$ *of system (4.1) converges.*

Proof. We have

$$\begin{aligned} \|G(\theta)\| &= \left\| G_0 + \sum_{i=1}^{m} \Omega_i G_i \right\| \\ &\leq \|G_0\| + \sum_{i=1}^{m} \|\Omega_i G_i\| \\ &\leq \|G_0\| + \sum_{i=1}^{m} \|\Omega_i\| \, \|G_i\| \\ &\leq \|G_0\| + \sum_{i=1}^{m} \|G_i\| \end{aligned}$$

Hence, $\|G(\theta)\|$ is bounded. Since $\Delta\mathbf{u}_k \to 0$ as $k \to \infty$, it leads to

$$G(\theta)\Delta\mathbf{u}_{k+1} \to 0 \text{ as } k \to \infty.$$

Equivalently,

$$(\mathbf{e}_k - \mathbf{e}_{k+1}) \to 0 \text{ as } k \to \infty.$$

This results in the convergence of $\{\mathbf{e}_k\}$. □

4.4 Numerical example

Consider a linear system whose output is

$$y(t) = y_0(t) + \theta_1(t)y_1(t) + \theta_2(t)y_2(t) \tag{4.30}$$

where y_0, y_1, y_2 are the corresponding outputs of the systems described by

$$\begin{aligned} G_0(s) &= \frac{1}{15s^2 + 8s + 1} \\ G_1(s) &= \frac{0.8e^{-s}}{5s + 1} \\ G_2(s) &= \frac{0.5e^{-s}}{2s + 1} \end{aligned}$$

where θ_1, θ_2 are the uncertain parameters over the interval $[-1, 1]$, and vary with time. The constraints of control input for system (4.30) are specified by

$$\mathbf{u}_l = -3, \mathbf{u}_h = 3, \delta\mathbf{u}_l = -5, \delta\mathbf{u}_h = 5, \Delta\mathbf{u}_l = -4, \Delta\mathbf{u}_h = 4. \tag{4.31}$$

The target trajectory is

$$r(t) = \begin{cases} 0, & t \in [0, 5] \cup [36, 40] \\ 0.1(t - 5), & t \in [6, 15] \\ 1, & t \in [16, 25] \\ 1 - 0.1(t - 25), & t \in [26, 35] \end{cases}$$

The design parameters are chosen as follows: $Q = 2I_1, R = 0.1I_2$ where I_1, I_2 are identity matrices with appropriate dimension. For the stopping criteria, we choose $\epsilon = 0.01$ and `iter_max` $= 10$. To illustrate the time responses of the ILC control system, we choose sampling time 1 second and the number of samples $N = 41$. In our work, we use the software `cvx` [38] to solve the LMI problem (4.25).

The design results are shown in the Figures 4.1–4.5. Figures 4.1, 4.2, 4.3 show that the control input converges and satisfies all constraints (4.31). Figure 4.2 displays the convergence of $\delta_k(t)$ and Figures 4.3 exhibits the convergence of $\Delta_k(t)$. Moreover, Figure 4.4 demonstrates that the output of system (4.30) converges to the desired reference trajectory. A monotonic convergence of the system error is illustrated in Figure 4.5.

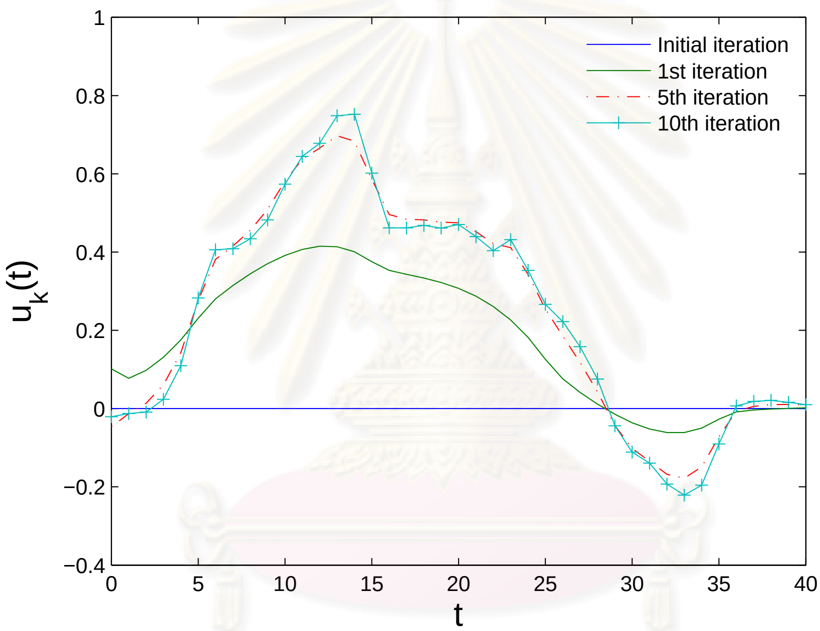

Figure 4.1: Control input of the robust ILC system with time-varying parametric uncertainties.

4.5 Conclusion

This chapter is devoted to reveal the design of robust ILC for linear systems with time-varying parametric uncertainties which includes of four sections. Section 1 presents the affine Markov model of the system and the worst-case performance analysis. Next, in Section 4.2, we obtain an ILC controller as the solution of a convex optimization problem over LMI constraints. Then, an LMI-based ILC algorithm is proposed. In Section 4.3, we provide the proof of the convergence of the control input and the system error. Finally, Section 4.4 exhibits the simulation results of a generic example which demonstrates the effectiveness of the algorithm.

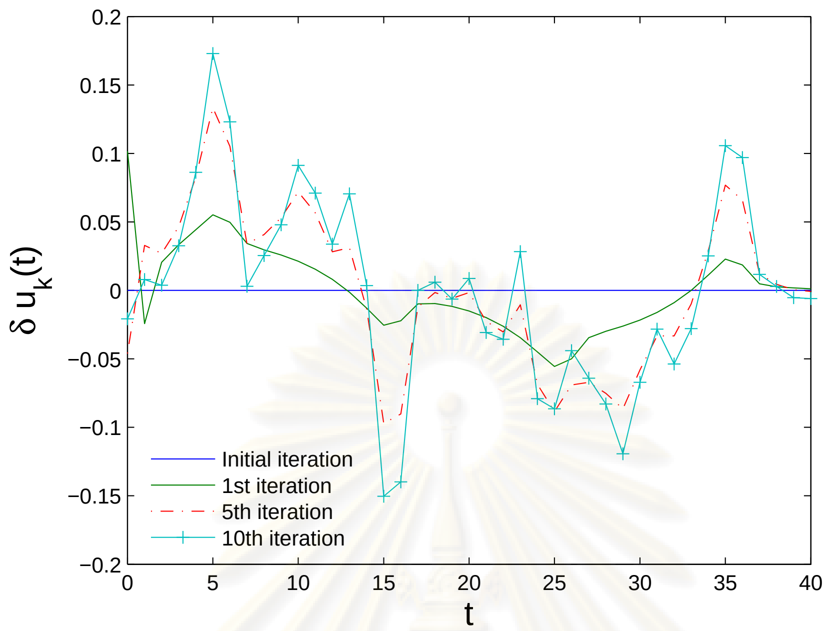

Figure 4.2: The difference of control input of the robust ILC system with time-varying parametric uncertainties w.r.t. time index.

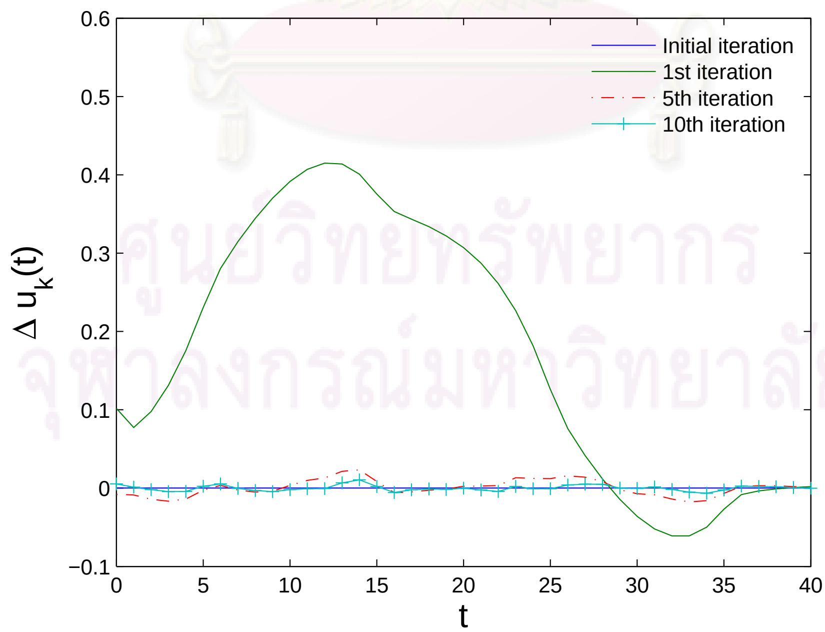

Figure 4.3: The difference of control input of the robust ILC system with time-varying parametric uncertainties w.r.t. iteration index.

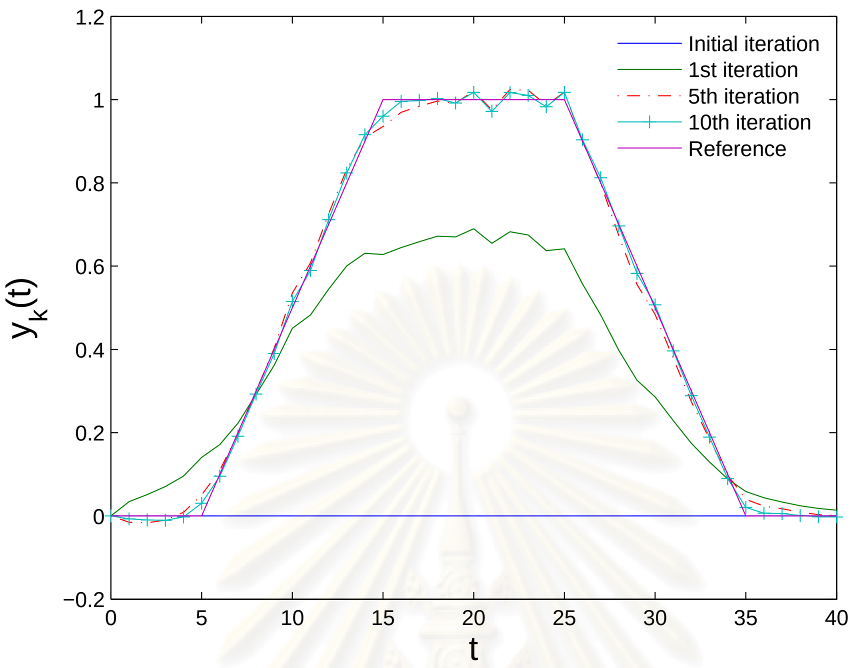

Figure 4.4: Output response of the robust ILC system with time-varying parametric uncertainties.

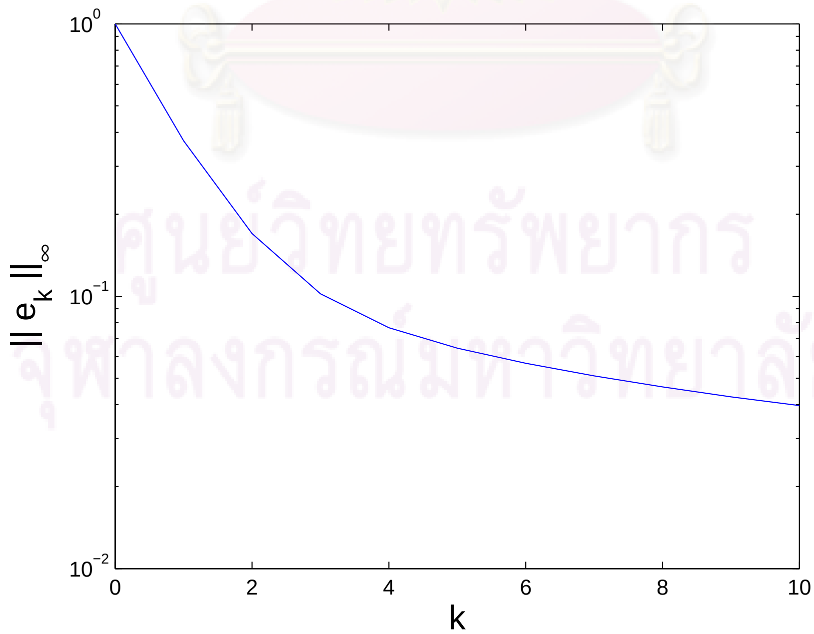

Figure 4.5: Infinity-norm of error of the robust ILC system with time-varying parametric uncertainties vs. the number of iterations.

CHAPTER V

ITERATION-VARYING PARAMETRIC UNCERTAINTIES

Within a broad range of ILC techniques, we focus on the robustness of the systems controlled under ILC framework. Among a large number of articles on the robust ILC design, the systems are subjected to time-invariant uncertainties or time-varying uncertainties. Since the systems controlled by ILC algorithms work in both time domain and iteration domain, it is also requisite to ensure the robustness of ILC systems in the iteration domain. In the design procedure, a dynamic model is typically obtained by the identification which is usually an approximation of the real system. When the system operates repetitively, the system dynamic may change with iterations in which some parameters vary. To the best of our knowledge, the robust ILC designs with iteration-varying uncertainties have been explored by Ahn, Moore and Chen in a series of articles [26–28] which also appear in a research monograph [2]. The plant is described by a linear model in a super-vector framework with iteration-varying uncertainty. Then, the Markov matrix is converted into a form of bounded additive model uncertainty. As a result, the Markov matrix belongs to an interval set and all the ILC updates are designed for the uncertain interval set.

In this chapter, we propose a new ILC algorithm for a class of linear systems whose transfer functions are affine of parametric uncertainties and are subjected to iteration-varying [40]. The system model and problem formulation are different from that in the previous works [26–28]. Our approach is built in a super-vector framework and the robust ILC design is formulated as a min-max problem using a quadratic performance criterion which has been utilized in [33,34,36]. Finding the least upper bound of the maximization problem, the initial min-max problem becomes a minimization problem which is easier to solve. Next, we derive a dual problem of the minimization problem and reformulate it as a convex optimization over linear matrix inequalities (LMIs). Note that the parametric uncertainty in our previous work [36] is time-invariant whilst in this paper, the parametric uncertainties are iteration-varying. Thus, the problem formulation and the algorithm derivation are different.

Now, we consider linear systems in the presence of iteration-varying uncertainties. Due to assumptions A1-A2, the system model is given by

$$G(\theta_k) = G_0 + G_1\theta_{k,1} + G_2\theta_{k,2} + \cdots + G_m\theta_{k,m} \qquad (5.1)$$

where G_0 is the nominal matrix, $G_1, \ldots, G_m$ are uncertain dynamic matrices, and $\theta_{k,i}$, $i = 1, \ldots, m$ are parametric uncertainties.

Hence, the error update model of the system (2.21) is

$$\begin{aligned}\mathbf{e}_k - \mathbf{e}_{k+1} &= \mathbf{y}_{k+1} - \mathbf{y}_k \\ &= G(\theta_{k+1})\mathbf{u}_{k+1} - G(\theta_k)\mathbf{u}_k \\ &= G_0\Delta\mathbf{u}_{k+1} + \sum_{i=1}^{m} G_i(\theta_{k+1,i}\mathbf{u}_{k+1} - \theta_{k,i}\mathbf{u}_k)\end{aligned}$$

Moreover,

$$\begin{aligned}\theta_{k+1,i}\mathbf{u}_{k+1} - \theta_{k,i}\mathbf{u}_k &= \theta_{k+1,i}(\mathbf{u}_{k+1} - \mathbf{u}_k) + (\theta_{k+1,i} - \theta_{k,i})\mathbf{u}_k \\ &= \theta_{k+1,i}\Delta\mathbf{u}_{k+1} + \Delta\theta_{k+1,i}\mathbf{u}_k\end{aligned}$$

where $\Delta\theta_{k+1,i} = \theta_{k+1,i} - \theta_{k,i}$. Since $|\theta_{k,i}| \le 1 \ \forall \ i = \overline{1,m}$, then, $|\Delta\theta_{k+1,i}| \le 2 \ \forall \ i = \overline{1,m}$. Therefore,

$$\mathbf{e}_k - \mathbf{e}_{k+1} = G_0\Delta\mathbf{u}_{k+1} + \sum_{i=1}^{m} G_i(\theta_{k+1,i}\Delta\mathbf{u}_{k+1} + \Delta\theta_{k+1,i}\mathbf{u}_k) \tag{5.2}$$

5.1 The worst-case performance analysis

Let us define

$$\begin{aligned}\sigma_{k+1} &= [\theta_{k+1,1}, \ldots, \theta_{k+1,m}, \Delta\theta_{k+1,1}, \ldots, \Delta\theta_{k+1,m}]^T \\ \widetilde{G} &= [G_1\Delta\mathbf{u}_{k+1}, \ldots, G_m\Delta\mathbf{u}_{k+1}, G_1\mathbf{u}_k, \ldots, G_m\mathbf{u}_k]\end{aligned}$$

Replacing into (5.2), we get

$$\mathbf{e}_{k+1} = \mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1} - \widetilde{G}\sigma_{k+1} \tag{5.3}$$

Substituting (5.3) into (2.33) results in

$$\begin{aligned}J_{k+1} &= (\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1} - \widetilde{G}\sigma_{k+1})^T Q(\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1} - \widetilde{G}\sigma_{k+1}) + \Delta\mathbf{u}_{k+1}^T R\Delta\mathbf{u}_{k+1} \\ &= (\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1})^T Q(\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1}) + \Delta\mathbf{u}_{k+1}^T R\Delta\mathbf{u}_{k+1} - 2\sigma_{k+1}^T\widetilde{G}^T Q(\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1}) \\ &\quad + \sigma_{k+1}^T\widetilde{G}^T Q\widetilde{G}\sigma_{k+1} \\ &= z^T H z\end{aligned}$$

where $z = \begin{bmatrix}\sigma_{k+1} \\ 1\end{bmatrix}$, $H = \begin{bmatrix}H_1 & H_2 \\ H_2^T & H_3\end{bmatrix}$, $H_1 = \widetilde{G}^T Q\widetilde{G}$, $H_2 = -\widetilde{G}^T Q(\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1})$,
$H_3 = (\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1})^T Q(\mathbf{e}_k - G_0\Delta\mathbf{u}_{k+1}) + \Delta\mathbf{u}_{k+1}^T R\Delta\mathbf{u}_{k+1}$.
Hence, the maximization problem in (2.33) can be rewritten as

$$\max_{z\in Z} z^T H z \tag{5.4}$$

where $Z = \left\{z = \begin{bmatrix}\sigma_{k+1} \\ 1\end{bmatrix}\right\}$.

If there exists a diagonal matrix T such that $T \geq H$, then

$$z^T Hz \leq z^T Tz = \sum_{i=1}^{m} t_i z_i^2 + t_{m+1} \leq 4 \sum_{i=1}^{m+1} t_i = 4 \text{ trace } T \tag{5.5}$$

where t_i's$, i = 1, 2, \ldots, m+1$ are elements on the diagonal of matrix T. Accordingly, the least upper bound of (5.4) can be found by solving the following minimization problem

$$\begin{aligned} \min \quad & \text{trace } T \\ \text{s.t.} \quad & T \geq H \\ & T \text{ is diagonal} \end{aligned} \tag{5.6}$$

5.2 LMI-based ILC algorithm

Replacing (5.6) into (2.32), and combining two minimization problems, the iterative input update $\Delta \mathbf{u}_{k+1}$ can be calculated by solving the following minimization problem.

$$\begin{aligned} \min \quad & \text{trace } T \\ \text{s.t.} \quad & T \geq H \\ & T \text{ is diagonal} \\ & \Delta \mathbf{u}_{k+1} \in U_{k+1} \end{aligned} \tag{5.7}$$

This is a minimization problem with variables T and $\Delta \mathbf{u}_{k+1}$. To solve (5.7), we consider its dual problem. We first reformulate (5.7) as follows. Rewrite T as

$$\sum_{i=1}^{2m+1} t_i F_i$$

where $F_i = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \cdots & 0 \\ 0 & \cdots & 1 & \cdots & 0 \\ 0 & \cdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix}$, with 1 is located at the ith position on the diagonal of matrix F_i.

Then, (5.7) becomes

$$\begin{aligned} \min \quad & \mathbf{1}^T t \\ \text{s.t.} \quad & H - \sum_{i=1}^{2m+1} t_i F_i \leq 0 \\ & \Pi \Delta \mathbf{u}_{k+1} \leq \phi \end{aligned} \tag{5.8}$$

where $t = [t_1, t_2, \ldots, t_{2m+1}]^T$, $\mathbf{1}$ is the $(2m+1) \times 1$ vector whose elements are all 1.

Define a Lagrangian

$$L(t, \Delta \mathbf{u}_{k+1}, W, \nu) = \mathbf{1}^T t + \text{tr} \left(\left(H - \sum_{i=1}^{2m+1} t_i F_i \right) W \right) + \nu^T (\Pi \Delta \mathbf{u}_{k+1} - \phi) \tag{5.9}$$

where $W \in \mathbf{S}_+^{2m+1}$, in which, $\mathbf{S}_+^{2m+1}$ is the set of symmetric semi-definite positive matrices with dimension $(2m+1) \times (2m+1)$, ν is a vector.

