1-1-2011

Quadrotor UAV control : online learning approach

Pong-In Pipatpaibul
Ryerson University

QUADROTOR UAV CONTROL: ONLINE LEARNING APPROACH

by

Pong-in Pipatpaibul
Bachelor of Aerospace Engineering
King Mongkut's University of Technology North Bangkok
Bangkok, Thailand, April 2009

A Thesis
presented to Ryerson University

in partial fulfillment of the
requirements for the degree of
Master of Applied Science
in the program of
Aerospace Engineering

Toronto, Ontario, Canada 2011

AUTHOR'S DECLARATIONS

I hereby declare that I am the sole author of this thesis or dissertation.

I authorize Ryerson University to lend this thesis or dissertation to other institutions or individuals for the purpose of scholarly research.

_____

I further authorize Ryerson University to reproduce this thesis or dissertation by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

ABSTRACT

QUADROTOR UAV CONTROL: ONLINE LEARNING APPROACH

Pong-in Pipatpaibul

A thesis for the degree of
Master of Applied Science, 2011

Department of Aerospace Engineering, Ryerson University

Quadrotor unmanned aerial vehicles (UAVs) are recognized to be capable of various tasks including search and rescue and surveillance for their agilities and small sizes. This thesis proposes a simple and robust trajectory tracking controller for a quadrotor UAV utilizing online Iterative Learning Control (ILC) that is known to be effective for tasks performed repeatedly. Based on a nonlinear model which considers basic aerodynamic and gyroscopic effects, the quadrotor UAV model is simulated to perform a variety of maneuvering such as take-off, landing, smooth translation and horizontal and spatial circular trajectory motions. PD online ILCs and PD online ILCs with switching gain (SPD ILCs) are studied, tested and compared. Simulation results prove the ability of the online ILCs to successfully perform certain missions in the presence of considerably large disturbances and SPD ILCs can obtain faster convergence rates.

ACKNOWLEDGEMENT

I would first and foremost like to express my sincere gratitude and appreciation toward my supervisor Dr. Puren Ouyang for his dedication to my study. I am genuinely grateful for his kindness and invaluable time guiding the way. His wisdom has greatly enlightened and enhanced my inexperienced knowledge and skills, through his persistence and patience despite my stubbornness. Without his generous educational and financial support, this thesis would have been nowhere near as it is.

I would like to thank the Graduate Program Director Dr. Jeffrey Yokota for his invaluable preliminary guideline throughout my graduate study. His advice led me to a wise decision of the path of my life. I would also like to thank the members of the advisory committee: Dr. Krishna D. Kumar and Dr. Bo Tan for their indispensable time and effort reviewing and advising this thesis.

I would like to thank my parents for their main financial support, love, care and encouragement despite the long distance.

My study in the Master of Applied Science was supported by the financial contribution of Ryerson Graduate Scholarship (RGS).

DEDICATED TO

My beloved parents

All the teachers and mentors in my life

Table of Contents

List of Figures

List of Tables

Acronyms

ASL	Adaptive Switching Learning
DC	Direct Current
HDPE	High Density Poly-Ethylene
HTA	Heavier Than Air
ILC	Iterative Learning Control
LQR	Linear Quadratic Regulator
LTA	Lighter Than Air
LTI	Linear Time-Invariant
MBPC	Model-Based Predictive Control
MIMO	Multiple Input Multiple Output
PID	Proportional Integral Derivative
R/C	Radio Control
SDRE	State-Dependent Riccati Equation
SISO	Single Input Single Output
SMC	Sliding Mode Control
SPD	Switching Gain PD
STARMAC	The Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take-off and Landing

List of Symbols

ϕ	Roll angle
ω	Angular velocity
ψ	Yaw angle
τ	Torque
ρ	Convergence factor
θ	Pitch angle
m	Mass
l	Quadrotor arm length
d	Drag coeeficient
b	Thrust coefficient
V	Linear velocity
T	Thrust/Lift
Q	Drag moment
I	Moment of inertia
F	Force
Ω	Rotor speed
$\mathcal{R}$	Rotation matrix
u_k	Control input of k^{th} iteration
e_k	Error of k^{th} iteration
K_{pon}	Proportional part online learning gain
K_{poff}	Proportional part offline learning gain
K_{don}	Derivative part online learning gain
K_{doff}	Derivative part offline learning gain
J_r	Rotor inertia
Ω_r	Total rotor speed
Ω_{max}	Maximum rotor speed

Chapter 1 Introduction

1.1 Motivation

Unmanned Aerial Vehicles (UAVs) have drawn much attention from many researchers for decades and their studies still keeps growing intensely. With the advanced technology, their existences became reality and even better each day. Because of being unmanned, not only that their sizes and cost can be minimized but also capability of various tasks that manned aircrafts cannot perform can be achieved. Some tasks that suit UAVs include surveillance, geographic monitoring and crop dusting in open-areas or even accessibility in urban areas, and many more. There exists many categories of UAVs, one of the most interesting and challenging, in terms of capability, is the type of vertical take-off and landing (VTOL) UAVs. The feature of vertical movement provides them with advantages over other types of UAVs in terms of accessibility, agility, low speed flight and hovering. Different configurations of VTOL UAVs were proposed and designed, among them are quadrotor UAVs. Having four rotors located at each end and heading upward makes them one of the simplest, yet most efficient configurations. Being underactuated systems (less actuators than degrees of freedom) and agile, however, result in natural instability and thus require a carefully designed control system for stabilization and path tracking.

Many of control methods have been proposed to satisfactorily deal with the control problems of UAVs. Most of them focused heavily on the stabilization problem which is the first step toward successful flights. Some also handled position tracking or velocity tracking in order to obtain certain maneuverings and thus full autonomous control. However, most of them relied on complicated dynamic model which might be unavailable in certain situations and some control methods require intense computation which might become a problem when applied on-board. Furthermore, only a few researches addressed disturbances in simulations or experiments, which should be a major concern in real applications.

Iterative Learning Controls (ILCs) were proposed to deal with tasks performed repeatedly, which are mostly used in industrial applications such as robotic manipulation and pneumatic systems. They are also well known for their simplicities to implement and robustness against

uncertainties and disturbances. These features fit well to quadrotor UAVs' maneuverings that mostly are performed repeatedly and oftentimes experience model uncertainties and noises from sensors. Motivated by the aforementioned advantages of ILCs, this research will focus on the control system design of a quadrotor UAV for various maneuverings.

1.2 Objectives and Methodology

This thesis intends to take advantage of simplicity and robustness of online-type ILCs and to develop a learning control system for a quadrotor UAV in order to obtain a good trajectory tracking performance. Main objectives of this thesis can be summarized as

- To verify the capability, performance and robustness against large disturbances of online ILCs on the implemented system of nonlinear dynamic model of a quadrotor UAV.
- To compare performance among online ILCs for future development.
- To pioneer online learning approach on quadrotor trajectory tracking and obtain guidelines for further improvement.
- To extend applicability of ILCs on wider range of applications in addition to industrial applications.

All the previously mentioned objectives can be achieved in this thesis via simulations done in MATLAB programming by

- Forming nonlinear dynamic model of a quadrotor and referring model parameters from a well-design quadrotor.
- Simulating major maneuverings which include take-off, landing, X-Y translation and circular trajectory.
- Adding large disturbances in all simulations to prove robustness of online ILCs
- Comparing two types of online ILCs in simulations and comparing other types by citing results from other sources.
- Indentifying effect of each control gain on the tracking performances.

1.3 Organization of Contents

The contents in this research are organized as follows:

Chapter 2: Literature Reviews - to provide general concepts of aircrafts, background of quadrotor UAVs including their history, designs and control methods implemented to quadrotor UAVs from various studies; and researches and development in ILCs.

Chapter 3: Quadrotor Modeling - to explore detailed concepts and modeling of a quadrotor UAV which includes basic motions of a quadrotor; assumptions adopted in modeling; kinematics; dynamics; and parameters used in the simulations.

Chapter 4: Control System Design - to present detailed concepts of ILCs and their implementations on the quadrotor model. These include convergence analyses, state-space modeling, implemented control laws of both PD online ILCs and SPD ILCs, and finally trajectory generation method.

Chapter 5: Simulation Results - to demonstrate performances, convergence rates and tracking errors in the presences of disturbances of the two ILCs implemented to the quadrotor model performing various maneuverings.

Chapter 6: Conclusion and Discussion - to provide general reviews of this thesis and to discusse some issues and possible future development

Chapter 2 Literature Reviews

This chapter is devoted to describe and summarized past studies and developments of quadrotor UAVs by various research groups and some investigation in iterative learning control. Firstly, basic principles and the main reasons for choosing quadrotor UAVs are established in Section 2.1. Brief history and studies in quadrotor UAVs are then shown in Section 2.2. Control algorithms for quadrotor UAVs and ILCs are investigated in Section 2.3 and 2.4, respectively. Finally, concluding remarks of this chapter are summarized in Section 2.5.

2.1 Aircraft and UAV

2.1.1 Aircraft Generalities and Principles

Aircrafts have been invented and developed long ago. Many designs and principles were proposed and implemented. In this modern era, aerial vehicles can be generally classified into two categories: Heavier Than Air (HTA) and Lighter Than Air (LTA) and consequently divided into several sub-categories as illustrated in Fig. 2-1.

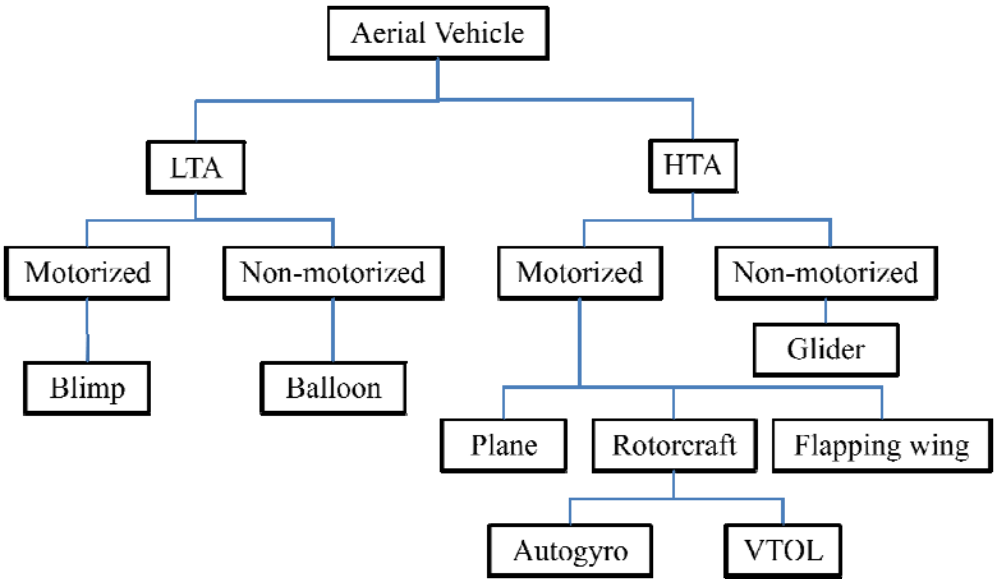

Fig. 2-1 Aircraft Classification

It is clear that gliders, planes and autogyros, which need forward flight in order to generate lift, are not capable of hovering and VTOL. Balloons and blimps are able to hover easily, owing to buoyant force from lighter-than-air gases they contain, but they are poorly maneuverable. Flapping-wing or bird-like (or insect-like) UAVs may be able to hover and VTOL at the expense of extremely complex mechanics and control. VTOL rotorcrafts are highly agile, maneuverable and able to hover at the expense of high energy consumption, hence seem to fit best to the aforementioned requirements.

There are several configurations of recognized VTOL rotorcrafts. One of the most common VTOL rotorcrafts is a conventional helicopter; one main rotor above the center of gravity for lift generating and smaller one on the tail boom for counter-torque. This configuration uses a quite large main rotor and takes more space from long tail and requires relatively complex mechanics from pitching rotor. Another configuration is double-rotor. This includes tandem rotors [***1***] and coaxial rotors[***2***]. These two configurations have advantages over conventional helicopters. For example, two rotors generate lift instead of only one main rotor, allowing more payloads, and the two rotors cancel out each other's torque that stabilizes the yaw direction motion. Two main rotors are located on each end of a tandem helicopter (Fig. 2-2, Fig. 2-3). This also allows wider range of the center of gravity. On the other hand, it needs a larger size body and a synchronous transmission between two rotors to avoid blade collision, making it more complicated.

This problem can be eliminated by having two rotors in a single shaft as in coaxial helicopters (Fig. 2-4), making this configuration quite compact. Although having advantages, it takes an even more complex mechanics in order to drive two coaxial rotors in opposite direction to balance torque and a more complicated aerodynamics that the two rotors affect each other, making it more difficult to design and control. However, coaxial configuration is more likely to be useful for miniaturization [***2***].

Fig. 2-2 Boeing CH-47 Tandem helicopter [*1*]

Fig. 2-3 Dragonfly Pictures DP-6 Tandem rotor UAV [*3*]

Unlike the previous VTOL rotorcraft configurations, a quadrotor is much simpler although requires a larger size. Having all four fixed-pitch rotors located at each end of the body, this configuration utilizes every rotor to generate lift, allowing more payloads. This also results in simple mechanics and design. Hence the quadrotor configuration is then chosen in this research for its simplicity, in terms of designs, mechanics and aerodynamics, and the capability of VTOL, hovering and low speed flight. More details in quadrotor concepts are explained in Chapter 3.

Fig. 2-4 CoaX Coaxial rotor UAV[*2*]

2.1.2 UAVs Applications

Since the early years of aircraft invention, most of them relied heavily, if not solely, on human being to control. This resulted in very large size vehicles in order to accommodate a pilot. Sometimes, due to technology limitations, the systems were even unreliable and too complicated to control by human being, sometimes even leading to tragic accidents.

Later on, with the dramatic growth in industry and improvement in technology, the possibility of small Unmanned Aerial Vehicles (UAVs), which eliminated the needs for pilot and thus decreasing weight, size and cost, seemed to be within grasp. This also opened doors for new applications including, for example, reconnaissance for military purpose, surveillance for geographic study, planetary exploration [***4***] and search and rescue.

Many types of UAVs were developed for certain range of missions, filling the gap for specific needs. However, hovering, low speed flight, high maneuverability and Vertical Take-off and Landing (VTOL) are preferred or required for many applications since those features would result in a more accessible UAV, which is especially needed in urban area missions where spaces are limited.

2.2 Quadrotor UAVs History and Development

The idea of vertical flight has been around in mankind since 400 BC by a toy called Chinese Top[***5***], consisting of feathers (or a propeller) at an end of a stick, as seen in Fig. 2-5, one can simply quickly spin the stick off hands and let it fly freely. The history of quadrotor aerial vehicles, however, only dates back in 1907 when the French brothers Louis and Jacques Breguet invented the first full-sized piloted quadrotor. The vehicle shown in Fig. 2-6, named Gyroplane No.1 [***1***], consisted of four long girders made of steel tubes. Each rotor had four blades and placed on each of four ends, summing up to 32 individual lifting surfaces. Due to lack of a high power engine and light-weight materials, it was reported to hop above ground for only a few moments.

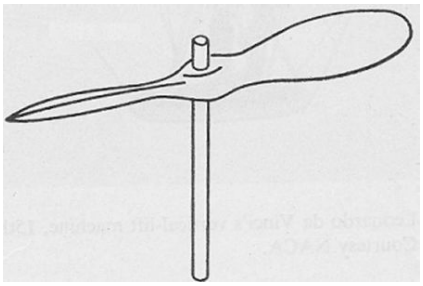

Fig. 2-5 Chinese top[*5*]

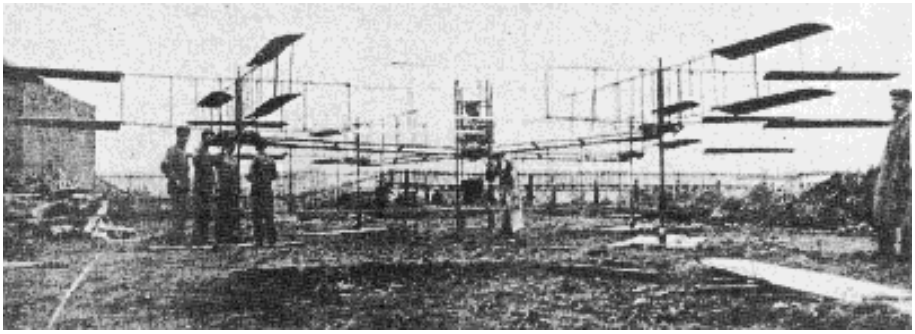

Fig. 2-6 Gyroplane No.1 by the Breguet brothers in 1907 [*1*]

Later on, some successful designs of quadrotor were achieved; one of them was Convertawings by D.H.Kaplan in the United States in 1956[***6***], shown in Fig. 2-7. The

vehicle was configured in ‘H’ shape rather than ‘+’ shape and weighed totally about 990 kg. It could successfully fly but then the development was ceased by the United States military.

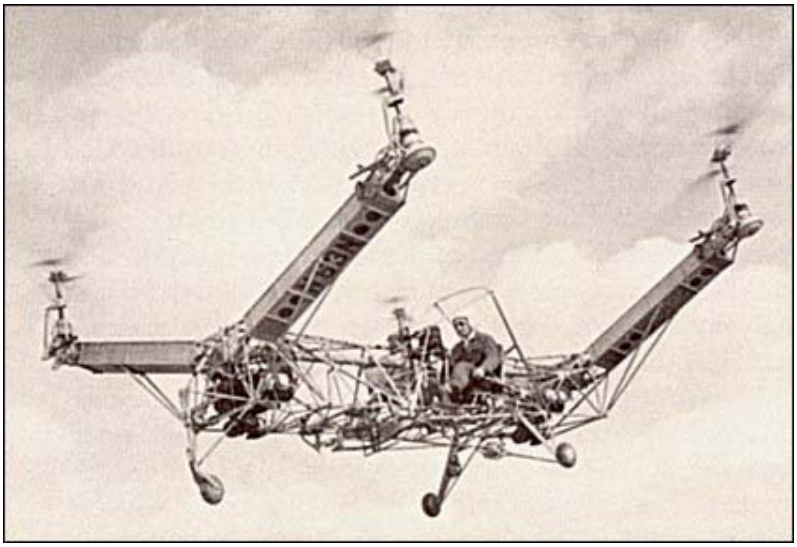

Fig. 2-7 Convertawings Model A in 1956[*6*]

In these modern days, many aspects of technology have developed dramatically. Much lighter materials were discovered, higher power engines were invented, and especially MEMS were introduced, making small, light-weight, unmanned aerial vehicle ideas become realistic. As a result of the aforementioned improvement, many quadrotor UAV projects were independently studied and developed.

One of the first quadrotor UAVs was ‘The Mesicopter’, shown in Fig. 2-8, developed by I. Kroo of Stanford University in 2001. With the span of the size of a coin, the mass of the UAV was approximately 3-15g and powered by DC motors. As a result of being very small, air viscosity, micro-manufacturing and power and control system integration become critical issues [*7*].

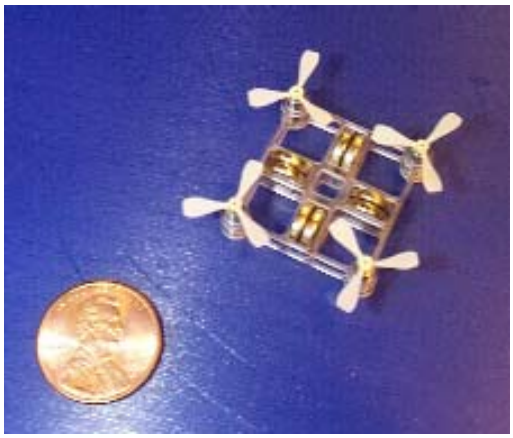

Fig. 2-8 the Mesicopter

In 2002, a visual feedback control was implemented in a quadrotor UAV development by E. Altug of Pennsylvania State University as seen in Fig. 2-9. Using image processing, the visual system used ground camera to estimate the position and attitude and then sent control input signals, derived using feedback linearization and backstepping method, to the on-board controller. The aircraft utilized in the study was the commercial R/C toy HMX-4 with the mass of 0.7kg and the span of 76cm. It was shown that relying only on ground camera was not enough for full control [***8***].

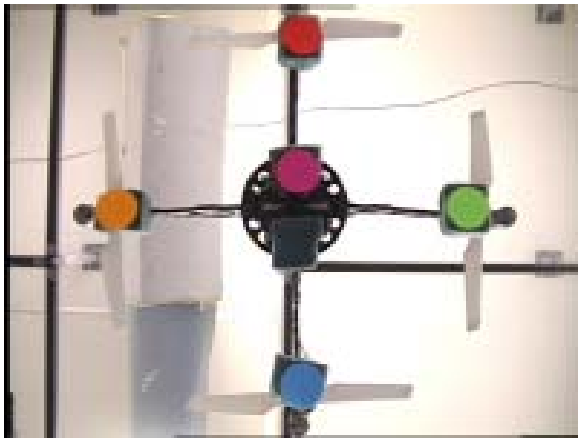

Fig. 2-9 Visual Feedback quadrotor by Pennsylvania State University[***8***]

X-4 Flyer project is another example of quadrotor UAVs studies. First developed in 2002 at Australian National University, the X-4 MkI weighed 2 kg with 70 cm in span. In the design of so called hub-and-spokes configuration, the hub was built from HDPE and spokes from a material used in hunting arrows. Simple gyro sensors were used. Then a new prototype of X-4 MkII was released in 2006, with a totally different configuration made of carbon-fiber and aluminum and weighed up to 5kg including payload. The rotors were placed below the frame instead of above as mostly designed. It was said to have high thrust-to-weight ratio and to be more robust against disturbances. Both prototypes are illustrated in Fig. 2-10 [***9***][***10***].