Next, to obtain the dual function, we find the minimum of the Lagrangian with respect to t and $\Delta\mathbf{u}_{k+1}$. We have

$$\begin{aligned}
\inf_{t,\Delta\mathbf{u}_{k+1}} L(t,\Delta\mathbf{u}_{k+1},W,\nu) &= \inf_t \left\{ \mathbf{1}^T t - \mathrm{tr}\left(\sum_{i=1}^{2m+1} t_i F_i W\right) - \nu^T\phi \right\} + \inf_{\Delta\mathbf{u}_{k+1}} \left\{ \mathrm{tr}\,(HW) + \nu^T \Pi \Delta\mathbf{u}_{k+1} \right\} \\
&= -\nu^T\phi + \inf_t \left\{ \sum_{i=1}^{2m+1} t_i (1 - \mathrm{tr}(F_i W)) \right\} + \inf_{\Delta\mathbf{u}_{k+1}} \left\{ \mathrm{tr}\,(HW) + \nu^T \Pi \Delta\mathbf{u}_{k+1} \right\} \\
&= -\nu^T\phi + \inf_{\Delta\mathbf{u}_{k+1}} \left\{ \mathrm{tr}\,(HW) + \nu^T \Pi \Delta\mathbf{u}_{k+1} \right\} \qquad (5.10)
\end{aligned}$$

Note that (5.10) is obtained when $\mathrm{tr}\,(F_i W) = 1 \;\forall\; i = \overline{1, 2m+1} \Leftrightarrow w_{ii} = 1 \;\forall\; i = \overline{1, 2m+1}$ with $w_{ii}'\mathrm{s}, i = 1, 2, \ldots, 2m+1$ are elements on the diagonal of matrix W.
Otherwise,

$$\begin{aligned}
\mathrm{tr}\,(HW) &= \sum_{i=1}^{2m+1} \sum_{j=1}^{2m+1} h_{ij} w_{ij} \\
&= \sum_{i=1}^{2m} \sum_{j=1}^{2m} h_{ij} w_{ij} + 2 \sum_{i=1}^{2m} h_{i,2m+1} w_{i,2m+1} + h_{2m+1,2m+1} w_{2m+1,2m+1} \qquad (5.11)
\end{aligned}$$

where $h_{ij}, w_{ij}, i, j = 1, 2, \ldots, m+1$ are elements of corresponding matrices H, W. In addition,

$$\begin{aligned}
\sum_{i=1}^{2m} \sum_{j=1}^{2m} h_{ij} w_{ij} &= \sum_{i=1}^{m} \sum_{j=1}^{m} h_{ij} w_{ij} + 2 \sum_{i=m+1}^{2m} \sum_{j=1}^{m} h_{ij} w_{ij} + \sum_{i=m+1}^{2m} \sum_{j=m+1}^{2m} h_{ij} w_{ij} \\
&= \sum_{i=1}^{m} \sum_{j=1}^{m} \Delta\mathbf{u}_{k+1}^T G_i^T Q G_j \Delta\mathbf{u}_{k+1} w_{ij} + 2 \sum_{i=1}^{m} \sum_{j=1}^{m} \mathbf{u}_k^T G_i^T Q G_j \Delta\mathbf{u}_{k+1} w_{i+m,j} \\
&\quad + \sum_{i=1}^{m} \sum_{j=1}^{m} \mathbf{u}_k^T G_i^T Q G_j \mathbf{u}_k w_{i+m,j+m} \\
\sum_{i=1}^{2m} h_{i,2m+1} w_{i,2m+1} &= -\sum_{i=1}^{m} \Delta\mathbf{u}_{k+1}^T G_i^T Q (\mathbf{e}_k - G_0 \Delta\mathbf{u}_{k+1}) w_{i,2m+1} \\
&\quad - \sum_{i=1}^{m} \mathbf{u}_k^T G_i^T Q (\mathbf{e}_k - G_0 \Delta\mathbf{u}_{k+1}) w_{i+m,2m+1} \\
h_{2m+1,2m+1} w_{2m+1,2m+1} &= (\mathbf{e}_k - G_0 \Delta\mathbf{u}_{k+1})^T Q (\mathbf{e}_k - G_0 \Delta\mathbf{u}_{k+1}) + \Delta\mathbf{u}_{k+1}^T R \Delta\mathbf{u}_{k+1}
\end{aligned}$$

Therefore,

$$\mathrm{tr}\,(HW) + \nu^T \Pi \Delta\mathbf{u}_{k+1} - \nu^T\phi = \Delta\mathbf{u}_{k+1}^T \widehat{G} \Delta\mathbf{u}_{k+1} + \beta^T \Delta\mathbf{u}_{k+1} + \alpha \qquad (5.12)$$

where

$$\widehat{G} = \sum_{i=1}^{m}\sum_{j=1}^{m} G_i^T Q G_j w_{ij} + 2\sum_{i=1}^{m} G_i^T Q G_0 w_{i,2m+1} + G_0^T Q G_0 + R \tag{5.13}$$

$$\begin{aligned}\beta &= \Pi^T \nu + 2\sum_{i=1}^{m}\sum_{j=1}^{m} G_j^T Q G_i \mathbf{u}_k w_{i+m,j} + 2\sum_{i=1}^{m} G_0^T Q G_i \mathbf{u}_k w_{i+m,2m+1} \\ &\quad -2\sum_{i=1}^{m} G_i^T Q \mathbf{e}_k w_{i,2m+1} - 2G_0^T Q \mathbf{e}_k \end{aligned} \tag{5.14}$$

$$\alpha = -\nu^T \phi + \mathbf{e}_k^T Q \mathbf{e}_k + \sum_{i=1}^{m}\sum_{j=1}^{m} \mathbf{u}_k^T G_i^T Q G_j \mathbf{u}_k w_{i+m,j+m} - 2\sum_{i=1}^{m} \mathbf{u}_k^T G_i^T Q \mathbf{e}_k w_{i+m,2m+1} \tag{5.15}$$

Consequently,

$$\inf_{\Delta \mathbf{u}_{k+1}} \left\{ \mathrm{tr}\,(HW) + \nu^T \Pi \Delta \mathbf{u}_{k+1} \right\} = -\frac{1}{4}\beta^T \widehat{G}^{-1}\beta + \alpha \tag{5.16}$$

with the optimal value of $\Delta \mathbf{u}_{k+1}$ is

$$\Delta \mathbf{u}_{k+1}^* = -\frac{1}{2}\widehat{G}^{-1}\beta \tag{5.17}$$

Thus,

$$\inf_{t,\Delta \mathbf{u}_{k+1}} L(t, \Delta \mathbf{u}_{k+1}, W, \nu) = -\frac{1}{4}\left(-4\alpha + \beta^T \widehat{G}^{-1}\beta\right) \tag{5.18}$$

Accordingly, the dual problem of (5.7) is

$$\begin{aligned} \max \quad & -\tfrac{1}{4}\left(-4\alpha + \beta^T \widehat{G}^{-1}\beta\right) \\ \text{s.t.} \quad & \widehat{G} > 0 \\ & W \in \mathbf{S}_+^{2m+1} \\ & w_{ii} = 1,\ \nu \geq 0 \end{aligned} \tag{5.19}$$

which is equivalent to the following optimization problem

$$\begin{aligned} \min \quad & \rho \\ \text{s.t.} \quad & -4\alpha + \beta^T \widehat{G}^{-1}\beta \leq \rho \\ & \widehat{G} > 0 \\ & W \in \mathbf{S}_+^{2m+1} \\ & w_{ii} = 1,\ \nu \geq 0 \end{aligned} \tag{5.20}$$

Using Schur complement [38], we can rewrite the dual problem (5.20) as the following LMI problem.

$$\begin{aligned} \min \quad & \rho \\ \text{s.t.} \quad & \begin{bmatrix} \widehat{G} & \beta \\ \beta^T & \rho + 4\alpha \end{bmatrix} \geq 0 \\ & W \in \mathbf{S}_+^{2m+1} \\ & w_{ii} = 1,\ \nu \geq 0 \end{aligned} \tag{5.21}$$

The problem (5.21) can be solved using available convex optimization solvers such as `cvx` [29]. The stopping criteria for iterative solution procedure are as follows.

$$\|\mathbf{e}_k\| \leq \epsilon \tag{5.22}$$

$$k = \texttt{iter\_max} \tag{5.23}$$

where ϵ is a tolerance chosen by the designer and `iter_max` is the maximum number of iterations.

Finally, the iterative input design can be summarized in the following algorithm.

Algorithm 3. *An LMI algorithm for linear systems with iteration-varying parametric uncertainties (ILC-IV)*

1. *Set* $k := 0$, $\boldsymbol{u}_k := 0$. *Measure* $\boldsymbol{e}_k$.
2. *Solve the LMI problem according to (5.21).*
3. *Calculate* $\Delta\boldsymbol{u}_{k+1}$ *according to (5.17).*
4. *Apply* $\boldsymbol{u}_{k+1}$ *to the system and measure* $\boldsymbol{e}_{k+1}$.
5. *If (5.22) or (5.23) is true, then, stop the iteration, else set* $k := k+1$, *return to step 2.*

Remark 5. The proposed robust ILC design may have conservatism since an upper bound of the worst-case performance is used in the maximization problem (5.4). Nevertheless, the algorithm appears to work well as we will demonstrate in the numerical example.

Remark 6. Weighting matrices Q, R and the bounds in constraint specifications C1-C3 can be considered as tuning parameters for the proposed method since their values affect to the solution of the LMI problem (5.21). Thus, in the implementation, the designer can vary these values to choose the fit ones.

5.3 Convergence properties

Theorem 11. *Under assumptions A1-A2 and constraints C1-C3, the control input* $\boldsymbol{u}_k$ *of system (5.1) converges.*

Proof. Let

$$V(\mathbf{e}_k) = \min_{\Delta\mathbf{u}_{k+1} \in U_{k+1}} \max_{\theta \in \Theta} J_{k+1} \tag{5.24}$$

with J_{k+1} is in (2.33). Then, $V(\mathbf{e}_k) \geq 0 \ \forall \ k$ since $J_{k+1} \geq 0 \ \forall \ k$.

We have,

$$V(\mathbf{e}_k) \leq J_{k+1}|_{\Delta\mathbf{u}_{k+1}=0} = \mathbf{e}_k^T Q \mathbf{e}_k \tag{5.25}$$

Suppose that θ_k^* is the optimizer of the maximization problem at the kth iteration. Hence,

$$\mathbf{e}_k^T Q \mathbf{e}_k \leq \mathbf{e}_k^T(\theta_k^*) Q \mathbf{e}_k(\theta_k^*) = V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k \tag{5.26}$$

Therefore,

$$V(\mathbf{e}_k) \leq V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k \tag{5.27}$$

Inequality (5.27) leads to

$$V(\mathbf{e}_k) + \sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \leq V(\mathbf{e}_0) \tag{5.28}$$

Since $V(\mathbf{e}_k) \geq 0$, we get

$$\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \leq V(\mathbf{e}_0) < \infty \tag{5.29}$$

Moreover, since R is positive definite, $\Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \geq 0 \ \forall\ i$, the sequence $\left\{\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i\right\}$ is non-decreasing. Combine with (5.29), it deduces that $\left\{\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i\right\}$ converges. Accordingly,

$$\begin{aligned}
\lim_{k\to\infty} \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k &= \lim_{k\to\infty} \left(\sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i - \sum_{i=1}^{k-1} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \right) \\
&= \lim_{k\to\infty} \sum_{i=1}^{k} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i - \lim_{k\to\infty} \sum_{i=1}^{k-1} \Delta\mathbf{u}_i^T R \Delta\mathbf{u}_i \\
&= 0.
\end{aligned}$$

It implies that $\Delta\mathbf{u}_k \to 0$ as $k \to \infty$. Thus, $\{\mathbf{u}_k\}$ converges. □

Theorem 12. *Under assumptions A1-A2 and constraints C1-C3, the error $\mathbf{e}_k$ of system (5.1) converges.*

Proof. From (5.25) and (5.26), we get

$$V(\mathbf{e}_k) \leq \mathbf{e}_k^T Q \mathbf{e}_k \leq V(\mathbf{e}_{k-1}) - \Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k \tag{5.30}$$

Moreover, $\Delta\mathbf{u}_k^T R \Delta\mathbf{u}_k \geq 0$ since R is positive definite. Hence,

$$V(\mathbf{e}_k) \leq \mathbf{e}_k^T Q \mathbf{e}_k \leq V(\mathbf{e}_{k-1}) \tag{5.31}$$

On the other hand, it can be deduced from (5.31) that the sequence $\{V(\mathbf{e}_k)\}$ is non-increasing and $V(\mathbf{e}_k) \geq 0 \ \forall\, k$. Consequently, $\{V(\mathbf{e}_k)\}$ converges. It leads to the convergence of $\{\mathbf{e}_k^T Q \mathbf{e}_k\}$. Thus, $\{\mathbf{e}_k\}$ converges. □

5.4 Numerical example

Consider the following system with transfer function

$$G(s) = \frac{1}{15s^2 + 8s + 1} + \theta_k \frac{0.8e^{-s}}{5s + 1} \tag{5.32}$$

where θ_k is the uncertain parameter: $\theta_k \in [-1, 1]$. The constraints of control input for system (5.32) are specified by

$$\mathbf{u}_l = -4, \mathbf{u}_h = 4, \delta\mathbf{u}_l = -5, \delta\mathbf{u}_h = 5, \Delta\mathbf{u}_l = -6, \Delta\mathbf{u}_h = 6. \tag{5.33}$$

Target trajectory is

$$r(t) = \begin{cases} 0, & t \in [0,5] \cup [36,40] \\ 0.1(t-5), & t \in [6,15] \\ 1, & t \in [16,25] \\ 1-0.1(t-25), & t \in [26,35] \end{cases} \tag{5.34}$$

The design parameters are chosen as follows: $Q = I_1, R = 0.05I_2$ where I_1, I_2 are identity matrices with appropriate dimension. For the stopping criteria, we choose $\epsilon = 0.01$ and `iter_max` $= 14$. To illustrate the time responses of the ILC control system, we choose sampling time 1 second and the number of samples $N = 41$. In our work, we use the software `cvx` [29] to solve the LMI problem (5.21).

The design results are shown in the Figures 5.1–5.5. Figures 5.1, 5.2, 5.3 show that the control input satisfies all input constraints (5.33). Moreover, in Figure 5.3, we plot the iterative input update $\Delta\mathbf{u}_{k+1}$ at the first iteration, 5th iteration, 10th iteration and 14th iteration to demonstrate that $\Delta\mathbf{u}_{k+1}$ decays. Therefore, the control input of system (5.32) converges. On the other hand, Figure 5.4 shows that the output of system (5.32) approaches to the desired reference trajectory and Figure 5.5 illustrates the convergence of the system error. It can be seen from Figure 5.5 that the infinity-norm of error converges but not monotonically. It is due to the variation of the uncertainty from one iteration to another. Note that in Theorem 12, we only prove the convergence of the system error, but the monotonic convergence of error needs to be further investigated.

5.5 Comparison of three proposed ILC algorithms

In this section, we aim to compare the proposed ILC algorithms applied to linear systems subject to various types of parametric uncertainty. In many practical systems, sometimes we do not exactly know the characteristic of the uncertainty in the system. Hence, it is not easy to determine that the uncertainty is time-invariant, time-varying or iteration-varying. Accordingly, we can test all the proposed ILC algorithms with the system and select the one which gives the best results. We will simulate three proposed ILC algorithms to the linear system (3.26) [33] when the uncertainty is time-invariant, time-varying, and iteration-varying. Table 5.1 describes the comparisons and corresponding responses.

When the uncertainty is time-invariant, Figures 5.6–5.7 show that the ILC-TI (Algorithm 1) converges fastest and tracks the reference input best while the ILC-IV (Algorithm 3) gives worst performance. The results are logical since the ILC-TI is designed for linear systems with time-invariant parametric uncertainty. Figure 5.8–5.9 display the best tracking

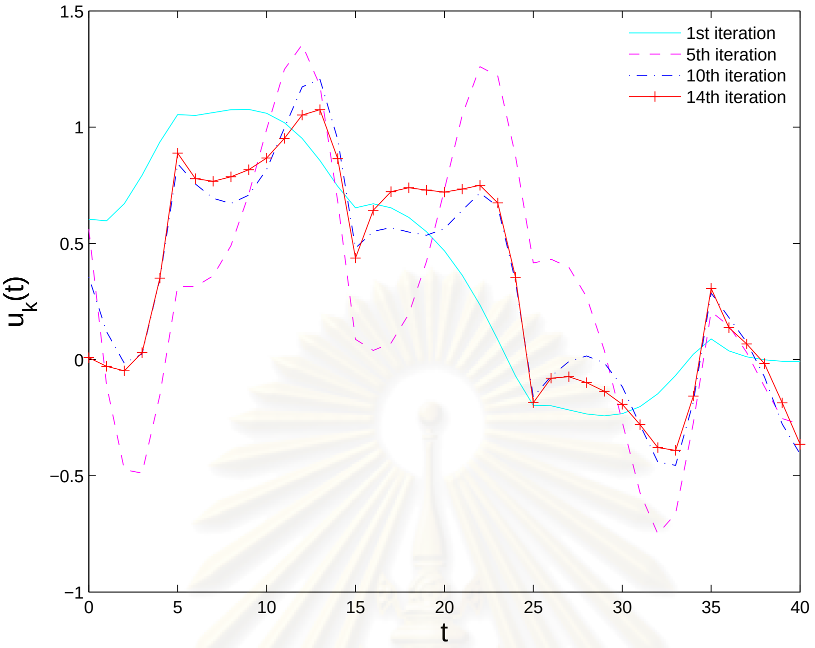

Figure 5.1: Control input of the robust ILC system with iteration-varying parametric uncertainty.

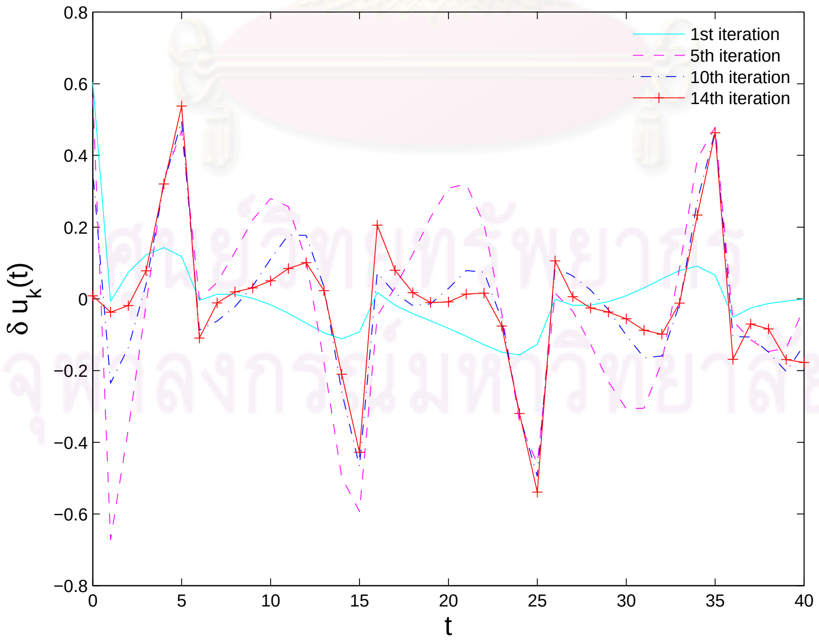

Figure 5.2: The difference of control input of the robust ILC system with iteration-varying parametric uncertainty w.r.t. time index.

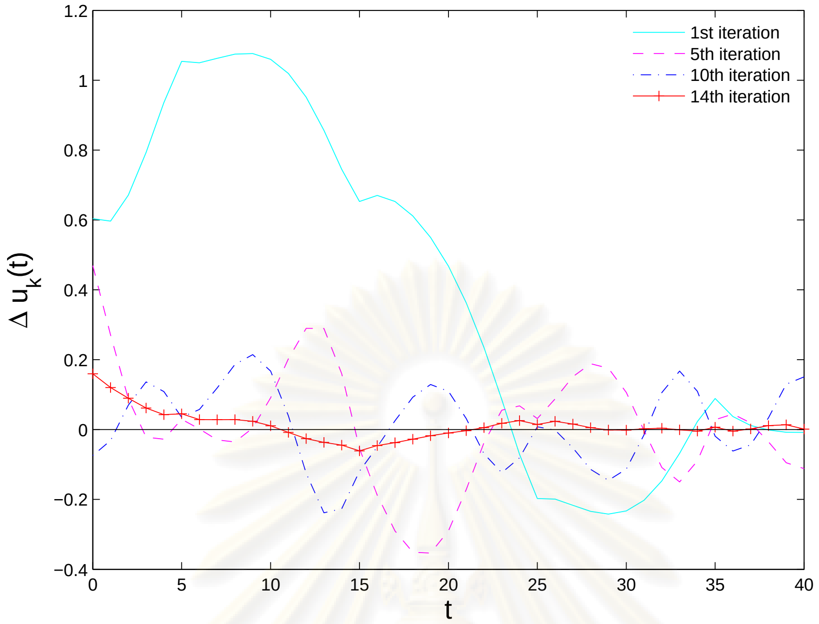

Figure 5.3: The difference of control input of the robust ILC system with iteration-varying parametric uncertainty w.r.t. iteration index.

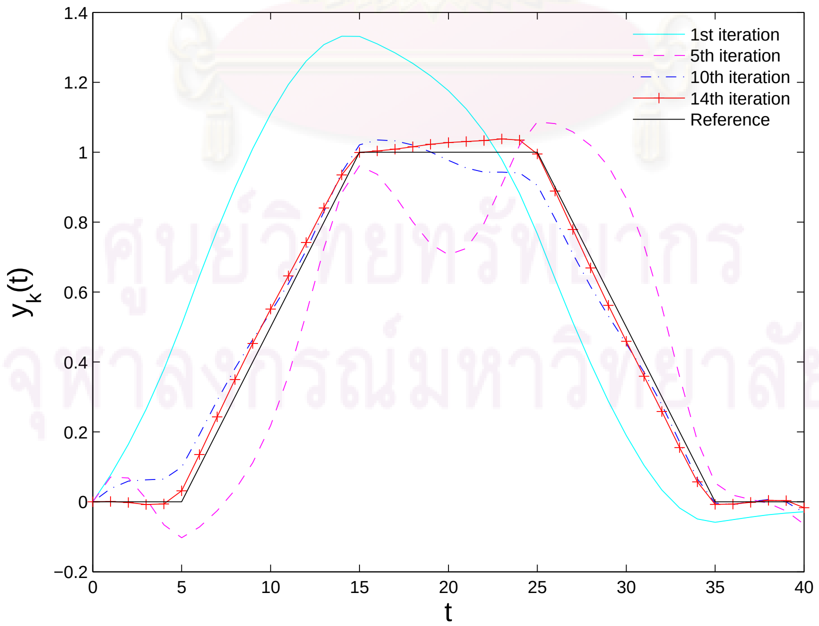

Figure 5.4: Output response of the robust ILC system with iteration-varying parametric uncertainty.

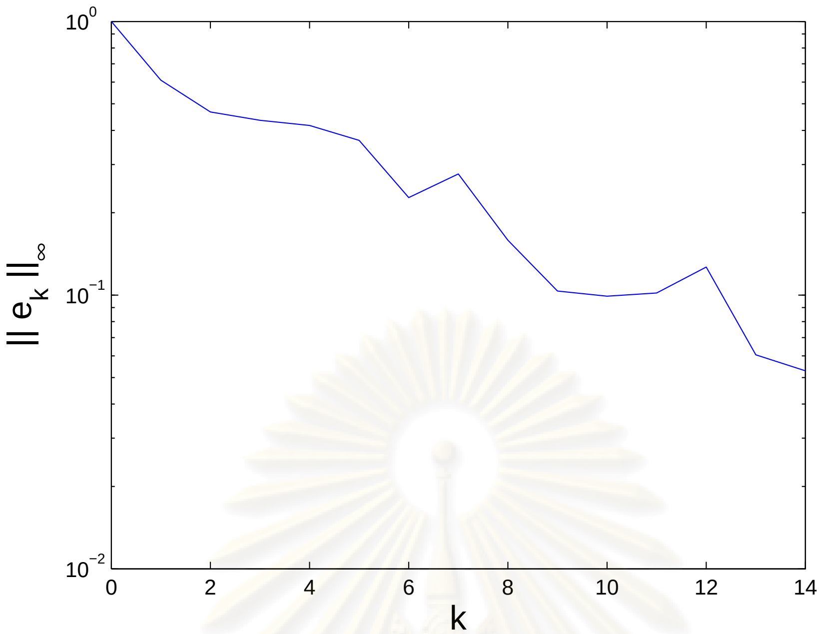

Figure 5.5: Infinity norm of error of the robust ILC system with iteration-varying parametric uncertainty vs. the number of iteration.