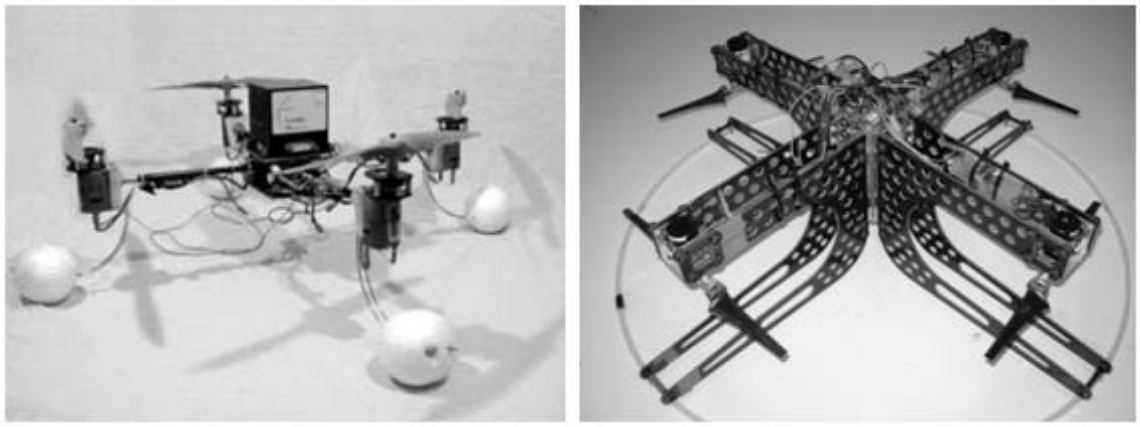

Fig. 2-10 ANU's X-4 Mk I(left) and X-4 Mk II(right)[*9*][*10*]

Another quadrotor UAV research was established in 2003 at University of British Columbia. The work focused on nonlinear modeling of a quadrotor UAV and utilizing H∞ and MBPC to stabilize the system and manipulate trajectory tracking. The study used the Draganflyer III, a commercial R/C toy, as the testbed and attached to a flying mill for flight testing, shown in Fig. 2-11[***11***].

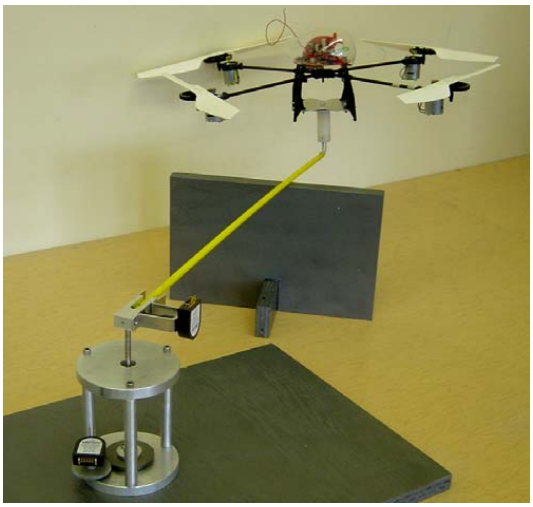

Fig. 2-11 University of British Columbia's quadrotor testbed[***11***]

In 2004, a study was done at Cornell University by E.B. Nice's thesis[***12***]. The work comprised overall design; structure, propulsion and control modules. Sigma Point Filter was used for state estimation and LQR control was used for stabilization. After assembling, the prototype, shown in Fig. 2-12, finally weighed 6.2kg and could only perform hovering due to hardware failure before maneuvering tests could be done[***12***].

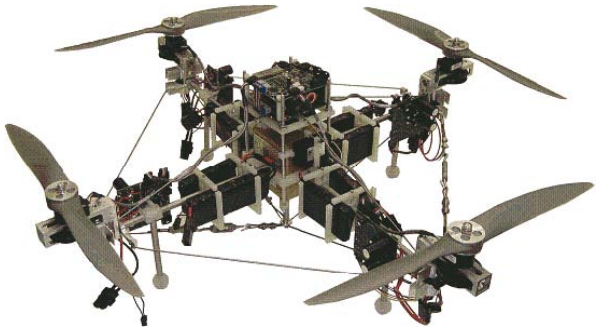

Fig. 2-12 a Cornell University's quadrotor project[***12***]

At about the same time, Stanford University launched a new quadrotor development project called 'STARMAC' in 2004. Its main goal was to provide testbed for algorithms verification in decentralized optimization so that each vehicle could compute optimal trajectory to avoid collision and obstacles. The project utilized the Draganflyer III as a testbed and would

communicate with a ground station. Linearized dynamic model was used along with the Kalman filter for state estimation. It was also reported that frame stiffening added to the testbed greatly improved attitude estimation. Then later in 2007, STARMAC II was introduced. This sequel included more intense aerodynamic effects study and reconfigured the testbed with higher thrust. STARMAC and its successor of STARMAC II are shown in Fig. 2-13 Fig. 2-14, respectively [***13***][***14***].

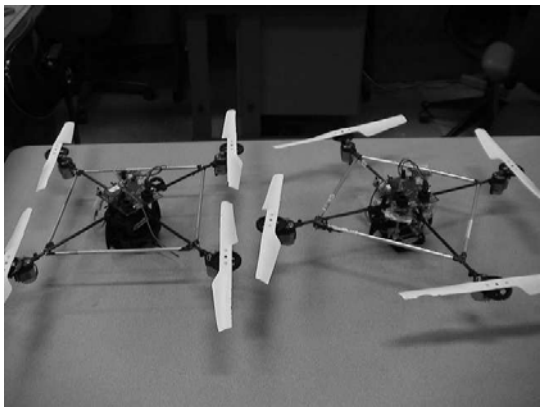

Fig. 2-13 Two Draganflyer III used in STARMAC[***13***]

Fig. 2-14 STARMAC II[***14***]

Another interesting quadrotor UAV study is the OS4 project [***15***], started in 2003 at Swiss Federal Institute of Technology. Focusing on design and control of a quadrotor, intensive designs and various control methods have been tried, compared and evaluated, including Sliding Mode Control, Backstepping Control, LQR and PID control. Then full control was achieved in 2007. The model weighed 520g and was able to take-off, land and avoid collision

automatically using PID control combined with backstepping control and more detailed aerodynamic effects were taken into account. The OS4 quadrotor is shown in Fig. 2-15[***16***].

Fig. 2-15 OS4 quadrotor of Swiss Federal Institute of Technology

In summary, many research and studies have independently implemented various control algorithms for full control. A number of them were structurally based on commercial quadrotor R/C toys while others designed their own, ranging from as light as a few grams to 6kg. All the aforementioned projects are listed in Table 2-1.

Table 2-1 List of quadrotor projects

Project	Institute	Testbed	Total weight	Goal
The Mesicopter	Stanford University	self-developed	3-15g	centimeter-scale UAV
STARMAC I, II	Stanford University	Draganflyer III	0.7kg	Multi-agent testbed
X-4 MkI, MkII	Australian National University	self-developed	2kg (MkI) 4kg (MkII)	full design and control
OS4	Swiss	self-	0.52kg	full design and

	Federal Institute of Technology	developed		control
E. Altug's thesis	Pennsylvania State University	HMX-4	0.7kg	visual control
E.B. Nice's thesis	Cornell University	self-developed	6.2kg	full design and control
M. Chen and M. Huzmezan's research	University of British Columbia	Draganflyer III	0.7kg	MBPC and H∞ implementation

2.3 Quadrotor UAVs Control Algorithm

Quadrotor UAVs have been studied for years. Many control algorithms were tried out and implemented in order to stabilize, maneuver and ultimately perform various missions. Some have different advantages and disadvantages over others. This section devotes to list and summarize major controlling schemes for a quadrotor, which are PID control, linear and nonlinear state-feedback control, sliding mode control (SMC), backstepping control, fuzzy control and neural network control.

Table **2-2** lists some control methods used in for controlling quadrotor UAVs. Detailed discussions are shown in the following sections.

Table 2-2 List of studies on quadrotor UAV control systems

Source	Control Method	Objectives	Model Type	Remarks
[***17***]	Adaptive-Fuzzy	attitude stabilization	Nonlinear	- w/ disturbance - gradually back to equilibrium - simulation
[***18***]	LQR	trajectory optimization and path following	Nonlinear	- perform various maneuverings - simulation
[***19***]	PID	trajectory tracking	Nonlinear	- 10-50cm tracking accuracy - indoor and outdoor experiment
[***13***]	SMC+LQR	attitude and altitude control	Linearized	- SMC for altitude control and LQR for attitude control - outdoor experiment
[***15***]	Backstepping, SMC	full control	Nonlinear	- w/o disturbance - simulation and experiment on test-bench
[***20***]	Backstepping+PID	full control	Nonlinear	- w/ disturbance - experiment and simulation

				- full control attained
[***21***]	Backstepping, SMC	full control	Nonlinear	- simulation and experiment - SMC robust against uncertainty
[***22***]	SDRE	attitude and velocity control	Nonlinear	- w/o disturbance - simulation
[***23***]	Feedback Linearization, PD and PD w/ partial diff.	attitude stabilization	Nonlinear	- w/o disturbance - simulation and experiment - compare the methods
[***24***]	Neural Network + PID	full control	Nonlinear	- w/ disturbance - simulation - NN trained by PID
[***25***]	Neural Network (CMAC method)	full control	Nonlinear	- w/ disturbance - simulation
[***26***]	Output Feedback Neural Network	full control	Nonlinear	- theoretical proof only
[***27***]	Output Feedback Neural Network	full control	Nonlinear	- w/ disturbance - simulation - NN for observer and controller

2.3.1 Classic PID Control

One of the simplest ways to control a system is to utilize the classic control theory, i.e., Proportional, Integral and Differential (PID) Control. It is easy to implement in the sense that the control law is a function of output errors so that the designer can choose and tune the control gains according to the desired output. This results in a simple system model but in some cases it might be too sensitive to uncertainties and disturbances, ending up with poor stabilization. PIDs have been widely used in many quadrotor developments. For example, a PID controller was implemented in the quadrotor work called STARMAC [***19***]. In the paper, PID and angular acceleration feedback were used in attitude control. For path tracking controller, PI was used for the along tracking direction and PID for the cross tracking direction. The result was so accurate that the path tracking errors were less than 10 cm for indoor flight and 50 cm for outdoor.

2.3.2 State-feedback and LQR Control

State-feedback control schemes were also utilized in many quadrotor UAV researches. This is done by forming state-space model of the system then calculating system's poles and thus control gains from the required performance, in the case of linear systems. For a nonlinear system, feedback-linearization method might be used as shown in [***23***]. In that paper, classic PD, PD with partial differential and feedback-linearization were evaluated in simulation and testing of attitude stabilization. The classic PD control failed to stabilize the quadrotor in the real test based on the linearized model. The proposed feedback-linearization outperformed the others. In addition to the previous method, Linear Quadratic Regulators (LQRs) were widely studied. Based on linear state-space models, LQRs try to minimize the tracking error by minimizing a cost function. Simulations of a quadrotor path following using LQR for various maneuver missions were conducted in [***18***]. The results showed accurate path following in three different maneuvering without violating control input saturation. A more complex method of LQR scheme is State-dependent Riccati Equation (SDRE) controller that can be applied to a nonlinear model. It realizes the model as a linear one at any fixed state and thus requires intense calculation. An example of SDRE appears in [***22***]. The paper made use of SDRE for attitude control and simple state-feedback for velocity control.

2.3.3 Sliding Mode and Backstepping Control

Sliding Mode Control (SMC) and Backstepping control were heavily studied and compared in many papers. In a SMC, the system's states are controlled to move along sliding surface, s=0, in phase plane which makes the system stable. The control inputs always switch at high frequency between two values and thus said to be non-continuous functions. In contrast, backstepping control has a recursive structure in which an initial control input is not the real control input for the system but rather a 'virtual control input' acting as another state and eventually the scheme relates to the real control law. Both SMC and backstepping can be effectively applied to nonlinear systems.The two aforementioned methods were compared in [***21***] and [***15***]. They illustrated both simulation and experimental results for both attitude and altitude control. In [***21***], it was shown that SMC and backstepping outperform feedback control under disturbances and both are approximately identical, while a better performance of backstepping over SMC was shown in [***15***], which sometimes caused chattering, in the condition of no disturbances. In [***13***], an SMC technique was proposed for altitude control and used ordinary LQR for attitude control but many problems occurred due to motor vibration at high thrust and chattering from SMC. A full control of a quadrotor was proposed in [***20***], i.e., attitude control, position control, take-off and landing, using their new technique of Integral Backstepping (IB) which combined the backstepping method with an integral term of PID. The test results showed that errors in the pitch and roll angles were very small despite various disturbances and the quadrotor could take-off, land and avoid obstacle smoothly.

2.3.4 Adaptive-Fuzzy and Neural Network Control

Adaptive-Fuzzy control was proposed in [***17***] to stabilize a quadrotor at hovering using a new adaptive method that prevents fuzzy membership function center drift without decreasing performance. This control method does not require an accurate system model and was shown robust against disturbances in the simulation. Neural network control methods were proposed in many papers. They are considered a type of learning control which, in many cases, can be done online or on-board with the real applications. Although intense computing is required, the method was proved to have robustness against noises and disturbances. In [***26***], the availability of a neural network control was mathematically proved and later the simulation results were revealed in [***27***]. Using neural network output feedback, the paper showed

effective tracking performance. A neural-adaptive control which utilized adaptive method to train the neural network online was introduced in [***25***]. Stability achievement was confirmed by simulation results. A more simple neural network trained by PID control was proposed in [***24***] and stability was proved in the simulations.

2.4 Iterative Learning Control (ILC)

Iterative learning control adopts the idea of human learning by doing the same task repeatedly and performing better as one learns each time, so this scheme is popularly used in systems that perform repetitive tasks, such as robot manipulators, owing to its simple structure and straightforward implementation. ILC was first proposed by S. Arimoto in [***28***]. Based on information from previous learning cycles, the system's performance improves as iteration number increases, reducing trajectory tracking error and leading to convergence. One of the key advantages of an ILC is that the system's dynamic model is not fully required, making it simple and widely applicable. Since a quadrotor UAV may perform some specific missions repeatedly, ILCs are then considered potential control methods in this research. There are mainly three types of ILCs: offline ILC (open loop), online ILC (closed loop) and combination of both (online-offline ILC). In this section, online ILC and online-offline ILC are merged into one discussion because, in most cases, they yield almost the same result, as shown in [***29***].

2.4.1 Offline ILC

Offline ILCs use only information from previous iteration. One can implement the classic proportional, integral and differential (PID) controller to ILCs; this includes their subtype such as P, PD, D, PI and PID. The original one from [***28***] was D-type ILC, only differential term is used in the control law. Implementing the idea of learning scheme, his work resulted in precise trajectory tracking of a linear time-invariant (LTI) system within just a few iterations of learning. Another open loop D-type ILC was proposed in [***30***]. The paper presented a new way to implementation of D-type ILC for nonlinear systems with unknown relative degree. As a D-type ILC is complicated to use with high relative degree systems, applying dummy model turns any system into a first-order one, making it easy to control and resulting in perfect trajectory tracking performance and monotonic convergence of the

controlled system. However, it's shown in [***31***] that D-type ILC is more sensitive to noise measurement than P-type. In [***32***], they proposed P-type ILC for a discrete-time system by mathematically proving necessary and sufficient condition for monotonic convergence. A study in P-type ILC with systems that contain resonance was done in [***33***]. It was shown that typical learning schemes could result in growing error amplitude, which could ultimately lead to instability of the system. Aliasing and some filtering methods were compared and aliasing was shown to outperform other methods. Another solution in learning behavior was presented in [***34***] by down-sampling and applying low-pass filter in frequency domain of the system, due to the fact that errors are accumulated in high frequency band. The result was a monotonic convergence and a high accuracy in tracking performance.

The combination among P, I and D is also widely seen in ILCs. A PD-type ILC for a pneumatic system with disturbances was proposed in [***35***]. The system was analyzed in discrete-time and then compared among P-type ILCs, PD-type ILCs and pre-saved control laws from a system that already converges, and the simulation result showed a better performance of PD-type ILCs. In [***36***], the optimal design for a monotonic convergent discrete-time system was proposed. Using averaged difference instead of one step backward difference proved to reduce high frequency noise. The paper mathematically showed the trade-off between noise reduction and the monotonic convergence rate. PD-type ILCs with anticipatory approach was presented in [***37***] by time shifting ahead and sampling data. A more intriguing scheme implemented to a LTI system was proposed in [***38***]. Instead of using information from previous adjacent iteration, the controller used all the previous iterations and averaged them, rendering it effective against initial state error. PI-type ILCs were also studied and proposed in [***39***] with the purpose of pointing out the effect of the integral term. It showed that the integral term increased convergence rate but would be useless when simulation time is short. A PID-type ILC was studied in [***40***] in the sense of robustness against initial state error by comparing among sets of learning gains. In [***41***], they also studied PID-type and proposed optimal gains for discrete-time SISO LTI systems. The result showed monotonic and increased rate of convergence comparing to non-optimal gains.

2.4.2 Online ILC & Offline-Online ILC

A more effective way to track system's desired output trajectory is to use online ILC scheme since it makes use of current iteration's error information, which is more up to date. In other words, it's a feedback control scheme applied to ILC, which is known and shown to have better performance in terms of error tracking, monotonic convergence speed and robustness as shown in [***29***][***42***]. A P-type online ILC for nonlinear system was proposed in [***43***] by proving error bound in each iteration, implying that the system converges. The paper also adopted the idea of forgetting factor, aiming to overcome uncertainties and disturbances by controlling priority of previous iteration's information. Aside from P-type, a D-type online ILC was proposed in [***44***] with the purpose to eliminate the requirement of known initial state. The paper showed the solution of iterating initial states for each iteration so that one can choose the initial state in the first iteration arbitrarily. The result was impressing that it solved the problem and could precisely track the desired trajectory. PD-type online ILC was proposed in [***45***]. The paper aimed to design a controller that is robust against state uncertainties and measurement disturbances. Not only it could perform the task well but also provided the controller a wide range of learning gains choices, making it very flexible to apply. Another form of PD-type was proposed in [***46***] by applying switching gain thus it's so called Adaptive Switching Gain ILC or ASL-PD ILC. Here the learning gains were not fixed but increased monotonically every iteration, resulting in monotonic convergence at a very fast rate. A PID-type was proposed in [***47***] for electro-hydraulic servo systems, which are very nonlinear, showing versatilities of ILCs.

Offline-online ILCs were also developed. They typically provide a better-quality result from online ones as the effect of an online scheme overwhelms that of an offline one. Presented in [***48***] was the sufficient conditions for a convergent PID open-closed-loop. The conditions were more relaxed than those in other papers and the proportional gain was shown to be independent of the conditions. A PD-type ILC was proposed in [***49***] for discrete-time systems. The paper pointed out that D-type component in any ILC cannot be directly used unless the output is predicted, thus conditions for convergence were given and proved through inductive method. In that paper, P-type feedback was used in closed-loop and D-type ILC was used in open-loop and consequently showed that P component is applicable to improve precision. PD-PD ILCs were proposed in [***42***] by applying PD-type to both the

closed-loop and open-loop, which could improve tracking error, attain a monotonic convergence rate and easily choose learning gains. A more complicated yet powerful idea of ILC was proposed in [***50***]. The paper applied switching gains scheme to a PD-PD ILC by adding switching gains to closed loop part, thus called a SPD-PD ILC. The key reasons of adding switching gains were to increase the convergence rate as most ILCs tend to converge more slowly and to avoid vibration of actuators in the applied systems. Also it was shown to reduce tracking error significantly. A comparison of ILCs was presented in [***29***]. The paper summarized performances of each ILC scheme and showed that SPD-PD outperforms other schemes despite the presence of state uncertainty and disturbances, both repetitive and varying noise. It also illustrated the monotonic convergence rate and small difference between online and offline-online ILCs.

Table 2-3 List of ILC studies

Source	Scheme	Type	Objective/ Application	Remarks
[***28***]	D	Offline	robot learning	---
[***30***]	D	Offline	unknown degree systems	use dummy model to reduce to 1st order
[***33***]	P	Offline	system w/resonance	use aliasing technique
[***31***]	P, D	Offline	optimal design, comparison	---
[***34***]	P	Offline	eliminate bad learning behavior	pseudo-downsampling
[***32***]	P	Offline	monotonic convergence for discrete-time systems	mathematical proof
[***35***]	PD	Offline	discrete-time pneumatic system	---

[***38***]	PD	Offline	variable initial state error (LTI)	use average operator
[***37***]	PD	Offline	continuous nonlinear systems	use causal pair and time shift
[***36***]	PD	Offline	optimal design of monotonic convergence for discrete-time systems	use average operator
[***39***]	PI	Offline	effect of integral part	---
[***40***]	PID	Offline	variable initial state error (LTI)	---
[***41***]	PID	Offline	optimal design for SISO LTI	mathematical proof
[***43***]	P	Online	initial state error for nonlinear systems	use forgetting factor
[***44***]	D	Online	nonlinear systems w/unknown initial states and inputs	use initial state iteration
[***47***]	PID	Online	nonlinear electrohydraulic systems	feedback control added
[***45***]	PD	Online	robustness for nonlinear time varying systems	---
[***46***]	ASL-PD	Online	fast monotonic convergence for nonlinear time varying systems	use switching gain
[***48***]	PID	ON-OFF	convergence for nonlinear time varying systems	mathematical proof

[***49***]	D-P	ON-OFF	discrete-time linear systems w/o U in output equation	D for online P for offline
[***42***]	PD-PD	ON-OFF	robustness for nonlinear time varying systems	---
[***50***]	SPD-PD	ON-OFF	fast monotonic convergence and robustness for nonlinear time varying systems	switching gain added in online part
[***29***]	---	ON-OFF	comparison study	---

2.5 Concluding Remarks

This chapter presented history, basic principles and advantages of quadrotor UAVs, previous research and development in quadrotor UAVs, major control methods for quadrotor UAVs and some basic ideas and studies in ILCs which consequently used in this work. From the literature reviews discussed earlier, it implies that studies in applications for full control of quadrotor UAVs are still in need to exploit their various missions precisely, which is the main objective of this research. Due to major concerns in control system, structural designs will not be taken into account in the following discussion.