Table 5.1: Comparison of proposed ILC algorithms to linear systems with various types of parametric uncertainty.

Type of parametric uncertainty	ILC-TI	ILC-TV	ILC-IV
Time-invariant	Figures 5.6–5.7		
Time-varying	Figures 5.8–5.9		
Iteration-varying	Figures 5.10–5.11		

of the ILC-TV (Algorithm 2) as the uncertainty is time-varying, whereas the ILC-IV (Algorithm 3) exhibits poor performance. Finally, the comparison of ILC algorithms when the uncertainty is iteration-varying is illustrated in Figure 5.10–5.11. The ILC-IV (Algorithm 3) appears to converge slowest but gives smallest tracking error whilst ILC-TI and ILC-TV algorithms converge faster but track the reference input worse.

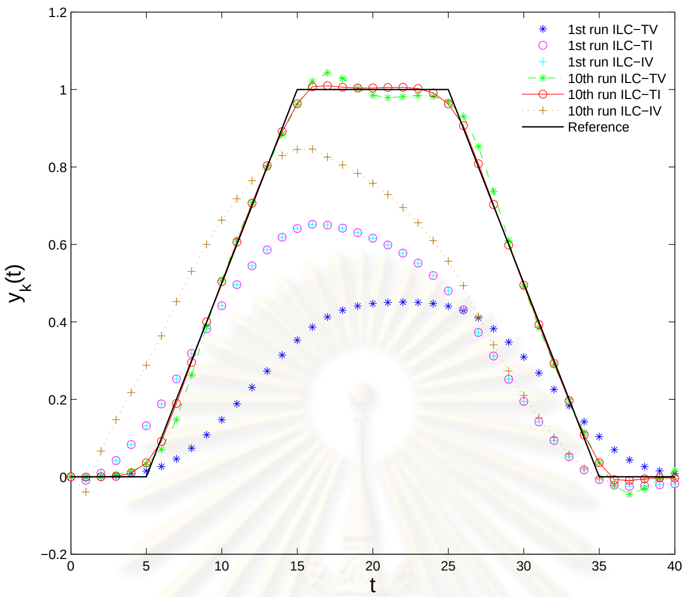

Figure 5.6: Output response of system with a time-invariant parametric uncertainty using three ILCs

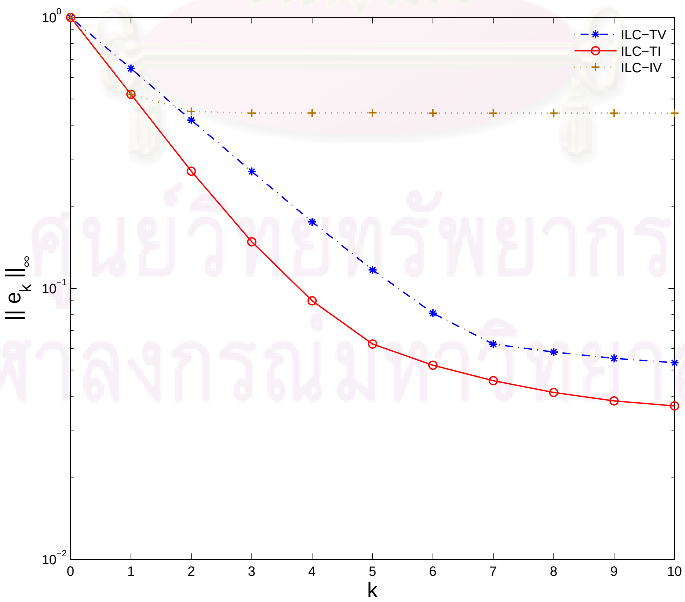

Figure 5.7: Infinity norm of system error with a time-invariant parametric uncertainty using three ILCs

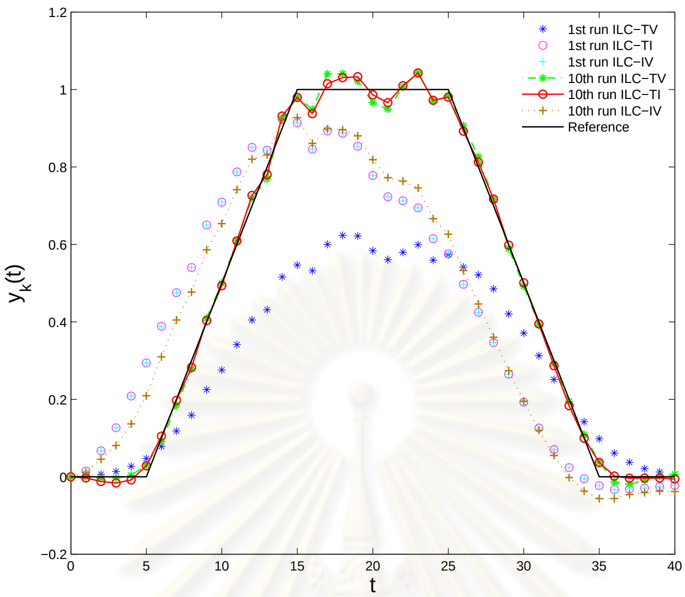

Figure 5.8: Output response of system with a time-varying parametric uncertainty using three ILCs

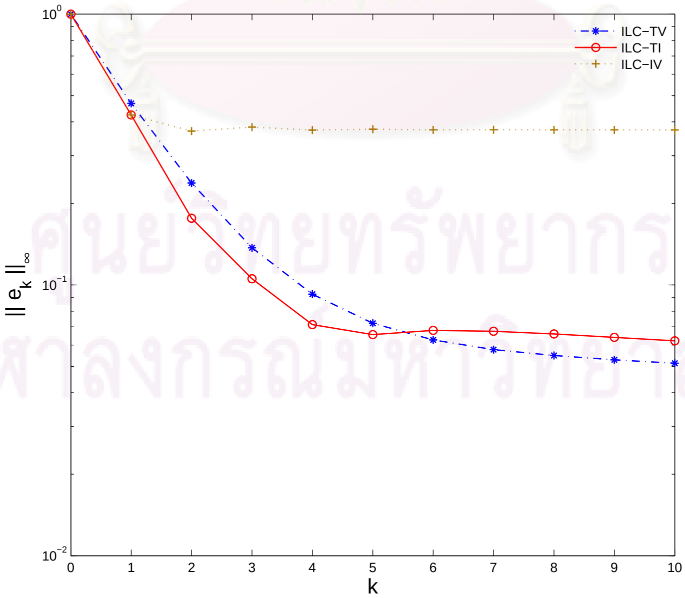

Figure 5.9: Infinity norm of system error with a time-varying parametric uncertainty using three ILCs

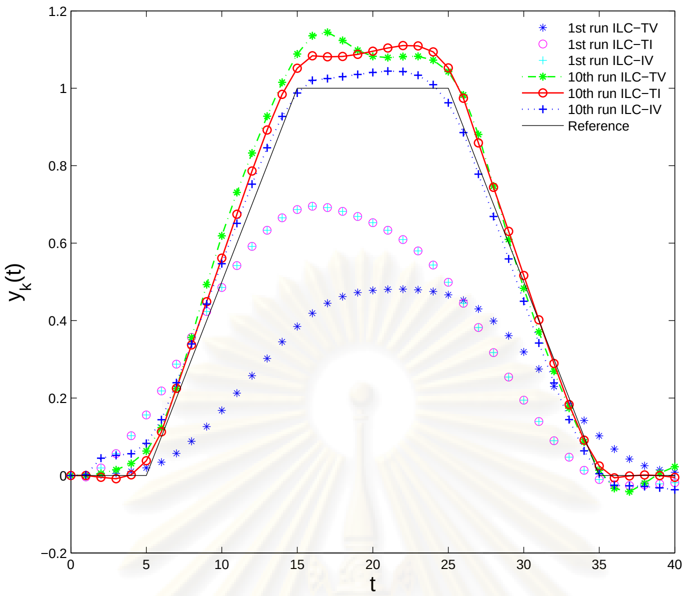

Figure 5.10: Output response of system with a iteration-varying parametric uncertainty using three ILCs

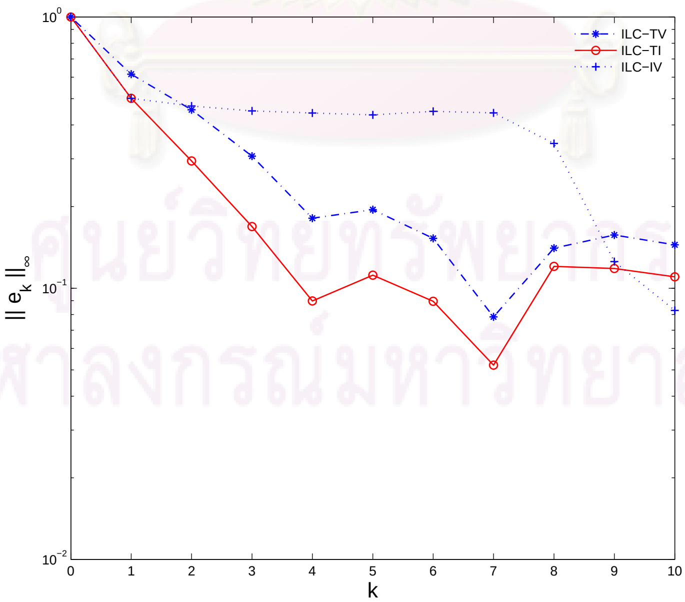

Figure 5.11: Infinity norm of system error with a iteration-varying parametric uncertainty using three ILCs

5.6 Conclusion

The robust ILC design for linear systems with iteration-varying parametric uncertainties is introduced in Chapter 5. First, the affine Markov model of the system is given and the worst-case performance analysis is presented to relax the initial min-max problem to a minimization one. Next, we investigate the dual problem of the minimization problem which can be described as a convex optimization problem over LMI constraints. An LMI-based ILC algorithm is provided afterward. Then, the convergence of the control input and the system error is proved. Consequently, the simulation results of a generic example are displayed to exhibit the effectiveness of the proposed algorithm. At the end, we show a comparison of the three proposed ILC algorithms in Chapter 3, Chapter 4 and Chapter 5 when applied to a same linear system in the presence of a time-invariant, time-varying and iteration-varying parametric uncertainty.

CHAPTER VI

APPLICATION TO PHYSICAL MODELS

6.1 Flexible link

Flexible links or flexible robot arms have been increasingly used in industry since they have several advantages over rigid links such as lighter weight, less power consumption, and faster response. In addition, they are encountered in spacecraft where the weight constraints result in flexible structures. Flexible links are modeled by highly non-linear and complex dynamics, and receive much attention in control engineering. Moreover, they become an experiment platform for research laboratories. In practice, a linear model of a flexible link is used in the control design. However, the linear model is only an approximation to the real system, whose parameters are subject to change due to working conditions [41]. Therefore, it is necessary to develop robust control design for industrial flexible link.

This session aims to develop a robust ILC algorithm to control the deflection of a single flexible link based on Quanser Inc. [42]. With the assumption that there is a parametric uncertainty in the flexible link system, we build a model which is an affine function of the uncertainty. Then, we develop a robust ILC algorithm which employs a quadratic performance criterion and formulates the design as a min-max optimization problem [33, 35, 36]. An upper-bound of the maximization problem is utilized, then the Lagrange dual problem of the minimization problem is considered. This dual problem is reformulated as a convex optimization problem over Linear Matrix Inequalities (LMIs) which can be efficiently solved.

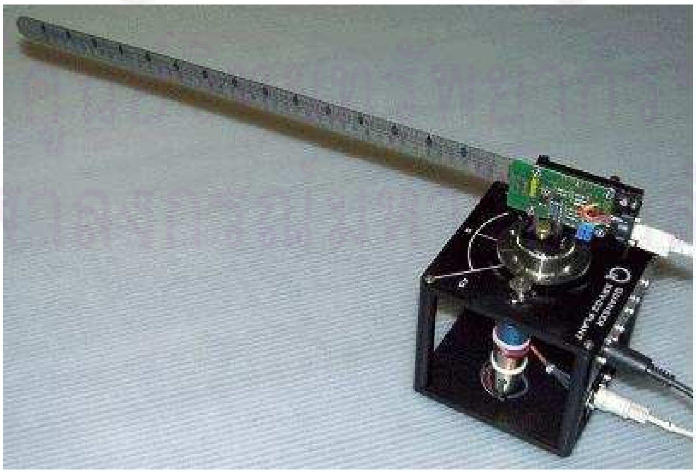

Figure 6.1: The single flexible link system.

Consider a single flexible link system which consists of a flexible beam and a servo DC

Table 6.1: Parameters of a single flexible link.

Parameter	Physical Meaning
R_m	Motor armature resistance
K_t	Motor torque constant
K_m	Motor Back-EMF constant
K_g	Total gear ratio
η_g	Gearbox efficiency
η_m	Motor efficiency
B_{eq}	Equivalent viscous damping coefficient as seen at the load
M	Mass of the flexible link
L	Length of the flexible link
J_{eq}	Equivalent inertia as seen at the load
$J_{\text{arm}} = \frac{ML^2}{3}$	Moment of inertia of the flexible link
ω_c	Damped natural frequency of flexible link

motor as in Figure 6.1. The motion for the flexible beam can be precisely described as an infinite-dimensional system. However, in many practical control designs, the flexible link is modeled as a finite order state-space equation (see [42] for details)

$$\begin{cases} \dot{x} = & Ax + Bu \\ y = & Cx \end{cases} \tag{6.1}$$

where

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{\omega_c^2 J_{\text{arm}}}{J_{\text{eq}}} & -\frac{\eta_m \eta_g K_t K_m K_g^2 + B_{\text{eq}} R_m}{J_{\text{eq}} R_m} & 0 \\ 0 & -\frac{\omega_c^2 (J_{\text{arm}} + J_{\text{eq}})}{J_{\text{eq}}} & \frac{\eta_m \eta_g K_t K_m K_g^2 + B_{\text{eq}} R_m}{J_{\text{eq}} R_m} & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{\eta_m \eta_g K_t K_g}{J_{\text{eq}} R_m} \\ -\frac{\eta_m \eta_g K_t K_g}{J_{\text{eq}} R_m} \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}.$$

The state vector of a flexible link system is defined as $x = \begin{bmatrix} \alpha & d & \dot{\alpha} & \dot{d} \end{bmatrix}^T$ where α is the angular displacement of the servo motor and d is the deflection angle of flexible link. The control input is defined as $u = V_m$ which is the voltage supplied to the servo motor. Note that Eq. (6.1) is a linear model with single input single output where the input is V_m and the output is d.

The parameters of the single flexible link are provided in Table 6.1.

We discretize (6.1) over the time interval $[0,T]$ and the number of samples is $N+1$. Thus, the sampling time is T/N. Let k be the iteration index. Using super-vector notation, the discretized model is described as follows [43]

$$\mathbf{y}_k = G\mathbf{u}_k \tag{6.2}$$

where

$$\begin{aligned}
\mathbf{y}_k &= \begin{bmatrix} y_k(1)^T & y_k(2)^T & \dots & y_k(N)^T \end{bmatrix}^T, \\
\mathbf{u}_k &= \begin{bmatrix} u_k(0)^T & u_k(1)^T & \dots & u_k(N-1)^T \end{bmatrix}^T, \\
G &= \begin{bmatrix} CB & 0 & 0 & 0 \\ CAB & CB & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{N-1}B & \cdots & CAB & CB \end{bmatrix}.
\end{aligned}$$

Next, we consider the mass of the flexible link as an uncertain parameter of system (6.1). In particular,

$$M = M_0(1 + q\theta) \tag{6.3}$$

where M_0 is the nominal mass, θ is a real uncertainty, and q is a positive real number. In this work, we assume that $\theta \in \Theta$ where $\Theta = \{\theta : |\theta| \leq 1\}$. Then, q has a physical meaning to be the bound of uncertainty. As the flexible link model (6.1) is subjected to parametric uncertainty, the discretized model G becomes a function of θ, *i.e.*, $G(\theta)$. Afterward, depend on the type of uncertainty, we will apply the corresponding designed ILC algorithm to control the flexible link.

6.1.1 Time-invariant parametric uncertainty

To apply the designed ILC algorithm to this flexible link system, we seek matrices G_0 and G_1 so that,

$$G(\theta) = G_0 + \theta G_1 \tag{6.4}$$

where G_0 represents the nominal system and G_1 is the uncertain dynamic matrix. Utilizing the procedure in the Section 2.4.1, the nominal matrix G_0 is determined as in (2.26) by setting the mass of the flexible link at its nominal value. Then, the uncertainty interval $[-1, 1]$ is discretized and the impulse responses of (6.1) corresponding to the discretized values of θ are calculated. Based on this set of impulse responses, we find out the bounds of them and then, obtain G_1 as in (2.29).

Now, let the system (6.1) be discretized by a sampling time 0.012 seconds, the number of samples is 101, *i.e.*, $N = 100$. The parameters of the flexible link are taken from Quanser [42]. Figure 6.2 show the impulse response of the nominal system as well as the upper bound and the lower bound of the impulse responses with $n = 40$.

The flexible link system is simulated using Algorithm 1 with the following simulation parameters. The mass of flexible link is assumed to have $\pm 20\%$ uncertainty, thus $q = 0.2$.

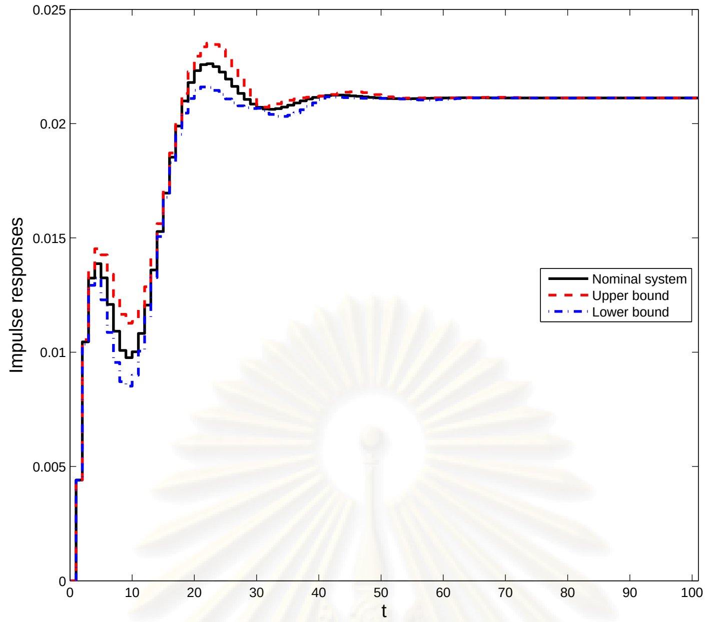

Figure 6.2: Impulse response of the nominal system and the bounds of impulse responses of system with time-invariant uncertainty.

The weighting matrices are $Q = I_1, R = 0.01I_2$ where I_1, I_2 are identity matrices with appropriate dimensions. For the stopping criteria, we choose $\epsilon = 0.01$ and `iter_max` $= 30$.

The desired reference trajectory is

$$r(t) = \begin{cases} 0, & t \in [0, 15] \cup [86, 100] \\ \frac{1}{25}(t-15), & t \in [16, 40] \\ 1, & t \in [41, 60] \\ 1 - \frac{1}{25}(t-60), & t \in [61, 85] \end{cases}$$

The constraints of control inputs are specified by

$$\mathbf{u}_l = -3, \mathbf{u}_h = 3, \delta\mathbf{u}_l = -4, \delta\mathbf{u}_h = 4, \Delta\mathbf{u}_l = -4, \Delta\mathbf{u}_h = 4. \quad (6.5)$$

To verify the effectiveness of the proposed robust ILC algorithm, the dynamic model (6.4) is used in steps 2 and 3 of Algorithm 1, whereas the flexible link in step 4 is modeled using the Finite Element Method [44]. In particular, the flexible beam is partitioned into 3 elements and 4 nodes which result in an 8th order model. Moreover, in the simulation, the uncertain parameter θ is randomly selected from the uncertainty interval and kept unchanged for all iterations.

In this thesis, we simulate the proposed ILC algorithms to physical models such as a flexible link in this section and a distillation column in next section using a computer with the following configuration: CPU Core 2 Duo 2.2 GHz, RAM 2Gb, MATLAB 7.4.0 (R2007a). The simulation results are presented in Figures 6.3–6.7. Figures 6.3–6.5 show that the control input satisfies all the constraints C1–C3 specified by (6.5). The convergence of the control input is demonstrated in Figure 6.5 where the control input update $\Delta\mathbf{u}_{k+1}$ goes to zero throughout executions. In addition, Figure 6.6 illustrates that the output of the flexible link system tracks the trapezoidal reference input. At the first iteration, the tracking performance is very bad when the system output is far different from the target trajectory, nonetheless, the tracking performance is improved through executions and is acceptable after 30 trials. The convergence of the system error is illustrated in Figure 6.7.

In addition, Figure 6.8 shows the computational time to solve the LMI problem (3.19) of flexible link with a time-invariant parametric uncertainty for all iterations. We can see that the computational time grows up as the number of iteration increases. It can be explained as follows. The more iterations are run, the better the tracking performance is, hence, the smaller the control input update $\Delta\mathbf{u}_{k+1}$ and the system error $\mathbf{e}_{k+1}$ are. It is conjectured that the solution of the LMI problem gets closer to the boundary of the constraints. As a result, the computational time is longer. It is noted that the sampling time is 0.012 second while the maximum computational time during 30 iterations is about 8.3 seconds, so, the LMI problem (3.19) should be solved off-line.

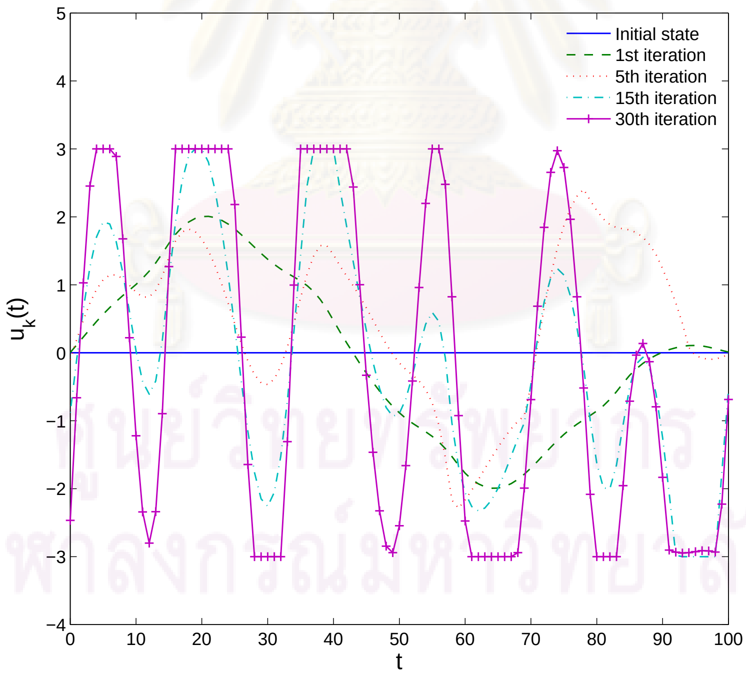

Figure 6.3: Control input of flexible link system with time-invariant parametric uncertainty.