From previously studied quadrotor UAV control methods' point of view, as shown earlier, full control, positioning accuracy and robustness are important for survivability and capability of various missions. Basic PID control alone was mostly proved to be inefficient in the presence of uncertainties and disturbances. LQR method was attempted and shown to be useful for path tracking, nevertheless disturbances were not addressed. SDRE and Neural Network Control were tried and performed well but intense computation is required. The Adaptive-Fuzzy method was implemented and proved to be robust against disturbances but only the stabilization problem was addressed. SMCs were used and found controllable but

might cause chattering which leads to control failure in some cases. The backstepping method was also tried and seemed very promising when combined with Integral term in PID control, at the expense of a little more complicated computation.

For ILCs, simplicity, in terms of implementation and plant model, is one of the main characteristics. In addition, they seem to fit well in applications in which tasks are done repeatedly and also robust against uncertainties and disturbances. Many types of ILCs were developed; mainly categorized as Offline, Online and Online-offline. Offline ILCs rely solely on errors occurred in past iterations, yielding very slow convergence rate, if not unstable, in the presence of uncertainties and disturbances and even initial state errors in many cases. Online ILCs depend only on current iteration, forming a feedback control instead of feed-forward as in Offline, dramatically increasing the convergence rate. Also when switching gain is used, monotonic convergence is then certainly assured. However, when feedback gains are chosen too high, the system might response poorly. Online-offline ILCs simply add both type together but results are not much different from Online type ILCs.

Although mainly implemented in industrial robotic manipulator applications, ILCs' advantages of simplicity, self-learning and robustness are appealing for quadrotor control, where automatic positioning and maneuvering are significant. In addition, many missions may be performed repeatedly, making it even more reasonable to exploit the advantages of ILCs for quadrotor UAVs.

Chapter 3 Quadrotor UAV Modeling

In this chapter, detailed discussions of a quadrotor UAV model are presented. The organization of this chapter is as follows: basic concepts and assumptions used in this thesis are introduced in Section 3.1. Kinematics and explanation on how a quadrotor rotate and translate and rotation matrix are presented in Section 3.2. Dynamics of a quadrotor including derivation using Newton-Euler method, the derived nonlinear equations of motion and their detailed analysis are shown in Section 3.3. The quadrotor model parameters used in the simulation are discussed in Section 3.4 and finally concluding remarks of this chapter are included in Section 3.5.

3.1 Concepts and Assumptions

As named, a quadrotor UAV contains four rotors producing thrust upward against its own weight and payloads. Oftentimes, payloads are place at the center of the body and rotors propel using DC motors, with or without gearbox. The rotors are divided into two pairs; one pair rotates in the opposite direction of the other pair in order to balance the angular momentum of the system. Increasing thrust on one side and decreasing on the other side of the same pair results in rotation in the pitch or roll direction and the quadrotor UAV will tend to translate toward the direction that it inclines to. To rotate the yaw angle while maintaining position and altitude, simply equally increase thrust in a pair and decrease thrust in the other. In the past studies, some researches were based on a linearized model, many relied on a nonlinear model yet some even incorporated more complicated aerodynamic and gyroscopic effects. In this thesis, due to aiming to utilize robustness against model uncertainty of ILCs, some aerodynamic effects are neglected and the nonlinear model of the quadrotor is based on these assumptions:

- Structure and rotors are assumedly rigid bodies

- Structure is approximately symmetric and the center of gravity is located at the center of the body, dividing the two arms equally in length

- Thrust and drag moment from a propeller are approximately proportional to the square of the rotor speed. This assumption was proved in real measurements in many researches

- Hub forces or resultant of horizontal forces acting on blade elements are neglected

- Rolling moment or blade flapping, in which the advancing blade generates more thrust than the retreating one in translational flight, is neglected

- Air friction in translational flight and ground effect are neglected

- Actuator and sensor delay time are not taken into account

3.2 Kinematics of Quadrotor UAV

The quadrotor configuration used in this model is illustrated in Fig. 3-1. As seen in the figure, three successive rotations, based on right hand rules, are

- Roll (ϕ) - rotation about X-body axis

- Pitch (θ) - rotation about Y-body axis

- Yaw (ψ) - rotation about Z-body axis

Numbering of rotor shown in Fig. 3-1 also states that rotor 1 and 3 rotate in the -ψ direction while rotor 2 and 4 rotate in the +ψ direction. Considering the Earth frame (E) and the Body frame (B), body orientation can be described in the Earth frame by rotation $\mathcal{R}$ from B to E where $\mathcal{R} \in SO3$ is the rotation matrix from the Body frame to the Earth frame. In some studies, the commonly used $\mathcal{R}_{ZYX}$ are adopted due to orientation of readily available sensors. In this thesis, since there is no sensors consideration involved in the simulation, the rotation

matrix $\mathcal{R}_{zxy}$ is employed instead, which simplifies the equations of motion in X and Y directions as will be shown later.

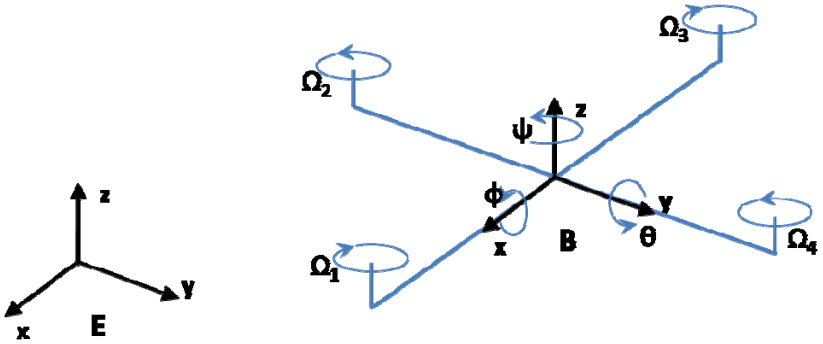

Fig. 3-1 Quadrotor configuration in this thesis

3.2.1 Rotation Matrix

As aforementioned, $\mathcal{R}_{zxy}$is adopted in the model, which means the order of orientation is rotation about the Z-body axis, the X-body axis and the Y-body axis, respectively. Each rotation can be separately described as

- $\mathcal{R}_x(\phi)$ - rotation about X-body axis
- $\mathcal{R}_y(\theta)$ - rotation about Y-body axis
- $\mathcal{R}_z(\psi)$ - rotation about Z-body axis

$$\begin{cases} \mathcal{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi & cos\phi \end{bmatrix} \\ \mathcal{R}_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \\ \mathcal{R}_z(\psi) = \begin{bmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{cases}$$

$\mathcal{R}_{zxy}$ is then the multiplication of all three rotation matrices and described as

$$\begin{aligned} &\mathcal{R}_{zxy} \\ &= \mathcal{R}_z \mathcal{R}_x \mathcal{R}_y \\ &= \begin{bmatrix} cos\psi cos\theta - sin\psi sin\phi sin\theta & -sin\psi cos\phi & cos\psi sin\theta + sin\psi sin\phi cos\theta \\ sin\phi cos\theta + cos\psi sin\phi sin\theta & cos\psi cos\phi & sin\psi sin\theta - cos\psi sin\phi cos\theta \\ -cos\phi sin\theta & sin\phi & cos\phi cos\theta \end{bmatrix} \end{aligned} \quad (3.1)$$

3.2.2 Quadrotor UAV Motion

Z-direction Translation

Moving vertically for a quadrotor UAV is very easy and simple, which is a major advantage of VTOL aircrafts. To translate in the +Z direction, from hovering, simply increase rotor speed equally to each rotor and vice versa in the -Z direction. Changes in rotor angular momentum in each pairs are equal and thus be canceled out. Fig. 3-2 illustrates the concept of a vertical translation.

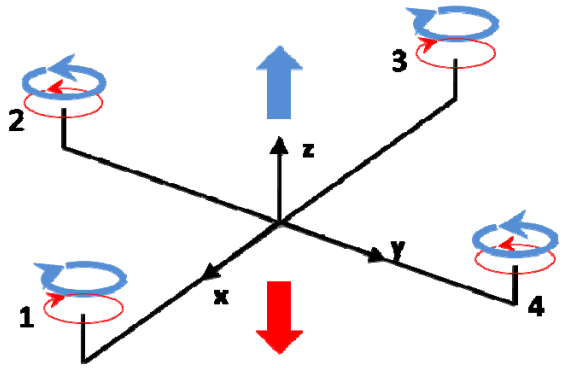

Fig. 3-2 Z-direction Translation

X-direction Translation and Pitching Motion

Since quadrotors are highly coupled, rotation in certain angles results in translation in a direction and that is fundamentally how a quadrotor UAV translates. This is also true to the pitching motion. Starting from hover, increasing rotor speed in rotor 3 and decreasing in rotor 1 while maintaining speeds in rotor 2 and 4 results in rotation in the $+\theta$ direction and translation in the +X direction of Earth frame and vice versa. Note that at this point the quadrotor UAV tilts in a small angle and thrust are approximately equal to weight, thus no translation in the Z direction. This is illustrated in Fig. 3-3.

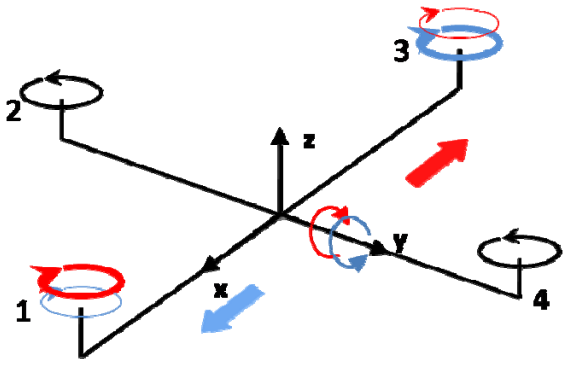

Fig. 3-3 X-direction Translation and Pitch

Y-direction Translation and Rolling Motion

Similar to pitching, rolling is coupled with the Y direction translation. In this case, starting from hover, increasing the rotor speed in rotor 4 and decreasing in rotor 2 while maintaining speeds in rotor 1 and 3 results in rotation in $+\phi$ direction and translation in -Y direction of the Earth frame and vice versa. This is also depicted in Fig. 3-4.

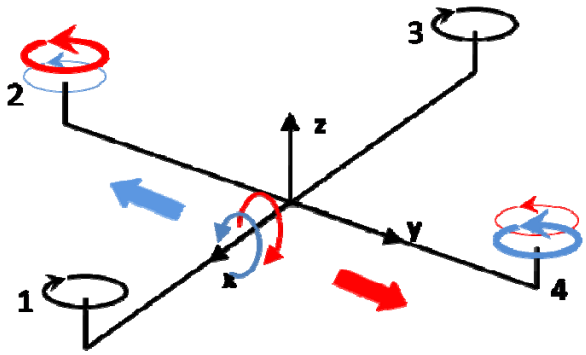

Fig. 3-4 Y-direction Translation and Roll

Yaw Rotation

Yawing is similar to vertical translation in that the motion is not coupled. Instead of cancelling out rotor angular momentum, thrust is balanced to maintain altitude in this case. To perform a pure yaw motion, starting from hover, simply equally increases speed in a pair of the same direction of propeller rotation and decreases in the other. Increasing the speed in rotor 2 and 4 while decreasing speeds in rotor 1 and 3 results in body rotation in $+\psi$ direction and vice versa. See Fig. 3-5 for more details.

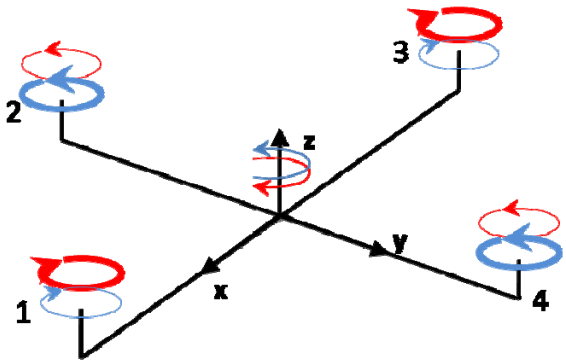

Fig. 3-5 Yaw

3.3 Dynamics of Quadrotor UAV

3.3.1 External Forces and Moments

In this model, the quadrotor UAV experiences several forces and moments from external sources. Free-body diagram of the quadrotor is presented in Fig. 3-6. Vectors marked in red are external forces and moments. All the considered external forces and moments are listed subsequently.

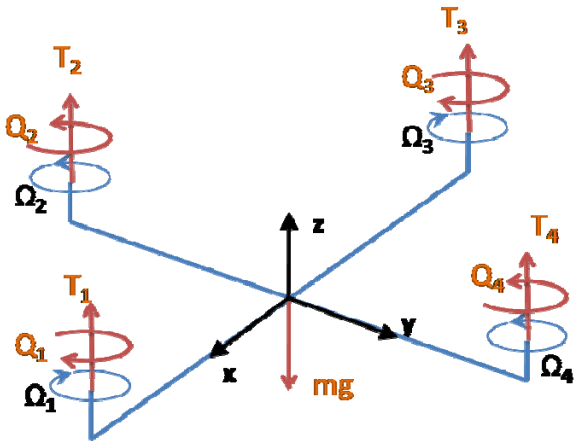

Fig. 3-6 Free-body Diagram of the quadrotor

Thrust/ Lift

Unlike any fixed-wing aircraft, a quadrotor relies on thrust (T) for both lifting its own weight and moving forward or backward. Thrust is generated upward by the four rotors and sometimes tilted toward the intended direction of translation. Thrust generated by each rotor relates to rotor speed via

$$T_i = b\Omega_i^2, \qquad for\ i = 1,2,3,4 \tag{3.2}$$

where b is the thrust coefficient and Ω_i is a rotor speed.

Weight

Weight is one of the most concerns for any aircrafts, no exception for a quadrotor. In many designs, most of overall weight is contributed by batteries. Necessary payloads such as sensors and mission-oriented payloads also add up the total weight. As stated earlier, a quadrotor generates thrust in the opposite direction of weight, hence requires high thrust propellers. Weight can also be reduced with a good design and appropriate materials as done in [***16***].

Drag Moment

Drag moment is caused by drag forces acting on propeller blade elements which are assumed to be equal on both blades but in opposite direction and thus generate pure moment about the rotor axis. Based on the aforementioned assumptions, yawing moment is mainly caused by drag moment. In each rotor, Drag moment is expressed as [***16***]

$$Q_i = d\Omega_i^2, \qquad for\ i = 1,2,3,4 \tag{3.3}$$

where d is the drag coefficient.

Rolling and Pitching Moment due to Actuators

Rolling and pitching moment are mainly generated by thrust from a certain pair of propellers. In this design, according to Fig. 3-6, rolling moment can be achieve by equally increasing and decreasing thrust in rotor 2 and 4 while pitching moment from rotor 1 and 3. These moments are not shown in Fig. 3-6 because they are caused by thrust, which is already included in the free-body diagram. Both moments relate to thrust as

$$\begin{cases} Rolling\ moment:\ \tau_\phi = l(-T_2 + T_4) \\ Pitching\ moment: \tau_\theta = \ l(T_1 - T_3) \end{cases} \tag{3.4}$$

where l is the quadrotor arm length.

3.3.2 Newton-Euler Formalism

To establish a nonlinear dynamic model for the quadrotor, in this thesis, the Newton-Euler method is utilized for both the main body and rotors. A general form of Newton-Euler equation is expressed as [***51***]

$$\begin{bmatrix} m\boldsymbol{I_{3\times 3}} & 0 \\ 0 & \boldsymbol{I} \end{bmatrix} \begin{Bmatrix} \dot{\boldsymbol{V}} \\ \dot{\boldsymbol{\omega}} \end{Bmatrix} + \begin{Bmatrix} \boldsymbol{\omega} \times m\boldsymbol{V} \\ \boldsymbol{\omega} \times \boldsymbol{I\omega} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{F} \\ \boldsymbol{\tau} \end{Bmatrix} \tag{3.5}$$

Note that eq. (3.5) is a general form of equations of motion which can be applied in any position in the coordinate system. In this case, the point of interest is the center of mass of the quadrotor and considering the body frame (B), eq.(3.5) reduces to

$$\begin{bmatrix} m\boldsymbol{I_{3\times 3}} & 0 \\ 0 & \boldsymbol{I} \end{bmatrix} \begin{Bmatrix} \dot{\boldsymbol{V}} \\ \dot{\boldsymbol{\omega}} \end{Bmatrix} + \begin{Bmatrix} \boldsymbol{0} \\ \boldsymbol{\omega} \times \boldsymbol{I\omega} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{F} \\ \boldsymbol{\tau} \end{Bmatrix} \tag{3.6}$$

3.3.3 Body's Dynamics

Considering the main body of the quadrotor UAV (Fig. 3-6) and eq.(3.6), one can describe translational dynamics of the quadrotor in the body frame (B) as

$$m\begin{Bmatrix}\ddot{X}\\ \ddot{Y}\\ \ddot{Z}\end{Bmatrix}_B = \begin{Bmatrix}0\\0\\ \sum_{i=1}^{4} b\Omega_i^2\end{Bmatrix} - \mathcal{R}_{zxy}^{-1}\begin{Bmatrix}0\\0\\mg\end{Bmatrix}$$

which can then be described in the earth frame (E) through eq. (3.1) as

$$\begin{aligned} m\begin{Bmatrix}\ddot{X}\\ \ddot{Y}\\ \ddot{Z}\end{Bmatrix}_E &= \mathcal{R}_{zxy} m\begin{Bmatrix}\ddot{X}\\ \ddot{Y}\\ \ddot{Z}\end{Bmatrix}_B \\ &= \mathcal{R}_{zxy}\begin{Bmatrix}0\\0\\ \sum_{i=1}^{4} b\Omega_i^2 - \mathcal{R}_{zxy}^{-1} mg\end{Bmatrix} \end{aligned}$$

$$m\begin{Bmatrix}\ddot{X}\\ \ddot{Y}\\ \ddot{Z}\end{Bmatrix}_E = \begin{Bmatrix} -sin\theta cos\phi \sum_{i=1}^{4} b\Omega_i^2 \\ sin\phi \sum_{i=1}^{4} b\Omega_i^2 \\ -mg + cos\theta cos\phi \sum_{i=1}^{4} b\Omega_i^2 \end{Bmatrix} \tag{3.7}$$

From eq.(3.6), the main body rotational dynamics can be described in the body frame (B) as

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}\begin{Bmatrix}\ddot{\phi}\\ \ddot{\theta}\\ \ddot{\psi}\end{Bmatrix} + \begin{Bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{Bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}\begin{Bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{Bmatrix} = \begin{Bmatrix}\tau_x\\ \tau_y\\ \tau_z\end{Bmatrix}$$

$$\begin{Bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\psi} \end{Bmatrix} + \begin{Bmatrix} \dot{\theta}\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\phi}\dot{\psi}(I_{xx} - I_{zz}) \\ \dot{\phi}\dot{\theta}(I_{yy} - I_{xx}) \end{Bmatrix} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_e + \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_r \tag{3.8}$$

where subscripts e and r refer to moments due to external forces, which ultimately caused by thrust and drag from rotors, and moments due to rotor gyro effect, respectively.

3.3.4 Rotor's Dynamics

Similarly, the dynamics of each rotor can be described using eq.(3.6) by considering coordinate system of each rotor, which is simply in the same plane as the body frame for X and Y axes while Z axis coincides with rotation of the rotor. Note that rotors do not translate relative to the body and, as aforementioned, hub forces and rolling moments are neglected, hence translational dynamics of the rotors is negligible. Considering rotational dynamics of each rotor in the form of Newton-Euler:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}_{r,i} \begin{Bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{Bmatrix}_{r,i} + \begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix}_{r,i} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}_{r,i} \begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix}_{r,i} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i}$$

$$\begin{Bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\psi} \end{Bmatrix}_{r,i} + \begin{Bmatrix} \dot{\theta}\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\phi}\dot{\psi}(I_{xx} - I_{zz}) \\ \dot{\phi}\dot{\theta}(I_{yy} - I_{xx}) \end{Bmatrix}_{r,i} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i}$$

for $i = 1,2,3,4$ and denotes i^{th} rotor. Since the rotors always rotate about their Z-axes at the rate of Ω with the moment of inertia about Z-axis of J_r and have very low masses, I_{xx} and I_{yy} can then be omitted and the dynamics of each rotor reduces to

$$\begin{Bmatrix} 0 \\ 0 \\ J_r\dot{\Omega} \end{Bmatrix}_i + \begin{Bmatrix} \dot{\theta}\Omega J_r \\ -\dot{\phi}\Omega J_r \\ 0 \end{Bmatrix}_i = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i} \tag{3.9}$$

Note that eq. (3.9) is a function of rotor speed Ω. Since rotor 1 and 3 rotate in the opposite direction of rotor 2 and 4, one can define the total rotor speed as:

$$\Omega_r = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 \tag{3.10}$$

From eq.(3.9) and (3.10), the total moment due to gyro effect from all rotors can be expressed as:

$$\begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_r = \sum_{i=1}^{4} \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i}$$

$$= \sum_{i=1}^{4} \left[\begin{Bmatrix} 0 \\ 0 \\ J_r\dot{\Omega} \end{Bmatrix}_i + \begin{Bmatrix} \dot{\theta}\Omega J_r \\ -\dot{\phi}\Omega J_r \\ 0 \end{Bmatrix}_i \right]$$

$$\begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_r = \begin{Bmatrix} \dot{\theta}J_r\Omega_r \\ -\dot{\phi}J_r\Omega_r \\ J_r\dot{\Omega}_r \end{Bmatrix} \tag{3.11}$$

3.3.5 Equations of Motion

Now that all necessary dynamics of the entire model has been established, one can write the complete equations of motion of the quadrotor. Combining eq. (3.2), (3.3), (3.4), (3.7), (3.8) and (3.11) yields

$$\begin{cases} m\ddot{X} &= -sin\theta cos\phi \sum_{i=1}^{4}[b\Omega_i^2] \\ m\ddot{Y} &= sin\phi \sum_{i=1}^{4}[b\Omega_i^2] \\ m\ddot{Z} &= -mg + cos\theta cos\phi \sum_{i=1}^{4}[b\Omega_i^2] \\ I_{xx}\ddot{\phi} &= \dot{\theta}\dot{\psi}\left(I_{yy} - I_{zz}\right) - \dot{\theta}\Omega_{\mathrm{r}}J_r + lb(-\Omega_2^2 + \Omega_4^2) \\ I_{yy}\ddot{\theta} &= \dot{\phi}\dot{\psi}(I_{zz} - I_{xx}) + \dot{\phi}\Omega_{\mathrm{r}}J_r + lb(\Omega_1^2 - \Omega_3^2) \\ I_{zz}\ddot{\psi} &= \dot{\theta}\dot{\phi}\left(I_{xx} - I_{yy}\right) + \dot{\Omega}_r J_r + \sum_{i=1}^{4}\left[(-1)^i d\Omega_i^2\right] \end{cases} \quad (3.12)$$

As seen from all the previous derivations, various effects from various sources affect the dynamic model. Summary of all considered internal and external forces and moments, their effects and sources is shown in Table 3-1.

Table 3-1 Summary of effects on the quadrotor dynamic model

Forces/Moments	Source	Effect	Terms
Thrust/Lift	Rotor rotation	Aerodynamic	$\sum_{i=1}^{4}[b\Omega_i^2]$
Weight	Center of mass	Gravitational	mg
Drag moment	Rotor rotation	Aerodynamic	$\sum_{i=1}^{4}\left[(-1)^i d\Omega_i^2\right]$
Rotating moment	rotor rotation/arm length	Aerodynamic	$\begin{cases} lb(-\Omega_2^2 + \Omega_4^2) \\ lb(\Omega_1^2 - \Omega_3^2) \end{cases}$
Body coupling moment	Body rotation	Gyroscopic	$\begin{cases} \dot{\theta}\dot{\psi}\left(I_{yy} - I_{zz}\right) \\ \dot{\phi}\dot{\psi}(I_{zz} - I_{xx}) \\ \dot{\theta}\dot{\phi}\left(I_{xx} - I_{yy}\right) \end{cases}$

Rotor coupling moment	Rotor and body rotation	Gyroscopic/Aerodynamic	$\begin{cases} \dot{\theta}\Omega_r J_r \\ \dot{\phi}\Omega_r J_r \\ \dot{\Omega}_r J_r \end{cases}$

3.4 Model Parameters

In this work, the chosen parameters for simulation solely rely on the study conducted in [***16***]. The quadrotor developed in that paper (OS4) was carefully and well designed, resulting in a light-weight model, relatively high thrust and a reasonably compact size. In addition, the quadrotor information was quite accessible and much detailed. Hence the OS4 model seems very promising to use as a model for the simulations in this research. Necessary parameters of the quadrotor are presented in Table 3-2. Also note that I_{xx} and I_{yy} are approximately identical, implying X-Y plane symmetry, which is in accordance with the assumptions in this work.

Table 3-2 Parameters of a quadrotor UAV

Parameters	**Description**	**Value**
l	quadrotor arm length	$0.232\ m$
b	rotor thrust coefficient	$3.13 \times 10^{-5}\ N \cdot s^2$
d	rotor drag coefficient	$7.5 \times 10^{-7} m\ \cdot s^2$
m	total quadrotor mass	$0.52\ kg$
I_{xx}	moment of inertia about X axis	$6.228 \times 10^{-3} kg \cdot m^2$
I_{yy}	moment of inertia about Y axis	$6.225 \times 10^{-3} kg \cdot m^2$
I_{zz}	moment of inertia about Z axis	$1.121 \times 10^{-2} kg \cdot m^2$

J_r	rotor inertia	$6 \times 10^{-5} kg \cdot m^2$
Ω_{max}	maximum rotor speed	$279\, rad/s$

3.5 Concluding Remarks

This chapter discussed about quadrotor modeling which includes basic and detailed concepts of quadrotor UAVs, rotation matrix, kinemics and dynamics and their derivations along with model parameters chosen in this thesis. The quadrotor UAV model in this thesis is based on basic aerodynamic and gyroscopic effects in order to make use of robustness of ILCs while considering it as a nonlinear system for the sake of accuracy. The rotation matrix adopted in this thesis is $\mathcal{R}_{zxy}$ due to the fact that it simplifies the dynamic model. Analysis on external forces and moments were intensely discussed and classified for clarification. The dynamic model derived in this chapter will be carried on and modified into state-space form in Chapter 4 and model parameters will be used in the simulations in Chapter 5.

Chapter 4 Control System Design

This chapter mainly presents detailed features of ILCs, including some derivations, and their utilization on the quadrotor model as done in this work. Firstly, detailed concepts, basic assumptions and uses of Offline, Online and Online-offline ILCs are presented in Section 4.1. Convergence analyses of PD-PD and SPD-PD ILCs are shown in Section 4.2. Comparison studies cited from [***52***] are partly illustrates in Section 4.3. Implementation of ILCs on quadrotor model is described in Section 4.4. Trajectory generation used by ILCs in this work is shown in Section 4.5, ending with remarks for this chapter in Section 4.6.

4.1 ILCs: Concepts and Principles

4.1.1 Problem Description

As stated earlier, Iterative Learning Control (ILC) can improve tracking performance each time a specific task is performed repeatedly. This self-learning characteristic also helps make it an easy and robust way to control a plant where the accurate detailed model is not available or exposed to relatively large model uncertainties and disturbances, which in most cases are nonlinear systems. To gain more insight in ILCs, consider a nonlinear time-varying system in the general form of

$$\begin{cases} \dot{x}_k(t) &= f(x_k(t),t) + B(t)u_k(t) + \eta_k(t) \\ y_k(t) &= C(t)x_k(t) + \xi_k(t) \end{cases} \tag{4.1}$$

where subscription k denotes iteration number, $x \in \Re^n, y \in \Re^r$ and $u \in \Re^m$ are the state, control input and output of the system, respectively. The function $\eta_k(t) \in \Re^n$ represents both the deterministic and random disturbances of the system, and $\xi_k(t) \in \Re^m$ is the measurement noise of the system. According to the aforementioned assumptions, it concludes that $f(x_k(t),t) \in \Re^n, B(t) \in \Re^{n\times r}$ and $C(t) \in \Re^{m\times n}$.

To control the system mentioned above, one can use ILCs. A general form of control law of ILCs implemented to the system can be written as

$$u_{k+1} = u_k + K_{pon}e_{k+1}(t) + K_{ion}\int_0^t e_{k+1}(t)dt + K_{don}\frac{de_{k+!}(t)}{dt} + K_{poff}e_k(t) + K_{ioff}\int_0^t e_k(t)dt + K_{doff}\frac{de_k(t)}{dt} \quad (4.2)$$

where $e(t) = y_d(t) - y(t)$ and $\dot{e}(t) = \dot{y}_d - \dot{y}$. $K_p, K_i\ and\ K_d$ are PID control gains and subscription on and off represent online(feedback, using errors from current iteration) and offline(feed-forward, using errors from previous iteration) learning gains, respectively. In most studies, integral part is not utilized as ILCs themselves have integration characteristic and thus makes not much improvement comparing to those without integral part. For ILCs, convergence condition and convergence rate are two main concerns and vary with types of ILC. Regardless of variations in each type, ILCs can be mainly divided into offline, online and online-offline.

Before moving on to detailed information on ILCs, it is safer to first establish norms used in this thesis for the sake of convergence analysis as follow:

$$\|f\| = \max_{l\le i\le n}|f_i|$$

$$\|h(t)\|_\lambda = \sup_{t\in[0,T]} e^{-\lambda t}\|h(t)\|, \lambda > 0$$

$$\|M(t)\| = \max_{l\le i\le m}\left(\sum_{j=1}^{n}\left|m_{i,j}\right|\right)$$

where $f = [f_1, \dots, f_n]^T$ is a vector, $M = \left[m_{i,j}\right] \in \Re^{m\times n}$ is a matrix, and $h(t)(t \in [0,T])$ is a real function where T is the time duration.

To restrict discussions, the following assumptions are introduced.

(A1) The desired trajectory $y_d(t)$ is first-order continuity for $t \in [0,T]$.

(A2) The control input matrix $B(t)$ is first-order continuity for $t \in [0,T]$.

(A3) The function $f(x(t),t)$ is globally uniformly Lipschitz in x for $t \in [0,T]$. That means $\left\| f(x_{k+1}(t),t) - f(x_k(t),t) \right\| \le c_f \left\| x_{k+1}(t) - x_k(t) \right\|$ where k is the iteration number and $c_f > 0$ is a constant.

(A4) The uncertainty and disturbance terms $\eta_k(t)$ and $\xi_k(t)$ are bounded as follows, $\forall\, t \in [0,T]$ and $\forall\, k$, $\left\| \eta_k(t) \right\| \le b_\eta$ and $\left\| \xi_i(t) \right\| \le b_\xi$

The assumptions can be explained as follows. A1 is a fundamental requirement for the designed control as the derivation of the output as a feedback signal is needed; A2 is for the purpose of the continuity of the controlled system; A3 is a basic requirement for the nonlinear system (1). In fact, the function $f(x,t)$ may be structurally unknown; A4 restricts the disturbance and noise to be bounded that is a reasonable assumption for real applications.

4.1.2 Types of ILCs

Offline ILCs

Traditional ILCs such as the one proposed in[***28***] are considered offline, where only errors from previous iteration have effects on the control law. In the case that only proportional part is utilized (P-ILCs), the control law reduces to

$$u_{k+1} = u_k(t) + K_{poff} e_k(t) \tag{4.3}$$

and formed up in the similar way for PD, PI, PID and so on. The convergence condition of P-type offline ILC can be derived as[***53***]

$$\rho_{poff} = \left\| I - K_{poff} CB \right\| < 1 \tag{4.4}$$

Whenever derivative part is involved, including D-ILC, PD-ILC and PID-ILC, the convergence rate becomes[***28***]

$$\rho_{doff} = \left\| I - K_{doff} CB \right\| < 1 \tag{4.5}$$

The two aforementioned convergence conditions imply that offline learning gains are upper bounded, which means only a certain range of control gain is valid, reducing choice of designs. Furthermore, because this type of ILCs relies only on past iterations, their convergence rate thus seems to be much slower comparing to other types and in some cases cannot reject disturbances. Oftentimes, they also experience 'bad-good-bad' learning problem, where errors decrease and increase repeatedly, taking much more iterations to reach acceptable range of error. In many real experiments, D type tends to perform better than P type in terms of convergence rate. Some papers, however, combined both together to achieve a better result. Although not used in this thesis, derivation of convergence conditions for offline ILCs are presented in section 4.1.2. A simplified control diagram of offline ILCs is depicted in Fig. 4-1.

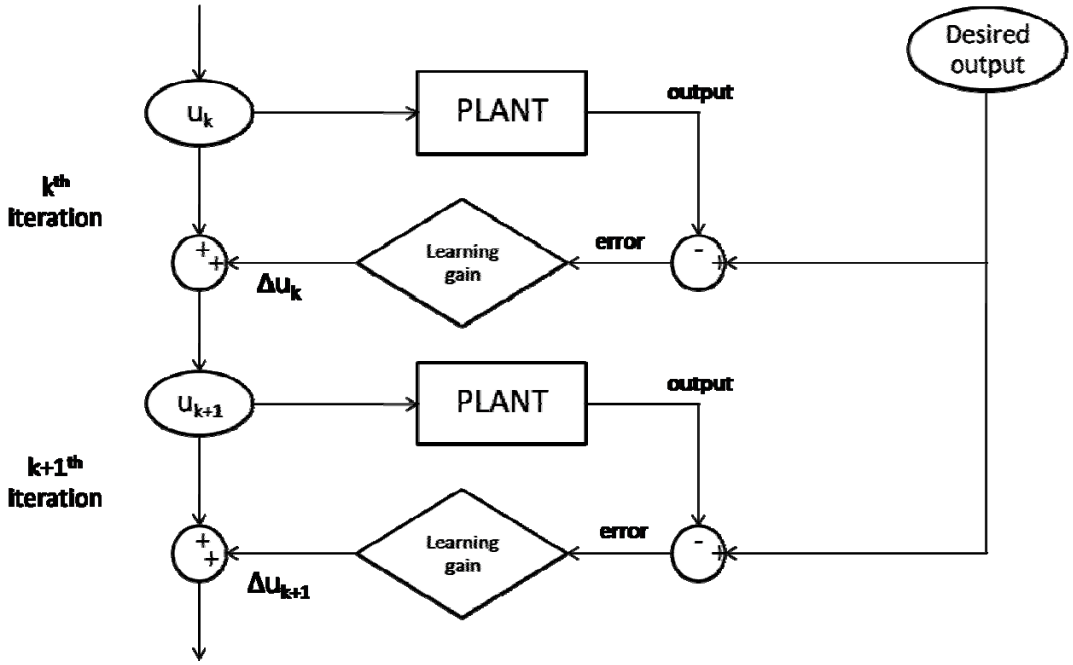

Fig. 4-1 General Offline ILC control diagram

Online ILCs

In contrary to offline ILCs, online ILCs rely on current iteration errors, making it simple like classic PID control while maintaining the self-learning feature. The convergence conditions are [***54***],[***55***]

$$\begin{aligned} \rho_{pon} &= \left\| \left(I + K_{pon} CB\right)^{-1} \right\| < 1 \quad \text{(P-type only)} \\ \rho_{pon} &= \| (I + K_{don} CB)^{-1} \| < 1 \quad \text{(D-type involved)} \end{aligned} \tag{4.6}$$

It is noticeable that, for online ILCs, the learning gains are unbounded; making it very flexible to choose control gains and the larger control gains, the faster the system converges. In the same way as offline ILCs, using D-type yield faster convergence rate than P-type alone plus P-type cannot reject disturbances well as shown in comparison study in [***52***]. However, relying only on derivative part might not be a wise choice when high noise level appears, thus combining both type seems to be a better option that they compensate each other drawbacks. Derivation of online PD ILCs convergence conditions are detailed subsequently. General control diagram of online ILCs can be demonstrated as seen in Fig. 4-2

In addition to ordinary online ILCs, one might utilize adaptive switching learning control, which dramatically improve convergence rate by increasing online learning gains every cycle, in this case, as a function of iteration index. The convergence conditions then become[***56***]

$$\begin{aligned} &\left\| \left(I + K_{pon}(0) CB\right)^{-1} \right\| < 1 \qquad \text{(P-type only)} \\ &\| (I + K_{don}(0) CB)^{-1} \| < 1 \qquad \text{(D-type involved)} \end{aligned} \tag{4.7}$$

Similar to offline ILCs presented earlier, proof of convergence is presented in section 4.1.2 which covers all types of ILCs mentioned in this thesis.

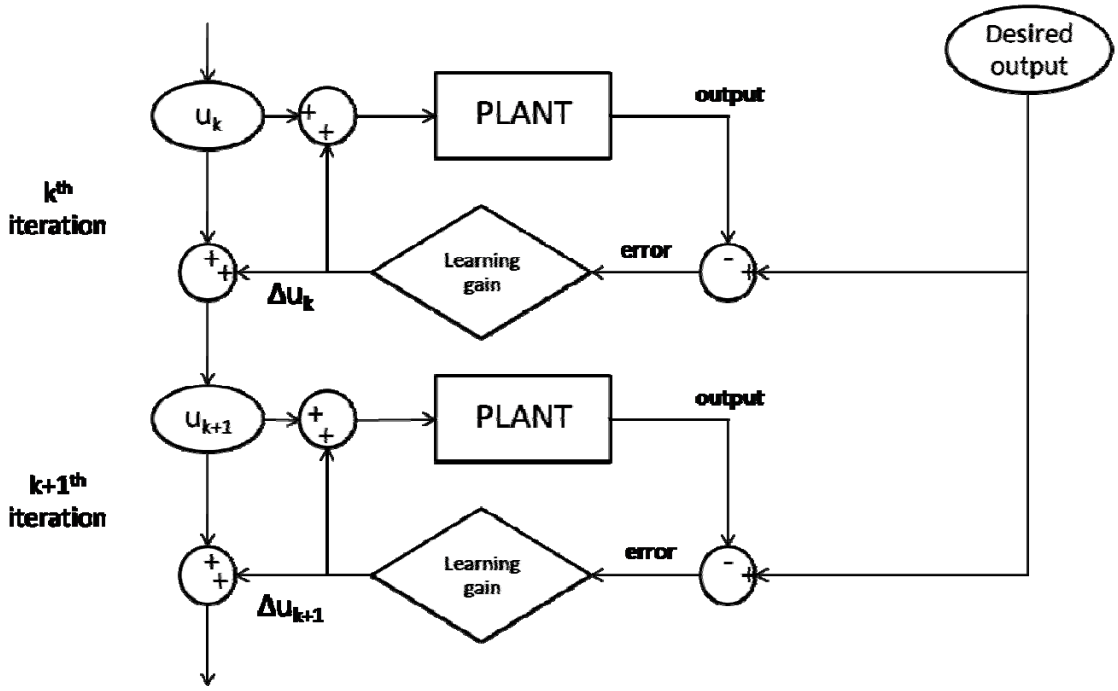

Fig. 4-2 General Online ILC control diagram

Online-offline ILCs

When online and offline ILCs are used simultaneously, it simply turns into online-offline ILCs. Convergence conditions are then simply combination of both types and can be expressed as [***42***]

$$\begin{cases}\rho_{poff} < 1 \\ \rho_{pon} < 1\end{cases} \quad \text{(P-type only)}$$

or (4.8)

$$\begin{cases}\rho_{doff} < 1 \\ \rho_{don} < 1\end{cases} \quad \text{(D-type involved)}$$

4.2 Convergence Analysis

The convergence analyses shown subsequently are of PD online-offline ILCs (PD-PD) and switching gain PD online-offline ILCs (SPD-PD), which are also shown in [***42***] and [***56***]. Both of them cover all types of ILCs discussed in previous section. In the following analyses, the λ –norm is used to examine the convergence of the tracking error. First of all, a relation between norm and λ –norm is represented by Lemma 1.

Lemma 1: Suppose that $x(t) = [x_1(t), x_2(t), \cdots, x_n(t)]^T$ is defined in $t \in [0, T]$. Then

$$\left(\int_0^t \|x(\tau)\| d\tau \right) e^{-\lambda t} \leq \frac{1}{\lambda} \|x(t)\|_\lambda \qquad \text{for } \lambda > 0 \tag{4.9}$$

Proof:

$$\left(\int_0^t \|x(\tau)\| d\tau \right) e^{-\lambda t} = \int_0^t \|x(\tau)\| e^{-\lambda \tau} e^{-\lambda (t-\tau)} d\tau$$

$$\leq \sup_{t \in [0,T]} \left\{ \|x(t)\| e^{-\lambda t} \right\} \int_0^t e^{-\lambda (t-\tau)} d\tau$$

$$\leq \sup_{t \in [0,T]} \left\{ \|x(t)\| e^{-\lambda t} \right\} \frac{1 - e^{-\lambda t}}{\lambda}$$

$$\leq \frac{1}{\lambda} \|x(t)\|_\lambda$$

[End of proof].