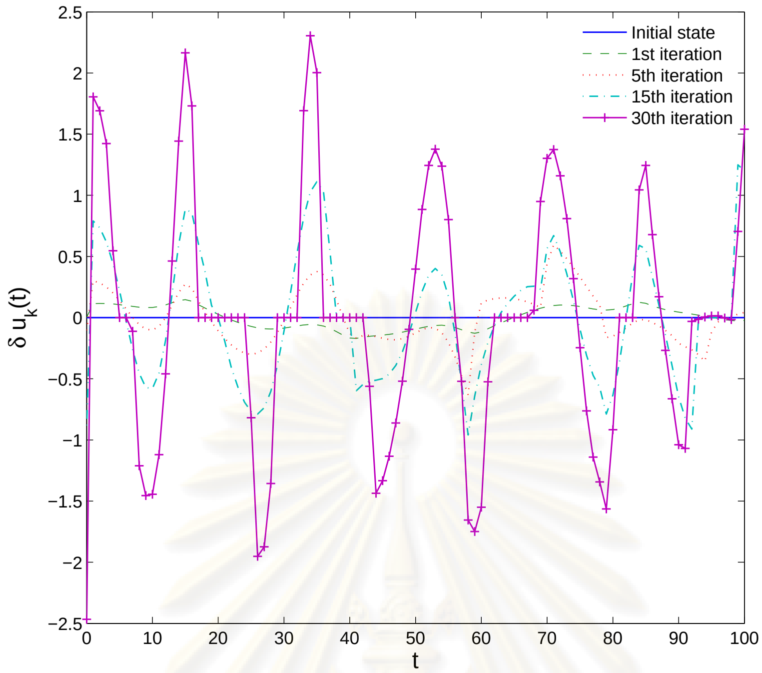

Figure 6.4: The difference of control input of flexible link system with time-invariant parametric uncertainty w.r.t. time index.

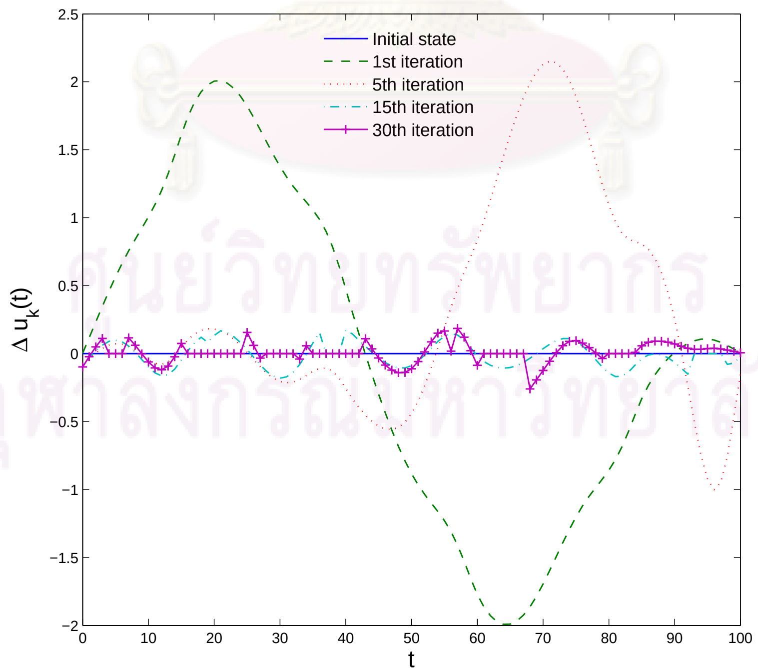

Figure 6.5: The difference of control input of flexible link system with time-invariant parametric uncertainty w.r.t. iteration index.

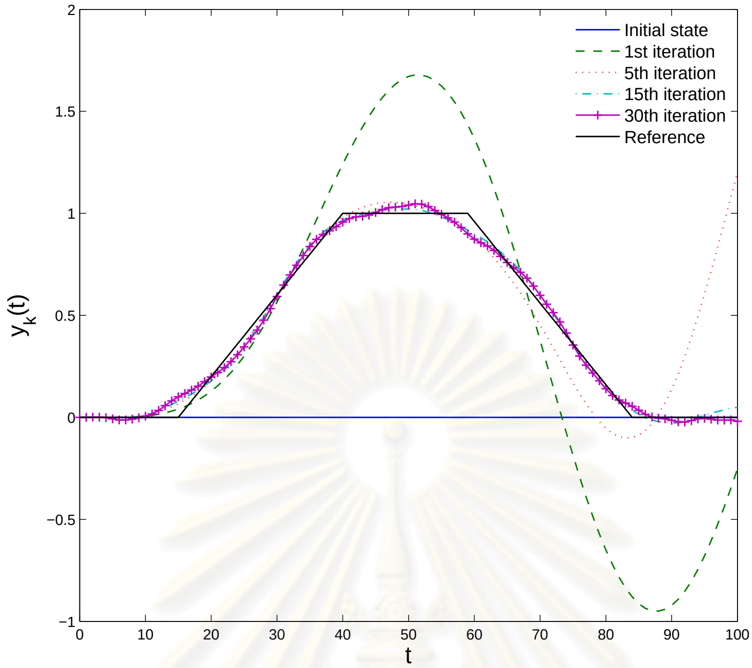

Figure 6.6: Output response of flexible link system with time-invariant parametric uncertainty.

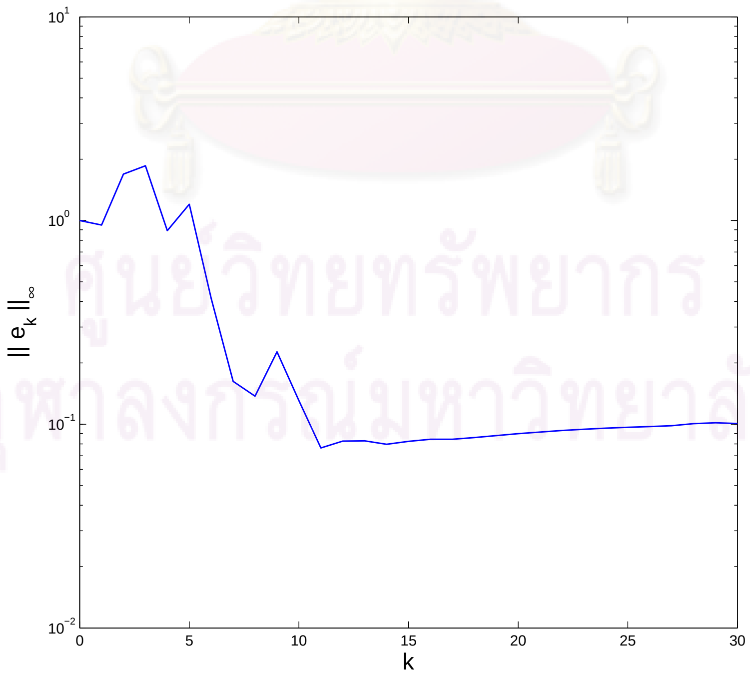

Figure 6.7: Infinity-norm of error of flexible link system with time-invariant parametric uncertainty.

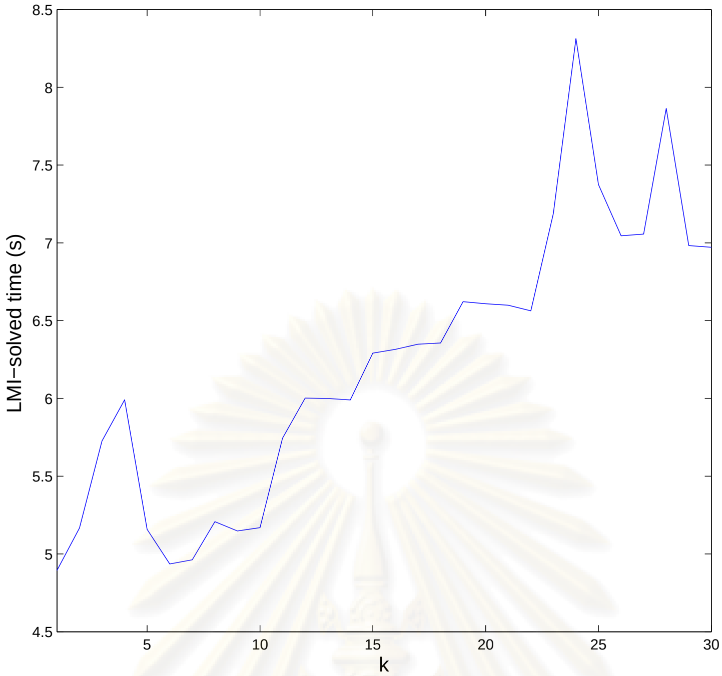

Figure 6.8: Computational time of flexible link system with time-invariant parametric uncertainty vs. the number of iterations.

6.1.2 Time-varying parametric uncertainty

In this part, the sampling time is set to be 0.012 sec. and the number of samples is 101. The parameters of the flexible link are taken from [42] and we consider the mass of flexible link has ±20% uncertainty, *i.e.*, $q = 0.2$. Now, we further assume that the mass of flexible link can be varied with time by picking up or dropping some weight on it during each iteration corresponding to the reference trajectory. The desired reference trajectory is

$$r(t) = \begin{cases} 0, & t \in [0, 15] \cup [86, 100] \\ \frac{1}{25}(t-15), & t \in [16, 40] \\ 1, & t \in [41, 60] \\ 1 - \frac{1}{25}(t-60), & t \in [61, 85] \end{cases}$$

In each iteration, the mass of flexible link is changed two times at the beginning of the iteration and at the sampling time 41. Hence, the uncertainty θ is time-varying respectively with the mass of flexible link. Figure 6.9 illustrates the mass of flexible link as a time-varying parameter.

Now, we still use the procedure in the Section 2.4.1 to determine the nominal Makov matrix G_0 and the uncertain Markov matrix G_1. Since the uncertainty is time-varying, the Markov matrix is different from the case of time-invariant uncertainty, hence, the impulse responses will be different, too. Following is Figure 6.10 which shows the impulse response of

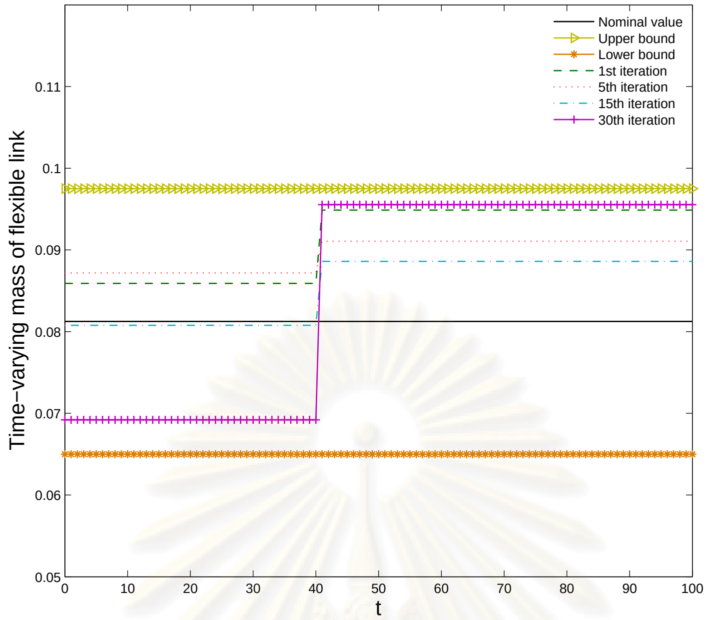

Figure 6.9: Mass of flexible link as a time-varying parameter.

the nominal system as well as the upper bound and the lower bound of the impulse responses with $n = 40$ in each change of the uncertainty.

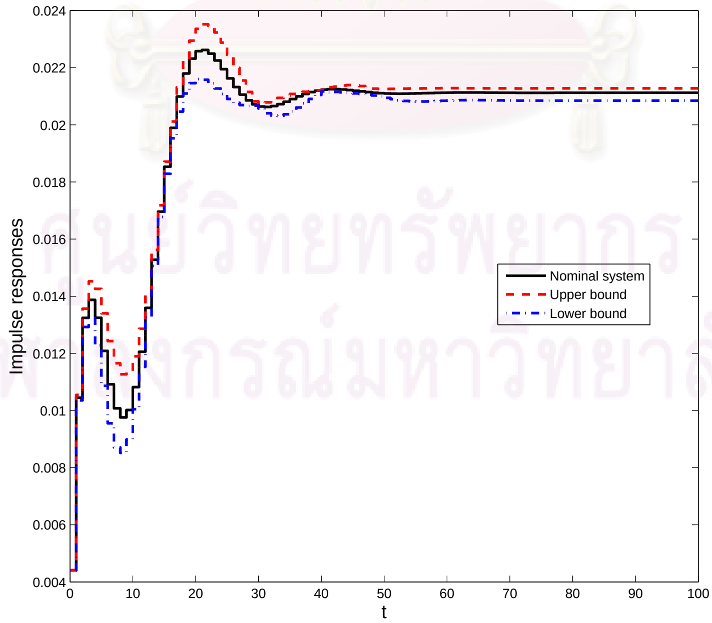

Figure 6.10: Impulse response of the nominal system and the bounds of impulse responses of system with time-varying uncertainty.

The constraints of control inputs are specified by

$$\mathbf{u}_l = -3, \mathbf{u}_h = 3, \delta\mathbf{u}_l = -4, \delta\mathbf{u}_h = 4, \Delta\mathbf{u}_l = -4, \Delta\mathbf{u}_h = 4. \tag{6.6}$$

The design parameters are chosen as follows: $Q = I_1, R = 0.01I_2$ where I_1, I_2 are identity matrices with appropriate dimension. Applying the proposed robust ILC design to the flexible link system, the 4th order model is used in steps 2 and 3 of the algorithm, whereas the flexible link in step 4 is modeled using the Finite Element Method [44]. In particular, the flexible beam is partitioned into 3 elements and 4 nodes which result in an 8th order model. In the simulation, the parameter θ is randomly selected from the uncertainty interval and is changed with time as described above. For the stopping criteria, we choose $\epsilon = 0.01$ and `iter_max` $= 30$.

The simulation results are presented in Figures 6.11–6.15. Figures 6.11–6.13 show that the control input converges and satisfies all the constraints C1–C3 specified by (6.6). Figure 6.14 illustrates that the output of the flexible link system tracks the trapezoidal reference input. Moreover, the system error in Figure 6.15 is fluctuated but tends to decrease . It is the change of the mass of flexible link as a time-varying parameter that leads to the variation of the system error. Figure 6.16 demonstrates the computational time to solve the LMI problem (4.25) of flexible link with a time-varying parametric uncertainty. The computational time increases throughout iterations. The explanation for this phenomenon is like in the case of time-invariant parametric uncertainty.

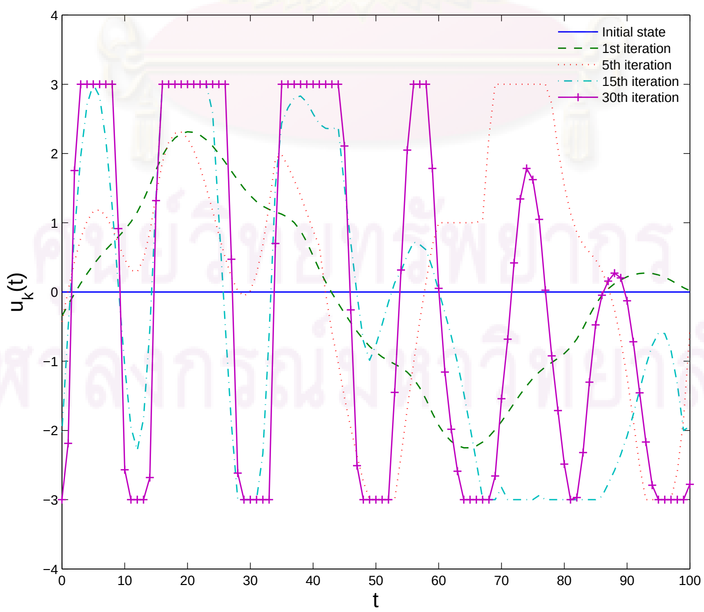

Figure 6.11: Control input of flexible link system with time-varying parametric uncertainty.

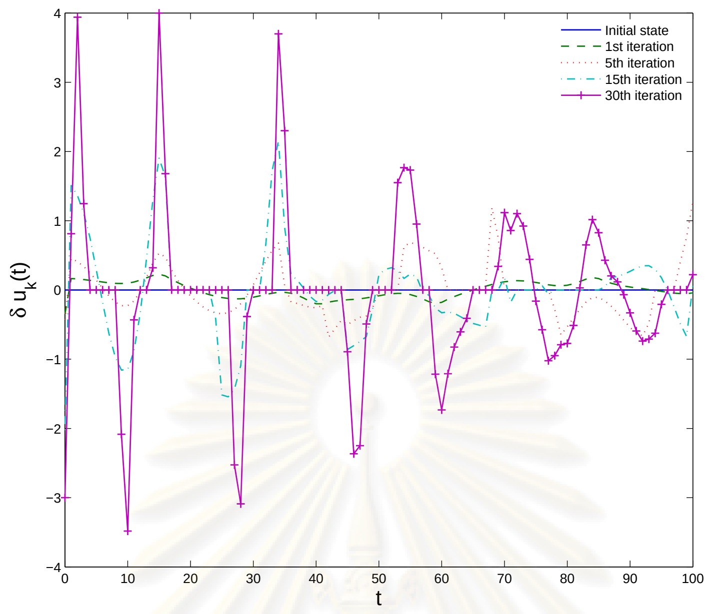

Figure 6.12: The difference of control input of flexible link system with time-varying parametric uncertainty w.r.t. time index.

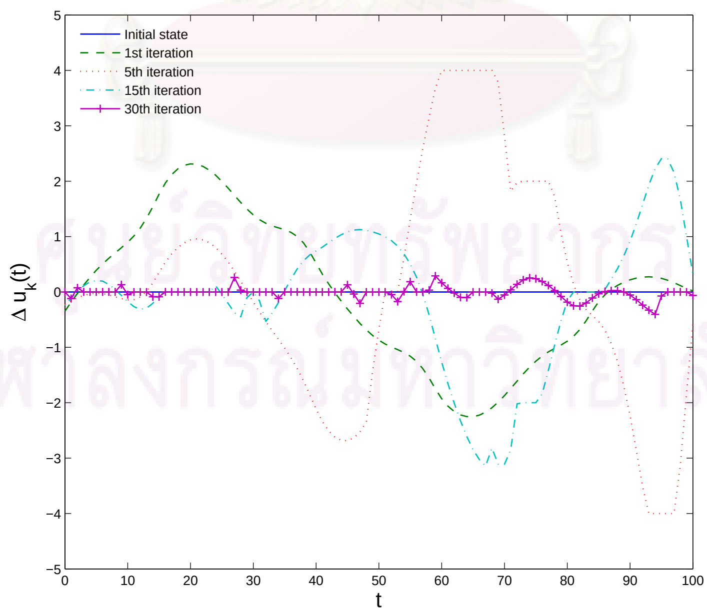

Figure 6.13: The difference of control input of flexible link system with time-varying parametric uncertainty w.r.t. iteration index.

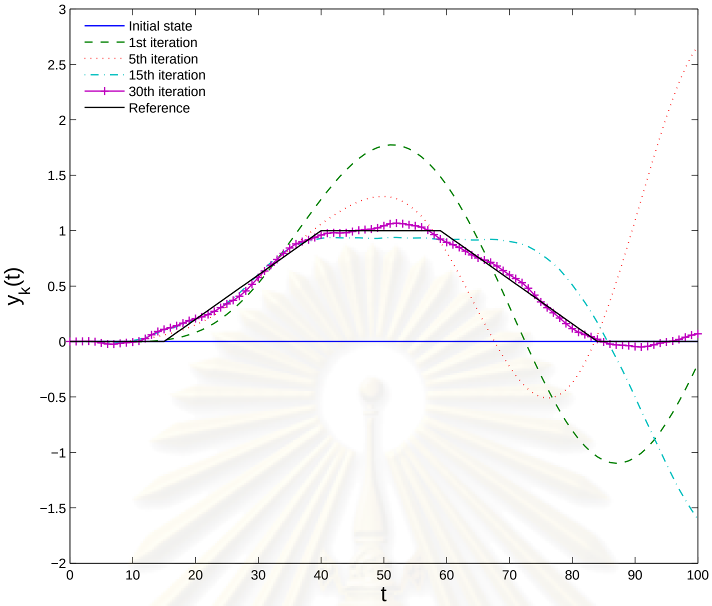

Figure 6.14: Output response of flexible link system with time-varying parametric uncertainty.

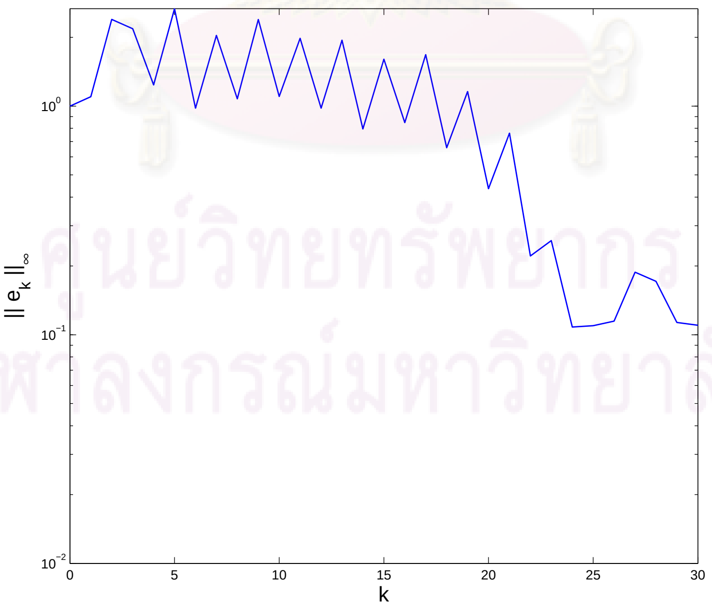

Figure 6.15: Infinity-norm of error of flexible link system with time-varying parametric uncertainty vs. the number of iterations.

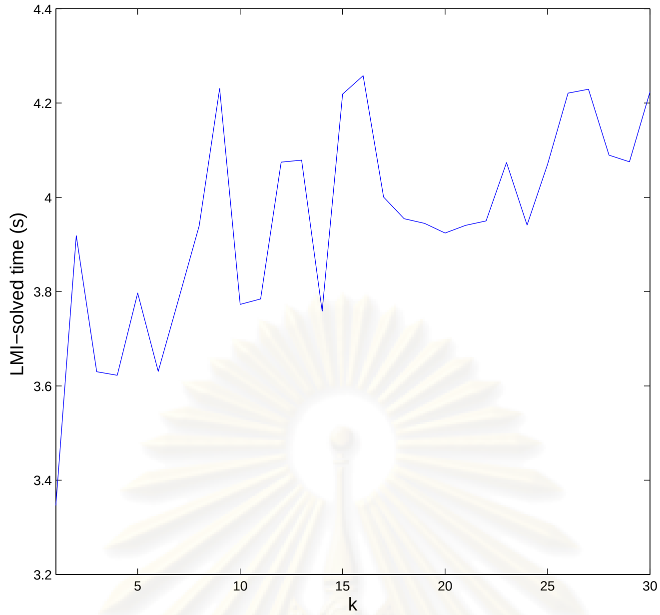

Figure 6.16: Computational time of flexible link system with time-varying parametric uncertainty vs. the number of iterations.