4.2.1 PD-PD Convergence Analysis

PD-PD control law is expressed as:

$$u_{k+1}(t) = u_k(t) + K_{p1}(t)e_{k+1}(t) + K_{d1}(t)\dot{e}_{k+1}(t) + K_{p2}(t)e_k(t) + K_{d2}(t)\dot{e}_k(t) \qquad (4.10)$$

Some notations are introduced first for the briefness of discussions:

$$B_{d1} = \max_{t\in[0,T]} \left\| B(t)K_{d1}(t) \right\|, \quad B_{d2} = \max_{t\in[0,T]} \left\| B(t)K_{d2}(t) \right\|$$

$$B_{pd1} = \max_{t\in[0,T]} \left\| B(t)K_{p1}(t) - \frac{d\left[B(t)K_{d1}(t)\right]}{dt} \right\|, \quad B_{pd2} = \max_{t\in[0,T]} \left\| B(t)K_{p2}(t) - \frac{d\left[B(t)K_{d2}(t)\right]}{dt} \right\|$$

$$\alpha = 1 - {c_f}/{\lambda}\ , \ B_c = \max_{t\in[0,T]} \left\| C(t) \right\|, \ K_{x\lambda 1} = \frac{\lambda B_{d1} + B_{pd1}}{\lambda - c_f}, \ K_{x\lambda 2} = \frac{\lambda B_{d2} + B_{pd2}}{\lambda - c_f}$$

$$\rho_1 = \max_{t\in[0,T]} \left\| \left(I_m + C(t)B(t)K_{d1}(t)\right)^{-1} \right\|, \ \rho_2 = \max_{t\in[0,T]} \left\| I_m - C(t)B(t)K_{d2}(t) \right\|$$

$$\rho = \frac{\rho_1}{1 - \frac{\rho_1}{\lambda}\left(c_f K_{x\lambda 1} + B_{pd1}\right)B_c}, \quad \beta = \left(\rho_2 + \frac{B_c B_{pd2} + B_c c_f K_{x\lambda 2}}{\lambda} \right)$$

For each iteration, the repeatability of the initial state setting is satisfied within an admissible deviation level, i.e.,

$$\left\| x_k(0) - x_0(0) \right\| \le \varepsilon_x \quad \text{for} \quad k = 1, 2, \ldots \qquad (4.11)$$

where ε_x is a small positive constant that represents the acceptable accuracy of the designed state vector, and $x_0(0)$ represents the desired initial state value. From eqn. (4.1), we have

$$y_d(t) = C(t)x_0(t) \tag{4.12}$$

Remark 1: The constraint in (4.11) is more flexible than the requirement of zero initialization error in most ILC studies. It can be seen that the zero initialization error is only a special case where $\varepsilon_x = 0$.

From (4.1), if the uncertainty and disturbance are bounded, then the following holds:

$$\|y_d(0) - y_k(0)\| \le \|C(0)\|\varepsilon_x + b_\xi \le B_c\varepsilon_x + b_\xi \quad \text{for} \quad k = 1,2,... \tag{4.13}$$

Define:

$$\phi = 2b_\xi + 2b_\eta T + \left(2 + B_c\left(B_{d1} + B_{d2}\right)\right)\varepsilon_x \tag{4.14}$$

Based on the above preparations, we have the following theorem.

Theorem: For the nonlinear time-varying system (4.1), if the PD-PD ILC law (4.10) is used and the initial state in each iteration follows (4.11), then the final output tracking error is bounded given by

$$\lim_{k\to\infty}\|e_k(t)\|_\lambda \le \frac{\rho B_c\left(1 + c_f/\left(\lambda\alpha\right)\right)\phi + 2\rho b_\xi}{1-\rho\beta} \quad \text{for all } t \in [0,T] \tag{4.15}$$

Provided the control gain $K_{d1}(t)$ and the learning gain $K_{d2}(t)$ are selected such that $I_m + C(t)B(t)K_{d1}(t)$ is non-singular, and

$$\max_{t\in[0,T]}\left\|\left(I+C(t)B(t)K_{d1}(t)\right)^{-1}\right\|\max_{t\in[0,T]}\left\|I-C(t)B(t)K_{d2}(t)\right\|<1 \tag{4.16}$$

For the k^{th} iteration, the state vector can be obtained from (4.1)

$$x_k(t)=x_k(0)+\int_0^t\left(f\left(x_k(\tau),\tau\right)+B(\tau)u_k(\tau)\right)d\tau+\int_0^t\eta_k(t)d\tau \tag{4.17}$$

For the k+1th iteration, from (4.1) we have

$$x_{k+1}(t)=x_{k+1}(0)+\int_0^t\left(f\left(x_{k+1}(\tau),\tau\right)+B(\tau)u_{k+1}(\tau)\right)d\tau+\int_0^t\eta_{k+1}(t)d\tau \tag{4.18}$$

Submitting control law (4.10) into (4.18) yields:

$$\begin{aligned}x_{k+1}(t)=x_{k+1}(0)&+\int_0^t f\left(x_{k+1}(\tau),\tau\right)d\tau+\int_0^t B(\tau)u_k(\tau)d\tau+\int_0^t\eta_{k+1}(\tau)d\tau\\&+\int_0^t B(\tau)\left[K_{p1}(\tau)e_{k+1}(\tau)+K_{d1}(\tau)\dot{e}_{k+1}(\tau)+K_{p2}(\tau)e_k(\tau)+K_{d2}(\tau)\dot{e}_k(\tau)\right]d\tau\end{aligned} \tag{4.19}$$

Submitting (4.19) from (4.17), we get

$$\begin{aligned}x_{k+1}(t)-x_k(t)=x_{k+1}(0)-x_k(0)&+\int_0^t\left[f\left(x_{k+1}(\tau),\tau\right)-f\left(x_k(\tau),\tau\right)\right]d\tau+\int_0^t\left[\eta_{k+1}(\tau)-\eta_k(\tau)\right]d\tau\\&+\int_0^t B(\tau)\left[K_{p1}(\tau)e_{k+1}(\tau)+K_{d1}(\tau)\dot{e}_{k+1}(\tau)+K_{p2}(\tau)e_k(\tau)+K_{d2}(\tau)\dot{e}_k(\tau)\right]d\tau\end{aligned} \tag{4.20}$$

Using the partial integration equation, we have

$$\begin{cases}\int_0^t B(\tau)K_{d1}(\tau)\dot{e}_{k+1}(\tau)d\tau=B(t)K_{d1}(t)e_{k+1}(t)-B(0)K_{d1}(0)e_{k+1}(0)-\int_0^t\dfrac{d\left(B(\tau)K_{d1}(\tau)\right)}{d\tau}e_{k+1}(\tau)d\tau\\\int_0^t B(\tau)K_{d2}(\tau)\dot{e}_k(\tau)d\tau=B(t)K_{d2}(t)e_k(t)-B(0)K_{d2}(0)e_k(0)-\int_0^t\dfrac{d\left(B(\tau)K_{d2}(\tau)\right)}{d\tau}e_k(\tau)d\tau\end{cases}$$

(4.21)

Submitting (4.21) into (4.20), we get

$$\begin{aligned} x_{k+1}(t)-x_k(t) &= x_{k+1}(0)-x_k(0)-B(0)K_{d1}(0)e_{k+1}(0)-B(0)K_{d2}(0)e_k(0) \\ &+B(t)K_{d1}(t)e_{k+1}(t)+B(t)K_{d2}(t)e_k(t) \\ &+\int_0^t\left[f\left(x_{k+1}(\tau),\tau\right)-f\left(x_k(\tau),\tau\right)\right]d\tau+\int_0^t\left[\eta_{k+1}(\tau)-\eta_k(\tau)\right]d\tau \\ &+\int_0^t\left[B(\tau)K_{p1}(\tau)-\frac{d\left(B(\tau)K_{d1}(\tau)\right)}{d\tau}\right]e_{k+1}(\tau)d\tau \\ &+\int_0^t\left[B(\tau)K_{p2}(\tau)-\frac{d\left(B(\tau)K_{d2}(\tau)\right)}{d\tau}\right]e_k(\tau)d\tau \end{aligned} \tag{4.22}$$

From (4.11), we have

$$\left\|x_{k+1}(0)-x_k(0)\right\|\le 2\varepsilon_x \tag{4.23}$$

From assumption A3, we can get

$$\left\|\int_0^t\left[\eta_{k+1}(\tau)-\eta_k(\tau)\right]d\tau\right\|\le\left\|\int_0^t\eta_{k+1}(\tau)d\tau\right\|+\left\|\int_0^t\eta_k(\tau)d\tau\right\|\le 2b_\eta t\le 2b_\eta T$$

where T is the integral interval. Applying assumptions A3 and A4, and submitting (4.13), (4.23) into (4.22), we get

$$
\begin{aligned}
\left\|x_{k+1}(t)-x_k(t)\right\| \le\ & 2b_\xi + 2b_\eta T + \left(1+\left\|B(0)K_{d1}(0)C(0)\right\|\right)\varepsilon_x + \left(1+\left\|B(0)K_{d2}(0)C(0)\right\|\right)\varepsilon_x \\
& + \left\|B(t)K_{d1}(t)\right\|\left\|e_{k+1}(t)\right\| + \left\|B(t)K_{d2}(t)\right\|\left\|e_k(t)\right\| \\
& + \int_0^t \left\| \left(B(\tau)K_{p1}(\tau) - \frac{d\left(B(\tau)K_{d1}(\tau)\right)}{d\tau} \right) \right\| \left\|e_{k+1}(\tau)\right\| d\tau \\
& + \int_0^t \left\| \left(B(\tau)K_{p2}(\tau) - \frac{d\left(B(\tau)K_{d2}(\tau)\right)}{d\tau} \right) \right\| \left\|e_k(\tau)\right\| d\tau \\
& + c_f \int_0^t \left\|x_{k+1}(\tau)-x_k(\tau)\right\| d\tau \\
\le\ & \phi + B_{d1}\left\|e_{k+1}(t)\right\| + B_{d2}\left\|e_k(t)\right\| + B_{pd1}\int_0^t \left\|e_{k+1}(\tau)\right\| d\tau \\
& + B_{pd2}\int_0^t \left\|e_k(\tau)\right\| d\tau + c_f \int_0^t \left\|x_{k+1}(\tau)-x_k(\tau)\right\| d\tau
\end{aligned}
\tag{4.24}
$$

Multiplying (4.24) by $e^{-\lambda t}$ where $\lambda > 1$ and $\lambda > c_f$, we have

$$
\begin{aligned}
\left\|x_{k+1}(t)-x_k(t)\right\|_\lambda \le\ & e^{-\lambda t}\phi + B_{d1}\left\|e_{k+1}(t)\right\|_\lambda + + B_{d2}\left\|e_k(t)\right\|_\lambda \\
& + B_{pd1}e^{-\lambda t}\int_0^t \left\|e_{k+1}(\tau)\right\| d\tau + B_{pd2}e^{-\lambda t}\int_0^t \left\|e_k(\tau)\right\| d\tau \\
& + c_f e^{-\lambda t}\int_0^t \left\|x_{k+1}(\tau)-x_k(\tau)\right\| d\tau
\end{aligned}
\tag{4.25}
$$

Applying **Lemma 1** to (4.25), we can get the following inequality

$$
\left(1-\frac{c_f}{\lambda}\right)\left\|x_{k+1}(t)-x_k(t)\right\|_\lambda \le \phi + \left(B_{d1}+\frac{B_{pd1}}{\lambda}\right)\left\|e_{k+1}(\tau)\right\|_\lambda + \left(B_{d2}+\frac{B_{pd2}}{\lambda}\right)\left\|e_k(\tau)\right\|_\lambda \tag{4.26}
$$

According to the notions introduced in Section 2, (4.26) can be rewritten as:

$$
\left\|x_{k+1}(t)-x_k(t)\right\|_\lambda \le K_{x\lambda 1}\left\|e_{k+1}(t)\right\|_\lambda + K_{x\lambda 2}\left\|e_k(t)\right\|_\lambda + \phi/\alpha \tag{4.27}
$$

So far, we build the relationship between the differences of state variables and the tracking errors in two sequential iterations. Now, we want to calculate the tracking errors from iteration to iteration. From (4.1), the tracking error at the k+1th iteration can be expressed as:

$$
\begin{aligned}
e_{k+1}(t) &= y_d(t) - y_{k+1}(t) \\
&= e_k(t) - \left(y_{k+1}(t) - y_k(t)\right) \\
&= e_k(t) - C(t)\left(x_{k+1}(t) - x_k(t)\right) - \left(\xi_{k+1}(t) - \xi_k(t)\right)
\end{aligned} \tag{4.28}
$$

Replacing (4.22) into (4.28), we have

$$
\begin{aligned}
e_{k+1}(t) = e_k(t) &- C(t)\int_0^t \left(f\left(x_{k+1}(\tau),\tau\right) - f\left(x_k(\tau),\tau\right)\right)d\tau - \left(\xi_{k+1}(t) - \xi_k(t)\right) \\
&- C(t)B(t)K_{d1}(t)e_{k+1}(t) - C(t)\int_0^t \left(B(\tau)K_{p1}(\tau) - \frac{d\left(B(\tau)K_{d1}(\tau)\right)}{d\tau}\right)e_{k+1}(\tau)d\tau \\
&- C(t)B(t)K_{d2}(t)e_k(t) - C(t)\int_0^t \left(B(\tau)K_{p2}(\tau) - \frac{d\left(B(\tau)K_{d2}(\tau)\right)}{d\tau}\right)e_k(\tau)d\tau \\
&+ C(t)\left[\left(x_{k+1}(0) - x_k(0)\right) - B(0)K_{d1}(0)e_{k+1}(0) - B(0)K_{d2}(0)e_k(0)\right]
\end{aligned} \tag{4.29}
$$

Reorganizing (4.29) gets

$$
\begin{aligned}
\left[I + C(t)B(t)K_{d1}(t)\right]e_{k+1}(t) = \left[I - C(t)B(t)K_{d2}(t)\right]e_k(t) &- C(t)\int_0^t \left(f\left(x_{k+1}(\tau),\tau\right) - f\left(x_k(\tau),\tau\right)\right)d\tau \\
&- C(t)\int_0^t \left(B(\tau)K_{p1}(\tau) - \frac{d\left(B(\tau)K_{d1}(\tau)\right)}{d\tau}\right)e_{k+1}(\tau)d\tau \\
&- C(t)\int_0^t \left(B(\tau)K_{p2}(\tau) - \frac{d\left(B(\tau)K_{d2}(\tau)\right)}{d\tau}\right)e_k(\tau)d\tau \\
&+ C(t)\left[\left(x_{k+1}(0) - x_k(0)\right) - B(0)K_{d1}(0)e_{k+1}(0) - B(0)K_{d2}(0)e_k(0)\right] \\
&- \left(\xi_{k+1}(t) - \xi_k(t)\right)
\end{aligned} \tag{4.30}
$$

From (4.30) we have

$$\begin{aligned}\left\|e_{k+1}(t)\right\| \le \rho_1 \Big\{ & \rho_2 \left\|e_k(t)\right\| + c_f B_c \int_0^t \left\|x_{k+1}(\tau) - x_k(\tau)\right\| d\tau \\ & + B_c B_{pd1} \int_0^t \left\|e_{k+1}(\tau)\right\| d\tau + B_c B_{pd2} \int_0^t \left\|e_k(\tau)\right\| d\tau \\ & + B_c \phi + 2 b_\xi \Big\}\end{aligned} \tag{4.31}$$

Multiplying (4.31) by $e^{-\lambda t}$ where $\lambda > c_f$, and using Lemma 1, we get

$$\begin{aligned}\left\|e_{k+1}(t)\right\|_\lambda \le \rho_1 \Big\{ & \rho_2 \left\|e_k(t)\right\|_\lambda + \frac{c_f}{\lambda} B_c \left\|x_{k+1}(t) - x_k(t)\right\|_\lambda \\ & + \frac{1}{\lambda} B_c B_{pd1} \left\|e_{k+1}(t)\right\|_\lambda + \frac{1}{\lambda} B_c B_{pd2} \left\|e_k(t)\right\|_\lambda \\ & + B_c \phi + 2 b_\xi \Big\}\end{aligned} \tag{4.32}$$

Reorganizing (4.32) gets

$$\left(1 - \frac{\rho_1 B_c B_{pd1}}{\lambda}\right) \left\|e_{k+1}(t)\right\|_\lambda \le \rho_1 \left(\rho_2 + \frac{B_c B_{pd2}}{\lambda}\right) \left\|e_k(t)\right\|_\lambda + \frac{\rho_1 c_f B_c}{\lambda} \left\|x_{k+1}(t) - x_k(t)\right\|_\lambda + \rho_1 B_c \phi + 2\rho_1 b_\xi \tag{4.33}$$

Submitting (4.27) into (4.33) and doing some simplification, we can get

$$\begin{aligned}\left(1 - \frac{\rho_1 B_c B_{pd1}}{\lambda}\right) \left\|e_{k+1}(t)\right\|_\lambda \le & \rho_1 \left(\rho_2 + \frac{B_c B_{pd2}}{\lambda}\right) \left\|e_k(t)\right\|_\lambda + \rho_1 B_c \phi + 2\rho_1 b_\xi \\ & + \frac{\rho_1 c_f B_c}{\lambda} \left(K_{x\lambda 1} \left\|e_{k+1}(t)\right\|_\lambda + K_{x\lambda 2} \left\|e_k(t)\right\|_\lambda + \phi / \alpha\right)\end{aligned} \tag{4.34}$$

Simplifying (4.34) obtains:

$$\left\|e_{k+1}(t)\right\|_{\lambda} \le \rho\beta\left\|e_k(t)\right\|_{\lambda} + \rho B_c\left(1+\frac{c_f}{\lambda\alpha}\right)\phi + 2\rho b_{\xi} \tag{4.35}$$

where $\rho = \dfrac{\rho_1}{1-\dfrac{\rho_1 B_c}{\lambda}\left(c_f K_{x\lambda 1} + B_{pd1}\right)}$, $\beta = \rho_2 + \dfrac{B_c\left(B_{pd2} + c_f K_{x\lambda 2}\right)}{\lambda}$

From (4.16), we have

$$\rho_1\rho_2 = \max_{t\in[0,T]}\left\|\left(I + C(t)B(t)K_{d1}(t)\right)^{-1}\right\| \max_{t\in[0,T]}\left\|I - C(t)B(t)K_{d2}(t)\right\| < 1 \tag{4.36}$$

If λ is set large enough, then the following relationship holds:

$$\lambda > \frac{\rho_1 B_c}{1-\rho_1\rho_2}\left(c_f\left(K_{x\lambda 1} + K_{x\lambda 2}\right) + B_{pd1} + B_{pd2}\right) \tag{4.37}$$

Then, we can guarantee

$$1 - \frac{\rho_1 B_c}{\lambda}\left(c_f K_{x\lambda 1} + B_{pd1}\right) > 0 \tag{4.38}$$

Finally, we can assure

$$\rho\beta < 1 \tag{4.39}$$

From (4.39) and (4.35), we can guarantee that the final tracking error is bounded as

$$\lim_{k\to\infty}\left\|e_k(t)\right\|_{\lambda} \le \frac{\rho B_c\left(1 + c_f / (\lambda\alpha)\right)\phi + 2\rho b_{\xi}}{1-\rho\beta} \quad \text{for all } t\in[0,T] \tag{4.40}$$

To increase the convergence rate in (4.40), we can reduce B_{pd1} and B_{pd2} in (4.35) by properly choosing the proportional control gains. According to (4.24), the optimal gains K_{p1} and K_{p2} can be chosen as follows:

$$\begin{cases} K_{p1}(t) = B^{-1}(t)\dfrac{d\left[B(t)K_{d1}(t)\right]}{dt} \\ K_{p2}(t) = B^{-1}(t)\dfrac{d\left[B(t)K_{d2}(t)\right]}{dt} \end{cases} \tag{4.41}$$

From (4.16), the optimal learning gain K_{d2} is:

$$K_{d2}(t) = B^{-1}(t)C^{-1}(t) \tag{4.42}$$

***Remark 2*:** If the following initial state updating law is used

$$x_{k+1}(0) = \left(1 + B(0)K_{d1}(0)C(0)\right)^{-1}\left[x_k(0) + B(0)K_{d1}(0)y_d(0) + B(0)K_{d2}(0)\left(y_d(0) - y_k(0)\right)\right] \tag{4.43}$$

then we have

$$\lim_{k\to\infty} x_k(0) = 0 \tag{4.44}$$

In such a way, we can get $\varepsilon_x = 0$. Therefore,

$$\lim_{k\to\infty} \left\| e_k(t) \right\|_\lambda \le \frac{\rho B_c \left(1 + c_f / (\lambda\alpha)\right)\left(2b_\xi + 2b_\eta T\right) + 2\rho b_\xi}{1 - \rho\beta} \tag{4.45}$$

Remark 3: If the uncertainty and disturbance are repeatable at every iteration, i.e., they are the same at every iteration. In this case, from (16) and (24), the final tracking error bound becomes:

$$\lim_{k\to\infty}\left\|e_k(t)\right\|_\lambda \le \frac{\rho B_c\left(1+c_f/(\lambda\alpha)\right)\left(2+B_c\left(B_{d1}+B_{d2}\right)\right)\varepsilon_x}{1-\rho\beta} \tag{4.46}$$

Remark 4: If the initial state updating law (4.43) is applied, and the uncertainty and disturbance are repeatable, then the final tracking error bound is $\lim_{k\to\infty}\left\|e_k(t)\right\|_\lambda = 0$. Such a conclusion can be derived directly from ***Remark 2*** and ***Remark 3***.