6.1.3 Iteration-varying parametric uncertainty

In our experiment, the sampling time is 0.012 sec. and the number of samples is 101. The parameters of the flexible link are taken from [42] and we assume the mass of flexible link has ±20% uncertainty, *i.e.*, $q = 0.2$. Following is Figure 6.17 which demonstrates the mass of flexible link as an iteration-varying parameter.

The desired reference trajectory is

$$r(t) = \begin{cases} 0, & t \in [0, 15] \cup [86, 100] \\ \frac{1}{25}(t-15), & t \in [16, 40] \\ 1, & t \in [41, 60] \\ 1 - \frac{1}{25}(t-60), & t \in [61, 85] \end{cases}$$

The constraints of control inputs are specified by

$$\mathbf{u}_l = -3, \mathbf{u}_h = 3, \delta\mathbf{u}_l = -4, \delta\mathbf{u}_h = 4, \Delta\mathbf{u}_l = -4, \Delta\mathbf{u}_h = 4. \quad (6.7)$$

The design parameters are chosen as follows: $Q = 0.9I_1, R = 0.01I_2$ where I_1, I_2 are identity matrices with appropriate dimension. Applying the proposed robust ILC design to the flexible link system, the 4th order model is used in steps 2 and 3 of the algorithm, whereas the flexible link in step 4 is modeled using the Finite Element Method [44]. In particular, the flexible beam is partitioned into 3 elements and 4 nodes which result in an 8th order model. In the simulation, the parameter θ is randomly selected from the uncertainty interval and is changed with iterations. For the stopping criteria, we choose $\epsilon = 0.01$ and `iter_max` $= 30$.

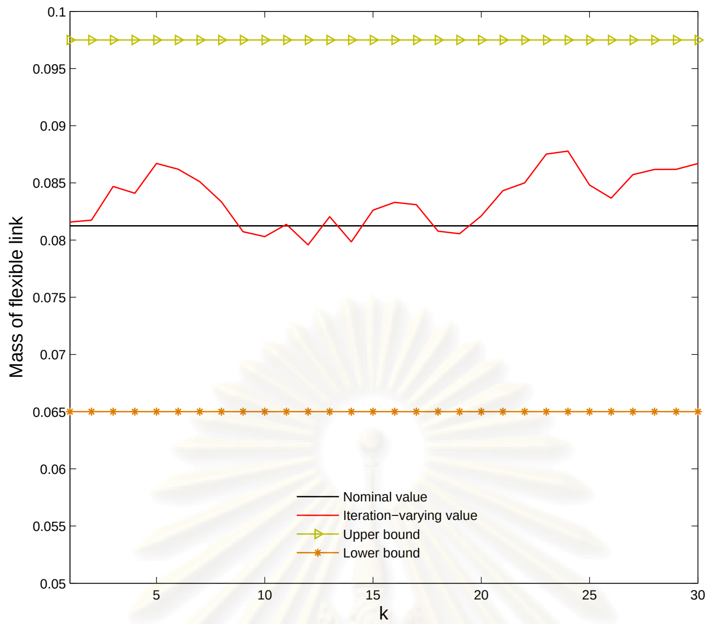

Figure 6.17: Mass of flexible link as an iteration-varying parameter.

The simulation results are presented in Figures 6.18–6.22. Figures 6.18–6.20 illustrate that the control input satisfies all the constraints C1–C3 specified by (6.7). In addition, the control input update $\Delta\mathbf{u}_{k+1}$ tends to decay as shown in Figures 6.20 which shows the convergence of the control input. On the other hand, Figure 6.21 demonstrates that the output of the flexible link system tracks the trapezoidal reference input. The convergence of the system error is displayed in Figure 6.22. It seems that the convergence of the system error in this case is worse than in the case when the mass of flexible link is time-invariant, but better than in the case when the mass of flexible link is changed as a time-varying parameter. Moreover, Figure 6.23 displays the computational time to solve the LMI problem (5.21) of flexible link with an iteration-varying parametric uncertainty. The computational time increases as more trials are executed. It can be explained as in the previous sections.

6.2 Distillation column

Consider a distillation column system which has the following LV-configuration

$$\begin{bmatrix} x_D \\ x_B \end{bmatrix} = H(s) \begin{bmatrix} L \\ V \end{bmatrix} \tag{6.8}$$

where x_D is the distillate product composition [mole fraction],
x_B is the bottom product composition [mole fraction],
L is the reflux flow [kmol/min],
V is the boilup flow [kmol/min].

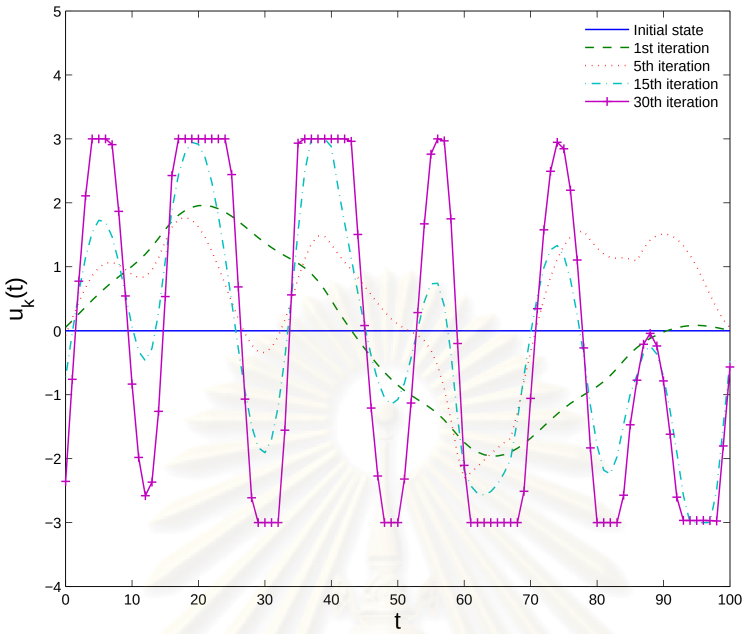

Figure 6.18: Control input of flexible link system with iteration-varying parametric uncertainty.

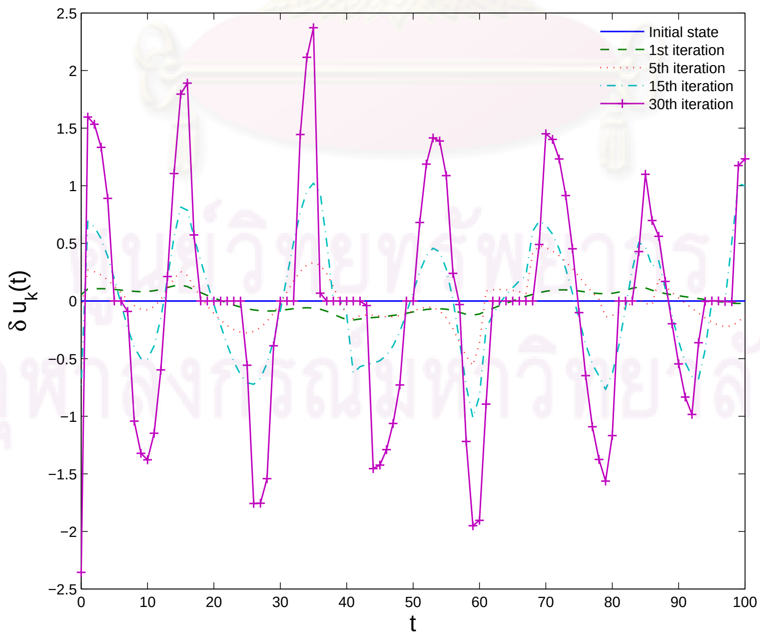

Figure 6.19: The difference of control input of flexible link system with iteration-varying parametric uncertainty w.r.t. time index.

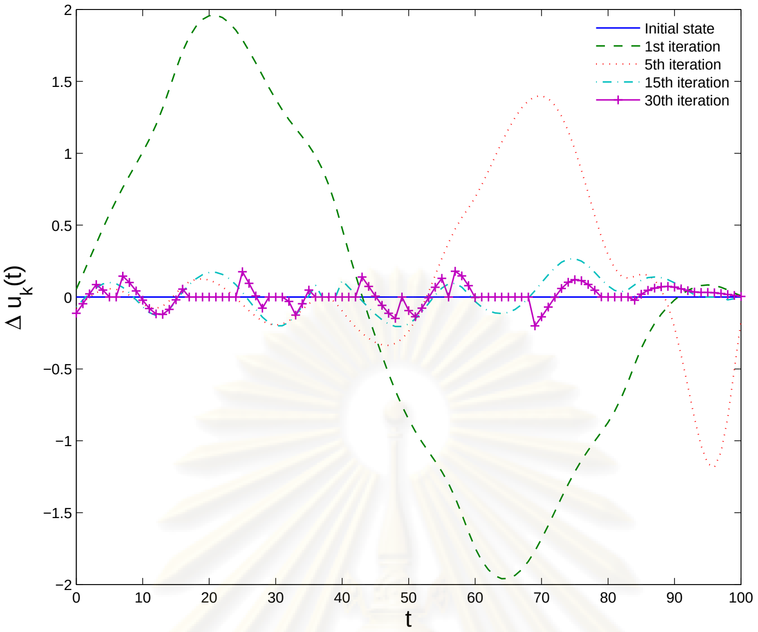

Figure 6.20: The difference of control input of flexible link system with iteration-varying parametric uncertainty w.r.t. iteration index.

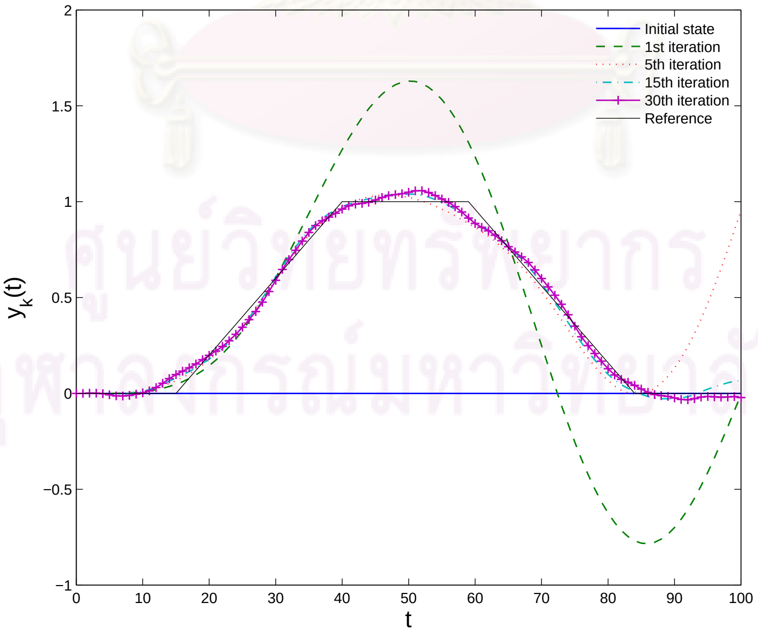

Figure 6.21: Output response of flexible link system with iteration-varying parametric uncertainty.

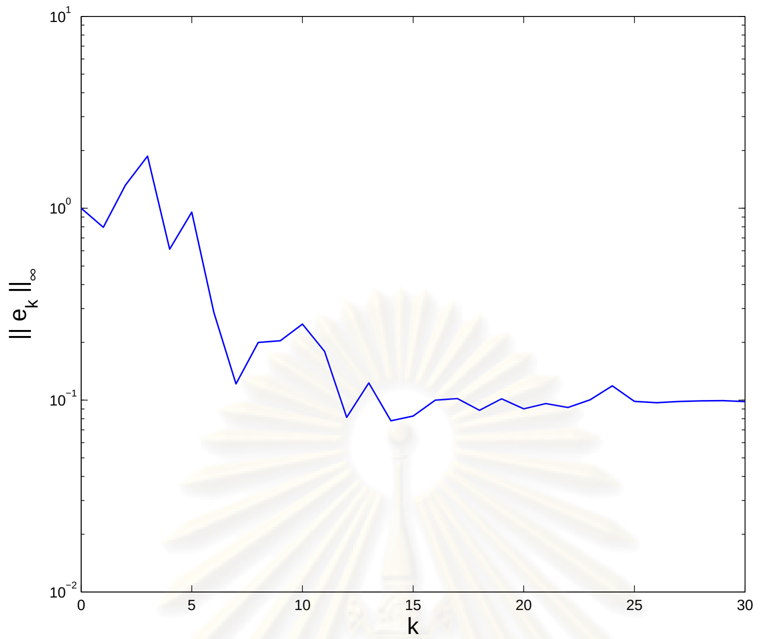

Figure 6.22: Infinity-norm of error of flexible link system with iteration-varying parametric uncertainty vs. the number of iterations.

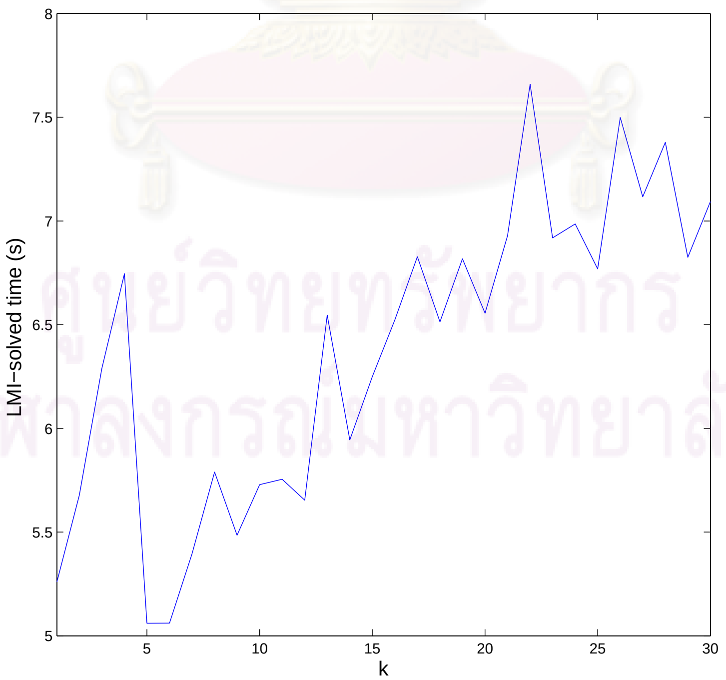

Figure 6.23: Computational time of flexible link system with iteration-varying parametric uncertainty vs. the number of iterations.

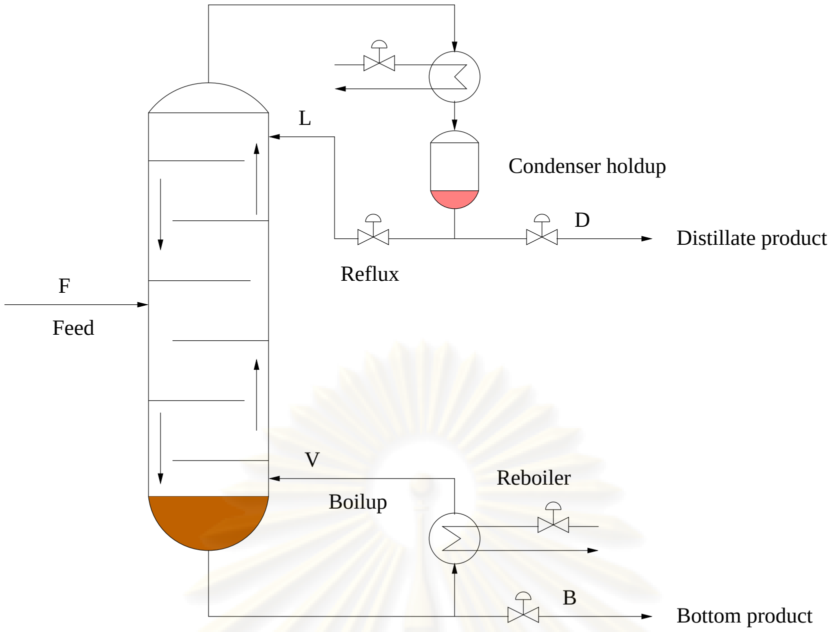

Figure 6.24: A 2-product distillation column system.

$$H(s) = \begin{bmatrix} h_{11}(s) & h_{12}(s) \\ h_{21}(s) & h_{22}(s) \end{bmatrix} \tag{6.9}$$

inwhich $h_{11}(s)$, $h_{12}(s)$, $h_{21}(s)$, $h_{22}(s)$ are first-order with time-delay systems,

$$h_{ij}(s) = \frac{k_{ij}e^{-\tau_{ij}s}}{T_{ij}s+1}, \;\; i,j = 1,2.$$

Suppose that K_p is the steady-state gain matrix of (6.8), then the Relative Gain Array (RGA) of this system is defined by

$$ij\text{th element of RGA} = (ij\text{th element of } K_p)(ji\text{th element of } K_p^{-1}) \tag{6.10}$$

When studying MIMO systems, we have to consider the interaction between the inputs and outputs of the system, otherwise, the control performance will not be as expected. The RGA matrix is one of the criteria to evaluate these interactions. The larger the elements of RGA matrix are, the more interactions in the system happen. Besides, if the values of elements of RGA matrix are around 0.5, the interactions in the system are much significant. In addition, the RGA matrix is useful for the decoupling of the control inputs or system outputs to avoid the interactions.

6.2.1 Time-invariant parametric uncertainty

To apply the designed ILC algorithm to this type of distillation column system, we assume that the time constant and time-delay constant of each block transfer function $h_{11}(s)$, $h_{12}(s)$,

$h_{21}(s)$, $h_{22}(s)$ are perturbed and these uncertain parameters can be represented by a parametric uncertainty vector $\theta = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8]^T$, *i.e.*,

$$T_{11} = \overline{T}_{11}(1 + q\theta_1),\ T_{12} = \overline{T}_{12}(1 + q\theta_2),\ T_{21} = \overline{T}_{21}(1 + q\theta_3),\ T_{22} = \overline{T}_{22}(1 + q\theta_4)$$

$$\tau_{11} = \overline{\tau}_{11}(1 + q\theta_5),\ \tau_{12} = \overline{\tau}_{12}(1 + q\theta_6),\ \tau_{21} = \overline{\tau}_{21}(1 + q\theta_7),\ \tau_{22} = \overline{\tau}_{22}(1 + q\theta_8)$$

where q is a positive real constant which has the meaning as the bound of uncertainty, $\theta \in \Theta = \{\theta : \|\theta\|_\infty \leq 1\}$.
The system (6.8) is discretized and reformulated in the super-vector framework. Then, the Markov matrix G of the system (6.8) is analysed to be an affine function of the uncertainty θ,

$$G(\theta) = G_0 + \sum_{i=1}^{8} \theta_i G_i \tag{6.11}$$

where G_0 represents the nominal system and G_i's, $i = \overline{1, 8}$ represent the uncertain dynamic matrices. Like in the previous example, each matrice $G_i, i = \overline{1, 8}$ is found by (2.29) in which θ_j's, $j \neq i, j = \overline{1, m}$ is kept at their nominal values whereas θ_i is varied in the uncertainty interval to calculate the corresponding impulse responses and obtain the upper bound as well as the lower bound of these impulse responses. Next, the affine model (6.11) is used in the step 2 and 3 of the Algorithm 1 whereas the linearized model (6.8) is the validating model in the step 4.

In this part, the Luyben-Vinante model of distillation column [45] is utilized where the block transfer functions in (6.9) are

$$\begin{aligned} h_{11}(s) &= \frac{-2.16e^{-s}}{8s + 1}, \qquad h_{12}(s) = \frac{1.26e^{-0.3s}}{9.5s + 1} \\ h_{21}(s) &= \frac{-2.75e^{-1.8s}}{9.5s + 1}, \quad h_{22}(s) = \frac{4.28e^{-0.35s}}{9.2s + 1} \end{aligned} \tag{6.12}$$

The steady-state gain matrix of Luyben-Vinante distillation column is

$$K_p = \begin{bmatrix} -2.16 & 1.26 \\ -2.75 & 4.28 \end{bmatrix}$$

Hence, RGA of Luyben-Vinante distillation column can be calculated by (6.10)

$$RGA = \begin{bmatrix} 1.5995 & -0.5995 \\ -0.5995 & 1.5995 \end{bmatrix}$$

Look at the elements in the anti-diagonal of RGA matrix of Luyben-Vinante distillation column, we can see there are some interactions in the systems.

The distillation column is considered to work around the operating point in which $x_D = 0.98,\ x_B = 0.01$. Let the sample time be 0.5 minute, the number of samples is 101. We assume that the time constant and the time-delay constant in each block transfer function of $G(s)$ has $\pm 15\%$ uncertainty, thus $q = 0.15$. The design parameters are chosen as follows: $Q = I_1, R = 0.01I_2$ where I_1, I_2 are identity matrices with appropriate dimension.

The desired reference trajectory is

$$r(t) = \begin{cases} (0,0)^T, & t \in [0,8] \\ (0.005,0)^T, & t \in [9,25] \\ (0.005,0.002)^T, & t \in [26,50] \end{cases}$$

The constraints of control inputs are specified by

$$\mathbf{u}_l = -0.05, \mathbf{u}_h = 0.05, \delta\mathbf{u}_l = -3, \delta\mathbf{u}_h = 3, \Delta\mathbf{u}_l = -0.1, \Delta\mathbf{u}_h = 0.1. \tag{6.13}$$

In the simulation, the parameter θ is randomly selected from the uncertainty interval and kept unchanged for all iterations. For the stopping criteria, we choose $\epsilon = 10^{-5}$ and `iter_max` $= 10$.

The simulation results are presented in Figures 6.25–6.29. Figures 6.25–6.27 show that the control input converges and satisfies all the constraints C1–C3 specified by (6.13). Figure 6.28 illustrates that the outputs of the distillation column system track the step reference inputs. We observe that there are abrupt changes of the output responses in Figure 6.28. They are due to the interaction between the inputs and outputs of the distillation column system as mentioned above. When the first output changes to a new set point at $t = 8$, a fluctuation occurs in the second output. It is interesting to note that this output fluctuation is decreased in the subsequent iterations. A similar phenomenon happens at $t = 25$ when the second output changes to a different set point. Moreover, a monotonic convergence of the system error is illustrated in Figure 6.29. On the other hand, Figure 6.30 shows the computational time to solve the LMI problem (3.19) of the distillation column with time-invariant parametric uncertainties. The computational time increases as more iterations are executed. Although the computational time in this case is large, it is acceptable since the distillation column has a slow dynamic.

6.2.2 Iteration-varying parametric uncertainty

We choose the Luyben-Vinante model of distillation column (6.12) to illustrate the effectiveness of the Algorithm 3. The distillation column is considered to work around the operating point in which $x_D = 0.98,\ x_B = 0.01$. The time delays and time constants of the four block transfer functions are assumed to have $\pm 15\%$ uncertainty, thus $q = 0.15$. Let the sample time be 0.5 minute, the number of samples is 101. Then, like in the previous part, the Markov matrix G of the system (6.12) is analysed to be an affine function of the uncertainty θ_k,

$$G(\theta) = G_0 + \sum_{i=1}^{8} \theta_{k,i} G_i \tag{6.14}$$

where G_0 represents the nominal system and G_i's, $i = \overline{1,8}$ represent the uncertain dynamic matrices. The matrices G_0, G_i's, $i = \overline{1,8}$ are the same with the case of time-invariant parametric uncertainties.

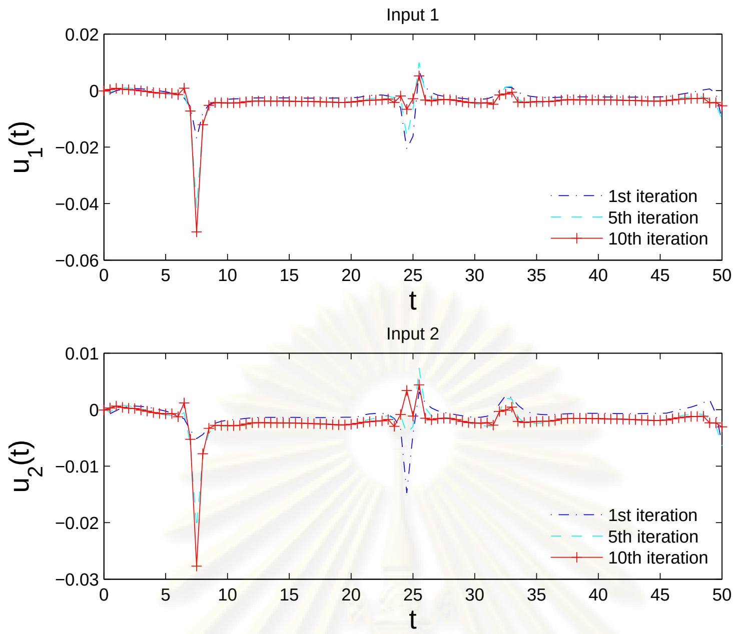

Figure 6.25: Control inputs of the distillation column system with time-invariant parametric uncertainty.

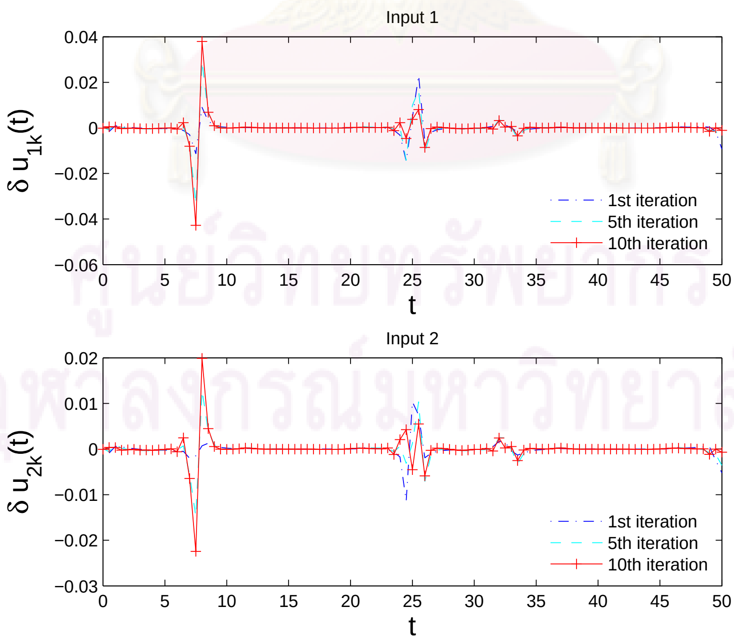

Figure 6.26: The difference of control inputs of the distillation column system with time-invariant parametric uncertainty w.r.t. time index.

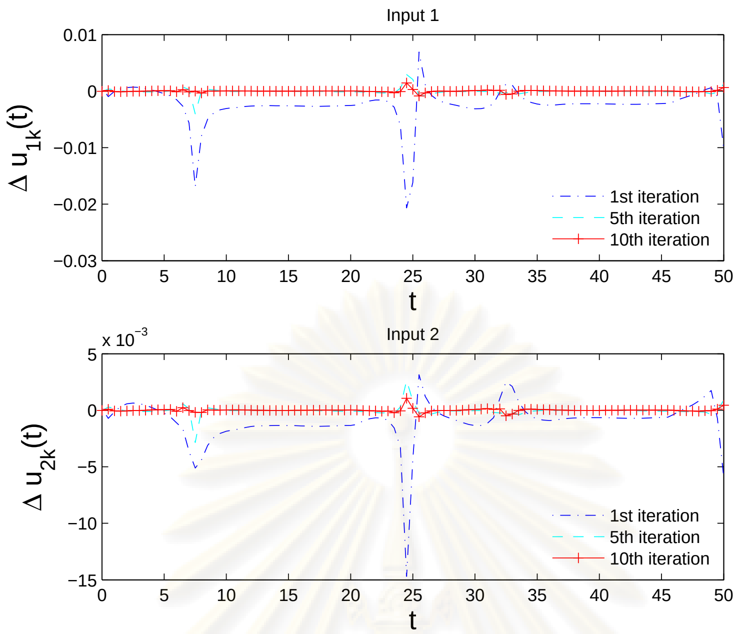

Figure 6.27: The difference of control inputs of the distillation column system with time-invariant parametric uncertainty w.r.t. iteration index.

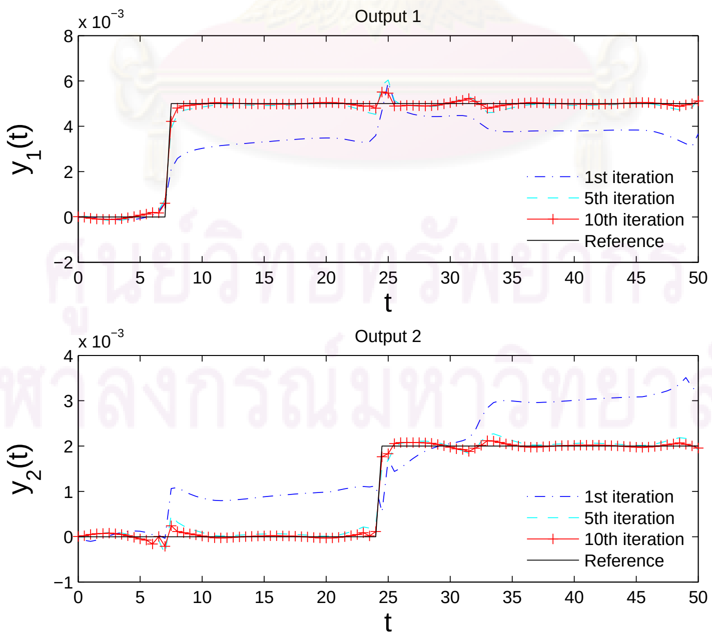

Figure 6.28: Output responses of the distillation column system with time-invariant parametric uncertainty.

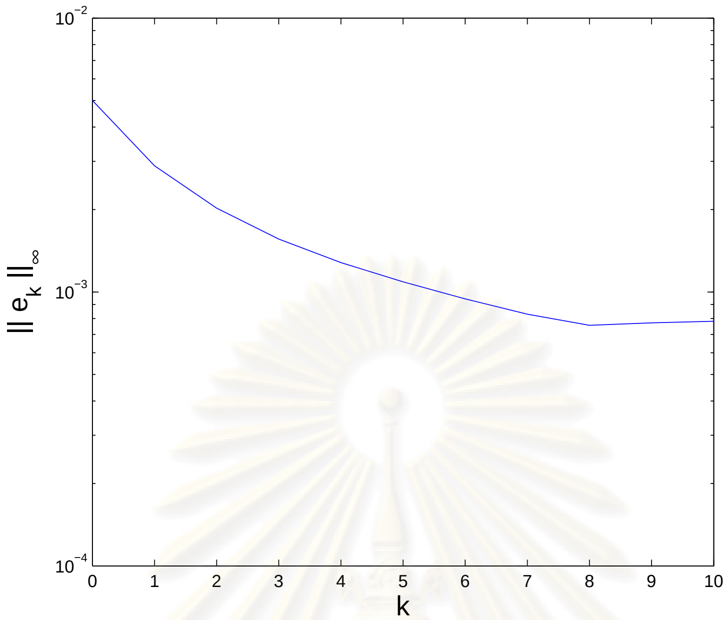

Figure 6.29: Infinity-norm of the error of the distillation column system with time-invariant parametric uncertainty.

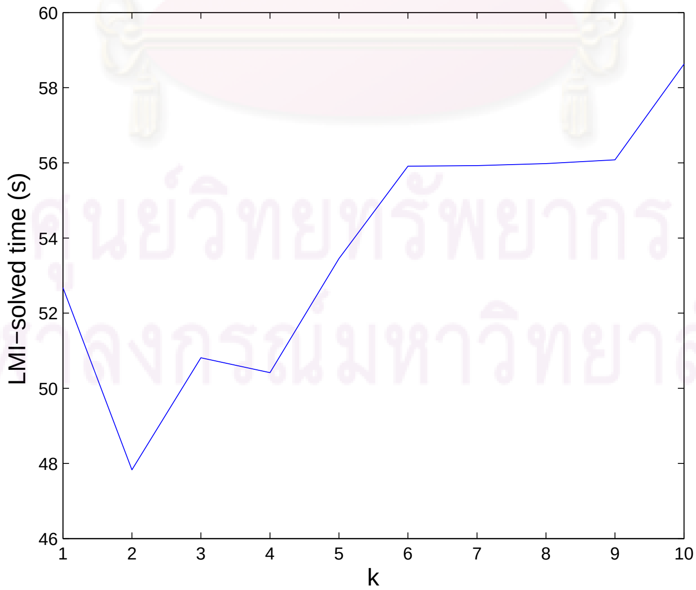

Figure 6.30: Computational time of the distillation column system with time-invariant parametric uncertainty.

The desired reference trajectory is

$$r(t) = \begin{cases} (0,0)^T, & t \in [0,8] \\ (0.005,0)^T, & t \in [9,25] \\ (0.005,0.002)^T, & t \in [26,50] \end{cases}$$

The constraints of control inputs are specified by

$$\mathbf{u}_l = -0.05, \mathbf{u}_h = 0.05, \delta\mathbf{u}_l = -3, \delta\mathbf{u}_h = 3, \Delta\mathbf{u}_l = -0.1, \Delta\mathbf{u}_h = 0.1. \tag{6.15}$$

The design parameters are chosen as follows: $Q = 0.085I_1, R = 0.012I_2$ where I_1, I_2 are identity matrices with appropriate dimension. In the simulation, the parameter θ is randomly selected from the uncertainty interval and varied each iteration. For the stopping criteria, we choose $\epsilon = 10^{-5}$ and `iter_max` $= 10$.

The simulation results are presented in Figures 6.31–6.35. Figures 6.31–6.33 show that the control inputs satisfy all the constraints C1–C3 specified by (6.15). In addition, the convergence of the control inputs is exhibited in Figure 6.33 where the control input update $\Delta\mathbf{u}_{k+1}$ comes to zero. On the other hand, Figure 6.34 illustrates that the outputs of the distillation column system track the step reference inputs but there are some abrupt changes of the output responses. They are due to the interaction between the inputs and outputs of the distillation column system. When the first output changes to a new set point at $t = 8$, a fluctuation occurs in the second output. It is worth noting that this output fluctuation is diminished in the next iterations. A similar phenomenon happens at $t = 25$ when the second output changes to a different set point. A monotonic convergence of the system error is demonstrated in Figure 6.35. Figure 6.36 shows the computational time to solve the LMI problem (5.21) of the distillation column with iteration-varying parametric uncertainties. The computational time increases as the number of iterations increases.

6.3 Conclusion

Chapter 6 proclaims the application of the proposed ILC algorithms to a flexible link and a distillation column. The flexible link is a SISO system whereas the distillation column in this section is a MIMO system. Section 6.1 shows the simulation results of flexible link in which, the mass of flexible link is considered as the uncertain parameter. Three cases of parametric uncertainty are considered, namely, time-invariant, time-varying and iteration-varying. In each case, we demonstrate how to experimentally analyse the impulse responses to derive an affine Markov model of flexible link using the proposed procedure, then, we apply the corresponding ILC algorithm to the flexible link. The computational time for each case of uncertainty is also investigated. On the other hand, in Section 6.2, we illustrate the application of proposed ILC algorithms to a distillation column where the time constants and time delays of the given transfer functions are considered to be time-invariant and iteration-varying uncertainties. The analysis of the impulse responses are introduced. Consequently,

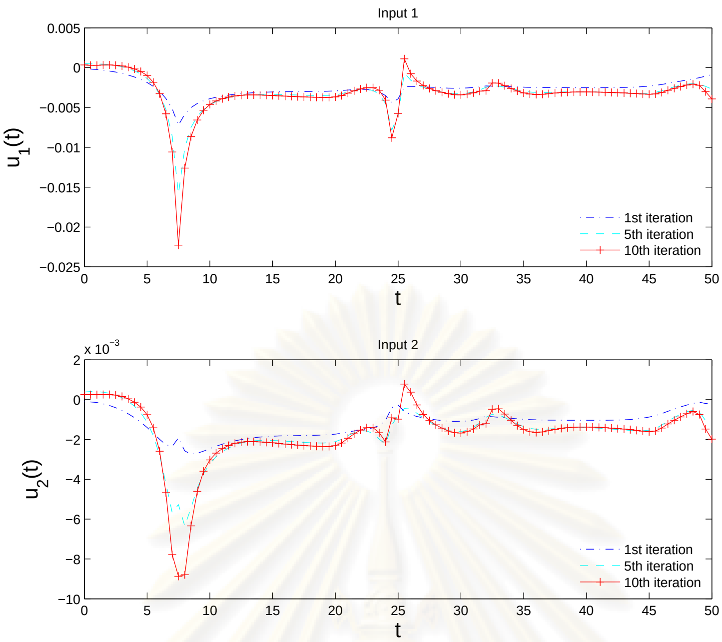

Figure 6.31: Control inputs of the distillation column system with iteration-varying parametric uncertainty.

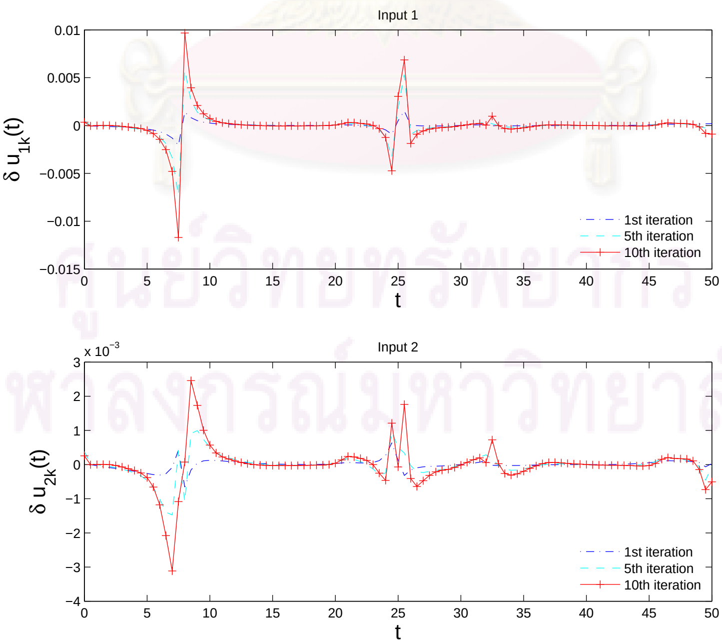

Figure 6.32: The difference of control inputs of the distillation column system with iteration-varying parametric uncertainty w.r.t. time index.

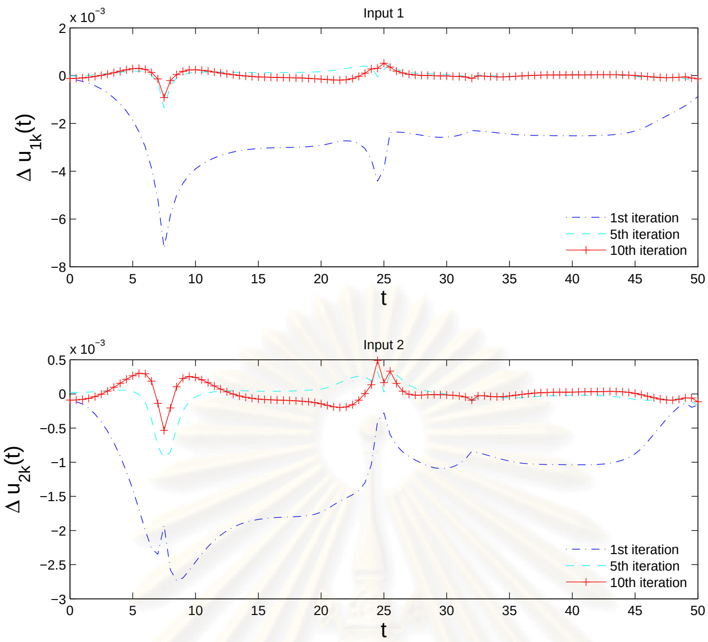

Figure 6.33: The difference of control inputs of the distillation column system with iteration-varying parametric uncertainty w.r.t. iteration index.

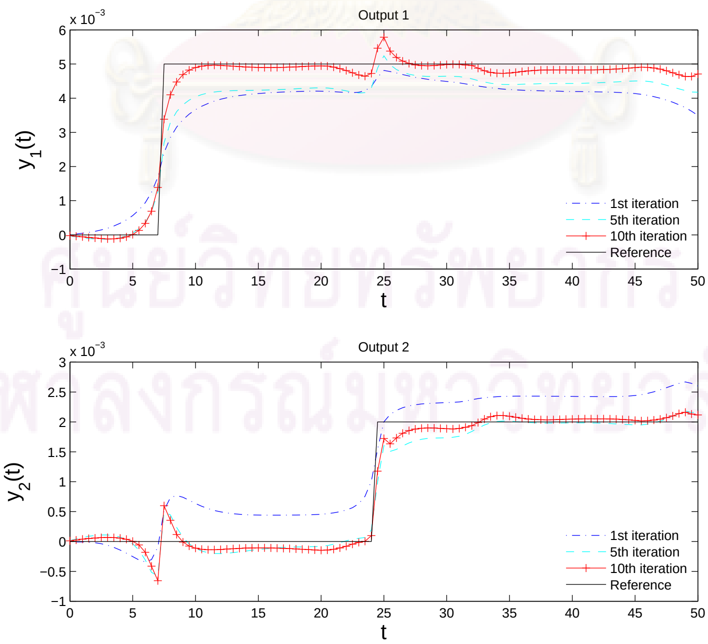

Figure 6.34: Output responses of the distillation column system with iteration-varying parametric uncertainty.

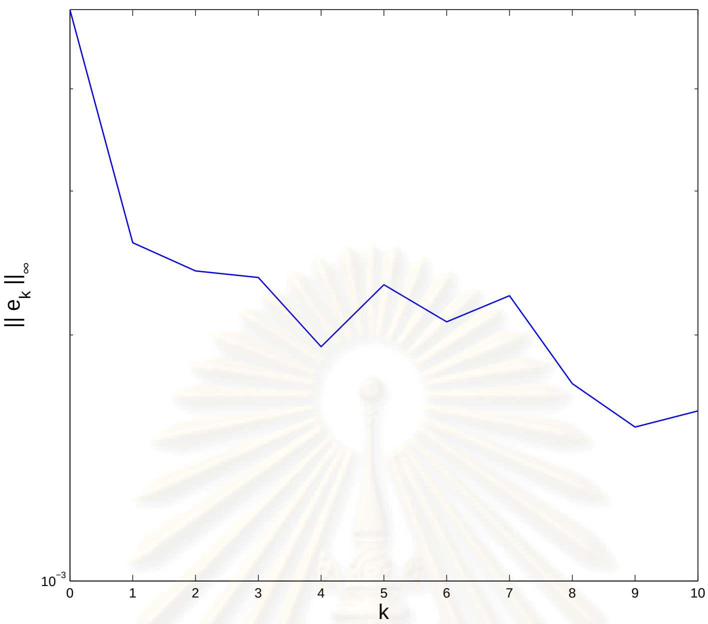

Figure 6.35: Infinity-norm of the error of the distillation column system with iteration-varying parametric uncertainty.

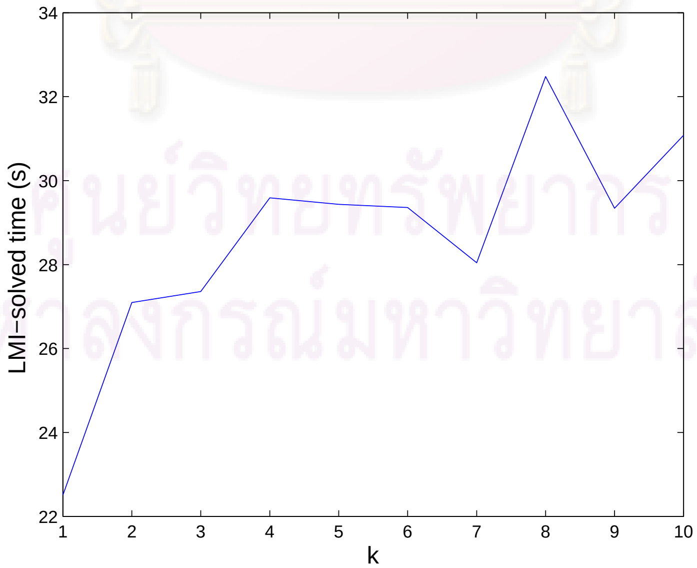

Figure 6.36: Computational time of the distillation column system with iteration-varying parametric uncertainty.

the simulation results and the computational time are displayed. The simulation results for both flexible link and distillation column reveal that the proposed ILC algorithm work quite well.

CHAPTER VII

CONCLUSIONS AND FUTURE WORKS

7.1 Conclusions

This thesis has proposed three new Iterative Learning Control algorithms for linear systems in the presence of parametric uncertainties. A unified, systematic and applicable methodology has been introduced to design those robust algorithms. Then, the effectiveness of the proposed ILCs has been demonstrated through some generic examples as well as applications to physical models. Next is the summary of the main points in each chapter to highlight the contents presented in the thesis.

Chapter 1 briefly introduces ILC, its definition, its key idea and the comparison between ILC with conventional feedback control. Next, the literature review is given to cover an overview of ILC as well as the state of the art of robust ILC design techniques. Afterward, we present the scope of our thesis and our contribution.