4.2.2 SPD-PD Convergence Analysis

PD-PD control law is expressed in the form of (4.10) with

$$\begin{cases} K_{p1}(k) = s(k)K_{p1}(0) \\ K_{d1}(k) = s(k)K_{d1}(0) \end{cases} \qquad with \quad s(k+1) > s(k) \tag{4.47}$$

The control law then becomes

$$\begin{aligned} u_{k+1}(t) = u_k(t) + K_{p1}(k+1)e_{k+1}(t) + K_{d1}(k+1)\dot{e}_{k+1}(t) \\ +K_{p2}(t)e_k(t) + K_{d2}(t)\dot{e}_k(t) \end{aligned} \tag{4.48}$$

Following notations are used in this analysis for the sake of briefness:

$$B_{d1} = \max_{t\in[0,T]}\left\|K_{d1}(0)C(t)\right\|, \quad B_{d2} = \max_{t\in[0,T]}\left\|K_{d2}C(t)\right\|$$

$$B_{pd1} = \max_{t\in[0,T]}\left\|K_{p1}(0)C + K_{d1}(0)\dot{C}\right\| \ , \quad B_{pd2} = \max_{t\in[0,T]}\left\|K_{p2}C + K_{d2}\dot{C}\right\|$$

$$B_{Kp1} = \max_{t\in[0,T]} \left\| K_{p1}(0) \right\|, \quad B_{Kp2} = \max_{t\in[0,T]} \left\| K_{p2} \right\|$$

$$B_{Kd1} = \max_{t\in[0,T]} \left\| K_{d1}(0) \right\|, \quad B_{Kd2} = \max_{t\in[0,T]} \left\| K_{d2} \right\|$$

$$B_B = \max_{t\in[0,T]} \left\| B(t) \right\|, \quad B_C = \max_{t\in[0,T]} \left\| C(t) \right\|, \quad B_s = \max \left\| s(k) \right\|$$

$$K_{B1} = \frac{B_{d1} + c_f B_{pd1}}{\lambda - c_f}, \; K_{B2} = \frac{B_{d2} + c_f B_{pd2}}{\lambda - c_f}$$

$$\rho_1 = \max_{t\in[0,T]} \left\| \left(I + K_{d1}(0)CB \right)^{-1} \right\|, \quad \rho_2 = \max_{t\in[0,T]} \left\| I_m - K_{d2}CB \right\|$$

$$\rho = \frac{\rho_1}{1 - \rho_1 B_s B_B K_{B1}}, \quad \beta = \rho_2 + B_B K_{B2}$$

Theorem: For the nonlinear time-varying system (4.1), if the SPD-PD type iterative learning control law (4.48) is applied and the switching gain algorithm (4.47) is adopted, then the final state error and the output tracking error are bounded and the boundednesses are given by

$$\begin{cases} \lim\limits_{k\to\infty} \left\| \delta x_k(t) \right\|_\lambda \le \dfrac{1}{\lambda - c_f} \left(\dfrac{\rho\Phi}{1-\rho\beta} + Tb_\eta + \varepsilon_x \right) \\ \lim\limits_{k\to\infty} \left\| e_k(t) \right\|_\lambda \le \dfrac{B_C}{\lambda - c_f} \left(\dfrac{\rho\Phi}{1-\rho\beta} + Tb_\eta + \varepsilon_x \right) + b_\xi \end{cases} \tag{4.49}$$

where $\Phi = \left(B_s K_{B1} + K_{B2} + B_s B_{d1} + B_{d2} \right) b_\eta + \left(B_s B_{Kp1} + B_{Kp2} \right) b_\xi + \left(B_s B_{Kd1} + B_{Kd2} \right) b_\xi + \left(B_s K_{B1} + K_{B2} \right) \varepsilon_x$.

Provided the control gain $K_{d1}(0)$ and the learning gain $K_{d2}(t)$ are selected such that $I_m + K_{d1}(0)B(t)C(t)$ is non-singular, and

$$\begin{cases} \max\limits_{t\in[0,T]}\left\|\left(I+K_{d1}(0)B(t)C(t)\right)^{-1}\right\|=\rho_1<1 \\ \max\limits_{t\in[0,T]}\left\|I-K_{d2}(t)B(t)C(t)\right\|=\rho_2<1 \end{cases} \tag{4.50}$$

Also we propose the following initial state learning algorithm:

$$x_{k+1}(0)=\left(1+B(0)K_{d1}(0)C(0)\right)^{-1}\left\{x_k(0)+B(0)K_{d1}(0)y_d(0)+B(0)K_{d2}(0)\left(y_d(0)-y_k(0)\right)\right\} \tag{4.51}$$

From (4.1), we can calculate the tracking errors as:

$$\begin{cases} e_k=C(t)\delta x_k-\xi_k \\ e_{k+1}=C(t)\delta x_{k+1}-\xi_{k+1} \end{cases} \tag{4.52}$$

The derivative of the tracking errors can be represented as

$$\begin{cases} \dot{e}_k=C(t)\delta\dot{x}_k+\dot{C}(t)\delta x_k-\dot{\xi}_k \\ \dot{e}_{k+1}=C(t)\delta\dot{x}_{k+1}+\dot{C}(t)\delta x_{k+1}-\dot{\xi}_{k+1} \end{cases} \tag{4.53}$$

From (4.48) we have:

$$\delta u_{k+1}=\delta u_k-s(k+1)K_{p1}(0)e_{k+1}(t)-s(k+1)K_{d1}(0)\dot{e}_{k+1}(t)-K_{p2}(t)e_k(t)-K_{d2}(t)\dot{e}_k(t) \tag{4.54}$$

Submitting (4.52) and (4.53) into (4.54) gets:

$$\begin{aligned}\delta u_{k+1}=\delta u_k&-K_{p2}\left\{C\delta x_k-\xi_k\right\}-K_{d2}\left\{C\delta\dot{x}_k+\dot{C}\delta x_k-\dot{\xi}_k\right\}-s(k+1)K_{p1}(0)\left\{C\delta x_{k+1}-\xi_{k+1}\right\}\\&-s(k+1)K_{d1}(0)\left\{C\delta\dot{x}_{k+1}+\dot{C}\delta x_{k+1}-\dot{\xi}_{k+1}\right\}\end{aligned} \tag{4.55}$$

Also from (4.1) we can get the following equations:

$$\begin{cases} \delta \dot{x}_k = \delta f_k + B\delta u_k - \eta_k \\ \delta \dot{x}_{k+1} = \delta f_{k+1} + B\delta u_{k+1} - \eta_{k+1} \end{cases} \tag{4.56}$$

Submitting (4.55) into (4.56) and reorganizing gets:

$$\begin{aligned} \left(I + s(k+1)K_{d1}(0)CB\right)\delta u_{k+1} = {} & \left(I - K_{d2}CB\right)\delta u_k - s(k+1)\left(K_{p1}(0)C + K_{d1}(0)\dot{C}\right)\delta x_{k+1} \\ & -\left(K_{p2}C + K_{d2}\dot{C}\right)\delta x_k - s(k+1)K_{d1}(0)\left(C\delta f_{k+1} - C\eta_{k+1} - \dot{\xi}_{k+1}\right) \\ & -K_{d2}\left(C\delta f_k - C\eta_k - \dot{\xi}_k\right) + s(k+1)K_{p1}(0)\xi_{k+1} + K_{p2}\xi_k \end{aligned} \tag{4.57}$$

From A3 we have $\|\delta f_k\| \le c_f \|\delta x_k\|$, and $\|\delta f_{k+1}\| \le c_f \|\delta x_{k+1}\|$. As $s(k+1) > 1$, to choose a proper control gain $K_{p1}(0)$ and from (4.50) we can ensure:

$$\left\| \left(I + s(k+1)K_{d1}(0)CB\right)^{-1} \right\| < \left\| \left(I + K_{d1}(0)CB\right)^{-1} \right\| = \rho_1 < 1 \tag{4.58}$$

Applying (4.58), (4.57) can be rewritten in the λ-norm and simplified as:

$$\begin{aligned} \|\delta u_{k+1}\|_\lambda \le \rho_1 \Big\{ & \rho_2 \|\delta u_k\|_\lambda + B_s\left(B_{pd1} + B_{d1}c_f\right)\|\delta x_{k+1}\|_\lambda + \left(B_{pd2} + B_{d2}c_f\right)\|\delta x_k\|_\lambda + \left(B_s B_{d1} + B_{d2}\right)b_\eta \\ & + \left(B_s B_{Kp1} + B_{Kp2}\right)b_\xi + \left(B_s B_{Kd1} + B_{Kd2}\right)b_{\dot{\xi}} \Big\} \end{aligned} \tag{4.59}$$

For the k^{th} iteration, the state vector can be written as:

$$x_k(t) = x_k(0) + \int_0^t \left(f\left(x_k(\tau),\tau\right) + B(\tau)u_k(\tau)\right)d\tau + \int_0^t \eta_k(t)d\tau \tag{4.60}$$

From (4.1), we also have:

$$x_d(t) = x_d(0) + \int_0^t \left(f\left(x_d(\tau), \tau\right) + B(\tau) u_d(\tau) \right) d\tau \tag{4.61}$$

From (4.60) and (4.61), we get:

$$\delta x_k = \int_0^t \left(f\left(x_d(\tau), \tau\right) - f\left(x(\tau), \tau\right) \right) d\tau + \int_0^t \left(B(t)\delta u_k - \eta_k \right) d\tau + \delta x_k(0) \tag{4.62}$$

Applying A3 to (4.62), we get

$$\delta x_k \le \int_0^t c_f \delta x_k d\tau + \int_0^t \left(B(t)\delta u_k - \eta_k \right) d\tau + \delta x_k(0) \tag{4.63}$$

Eqn. (4.63) can be written in the norm form as:

$$\left\| \delta x_k \right\| \le \int_0^t c_f \left\| \delta x_k \right\| d\tau + \int_0^t \left(\left\| B(t)\delta u_k \right\| + \left\| \eta_k \right\| \right) d\tau + \left\| \delta x_k(0) \right\| \tag{4.64}$$

According to the definition of λ-norm, for $\lambda > c_f$, applying **Lemma 1** to (4.64) obtains:

$$\left\| \delta x_k \right\|_\lambda \le \frac{B_B}{\lambda - c_f} \left\| \delta u_k \right\|_\lambda + \frac{T}{\lambda - c_f} b_\eta + \frac{\varepsilon_x}{\lambda - c_f} \tag{4.65}$$

For the $k+1^{th}$ iteration, we can get a very similar result:

$$\left\| \delta x_{k+1} \right\|_\lambda \le \frac{B_B}{\lambda - c_f} \left\| \delta u_{k+1} \right\|_\lambda + \frac{T}{\lambda - c_f} b_\eta + \frac{\varepsilon_x}{\lambda - c_f} \tag{4.66}$$

Submitting (4.65) and (4.66) into (4.59) and simplifying it gets:

$$\left(1 - \rho_1 B_s B_B K_{B1}\right) \left\| \delta u_{k+1} \right\|_\lambda \le \rho_1 \left(\rho_2 + B_B K_{B2}\right) \left\| \delta u_k \right\|_\lambda + \rho_1 \Phi \tag{4.67}$$

Eqn. (4.67) can be simplified as

$$\left\|\delta u_{k+1}\right\|_{\lambda} \leq \rho\beta\left\|\delta u_{k}\right\|_{\lambda} + \rho\Phi \tag{4.68}$$

From (4.50) we have $\rho_1\rho_2 < 1$. If we choose

$\lambda > \dfrac{\rho_1 B_B\left(c_f\left(B_s B_{pd1} + B_{pd2}\right) + B_s B_{d1} + B_{d2}\right)}{1-\rho_1\rho_2} + c_f$, then we can guarantee $\rho\beta < 1$.

From (4.68) we can get:

$$\lim_{k\to\infty}\left\|\delta u_k(t)\right\|_{\lambda} = \frac{\rho\Phi}{1-\rho\beta} \tag{4.69}$$

From (4.69),we can see that the control input is bounded and is close to the desired control input.

Submitting (4.69) into (4.65), we can get:

$$\lim_{k\to\infty}\left\|\delta x_k(t)\right\|_{\lambda} = \frac{1}{\lambda - c_f}\left(\frac{\rho\Phi}{1-\rho\beta} + Tb_{\eta} + \varepsilon_x\right) \tag{4.70}$$

Eqn. (4.70) proves that the state error is bounded.

Finally, from (4.52) we can get:

$$\lim_{k\to\infty}\left\|e_k(t)\right\|_{\lambda} = \frac{B_C}{\lambda - c_f}\left(\frac{\rho\Phi}{1-\rho\beta} + Tb_{\eta} + \varepsilon_x\right) + b_{\xi} \tag{4.71}$$

Eqn. (4.71) demonstrates that the output error is bounded.

***Remark 1*:** If the initial state updating law (4.51) is used, we will ensure $\lim_{k\to\infty} x_k(0) = x_0(0)$. In this manner, we can get $\lim_{k\to\infty} \varepsilon_x = 0$. Therefore,

$$\lim_{k\to\infty} \left\| e_k(t) \right\|_\lambda = \frac{B_C}{\lambda - c_f}\left(\frac{\rho\Phi}{1-\rho\beta} + Tb_\eta\right) + b_\xi \tag{4.72}$$

Remark 2: If there is no uncertainty and disturbance in (4.1), then the final tracking error bound becomes:

$$\lim_{k\to\infty} \left\| e_k(t) \right\|_\lambda = \frac{B_C \varepsilon_x}{\lambda - c_f} \tag{4.73}$$

Remark 3: If the initial state updating law (4.51) is applied, and there is no uncertainty and disturbance, then the final tracking error is $\lim_{k\to\infty} \left\| e_k(t) \right\|_\lambda = 0$. Such a conclusion can be derived directly from ***Remark 1*** and ***Remark 2***.

Remark 4: The convergence condition (4.50) doesn't include the proportional gains. Therefore, it provides some extra freedom for the choices of K_{p1} and K_{p2} in the proposed control law (4.48).

4.3 Comparison Study on ILCs

A comparison study on types of ILCs applied to a nonlinear time-varying system in the presence of repetitive noise (noise as a constant of iteration) and varying noise (noise as a function of iteration) has been well established in [***52***]. To generally demonstrate performance of online PD and SPD ILCs, some results in the paper are cited and shown afterward. Consider the following MIMO nonlinear system

$$\begin{bmatrix} \dot{x}_{1k}(t) \\ \dot{x}_{2k}(t) \end{bmatrix} = \begin{bmatrix} \sin\left(x_{2k}(t)\right) & 1+\sin\left(x_{1k}(t)\right) \\ 2-5t & -3-2t \end{bmatrix} \begin{bmatrix} x_{1k}(t) \\ x_{2k}(t) \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} u_{1k}(t) \\ u_{2k}(t) \end{bmatrix} + \left(0.5 + k\alpha_0\right) \begin{bmatrix} \cos\left(2\pi f_0 t\right) \\ 2\cos\left(4\pi f_0 t\right) \end{bmatrix}$$

$$\begin{bmatrix} y_{1k}(t) \\ y_{2k}(t) \end{bmatrix} = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1k}(t) \\ x_{2k}(t) \end{bmatrix} + (0.5 + k\alpha_0) \begin{bmatrix} \sin(2\pi f_0 t) \\ 2\sin(4\pi f_0 t) \end{bmatrix}$$

With $f_0 = 5$ Hz, α_0 is set to 0 for repetitive noise and 5 for varying noise. The desired tracking trajectories are set as

$$y_{1d}(t) = y_{2d}(t) = 12t^2(1-t) \qquad for\ t \in [0,1]$$

Using the general form of PD online-offline ILCs in eqn. (4.10), the chosen gains in the paper are

$$\begin{cases} K_{poff} = K_{doff} = K_{off} \begin{bmatrix} 0.25 & 0 \\ 0 & 0.5 \end{bmatrix} \\ K_{pon} = K_{don} = K_{on} \begin{bmatrix} 0.5 & 0 \\ 0 & 2.5 \end{bmatrix} \end{cases}$$

where K_{off} and K_{on} are control gain factor and are set to 0.5,1 and 1.5 for K_{off} and 1, 5 and 10 for K_{on}. Note that $K_{off} = 1$ is the optimal choice for offline learning gains.

Online ILCs: P-type vs D-type

Comparison between pure P-type and D-type online ILCs in the presence of repetitive noises are clearly illustrated in Fig. 4-3. The figure shows errors of output y_1 of the system. The blue, green and red lines indicate the value of K_{on} of 1, 5 and 10, respectively, for both P-type and D-type ILCs. At the same gain factor, it is obvious that D-type converges much faster with less maximum error in every iteration comparing to P-type. On the other hand, P-type alone cannot achieve monotonic convergence with gains chosen too small. Note that both types converge faster at higher gain factor.

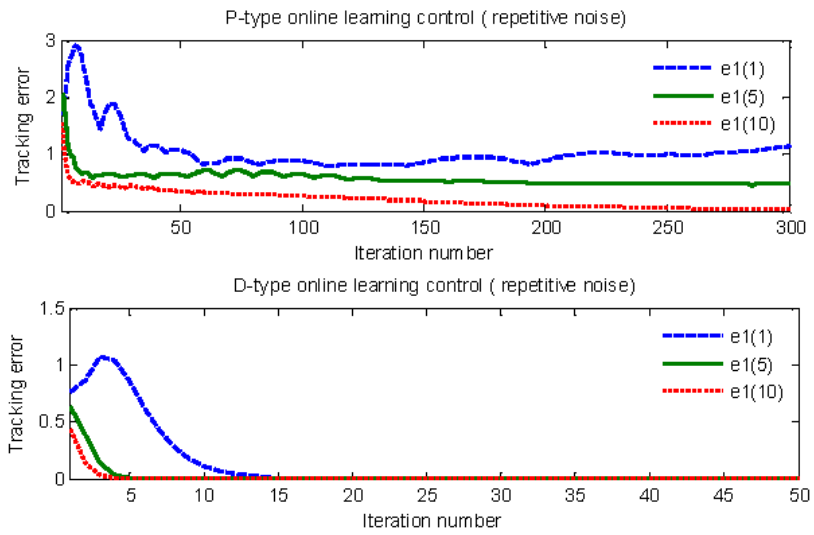

Fig. 4-3 Comparison between P-type and D-type online ILCs

PD ILCs: offline vs online vs switching gain

Performance of PD offline, online, online-offline and SPD are shown in Fig. 4-4. It can be seen that PD offline ILCs are the worst among them with highest maximum errors and lowest convergence speed. SPD ILCs converge fastest with smallest maximum errors while PD online and online-offline are approximately the same. The figure also shows better performance of PD-type over P-type and D-type alone, making more sense to choose PD-type in this thesis.

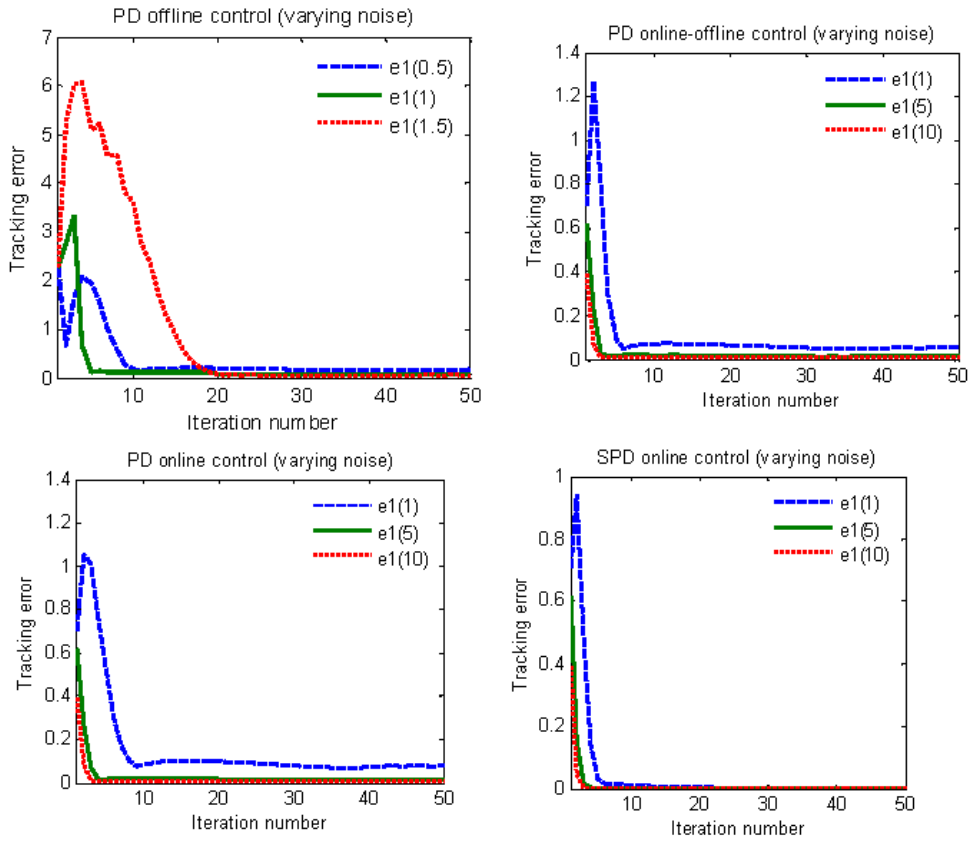

Fig. 4-4 Comparison among PD ILCs

4.4 ILCs Implementation on Quadrotor

To design the control system for a quadrotor UAV, it is more convenient to first establish a state-space model. In this work, the states of the system are chosen as

$$\begin{aligned} \boldsymbol{x} &= [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \;\; x_6 \;\; x_7 \;\; x_8 \;\; x_9 \;\; x_{10} \;\; x_{11} \;\; x_{12}]^T \\ &= \left[X \;\; Y \;\; Z \;\; \phi \;\; \theta \;\; \psi \;\; \dot{X} \;\; \dot{Y} \;\; \dot{Z} \;\; \dot{\phi} \;\; \dot{\theta} \;\; \dot{\psi}\right]^T \end{aligned} \tag{4.74}$$

Recalling from dynamic model formulated in Chapter 3 along with the state vector expressed in (4.74), the state-space model with disturbances can then be described in the form of

$$\dot{\boldsymbol{x}}(t) = \frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \end{pmatrix} = \begin{pmatrix} x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ -sinx_5 cosx_4 \left(\frac{b}{m}\right) u_1 \\ sinx_4 \left(\frac{b}{m}\right) u_1 \\ -g + cosx_5 cosx_4 \left(\frac{b}{m}\right) u_1 - d_1(t) \\ a_1 x_{11} x_{12} - a_2 \Omega_r x_{11} - a_{11} u_2 - d_2(t) \\ a_3 x_{10} x_{12} + a_4 \Omega_r x_{10} - a_{22} u_3 - d_3(t) \\ a_5 x_{10} x_{11} + a_{33} u_4 - d_4(t) \end{pmatrix} \tag{4.75}$$

and

$$\boldsymbol{y}(t) = \boldsymbol{I}_{12\times 12}\boldsymbol{x}(t) \tag{4.76}$$

where $u_1 \ldots u_4$ are virtual inputs, which simplifies the control laws shown later, and relates to thrust generated by rotors and ultimately rotor speed via

$$\boldsymbol{u} = \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{Bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{Bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{Bmatrix} \tag{4.77}$$

and

$$a_1 = \frac{I_y - I_z}{I_x}, \qquad a_2 = \frac{J_r}{I_x}, \qquad a_3 = \frac{I_z - I_x}{I_y}, \qquad a_4 = \frac{J_r}{I_y}, \qquad a_5 = \frac{I_x - I_y}{I_z}$$

$$a_{11} = \frac{l}{I_x}, \qquad a_{22} = \frac{l}{I_y}, \qquad a_{33} = \frac{d}{I_z}$$

and $d_1 \dots d_4$ are disturbances in the Z-direction and the three rotations. In the simulations, no additional disturbances are added in X and Y directions due to the fact that disturbances in rotation also affect position because when the quadrotor tilts, the lift force then contributes a horizontal force component, dragging the body to the tilt direction. The disturbances simulated will be discussed and presented in Chapter 5.

In this thesis, as stated earlier, integral part of PID will be neglected as ILC itself has a characteristic of integration, by adding control law from previous iteration to the current one. Furthermore, offline and online-offline ILCs are not considered here. Only ordinary PD online ILCs and SPD ILCs will be tested and compared.

4.4.1 PD ILCs

Control inputs are calculated using iterative learning control as

$$u_{i,k+1}(t) = u_{i,k}(t) + \left(K_{pon}\right)_{i+2}\left(x_{i+2,d}(t) - x_{i+2}(t)\right)_{k+1} + (K_{don})_{i+2}\left(x_{i+8,d}(t) - x_{i+8}(t)\right)_{k+1} \tag{4.78}$$

for $i = 1, \dots, 4$. From (4.75), $x_{3,d}$ and $x_{9,d}$ are the only desired states in the Z-direction for take-off and landing missions that are determined by desired trajectories. In order to find proper desired states as a function of time for the rest, two dummy control inputs are introduced and defined as

$$u_{y,k+1}(t) = u_{y,k}(t) + \left(K_{pon}\right)_{1}\left(x_{2,d}(t) - x_{2}(t)\right)_{k+1} + (K_{don})_{1}\left(x_{8,d}(t) - x_{8}(t)\right)_{k+1} \tag{4.79}$$

and

$$u_{x,k+1}(t) = u_{x,k}(t) + \left(K_{pon}\right)_2 \left(x_{1,d}(t) - x_1(t)\right)_{k+1} + (K_{don})_2 \left(x_{7,d}(t) - x_7(t)\right)_{k+1} \tag{4.80}$$

where $x_{1,d}, x_{2,d}, x_{7,d}$ and $x_{8,d}$ are determined by desired trajectories in X and Y directions and their derivatives, respectively. Hence the desired angles and rates can be defined as

$$\begin{aligned} \phi_d(t) &= x_{4,d}(t) = \text{asin}\left(u_y\right) \\ \dot{\phi}_d(t) &= x_{10,d}(t) = \frac{1}{\sqrt{1-u_y^2}} \dot{u_y} \end{aligned} \tag{4.81}$$

and

$$\begin{aligned} \theta_d(t) &= x_{5,d}(t) = -\text{asin}\left(\frac{u_x}{\cos(x_4)}\right) \\ \dot{\theta}_d(t) &= x_{11,d}(t) = -\frac{\dot{u}_x + \tan(x_4)\, x_{10} u_x}{\cos(x_4)\sqrt{1-\left(\frac{u_x}{\cos(x_4)}\right)^2}} \end{aligned} \tag{4.82}$$

Since a quadrotor is very agile and is able to maneuver in any direction, controlling yaw angle is not very necessary, but it is preferable to maintain zero yaw rate to prevent undesirable gyroscopic and aerodynamic effects, leading to inefficiency, if not instability, and thus it is defined as

$$\psi_d(t) = x_{6,d}(t) = \dot{\psi}_d(t) = x_{12,d}(t) = 0 \tag{4.83}$$

4.4.2 SPD ILCs

When a switching gain (SPD ILC) is used, online learning gains increase monotonically each iteration and can be written as

$$\begin{cases} \left(K_{pon}\right)_i(k) = s(k) \times \left(K_{pon}\right)_i(0) \\ \left(K_{don}\right)_i\,(k) = s(k) \times \left(K_{don}\right)_i(0) \end{cases} \qquad where\ s(k) = k, k \geq 1 \qquad (4.84)$$

where subscript i indicates learning gain corresponding to each control law. Hence control laws are then a function of iteration number and can be expressed as

$$\begin{aligned} u_{i,k+1}(t) = u_{i,k}(t) + \left(K_{pon}\right)_{i+2}(k)\left(x_{i+2,d}(t) - x_{i+2}(t)\right)_{k+1} \\ +\left(K_{don}\right)_{i+2}(k)\left(x_{i+8,d}(t) - x_{i+8}(t)\right)_{k+1} \end{aligned} \qquad (4.85)$$

Note that the switching here occurs in iteration domain rather than time domain for traditional switching control that could cause trouble in transient process of the switched system which does not apply to this case.

4.5 Trajectory Generation

To control a plant using ILCs, trajectory tracking, where desired position varies with time, is considered rather than point-to-point control, which is used in some quadrotor studies such as[***21***]. In order to utilize trajectory tracking to alter to different states, one cannot simply use a first order velocity equation, which would require a huge input to satisfy the control rule, resulting in an inaccurate tracking and would damage actuators if used in a real application. The problem can be solved by choosing a higher order polynomial equation for trajectory planning. One can literally choose the order of a polynomial as high as desired in obtain a smooth trajectory, i.e., continuous and gradual change in position, velocity and acceleration. In this thesis, a 5th order polynomial function of time is proved to be sufficient and thus used in every mission simulation. With the boundary conditions of $f(0) = 0, f(T) = 1, f'(0) = f'(T) = f''(0) = f''(T) = 0$, a unit trajectory can be obtained as [***57***]

$$f(t) = 10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5 \qquad (4.86)$$

where $f(t) \in [0,1]$ and $t \in [0,T]$. It can be seen that it is a function of time within a period, hence can be used in any task perform repeatedly, maneuvering in the case of this work. Fig. 4-5 demonstrates how the function results in a smooth trajectory, for position, velocity and acceleration. The velocity and acceleration gradually increase from zero as position starts to change, and then steadily decrease back to zero when reaching the destination. This is clearly an effective trajectory to track due to its smoothness which makes tracking more easily and accurately and also avoids actuator saturation.

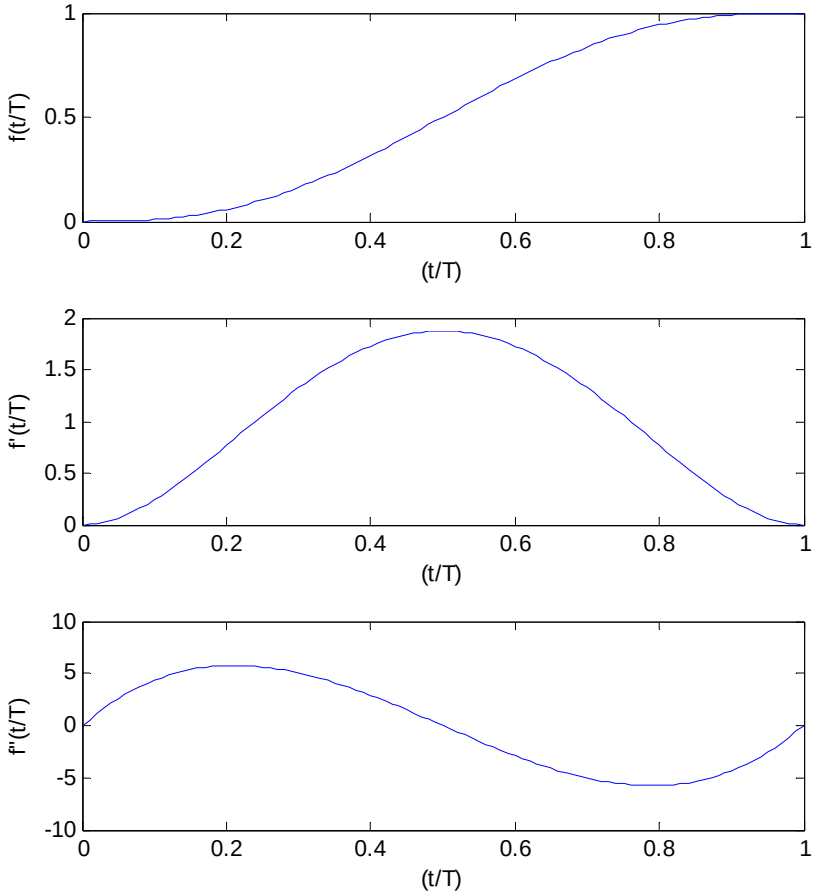

Fig. 4-5 Position, velocity and acceleration plotting of the 5th order polynomial

4.6 Concluding Remarks

This chapter has described important principles of general ILCs including offline ILCs, online ILCs and online-offline ILCs. Convergence analysis for PD online ILC and SPD online ILC were conducted and proved, showing advantages, drawbacks and reasons for candidacy of both types of ILCs to perform simulations. Implementation of both PD and SPD ILCs on the quadrotor nonlinear model with disturbances was demonstrated. A summarized control diagram of ILC implemented in a quadrotor plant is illustrated in Fig. 4-6. Finally, an appropriate choice of smooth trajectory was chosen for simulations and will be carried on to the next chapter of Simulation Results.

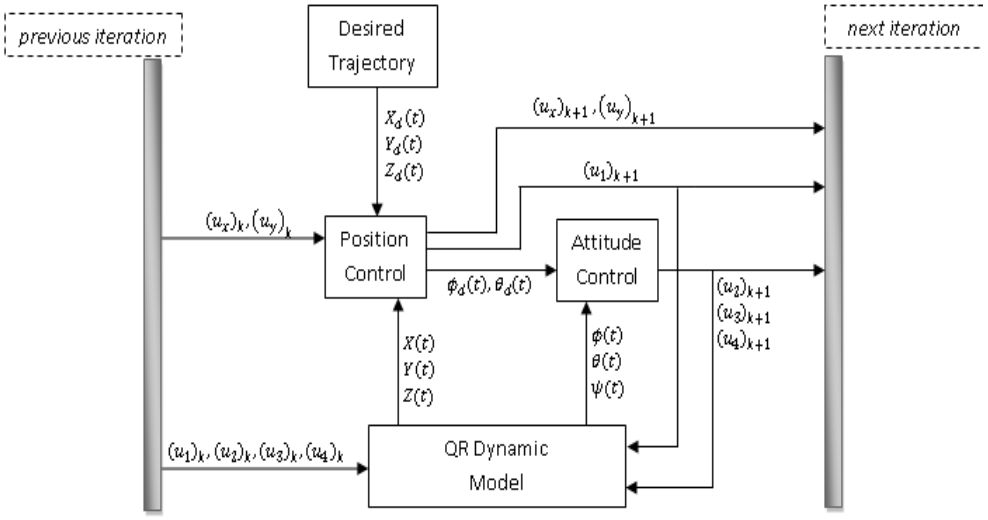

Fig. 4-6 Control diagram of the quadrotor model implemented with PD and SPD online ILCs

Chapter 5 Simulation Results

This chapter mainly presents some simulation results for trajectory tracking of a quadrotor UAV in the presence of considerable disturbances, using PD and SPD online ILCs. All the simulation results are obtained solely using MATLAB programming (see Appendix for more detail). The organization of this chapter is as follows: control gains and disturbances used in the simulations are introduced in Section 5.1. Simulation results of vertical take-off are shown in Section 5.2. This section also discusses the effect of changing each individual and the whole set of control gains and demonstrates the effectiveness of PD online and SPD-ILCs. Simulation results of landing, horizontal smooth translation, horizontal circular translation and spatial circular trajectory are presented in sections 5.3, 5.4, 5.5 and 5.6, respectively. Finally, concluding remarks for this chapter are then summarized in Section 5.7.

5.1 Introduction of Control Gains and Disturbances

Throughout the simulation, a standard set of P and D online learning gains is used in all maneuvering and chosen as

$$\begin{aligned} K_{pon} &= \left[(K_{pon})_1 \; (K_{pon})_2 \; (K_{pon})_3 \; (K_{pon})_4 \; (K_{pon})_5 \; (K_{pon})_6 \right] \\ &= K_{on}[0.1 \;\; 0.1 \;\; 50 \;\; 5 \;\; 5 \;\; 500] \\ K_{don} &= 1.5 K_{pon} \end{aligned} \tag{5.1}$$

where $K_{on} = 0.5, 1, 1.5$ is used to demonstrate the effect of increasing-decreasing the whole gain set by a factor. For most of the simulations here, $K_{on} = 1$ unless otherwise stated. The control gains $(K_{pon})_1 \dots (K_{pon})_6$ correspond to $x_1 \dots x_6$ respectively and K_{don} correspond to $x_7 \dots x_{12}$. Due to the fact that, in this thesis, the main goal is to demonstrate validity of online ILCs implementation on a quadrotor model, the control gains above were found using trial and error with regard to high stability and trajectory tracking performance priority and consequently neglecting power consumption optimization. First of all, note that the first two control gains of $(K_{pon})_1$ and $(K_{pon})_2$ are chosen very small compared to others for two reasons: firstly, to clearly demonstrate the learning characteristics of online ILCs and how

much improvement obtained despite fairly large errors in X and Y directions for some maneuverings in the first iterations as will be shown later. Secondly, to clearly show the better performances of SPD ILCs over PD online ILCs as at very high gains, their differences are less obvious. Smaller errors in the first iterations can be obtained using larger gains. The gain $\left(K_{pon}\right)_3 = 50$ was obtained through simulation and found to be the lowest gain that yielded a fast enough response to counter with its own weight at hover. The considerably large gain of 500 for $\left(K_{pon}\right)_6$ was chosen to obtain a sensible response due to very low moment of inertia of rotors and their opposite direction of rotation while moment of inertia about the Z-axis of the quadrotor UAV (I_{zz}) is much larger comparing to I_{xx} and I_{yy}(see section 3.4). The relation $K_{don} = 1.5K_{pon}$ was simply chosen arbitrarily and comparison in choices of this factor will be discussed later.

Before moving on to next section, the disturbances affecting the quadrotor model in each direction are introduced first. Throughout the simulations in this work, as stated earlier in previous chapter, disturbances appear only in Z-direction and the three angles, which are chosen as

$$\begin{cases} d_1 = 0.1D \\ d_2 = D \\ d_3 = D \\ d_4 = 0.001D \end{cases}, \qquad D = 10 + 20\sin(2\pi t) \tag{5.2}$$

Note that, recalling from the dynamic model and its parameters in Chapter 3, maximum disturbance in the Z direction (3 m/s^2) is fairly large comparing to maximum thrust generated by all propellers per total mass of 18.7 m/s^2, which is approximately 16% of maximum thrust. In pitch and roll direction, maximum disturbances are then 30 rad/s^2 which is a considerable amount. Finally in yaw mode, the range of disturbance is [-0.01, 0.03] rad/s^2. Although this might seem small, it may cause significant oscillation when simultaneously occurs with large disturbances in pitch and roll mode. Thus the chosen disturbances are a fine proof of the robustness for online ILCs.

5.2 Take-off Tracking

In this mission, the quadrotor UAV is commanded to smoothly take-off and ultimately hover at a desired height $h_d = 10m$ in this simulation within the time $= 5s$. The desired trajectory is defined as:

$$\begin{cases} X_d(t) = 0 \\ Y_d(t) = 0 \\ Z_d(t) = h_d \left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5 \right] \end{cases} \tag{5.3}$$

$$Z_d \in [0, h_d], \qquad t \in [0, T]$$

To clarify how each control gain has effect on trajectory tracking performance in each direction, variation of learning gains and its trajectory tracking performance based on take-off mission are tested with PD online ILCs and shown in Table 5-1. Similar results can also be obtained using SPD ILCs, excepted the faster convergence rates will be achieved.

Table 5-1 Effect of each control gain on tracking errors

Control Gains	**k**	$\|\boldsymbol{e_x}\|(\boldsymbol{m})$	$\|\boldsymbol{e_y}\|(\boldsymbol{m})$	$\|\boldsymbol{e_z}\|(\boldsymbol{m})$	$\|\boldsymbol{\phi}\|(\boldsymbol{deg})$	$\|\boldsymbol{\theta}\|(\boldsymbol{deg})$	$\|\boldsymbol{\psi}\|(\boldsymbol{deg})$
[0.1 0.1 50 5 5 500] $K_{on} = 1$	1	5.4230e-1	5.4233e-1	2.2288e-2	1.2445e0	1.2443e0	6.8584e0
	3	5.8570e-2	5.8567e-2	1.1571e-5	4.6324e-1	4.6363e-1	4.7586e0
	5	1.5329e-2	1.5293e-2	1.1513e-5	3.2871e-1	3.2868e-1	3.0638e0
	9	5.2615e-3	5.2580e-3	1.1418e-5	2.4151e-1	2.4150e-1	1.3656e0
[0.01 0.01 50 5 5 500] $K_{on} = 1$	1	7.4726e0	7.4710e0	2.2293e-2	2.3313e0	2.3303e0	6.8584e0
	3	2.6273e0	2.6271e0	2.0997e-5	2.1019e0	2.1048e0	4.7586e0

	5	1.2043e0	1.2045e0	1.6938e-5	1.2987e0	1.2988e0	3.0638e0
	9	5.7488e-1	5.7509e-1	1.2446e-5	1.7279e0	1.7272e0	1.3656e0
[0.1 0.1 5 5 5 500] $K_{on} = 1$	1	5.4294e-1	5.4296e-1	2.2143e-1	1.2439e0	1.2437e0	6.8584e0
	3	5.9510e-2	5.9508e-2	2.4145e-4	4.6384e-1	4.6422e-1	4.7586e0
	5	1.5699e-2	1.5661e-2	3.0327e-5	3.3077e-1	3.3073e-1	3.0638e0
	9	5.2814e-3	5.2783e-3	1.7845e-5	2.4423e-1	2.4423e-1	1.3656e0
[0.1 0.1 50 0.5 0.5 500] $K_{on} = 1$	1	5.1328e0	5.1527e0	2.2310e-2	1.2963e1	1.2887e1	6.8584e0
	3	5.9542e-1	5.9551e-1	1.2241e-4	5.5995e0	5.6874e0	4.7586e0
	5	1.4878e-1	1.4534e-1	2.5605e-5	3.9624e0	3.8798e0	3.0638e0
	9	5.1731e-2	5.1301e-2	1.2984e-5	2.9552e0	2.8919e0	1.3656e0
[0.1 0.1 50 5 5 50] $K_{on} = 1$	1	5.4217e-1	5.4223e-1	2.2137e-2	1.2476e0	1.2473e0	7.9413e0
	3	5.9406e-2	5.9384e-2	1.0859e-5	4.6272e-1	4.6324e-1	7.6792e0
	5	1.5774e-2	1.5685e-2	1.0623e-5	3.2723e-1	3.2732e-1	7.4223e0
	9	5.3604e-3	5.3157e-3	1.0622e-5	2.4210e-1	2.4249e-1	6.9237e0

As seen in Table 5-1, reducing $(K_{pon})_1$ and $(K_{pon})_2$ by a factor of 10 significantly reduces the convergence rate and yields much larger errors in the first three iterations. Likewise, decreasing $(K_{pon})_4$ and $(K_{pon})_5$ by a factor of 10 yields the similar result. This is due to slower reaction in attitude control that keeps the quadrotor away from the desired position in the presence of disturbances. Lastly, reducing $(K_{pon})_6$ by a factor of 10 shows considerable decrease in yaw angle stability and slower convergence rates.

The effect of changes in K_{don} is illustrated in Fig. 5-1. For the X and Y directions, when $K_{don} = K_{pon}$ the system yields larger errors and tends to produce steady errors, while when $K_{don} = 1.5K_{pon}$ and $K_{don} = 3K_{pon}$ the results are much better in terms of maximum errors and less steady errors, this strongly shows the importance of the derivative gains. Errors in the first iteration can be reduced with larger gains chosen. For the Z direction, the results are merely identical.

It is also noticeable that the roll and pitch responses are not plotted in all the following figures. This is because responses in the X, Y and Z directions are heavily affected by attitude control, hence successful maneuvering results from successful attitude control and thus only responses in the X, Y, Z and yaw directions are plotted and illustrated.