In Chapter 2, the first three sections are devoted to the mathematical tools such as convex optimization, Lagrangian duality, linear matrix inequalities which are used in the development of our algorithms. The remaining section includes a formulation of the robust ILC design problem, the system description and modeling, and the methodology to determine the solution.

Consequently, Chapter 3 presents the detail steps in the design of a robust ILC algorithm for linear systems with time-invariant parametric uncertainties. An upper bound of the worst-case performance is derived, then, the min-max problem is relaxed to a minimization problem. Employing Lagrangian duality, the dual problem of this minimization one is considered which then c an be reformulated as a convex optimization problem with LMI constraints utilizing Schur complement. The convergence of the control input and the system error is proved. Finally, a generic example is provided to demonstrate the effectiveness of the proposed algorithm.

Chapter 4 and Chapter 5 propose other two robust ILC algorithms for linear systems in the presence of time-varying and iteration-varying parametric uncertainties relatively. Like in Chapter 3, the design procedures in these chapters are based on the general methodology presented in Chapter 2. Due to the difference on the type of parametric uncertainties, the details of the system descriptions as well as the solutions of the optimization problems are not the same. However, the main steps in the designs are similar as described in the methodology. In addition, the convergence of the control inputs and the system errors is also proved in each chapter. , we introduce one generic example in each chapter to show the efficiency of the designed algorithms.

Chapter 6 presents the simulation results of some physical plants such as flexible link and distillation column controlled by the proposed ILC algorithms. Given the linearized models of the plants, we first assume that there are some parametric uncertainties in these models, then, we discretize the continuous models and analyse the impulse responses of the discretized systems to obtain the Markov matrices as affine functions of the uncertainties. Then, these affine Markov matrices are incorporated in the robust ILC algorithms. On the other hand, the parametric uncertainties are changed to be time-invariant, time-varying or iteration-varying to verify the corresponding ILC algorithms. The simulations exhibit acceptable results which are reliable resources for real implementation in the future.

7.2 Future works

As remarked in the designs of the three ILC algorithms, there are some conservatism or limitations in our thesis. Therefore, they could be the issues for the future improvements.

First of all, the system errors are proved to converge, but not monotonically. It is noticed that the monotonic convergence is an advance to ensure the ILC algorithms converge faster and the system error decays [5], [34]. Thus, the subsequent work should aim to derive a monotonic convergence of the system error.

Next, the design of robust ILC algorithms in this thesis is based on the assumption of the affine Markov matrix, hence, the design applications might be limited. The future work should focus on a larger class of linear systems and perhaps nonlinear systems.

In this thesis, we consider the uncertainty with bounded magnitude, but the rate of change of the uncertainty is not employed. In numerous practical systems, the bounds of rate of change of uncertainty are sometimes available [46], [47]. Hence, in the future research, this information of uncertainty should be incorporated in the controller design, so that the robust controller can handle with the uncertainty better.

The proposed algorithms seem to work well with the input-output delay as demonstrated in the numerical examples, but in our problem formulation, the time delay has not been explicitly considered. Actually, the advantage of ILC lies in the utilization of the system information of previous iterations, so, ILC algorithm can handle the delay in the system [48], [49]. Therefore, the future work should consider the time delay in the system such as input-delay and state-delay, and design the corresponding ILC.

REFERENCES

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

[2] H. S. Ahn, K. L. Moore, and Y. Q. Chen. *Iterative learning control: Robustness and monotonic convergence for interval systems*. Springer. 2007.

[3] M. Verwoerd. *Iterative Learning Control: A critical review*. PhD thesis. Dutch Institute of Systems and Control, Netherlands. 2005.

[4] D. A. Bristow, M. Tharayil, and A. G. Alleyne. A survey of iterative learning control: A learning-based method for high performance tracking control. *IEEE Control System Magazine*. 26. (2006): 96–114.

[5] H. S. Ahn, Y. Q. Chen, and K. L. Moore. Iterative learning control: Brief survey and categorization. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*. 37. (2007): 1099–1121.

[6] J. A. Frueh and M. Q. Phan. Linear quadratic optimal learning control. *International Journal of Control*. 73. (2000): 832–839.

[7] N. Amann, D. H. Owen, and E. Rogers. Iterative learning control using optimal feedback and feedforward actions. *International Journal of Control*. 65. (1996): 277–293.

[8] N. Amann, D. H. Owen, and E. Rogers. Predictive optimal iterative learning control. *International Journal of Control*. 69. (1998): 203–226.

[9] J. Hätönen. *Issues of Algebra and Optimality in Iterative Learning Control*. PhD thesis. University of Oulu, Finland. 2004.

[10] K. L. Moore, Y. Q. Chen, and H. S. Ahn. Iterative learning controller-Algebraic analysis and optimal design. in *IEEE International Conference on Mechatronics and Automation Tutorial Workshop*.

[11] S. Gunnarsson and M. Norrlöf. On the design of ILC algorithms using optimization. *Automatica*. 37. (2001): 2011–2016.

[12] D. H. Owens and J. Hätönen. Iterative learning control-An optimization paradigm. *Annual Reviews in Control*. 29. (2005): 57–70.

[13] J. Hätönen, D. H. Owens, and K. Feng. Basis functions and parameter optimisation in high-order iterative learning control. *Automatica*. 42. (2006): 287–294.

[14] A. Tayebi, S. Abdul, and M. Zaremba. Robust iterative learning control design via μ-synthesis. in *Proc. 2005 IEEE Conference on Control Applications.* (2005): 416–421.

[15] D. de Roover. Synthesis of a robust iterative learning controller using an H_∞ approach. in *Proc. IEEE Conf. on Decision and Control.* (1996): 3044–3049.

[16] K. L. Moore, Y. Q. Chen, and H. S. Ahn. Algebraic H_∞ design of higher order iterative learning controllers. in *Proc. 2005 IEEE International Symposium on Intelligent Control.* (2005): 1213–1218.

[17] J. Xu, M. Sun, and L. Yu. LMI-based robust iterative learning controller design for discrete linear uncertain systems. *Journal of Control Theory and Applications.* 3. (2005): 259–265.

[18] J. H. Moon, T. Y. Doh, and M. J. Chung. A Robust approach to iterative learning control design for uncertain systems. *Automatica.* 34. (1998): 1001–1004.

[19] J. Shi, F. Gao, and T. J. Wu. Robust design of integrated feedback and iterative learning control of a batch process based on a 2D roesser system. *Journal of Process Control.* 15. (2005): 907–924.

[20] A. Tayebi and J. Xu. Observer-based iterative learning control for a class of time-varying nonlinear systems. *IEEE Transaction on Circuits and Systems.* 50. (2003): 452–455.

[21] C.-J. Chien. A discrete iterative learning control for a class of nonlinear time-varying systems. *IEEE Transaction on Automatic Control.* 43. (1998): 748–752.

[22] R. Chi, S. Sui, and Z. Hou. A new discrete-time adaptive ILC for nonlinear systems with time-varying parametric uncertainties. *Acta Automatica Sinica.* 34. (2008): 805–808.

[23] M. Arif, T. Ishihara, and H. Inooka. Iterative learning control utilizing the error prediction method. *Journal of Intelligent and Robotic Systems.* 25. (1999): 95–108.

[24] K. Tan, S. Huang, T. Lee, and S. Lim. A discrete-time iterative learning algorithm for linear time-varying systems. *Engineering Applications of Artificial Intelligence.* 16. (2003): 185–190.

[25] L. Hladowski, K. Galkowski, E. Rogers, P. L. Lewin, and C. T. Freeman. A new iterative learning control scheme for linear time-varying discrete systems. in *Proc. 3rd IFAC Workshop on Periodic Control Systems.*

[26] H. S. Ahn, K. L. Moore, and Y. Q. Chen. Monotonic convergent iterative learning controller design with iteration varying model uncertainty. in *Proc. IEEE International Conference on Mechatronics and Automation.* (2005): 572–577.

[27] H. S. Ahn, K. L. Moore, and Y. Q. Chen. Monotonic convergent iterative learning controller design based on interval model conversion. in *Proc. 2005 IEEE International Symposium on Intelligent Control.* (2005): 1207–1212.

[28] H. S. Ahn, K. L. Moore, and Y. Q. Chen. Stability analysis of discrete-time iterative learning control systems with interval uncertainty. *Automatica.* 43. (2007): 892–902.

[29] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge University Press. 2004.

[30] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalties in System and Control.* SIAM. 1994.

[31] C. Scherer and S. Weiland. *Linear matrix inequalties in Control.* Delft University of Technology, Eindhoven University of Technology, The Netherlands. 2005. [Online] Available at http://w3.ele.tue.nl/nl/cs/education/courses/DISClmi.

[32] U. T. Jonsson. *A Lecture on the S-Procedure.* Royal Institute of Technology, Sweden. 2006. [Online] Available at http://www.math.kth.se/∼ulfj/5B5746.

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

[34] N. Amann, D. H. Owen, and E. Rogers. Iterative learning control for discrete-time system with exponential rate of convergence. *IEE Proc.: Control Theory and Applications.* 143. (1996): 217–224.

[35] D. H. Nguyen and D. Banjerdpongchai. An LMI approach for robust iterative learning control with quadratic performance criterion. in *Proc. 10th International Conference on Control, Automation, Robotics and Vision.* (2008): 1805–1810.

[36] D. H. Nguyen and D. Banjerdpongchai. An LMI approach for robust iterative learning control with quadratic performance criterion. *Journal of Process Control.* 19. (2009): 1054–1060.

[37] D. H. Nguyen and D. Banjerdpongchai. Robust iterative learning control for linear systems with time-invariant parametric uncertainties. in *Proc. ICROS-SICE International Joint Conference.* (2009): 4178–4183.

[38] M. Grant, S. Boyd, and Y. Ye. *cvx: Matlab software for disciplined convex programming.* Stanford University. 2008. [Online] Available at http://www.stanford.edu/∼boyd/cvx.

[39] D. H. Nguyen and D. Banjerdpongchai. Robust iterative learning control for linear systems with time-varying parametric uncertainties. in *Proc. 48th Control Conference on Decision and Control 2009 (Accepted)*.

[40] D. H. Nguyen and D. Banjerdpongchai. Robust iterative learning control for linear systems with iteration-varying parametric uncertainties. in *Proc. 7th Asian Control Conference*. (2009): 716–721.

[41] P. Chaichanavong and D. Banjerdpongchai. A Case Study of Robust Control Experiment on One-link Flexible Robot Arm. in *Proc. 38th Conference on Decision and Control*. (1999): 4319–4324.

[42] Quanser Inc.. *Flexible Link: User Manual*. 2004.

[43] D. H. Nguyen and D. Banjerdpongchai. Robust iterative learning control for flexible link under parametric uncertainty. in *Proc. 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications, and Information Technology*. (2009): 376–379.

[44] D. J. Inman. *Engineering Vibration*. Prentice Hall. 2001.

[45] W. L. Luyben and C. D. Vinante. Experimental studies of distillation decoupling. *Kem. Teollisuus*. 29. (1972): 499–514.

[46] P. Park and S. C. Jeong. Constrained RHC for LPV systems with bounded rates of parameter variations. *Automatica*. 40. (2004): 865–872.

[47] R. C. Oliveira and P. L. Peres. Time-varying discrete-time linear systems with bounded rates of variation: Stability analysis and control design. *Automatica*. DOI:10.1016/j.automatica.2009.07.015.

[48] X.-D. Li, T. W. S. Chow, and J. K. L. Ho. 2-D system theory based iterative learning control for linear continuous systems with time delays. *IEEE Transactions on Circuits and SystemsI: Regular Papers*. 52. (2005): 1421–1430.

[49] D. Meng, Y. Jia, J. Du, and F. Yu. Robust design of a class of time-delay iterative learning control systems with initial shifts. *IEEE Transactions on Circuits and SystemsI: Regular Papers*. 56. (2009): 1744–1757.

Appendix

Source Code of ILC algorithms Implemented in MATLAB

In this appendix, we will provide some source files of the simulations presented in the thesis. The following table give the description of the source files.

File name	Description
AffineModel.m	System modeling to obtain an affine Markov model of uncertainty
FigureDisplay.m	Plot the system responses after running the simulation
FlexibleLink_TI.m	Simulation of flexible link with a time-invariant parametric uncertainty using ILC-TI Algorithm 1
FlexibleLink_TV.m	Simulation of flexible link with a time-varying parametric uncertainty using proposed ILC-TV Algorithm 2
FlexibleLink_IV.m	Simulation of flexible link with an iteration-varying parametric uncertainty using proposed ILC-IV Algorithm 3
Initialization.m	Initializing the system
plantFEM.m	System modeling to obtain a high order model of flexible link using Finite Element Method

AffineModel.m

```
%==========================================================================
%System modeling to obtain the affine Markov model of uncertainty

%Inputs:

%Outputs: G0, G1: Matrices of Affine Markov model
%==========================================================================
%--------------------------------------------------------------------------
%Author |   Nguyen Dinh Hoa
%Date    |   13/7/2009
%Work   |   Control System Research Laboratory, Chulalongkorn University
%--------------------------------------------------------------------------

N=101; %number of samples
theta_l=-.2; theta_u=.2;
theta=(theta_u-theta_l)*rand+theta_l;
h=.01;
m=round((theta_u-theta_l)/h);

%Flexible link parameters
FlexibleLink_para;

% State-space model
A( 1, 1 ) = 0; A( 1, 2 ) = 0; A( 1, 3 ) = 1; A( 1, 4 ) = 0;
A( 2, 1 ) = 0; A( 2, 2 ) = 0; A( 2, 3 ) = 0; A( 2, 4 ) = 1;
A( 3, 1 ) = 0; A( 3, 2 ) = K_Stiff/Jeq; A( 3, 3 ) = -(Eff_M*Eff_G*Kt*Kg^2*Km+Beq*Rm)/Jeq/Rm; A( 3, 4 ) = 0;
A( 4, 1 ) = 0; A( 4, 2 ) = -K_Stiff*(Jeq+Jarm)/Jeq/Jarm; A( 4, 3 ) = (Eff_M*Eff_G*Kt*Kg^2*Km+Beq*Rm)/Jeq/Rm; A( 4, 4 ) = 0

B( 1, 1 ) = 0; B( 2, 1 ) = 0; B( 3, 1 ) = Eff_M*Eff_G*Kt*Kg/Jeq/Rm; B( 4, 1 ) = -Eff_M*Eff_G*Kt*Kg/Jeq/Rm;

C( 1, 1 ) = 1; C( 1, 2 ) = 0; C( 1, 3 ) = 0; C( 1, 4 ) = 0;

D = 0;

Jarm = Link_M_0 * Link_L ^ 2 / 3;
K_Stiff = Wc^2 * Jarm;
A( 3, 2 ) = K_Stiff/Jeq; A( 4, 2 ) = -K_Stiff*(Jeq+Jarm)/Jeq/Jarm;
sys=c2d(ss(A,B,C,D),.012);
G0=zeros(N);
for i=1:N
    for k=1:i
        G0(i,k)=sys.c*sys.a^(i-k)*sys.b;
    end
end

t=0:N;
y0=impulse(sys,.012*N);
stairs(t,y0,'r','LineWidth',2);hold on;
y=zeros(m+1,N+1);
for j=0:m
    theta=theta_l+j*h;
    Link_M = Link_M_0*(1+theta);
    Jarm = Link_M * Link_L ^ 2 / 3;
    K_Stiff = Wc^2 * Jarm;
    A( 3, 2 ) = K_Stiff/Jeq;
    A( 4, 2 ) = -K_Stiff*(Jeq+Jarm)/Jeq/Jarm;
    sysm=c2d(ss(A,B,C,D),.012);
```

```
    y(j+1,:)=impulse(sysm,.012*N);
end

upper=max(y); lower=min(y);
stairs(t,upper,'--k','LineWidth',2); hold on;
stairs(t,lower,'-.k','LineWidth',2); hold off;
h=legend('Nominal system','Upper bound','Lower bound');
xlabel('Samples','FontSize',12);
ylabel('Impulse responses','FontSize',12);

upper_bound=max(y(:,2:N+1));
lower_bound=min(y(:,2:N+1));

G_l=zeros(N); G_u=zeros(N);
G_u(:,1)=upper_bound(1:N)';
G_l(:,1)=lower_bound(1:N)';
for i=2:N
    G_u(:,i)=[zeros(i-1,1);upper_bound(1:N-i+1)'];
    G_l(:,i)=[zeros(i-1,1);lower_bound(1:N-i+1)'];
end
G1=(G_u-G_l)/2;

save G0.mat G0
save G1.mat G1

%*************************************************************************
%End of program
%*************************************************************************
```

FigureDisplay.m

```
%==========================================================================================
%Plot the results after running the simulation

%Inputs:    indis:      Control input in all iterations
%           outdis:     System output in all iterations
%           TimeIndex:  Control input rate w.r.t. time index in all iterations
%           IterIndex:  Control input rate w.r.t. iteration index in all iterations
%           e:          Infinity norm of system error in all iterations
%           ee:         Q-norm of system error of all iterations
%           theta:      The uncertainty in all iterations

%Outputs:   Figures of Control input, System output, Control input rate w.r.t. time index
%           Control input rate w.r.t. iteration index, Infinity norm of system error,
%           Q-norm of system error in all iterations.
%===========================================================================================
%-------------------------------------------------------------------------
%Author |   Nguyen Dinh Hoa
%Date   |   25/2/2009
%Work   |   Control System Research Laboratory, Chulalongkorn University
%-------------------------------------------------------------------------
```

```
%Plot the output at iteration 1, 2, 5 & last
figure
%subplot(2,1,1);
plot((0:N-1)',outdis(:,1),(0:N-1)',outdis(:,2),'--',(0:N-1)',outdis(:,6),':',(0:N-1)',outdis(:,16),'-.',(0:N-1)',
outdis(:,31),'+-',(0:N-1)',ref,'k');
h=legend('Initial state','1st iteration','5th iteration','15th iteration','30th iteration','Reference');
set(h,'Box','off');
xlabel('t','FontSize',16);
ylabel('y_k(t)','FontSize',16);

%Plot the input at iteration 1, 2, 5 & last
figure
%subplot(2,1,2);
plot((0:N-1)',indis(:,1),(0:N-1)',indis(:,2),'--',(0:N-1)',indis(:,6),':',(0:N-1)',indis(:,16),'-.',(0:N-1)',
indis(:,31),'+-');
h=legend('Initial state','1st iteration','5th iteration','15th iteration','30th iteration');
set(h,'Box','off');
xlabel('t','FontSize',16);
ylabel('u_k(t)','FontSize',16);

%Plot the constraint of the input w.r.t. time index
figure
plot((0:N-1)',TimeIndex(:,1),(0:N-1)',TimeIndex(:,2),'--',(0:N-1)',TimeIndex(:,6),':',(0:N-1)',TimeIndex(:,16),'-.',
(0:N-1)',TimeIndex(:,31),'+-');
h=legend('Initial state','1st iteration','5th iteration','15th iteration','30th iteration');
set(h,'Box','off');
xlabel('t','FontSize',16);
ylabel('\delta u_k(t)','FontSize',16);

%Plot the constraint of the input w.r.t. iteration index
figure
plot((0:N-1)',IterIndex(:,1),(0:N-1)',IterIndex(:,2),'--',(0:N-1)',IterIndex(:,6),':',(0:N-1)',IterIndex(:,16),'-.',
(0:N-1)',IterIndex(:,31),'+-');
h=legend('Initial state','1st iteration','5th iteration','15th iteration','30th iteration');
set(h,'Box','off');
xlabel('t','FontSize',16);
ylabel('\Delta u_k(t)','FontSize',16);

%Plot the Infinity-norm of error versus iterative process
figure
semilogy(0:k,e(1:k+1));
xlabel('k','FontSize',16);
ylabel('|| e_k ||_{\infty}','FontSize',16);

%Plot the Q-norm of error versus iterative process
figure
semilogy(0:k,ee(1:k+1));
xlabel('k','FontSize',16);
ylabel('|| e_k ||_{Q}','FontSize',16);

%**************************************************************************
%End of program
%**************************************************************************
```

FlexibleLink_TI.m

```
%================================================================================
%Design Robust ILC algorithm for Flexible Link robot with time-invariant uncertainty

%Inputs:    plantFEM.m      Higher model of flexible link using FEM
%           G0, G1:         Matrices of Affine Markov model

%Outputs:   indis:          Control input in all iterations
%           outdis:         System output in all iterations
%           TimeIndex:      Control input rate w.r.t. time index in all iterations
%           IterIndex:      Control input rate w.r.t. iteration index in all iterations
%           e:              Infinity norm of system error in all iterations
%           ee:             Q-norm of system error of all iterations
%           theta:          The uncertainty in all iterations
%================================================================================
%-------------------------------------------------------------------------
%Author |   Nguyen Dinh Hoa
%Date   |   25/2/2009
%Work   |   Control System Research Laboratory, Chulalongkorn University
%-------------------------------------------------------------------------
clear;clc

load G0.mat
load G1.mat

theta1=-.2; theta2=.2;
G1=.2*G1;

theta=(theta2-theta1)*rand+theta1;

% Length of the Link
Link_L = 0.0254 * 15;
% Mass of the Link
Link_M = (1+theta)*0.065/.8;

%Pertubed model of system using FEM method
[A,B,C,D]=plantFEM(3,Link_L,Link_M);

sys=c2d(ss(A,B,C,D),.012);
G=zeros(N);
for i=1:N
    for k=1:i
        G(i,k)=sys.c*sys.a^(i-k)*sys.b;
    end
end

%*************************************************************************
%Initialize the system
%*************************************************************************

Initialization;
%*************************************************************************
%Iterative process
%*************************************************************************
iter_max=30; %Maximum number of iteration
time=zeros(iter_num,1); %Computational time

for k=1:iter_max
```

```
        %Current iteration
        %------------------------------------------------------------------
        %Step 1: solve SDP problems
        l=[-max(u_low-u,de_u_low);min(u_high-u,de_u_high);-du_low+J*u;du_high-J*u];

        tic
        cvx_begin sdp
            variables rho z12
            variable ld(4*N)
            minimize (rho)
            [(G0'*Q*G1+G1'*Q*G0)*z12+G1'*Q*G1+G0'*Q*G0+R,F'*ld-2*(G1'*Q*error*z12+G0'*Q*error);
            (F'*ld-2*(G1'*Q*error*z12+G0'*Q*error))',-4*ld'*l+rho] >= 0;
            [1 z12;z12 1] >= 0;
            ld >= 0;
        cvx_end
        toc

        delta_u=-1/2*inv((G0'*Q*G1+G1'*Q*G0)*z12+G1'*Q*G1+G0'*Q*G0+R)*(F'*ld-2*(G1'*Q*error*z12+G0'*Q*error));

        %Calculate the derivative of the input w.r.t iteration index
        IterIndex(:,k+1)=delta_u;
        %------------------------------------------------------------------
        %Next iteration
        u=delta_u+u;
        indis(:,k+1)=u;
        y=G*u;
        y=[0;y(1:N-1)];
        outdis(:,k+1)=y;
        error=ref-y;
        e(k+1)=norm(error,inf);
        ee(k+1)=sqrt(error'*Q*error);
        %Calculate the derivative of the input w.r.t time index
        TimeIndex(1,k+1)=u(1);
        for i=2:N
            TimeIndex(i,k+1)=u(i)-u(i-1);
        end
    if norm(error,inf) <= epsilon
        break;
    end
end

%***************************************************************************
%Save the results
%***************************************************************************
save FL_input.mat indis
save FL_output.mat outdis
save FL_TimeConstraint.mat TimeIndex
save FL_IterConstraint.mat IterIndex
save FL_H_Error.mat e
save FL_Q_Error.mat ee
savefile= 'FL_uncertainty.mat';
save(savefile, 'theta');
savefile='computational_time.mat';
save(savefile,'time');

%***************************************************************************
%Display the results
%***************************************************************************
```

```
disp('Number of iteration is:');
disp(k);

FigureDisplay

%***************************************************************************
%End of program
%***************************************************************************
```

FlexibleLink_TV.m

```
%=========================================================================================
%Design Robust ILC algorithm for Flexible Link robot with time-varying uncertainty

%Inputs:    plantFEM.m      Higher model of flexible link using FEM
%           G0, G1:         Matrices of Affine Markov model

%Outputs:   indis:          Control input in all iterations
%           outdis:         System output in all iterations
%           TimeIndex:      Control input rate w.r.t. time index in all iterations
%           IterIndex:      Control input rate w.r.t. iteration index in all iterations
%           e:              Infinity norm of system error in all iterations
%           ee:             Q-norm of system error of all iterations
%           theta:          The uncertainty in all iterations
%=========================================================================================
%-------------------------------------------------------------------------
%Author |   Nguyen Dinh Hoa
%Date    |   25/2/2009
%Work   |   Control System Research Laboratory, Chulalongkorn University
%-------------------------------------------------------------------------

clear;clc;

load G0.mat
load G1.mat

theta_l=-.2; theta_u=.2; % size of uncertainty
G1=.2*G1;

% Length of the Link
Link_L = 0.0254 * 15;
% Nominal mass of the link
Link_M_0 = 0.065/.8;
% The first uncertain mass of the Link
theta1=(theta_u-theta_l)*rand+theta_l;
Link_M_1 = (1+theta1)*Link_M_0;
% The second uncertain mass of the Link
theta2=(theta_u-theta_l)*rand+theta_l;
Link_M_2 = (1+theta2)*Link_M_0;

%Time-varying model of flexible link using FEM method
[A1,B1,C1,D1]=plantFEM(3,Link_L,Link_M_1);
sys1=c2d(ss(A1,B1,C1,D1),.012);

[A2,B2,C2,D2]=plantFEM(3,Link_L,Link_M_2);
sys2=c2d(ss(A2,B2,C2,D2),.012);

yy=zeros(N,1);
for i=0:40
    yy(i+1)=sys1.c*sys1.a^i*sys1.b;
end
for i=1:60
    yy(i+41)=sys2.c*sys2.a^i*sys1.a^40*sys2.b;
end

G=zeros(N);
for i=1:N
    for k=1:i
```

```
        G(i,k)=yy(i-k+1);
    end
end

tt1=zeros(50,1); tt1(1)=theta1;
tt2=zeros(50,1); tt2(1)=theta2;

%***************************************************************************
%Initialize the system
%***************************************************************************

Initialization;

A=2*G1'*Q*G1+2*G0'*Q*G0+R;
%***************************************************************************
%Iterative process
%***************************************************************************
iter_max=30; %Maximum number of iteration
time=zeros(iter_num,1); %Computational time

for k=1:iter_max
        %Current iteration
        %-------------------------------------------------------------------
        %Step 1: solve SDP problems
        b=-4*G0'*Q*error;
        l=[-max(u_low-u,de_u_low);min(u_high-u,de_u_high);-du_low+J*u;du_high-J*u];

        tic
        cvx_begin sdp
            variable gm
            variable ld(4*N)
            minimize (gm)
            [A,b+F'*ld;(b+F'*ld)',-4*ld'*l+gm] >= 0;
            ld >= 0;
        cvx_end
        time(k)=toc;

        %Step 2: calculate the value of delta_u(k+1)
        delta_u=-1/2*inv(A)*(b+F'*ld);
        %Calculate the derivative of the input w.r.t iteration index
        IterIndex(:,k+1)=delta_u;
        %-------------------------------------------------------------------
        %Next iteration
        theta_next1=(theta_u-theta_l)*rand+theta_l;
        while (theta_next1-theta1 > 0.04)||(theta_next1-theta1 < -0.04)
            theta_next1=(theta_u-theta_l)*rand+theta_l;
        end
        theta_next2=(theta_u-theta_l)*rand+theta_l;
        while (theta_next2-theta2 > 0.04)||(theta_next2-theta2 < -0.04)
            theta_next2=(theta_u-theta_l)*rand+theta_l;
        end
        % Length of the Link
        Link_L = 0.0254 * 15;
        % Mass 1 of the Link
        Link_M_1 = (1+theta_next1)*0.065/.8;
        % Mass 2 of the Link
        Link_M_2 = (1+theta_next2)*0.065/.8;
```

```
        %Pertubed model of system using FEM method
        [A1,B1,C1,D1]=plantFEM(3,Link_L,Link_M_1);
        sys1=c2d(ss(A1,B1,C1,D1),.012);

        [A2,B2,C2,D2]=plantFEM(3,Link_L,Link_M_2);
        sys2=c2d(ss(A2,B2,C2,D2),.012);

        for i=0:40
            yy(i+1)=sys1.c*sys1.a^i*sys1.b;
        end
        for i=1:60
            yy(i+41)=sys2.c*sys2.a^i*sys1.a^40*sys2.b;
        end

        G=zeros(N);
        for i=1:N
            for l=1:i
                G(i,l)=yy(i-l+1);
            end
        end

        u=delta_u+u;
        indis(:,k+1)=u;
        y=G*u;
        y=[0;y(1:N-1)];
        outdis(:,k+1)=y;
        error=ref-y;
        e(k+1)=norm(error,inf);
        ee(k+1)=sqrt(error'*Q*error);

        theta1=theta_next1;
        tt1(k+1)=theta1;
        theta2=theta_next2;
        tt2(k+1)=theta2;

        %Calculate the derivative of the input w.r.t time index
        TimeIndex(1,k+1)=u(1);
        for i=2:N
            TimeIndex(i,k+1)=u(i)-u(i-1);
        end
    if norm(error,inf) <= epsilon
        break;
    end
end

%***************************************************************************
%Save the results
%***************************************************************************
save FL_input1.mat indis
save FL_output1.mat outdis
save FL_TimeConstraint1.mat TimeIndex
save FL_IterConstraint1.mat IterIndex
save FL_H_Error1.mat e
save FL_Q_Error1.mat ee
savefile= 'FL_uncertainty11.mat';
save(savefile, 'tt1');
savefile= 'FL_uncertainty12.mat';
save(savefile, 'tt2');
```

```
savefile='computational_time.mat';
save(savefile,'time');

%**************************************************************************
%Display the results
%**************************************************************************
disp('Number of iteration is:'); disp(k);

FigureDisplay
%**************************************************************************
%End of program
%**************************************************************************
```

FlexibleLink_IV.m

```
%=====================================================================================
%Design Robust ILC algorithm for Flexible Link robot with iteration-varying uncertainty

%Inputs:    plantFEM.m      Higher model of flexible link using FEM
%           G0, G1:         Matrices of Affine Markov model

%Outputs:   indis:          Control input in all iterations
%           outdis:         System output in all iterations
%           TimeIndex:      Control input rate w.r.t. time index in all iterations
%           IterIndex:      Control input rate w.r.t. iteration index in all iterations
%           e:              Infinity norm of system error in all iterations
%           ee:             Q-norm of system error of all iterations
%           theta:          The uncertainty in all iterations
%===========================================================================
%--------------------------------------------------------------------------
%Author |   Nguyen Dinh Hoa
%Date    |   25/2/2009
%Work   |   Control System Research Laboratory, Chulalongkorn University
%--------------------------------------------------------------------------

clear;clc

load G0.mat
load G1.mat

theta1=-.2; theta2=.2;
G1=.2*G1;

theta=(theta2-theta1)*rand+theta1;
tt=zeros(50,1); tt(1)=theta;

% Length of the Link
Link_L = 0.0254 * 15;
% Mass of the Link
Link_M = (1+theta)*0.065/.8;

%Pertubed model of system using FEM method
[A,B,C,D]=plantFEM(3,Link_L,Link_M);

sys=c2d(ss(A,B,C,D),.012);
G=zeros(N);
for i=1:N
    for k=1:i
        G(i,k)=sys.c*sys.a^(i-k)*sys.b;
    end
end

%**************************************************************************
%Initialize the system
%**************************************************************************
Initialization;

%**************************************************************************
%Iterative process
%**************************************************************************
iter_max=30; %Maximum number of iteration
time=zeros(iter_num,1); %Computational time
```

```
for k=1:iter_max
        %Current iteration
        %-----------------------------------------------------------------
        %Step 1: solve SDP problems
        l=[-max(u_low-u,de_u_low);min(u_high-u,de_u_high);-du_low+J*u;du_high-J*u];

        tic
        cvx_begin sdp
            variables rho z12 z13 z23
            variable ld(4*N)
            minimize (rho)
            [(G1'*Q*G0+G0'*Q*G1)*z13+G1'*Q*G1+G0'*Q*G0+R,F'*ld-2*G1'*Q*error*z13-2*G0'*Q*error...
            +2*G1'*Q*G1*u*z12+2*G0'*Q*G1*u*z23;
            (F'*ld-2*G1'*Q*error*z13-2*G0'*Q*error+2*G1'*Q*G1*u*z12+2*G0'*Q*G1*u*z23)',...
            -4*ld'*l-8*u'*G1'*Q*error*z23+rho] >= 0;
            [1 z12 z13;z12 1 z23;z13 z23 1] >= 0;
            ld >= 0;
        cvx_end
        time(k)=toc;

        %Step 2: calculate the value of delta_u(k+1)
        delta_u=-1/2*inv((G1'*Q*G0+G0'*Q*G1)*z13+G1'*Q*G1+G0'*Q*G0+R)*...
        (F'*ld-2*G1'*Q*error*z13-2*G0'*Q*error+2*G1'*Q*G1*u*z12+2*G0'*Q*G1*u*z23);

        %Calculate the derivative of the input w.r.t iteration index
        IterIndex(:,k+1)=delta_u;
        %-----------------------------------------------------------------
        %Next iteration

        theta_next=(theta2-theta1)*rand+theta1;
        while (theta_next-theta > 0.04)||(theta_next-theta < -0.04)
            theta_next=(theta2-theta1)*rand+theta1;
        end

        % Length of the Link
        Link_L = 0.0254 * 15;
        % Mass of the Link
        Link_M = (1+theta_next)*0.065/.8;

        %Pertubed model of system using FEM method
        [A,B,C,D]=plantFEM(3,Link_L,Link_M);

        sys=c2d(ss(A,B,C,D),.012);
        G=zeros(N);
        for i=1:N
            for l=1:i
                G(i,l)=sys.c*sys.a^(i-l)*sys.b;
            end
        end

        u=delta_u+u;
        indis(:,k+1)=u;
        y=G*u;
        y=[0;y(1:N-1)];
        outdis(:,k+1)=y;
        error=ref-y;
        e(k+1)=norm(error,inf);
```

```
        ee(k+1)=sqrt(error'*Q*error);

        theta=theta_next;
        tt(k+1)=theta;

        %Calculate the derivative of the input w.r.t time index
        TimeIndex(1,k+1)=u(1);
        for i=2:N
            TimeIndex(i,k+1)=u(i)-u(i-1);
        end
    if norm(error,inf) <= epsilon
        break;
    end
end

%**************************************************************************
%Save the results
%**************************************************************************
save FL_IV_input.mat indis
save FL_IV_output.mat outdis
save FL_IV_TimeConstraint.mat TimeIndex
save FL_IV_IterConstraint.mat IterIndex
save FL_IV_H_Error.mat e
save FL_IV_Q_Error.mat ee
savefile= 'FL_IV_uncertainty.mat';
save(savefile, 'tt');
savefile='computational_time.mat';
save(savefile,'time');

%**************************************************************************
%Display the results
%**************************************************************************
disp('Number of iteration is:');
disp(k);

FigureDisplay

%**************************************************************************
%End of program
%**************************************************************************
```

FlexibleLink_Para.m

```
%==========================================================================
%File contains the physical parameters of a flexible link
%Inputs:

%Outputs: parameters of flexible link
%==========================================================================
%--------------------------------------------------------------------------
%Author |   Nguyen Dinh Hoa
%Date   |   22/9/2009
%Work   |   Control System Research Laboratory, Chulalongkorn University
%--------------------------------------------------------------------------

% Armature Resistance (Ohm)
Rm = 2.6;
% Motor Torque Constant (N.m/A)
Kt = .00767;
% Motor Back-EMF Constant (V.s/rd)
Km = .00767;
% Internal Gear Ratio (of the Planetary Gearbox)
Kgi = 14;
% Gearbox Efficiency
Eff_G =0.9;
% Motor ElectroMechanical Efficiency
Eff_M =0.69;
% J24: 24-tooth Gear Inertia (on the Motor Shaft)
m24 = .005; % mass (kg)
r24 = 0.5 / 2 * 0.0254; % radius (m)
J24 = m24 * r24^2 / 2;
% J72: 72-tooth Gear Inertia (on the Potentiometer Shaft)
m72 = .030; % mass (kg)
r72 = 1.5 / 2 * 0.0254; % radius (m)
J72 = m72 * r72^2 / 2;
% J120: 120-tooth Gear Inertia (on the Load Shaft)
m120 = .083; % mass (kg)
r120 = 2.5 / 2 * 0.0254; % radius (m)
J120 = m120 * r120^2 / 2;
% Rotor Inertia (kg.m^2)
Jm = 3.9e-7;
% High Gear Configuration: (1x) 24-tooth gear, (2x) 72-tooth gear, (1x) 120-tooth gear
Kge = 5;
Kg = Kgi * Kge;
Jeq = J24 + 2 * J72 + J120 + Kg^2 * Jm * Eff_G;
% Equivalent Viscous Damping Coefficient as seen at the Load (N.m.s/rd)
Beq = 4e-3;
% Length of the Link is 15 inches
Link_L = 0.0254 * 15;
% Uncertain weight of the Link
Link_M = (1+theta)*0.065/.8;
% Nominal weight of the Link
Link_M_0 = 0.065/.8;
% Calculte the Moment of Inertia of a Link (Assumed Rigid)
Jarm = Link_M * Link_L ^ 2 / 3;
% Natural Frequency was experimentally determined to be 3 Hz
Wc = 2 * pi * 3;
% Estimate the Stiffness of the simplified Link Model
K_Stiff = Wc^2 * Jarm;
```

Initialization.m

```
N=101; %number of samples

ref=zeros(N,1);  %Target trajectory

for k=17:41
    ref(k)=1/25*(k-16);
end
for k=42:60
    ref(k)=1;
end
for k=61:85
    ref(k)=1-1/25*(k-60);
end

u=zeros(N,1);                                           %Zero initial input
y=G*u;                                                  %Initial output
y=[0;y(1:N-1)];
error=ref-y;

u_low=-3*ones(N,1); u_high=3*ones(N,1);                 %Constrains of the input magnitude
du_low=-4*ones(N,1); du_high=4*ones(N,1);               %Constrains of the input rate
de_u_low=-4*ones(N,1); de_u_high=4*ones(N,1);           %Constrains of the input changes between iterations

Q=.9*eye(N); R=.01*eye(N);

J=eye(N);
for k=2:N
    J(k,k-1)=-1;
end
F=[-eye(N);eye(N);-J;J];

indis=zeros(N,1000);
indis(:,1)=u;
outdis=zeros(N,1000);
outdis(:,1)=y;
IterIndex=zeros(N,1000);
TimeIndex=zeros(N,1000);
e=zeros(1000,1);
ee=zeros(1000,1);
e(1)=norm(error,inf);
ee(1)=sqrt(error'*Q*error);

epsilon=1e-2; %Tolerance

%***************************************************************************
%End of program
%***************************************************************************
```

plantFEM.m

```
%=========================================================================
% Perturbed plant transfer function using finite element model
%  [A,B,C,D]=plantFEM(N,M_tip_po,M_tip);
% M_tip_po is the M_tip position (not necessary to be equal to the length
% of the beam), M_tip is the weight of the Flexible Link
% Example:
%       [A,B,C,D]=plantFEM(1,0.45,0.05);
%=========================================================================

%-------------------------------------------------------------------------
%Author     |   Nguyen Dinh Hoa
%Date       |   25/2/2009
%Work       |   Control System Research Laboratory, Chulalongkorn University
%Version    |   Modified version of previous plantFEM.m
%-------------------------------------------------------------------------


function [A,B,C,D]=plantFEM(n,M_tip_po,M_tip)

%--------------------Physical Parameters--------------------------
FlexibleLink_para;

%---------------------------Finite Element Method-------------------------
li = Link_L/n; %length of element

%Stiffness matrix of one element
Ki = E*I/li^3* [12 6*li -12 6*li; 6*li 4*li^2 -6*li 2*li^2; -12 -6*li 12 -6*li; 6*li 2*li^2 -6*li 4*li^2];
%Mass matrix of one element
Mi = Link_M_O/(n*420)* [156 22*li 54 -13*li; 22*li 4*li^2 13*li -3*li^2; 54 13*li 156 -22*li;
-13*li -3*li^2 -22*li 4*li^2];

Mhat = zeros(2*n+2, 2*n+2);
Khat = zeros(2*n+2, 2*n+2);

for i = 1:n
Mhat(2*i-1:2*i+2, 2*i-1:2*i+2) = Mhat(2*i-1:2*i+2, 2*i-1:2*i+2) + Mi;
Khat(2*i-1:2*i+2, 2*i-1:2*i+2) = Khat(2*i-1:2*i+2, 2*i-1:2*i+2) + Ki;
end

ep=ceil(M_tip_po*n/Link_L);
xp=M_tip_po-(ep-1)*Link_L/n;

%The displacement of the elements
A1=1-3*xp^2/li^2+2*xp^3/li^3;
A2=li*(xp/li-2*xp^2/li^2+xp^3/li^3);
A3=3*xp^2/li^2-2*xp^3/li^3;
A4=li*(-xp^2/li^2+xp^3/li^3);

AA=[A1 A2 A3 A4];

Mhat(2*ep-1:2*ep+2, 2*ep-1:2*ep+2)=Mhat(2*ep-1:2*ep+2, 2*ep-1:2*ep+2)+M_tip*AA'*AA;

Maa = Mhat(3:2*n+2, 3:2*n+2);
Kaa = Khat(3:2*n+2, 3:2*n+2);

M11 = Maa(1:2:2*n-1, 1:2:2*n-1);
M12 = Maa(1:2:2*n-1, 2:2:2*n);
```

```
M21 = Maa(2:2:2*n, 1:2:2*n-1);
M22 = Maa(2:2:2*n, 2:2:2*n);
K11 = Kaa(1:2:2*n-1, 1:2:2*n-1);
K12 = Kaa(1:2:2*n-1, 2:2:2*n);
K21 = Kaa(2:2:2*n, 1:2:2*n-1);
K22 = Kaa(2:2:2*n, 2:2:2*n);

M = M11 - K21'*inv(K22)*M21 - M12*inv(K22)*K21 + K21'*inv(K22)*M22*inv(K22)*K21;
K = K11 - K12*inv(K22)*K21;
Q = [eye(n,n);-inv(K22)*K21];

lhat = li:li:Link_L; lhat=lhat';

Ie = Ih+lhat'*M*lhat;
M1 = M*lhat;

temp1 = (1/Ie)*M1'*inv(M-M1*M1'/Ie)*K;
temp2 = -inv(M-M1*M1'/Ie)*K;
temp = [temp1; temp2];

%SISO model
At = [zeros(n+1,n+1) eye(n+1,n+1); zeros(n+1,1) temp zeros(n+1,n+1)];
Bt = [zeros(n+1,1); (1/Ie^2)*M1'*inv(M-M1*M1'/Ie)*M1+1/Ie; -(1/Ie)*inv(M-M1*M1'/Ie)*M1];

B=zeros(size(Bt,1),1);
A=At;
for I=1:size(Bt,1),
  B(I,1)=Bt(I,1)*Eff_M*Eff_G*Kt*Kg/Jeq/Rm;
  A(I,n+2)=At(I,n+2)-Bt(I,1)*(Eff_M*Eff_G*Kt*Kg^2*Km+Beq*Rm)/Jeq/Rm;
end
C = zeros(1,2*n+2);
C(1,2*ep-1:2*ep+2)=AA;
C=C(1,3:2*n+2);
C1=C(1,1:2:2*n-1);
C2=C(1,2:2:2*n);
C=[C1 C2];
C=C*Q;
C=[1 C/M_tip_po zeros(1,n+1)];
D=0;

return

%*************************************************************************
%End of program
%*************************************************************************
```

Biography

Dinh Hoa Nguyen was born in 1984 at Phutho, Vietnam. He received his Bachelor's degree in Automatic Control from Faculty of Electrical Engineering, Hanoi University of Technology, Vietnam, in August 2007. He has been granted a scholarship by the AUN/SEED-Net (www.seed-net.org) to pursue his Master's degree in Electrical Engineering at Chulalongkorn University, Thailand, since October 2007. He has conducted his graduate study of Master's degree with the Control Systems Research Laboratory, Department of Electrical Engineering, Faculty of Engineering, Chulalongkorn University from October 2007 to October 2009. His research interest includes iterative learning control, robust control, optimal control, convex optimization, linear matrix inequalities and its applications in control.

List of Publications

1. D. H. Nguyen, D. Banjerdpongchai, "An LMI approach for Robust Iterative Learning Control with Quadratic Performance Criterion", *Journal of Process Control*, vol. 19, pp. 1054–1060, 2009.

2. D. H. Nguyen, D. Banjerdpongchai, "Robust Iterative Learning Control for Linear Systems with Time-Varying Parametric Uncertainties", in *Proc. 48th IEEE CDC*, Shanghai, China, December 16-18, 2009 (Accepted).

3. D. H. Nguyen, D. Banjerdpongchai, "Robust Iterative Learning Control for Linear Systems with Iteration-Varying Parametric Uncertainties", in *Proc. 7th ASCC*, pp. 716-721, Hong Kong, August 27-29, 2009.

4. D. H. Nguyen, D. Banjerdpongchai, "Robust Iterative Learning Control for Linear Systems with Time-Invariant Parametric Uncertainties", in *Proc. ICROS-SICE International Joint Conference*, pp. 4178-4183, Fukuoka, Japan, August 18-21, 2009.

5. D. H. Nguyen, D. Banjerdpongchai, "Robust Iterative Learning Control for Flexible Link under Parametric Uncertainty", in *Proc. 6th ECTI-CON*, pp. 376-379, Pattaya, Thailand, May 6-9, 2009.

6. D. H. Nguyen, D. Banjerdpongchai, "An LMI approach for Robust Iterative Learning Control with Quadratic Performance Criterion", in *Proc. 10th ICARCV*, pp. 1805-1810, Hanoi, Vietnam, December 17-20, 2008.