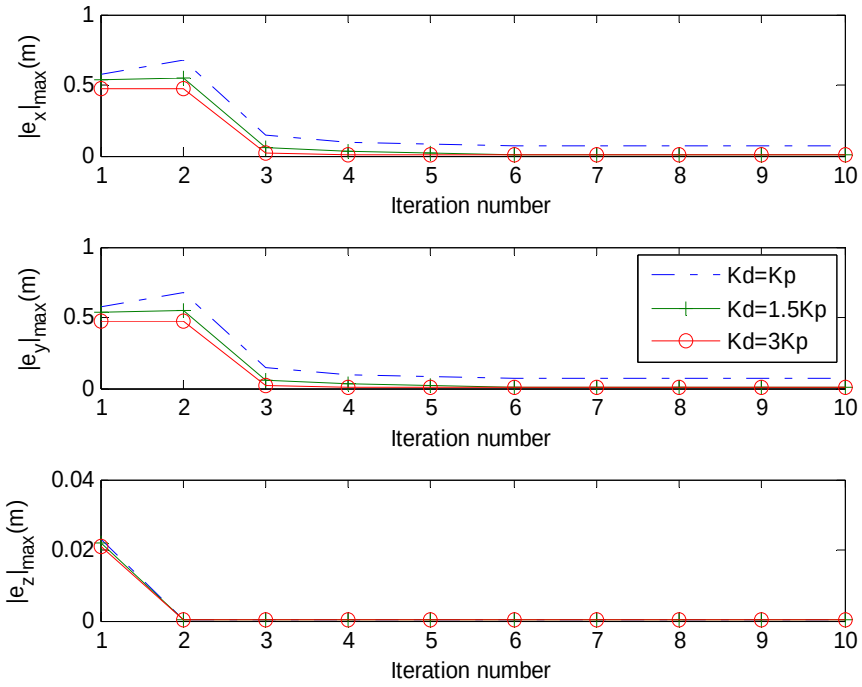

Fig. 5-1 Effect of Kd on tracking errors for online ILCs

Fig. 5-2 shows how changing the whole set of gains by a factor affects trajectory tracking performance. With $K_{on} = 0.5$, for X and Y direction, it is obvious that the control gains are not high enough to ensure monotonic convergence and yield high errors. On the other hand,

when $K_{on} = 1$ and $K_{on} = 1.5$ the system is monotonically convergent and even faster for the latter case. And again, the effect is hardly detected in the Z direction. This is because control in the Z direction depends solely on u_1 while X and Y directions also depend on attitude of the quadrotor UAV.

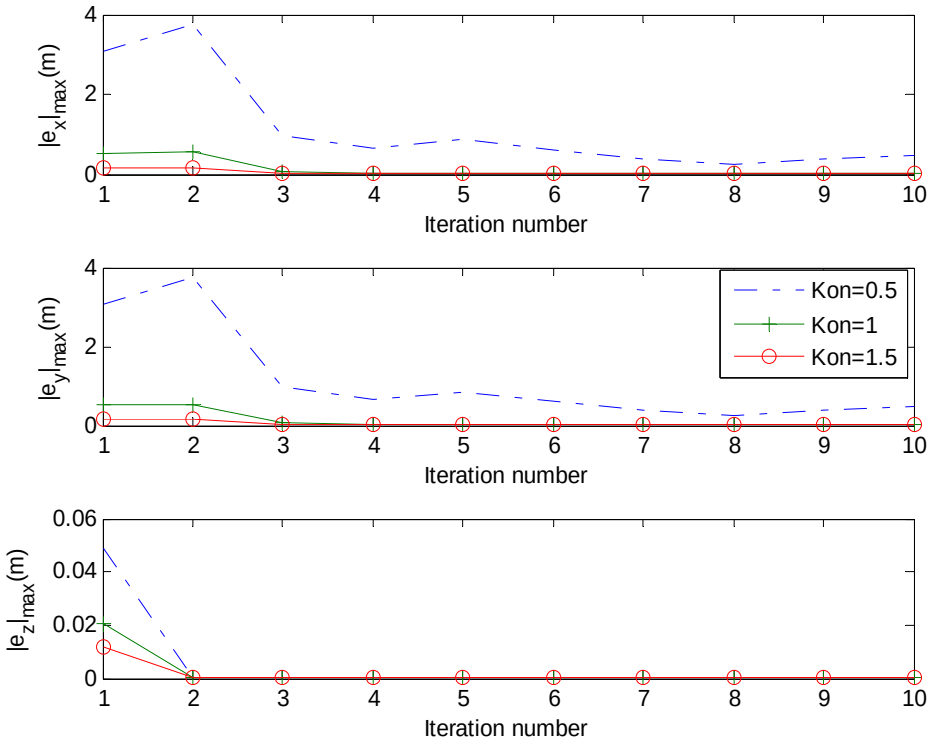

Fig. 5-2 Effect of changing the whole set of control gains by a factor of 0.5 and 1.5

To further demonstrate the effects of online learning gains on inner loop controller, another arbitrarily chosen set of learning gains of $K_{pon} = [0.1\ 0.1\ 35\ 15\ 2\ 75]$ is compared to the standard set of gains and shown in Fig. 5-3. From the figure, it can be seen that in the X direction, the one with the standard set of gains converges faster due to a larger gain corresponding to the θ angle (5 comparing to 2). This is also true in the Y direction, where the one with the standard set converges slower (control gain of 5 comparing to 15), and in the Z direction where the one with the standard set converge faster (control gain of 50 comparing to 35). The trajectory tracking performances of some iterations for PD online ILCs with the

random set of gains are shown in Fig. 5-4. Note that with this set of gains, tracking errors in the yaw direction are relatively large even at iteration 9.

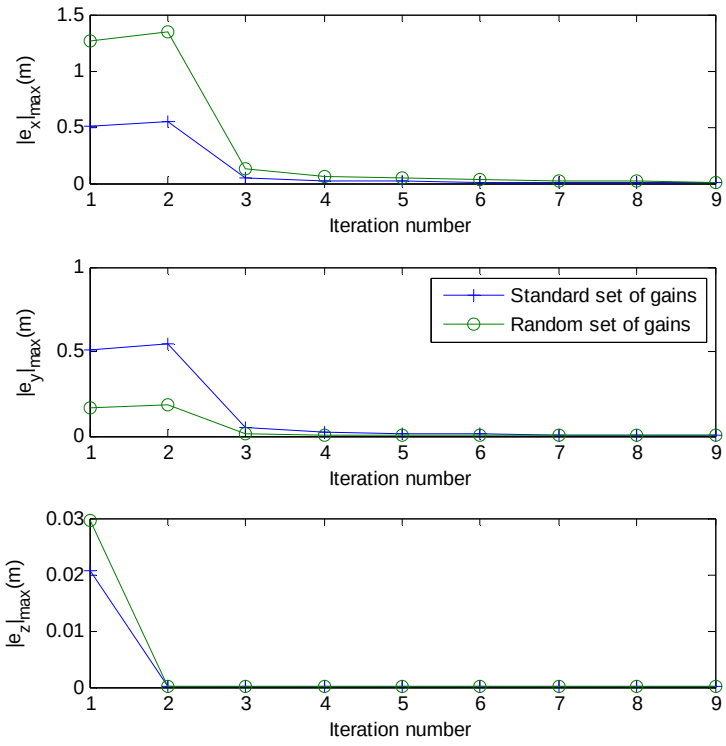

Fig. 5-3 Comparison between maximum errors in each iteration of PD online ILCs with a random set of gains and the standard set of gains

Fig. 5-5 shows the convergence rates of both PD online ILCs and SPD ILCs. In the presence of disturbances, SPD ILCs outperform PD online ILCs with their incredibly fast convergence rates, reaching approximately zero error at iteration 3 while PD online ILCs take 7 iterations. Trajectory tracking performances in X, Y, Z and ψ of PD online ILCs and SPD ILCs are shown in Fig. 5-6 and Fig. 5-7, respectively. It can be seen that the quadrotor can smoothly take-off, with some errors in X-Y directions in the first two iterations then later converges rapidly and SPD ILCs perform better, obviously, in X-Y directions. It is also noticeable that errors in yaw motion for both types are considered small, however, SPD ILCs can reject disturbance in yaw angle much better. Rotor speeds in iteration 10 of PD online ILCs are

depicted in Fig. 5-8, showing that all four rotors operate in the range of maximum 279 rad/s constrained by motor speed and propellers.

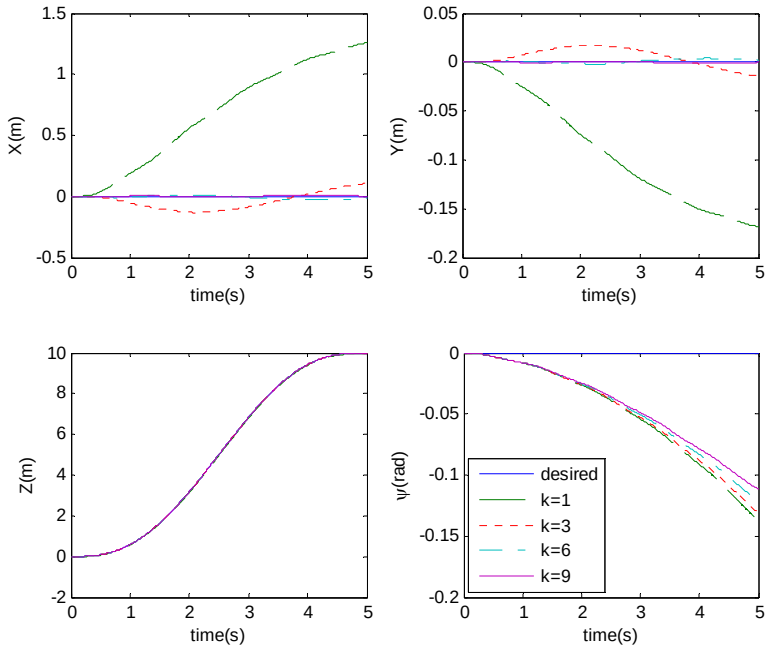

Fig. 5-4 Trajectory tracking performance of PD online ILCs with the random set of gains of K_{pon} =[0.1 0.1 35 15 2 75]

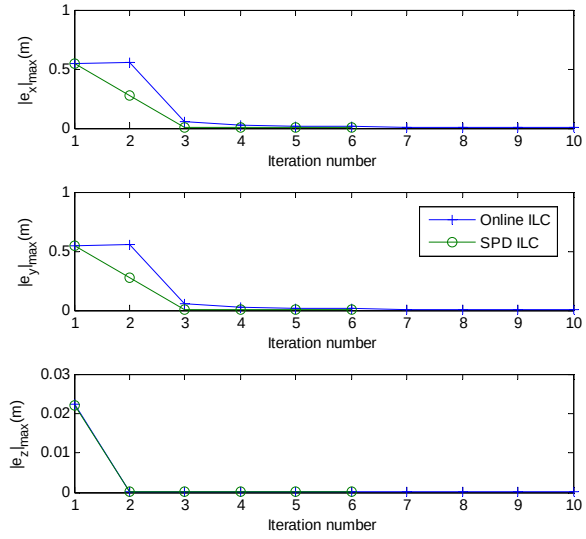

Fig. 5-5 Trajectory tracking performance comparison between PD Online ILCs and SPD ILCs for take-off

A similar simulation in take-off trajectory was performed in [***18***]. In that paper, a trajectory generation method was proposed and controlled by the LQR method and obtained a similar result for this kind of trajectory. However, no disturbances were addressed in that simulation. In addition, the results of the first iterations for both PD and SPD simply are classic PD control. Hence superior performance of online ILCs over classic PD control and LQR method, at least in this example, are proved here.

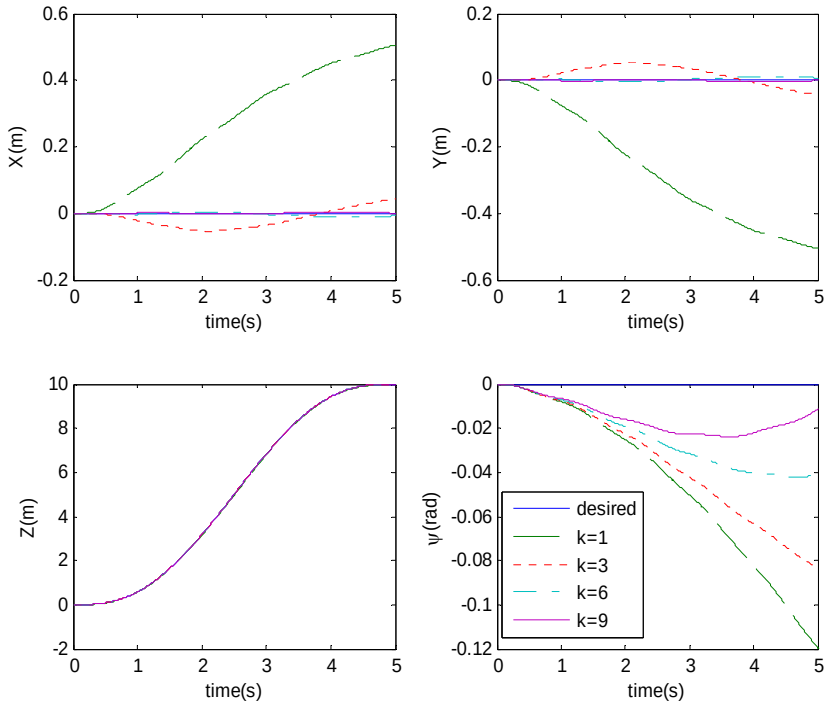

Fig. 5-6 Trajectory tracking performance of PD Online ILCs in iteration 1, 3, 6 and 9 for take-off

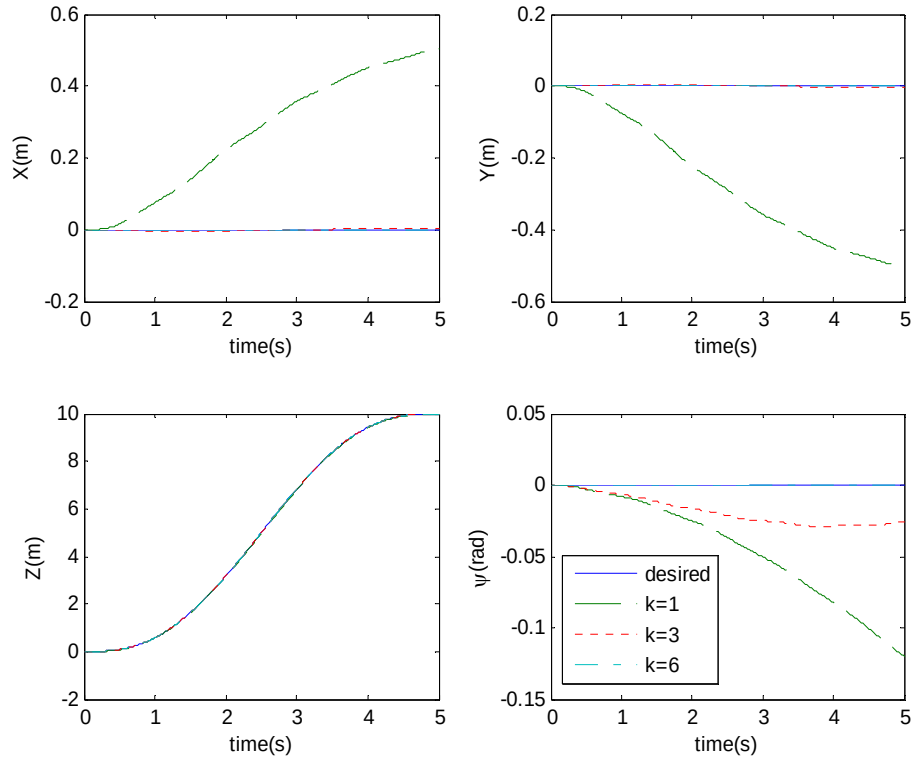

Fig. 5-7 Trajectory tracking performance of SPD ILCs in iteration 1, 3 and 6 for take-off

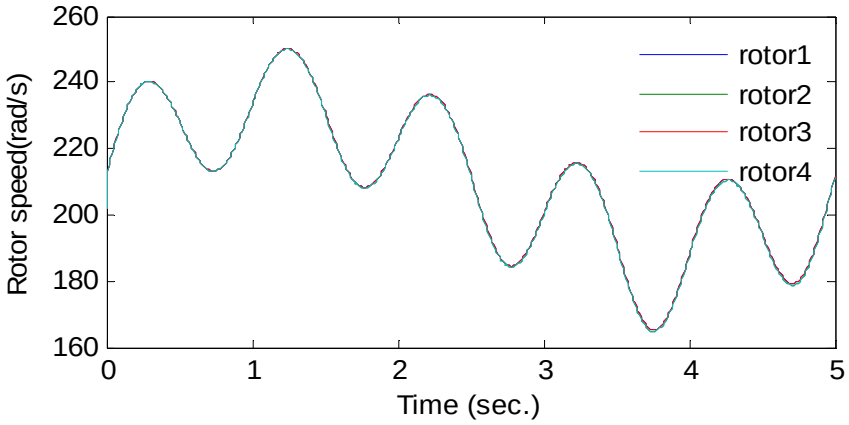

Fig. 5-8 Rotor speed at iteration 10 of PD Online ILCs for take-off

5.3 Landing Tracking

As opposed to take-off, here the quadrotor UAV descends from initial height Z_0 of 10m to 0m within 5s. The desired trajectories, modified from eq. (4.86), are

$$\begin{cases} X_d(t) = 0 \\ Y_d(t) = 0 \\ Z_d(t) = Z_0 \left[1 - \left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5 \right] \right] \end{cases} \tag{5.4}$$

$$Z_d \in [0, Z_0], \qquad t \in [0, T]$$

As seen in Fig. 5-9 Trajectory tracking performance comparison between PD Online ILCs and SPD ILCs for landing, convergence rates are similar to those of take-off. Fig. 5-10 and Fig. 5-11 show tracking performances of both types of ILCs, the quadrotor can land smoothly and again, SPD ILCs outperform PD online ILCs in terms of X-Y and yaw angle errors. Fig. 5-12 confirms that the rotors operate in the limited range.

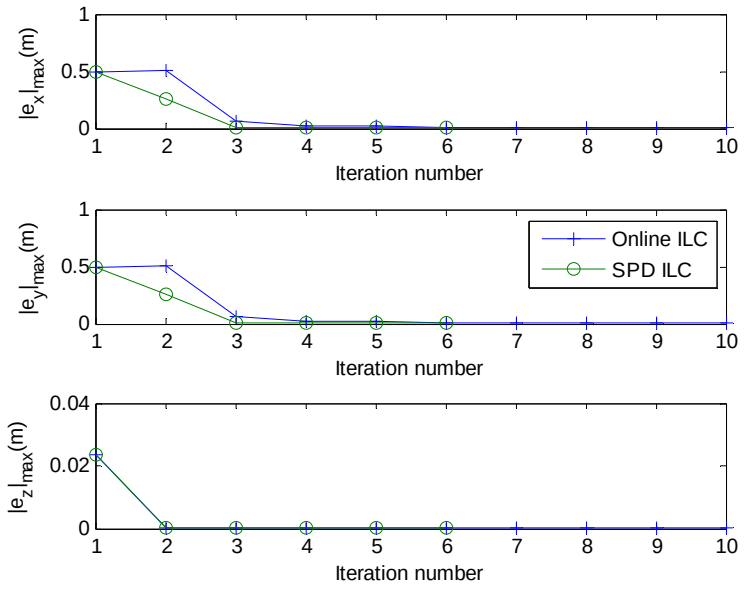

Fig. 5-9 Trajectory tracking performance comparison between PD Online ILCs and SPD ILCs for landing

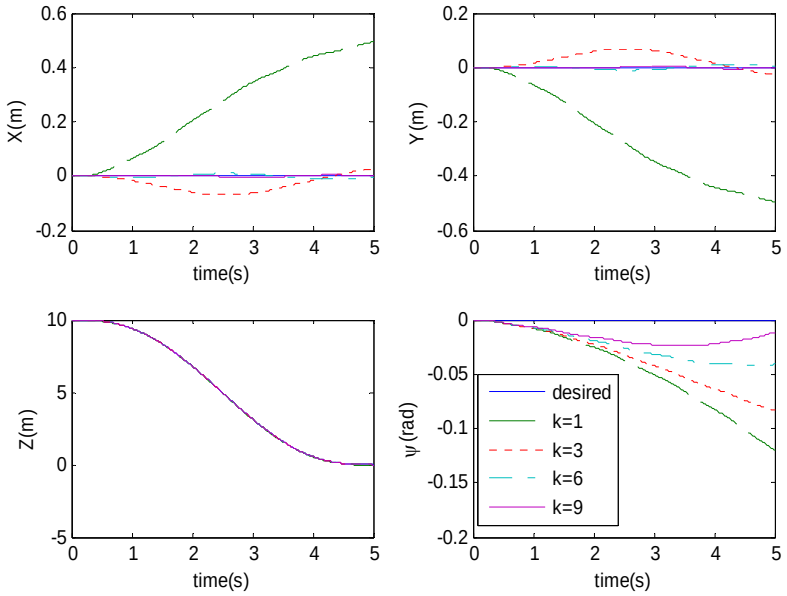

Fig. 5-10 Trajectory tracking performance of PD Online ILCs in iteration 1, 3, 6 and 9 for landing

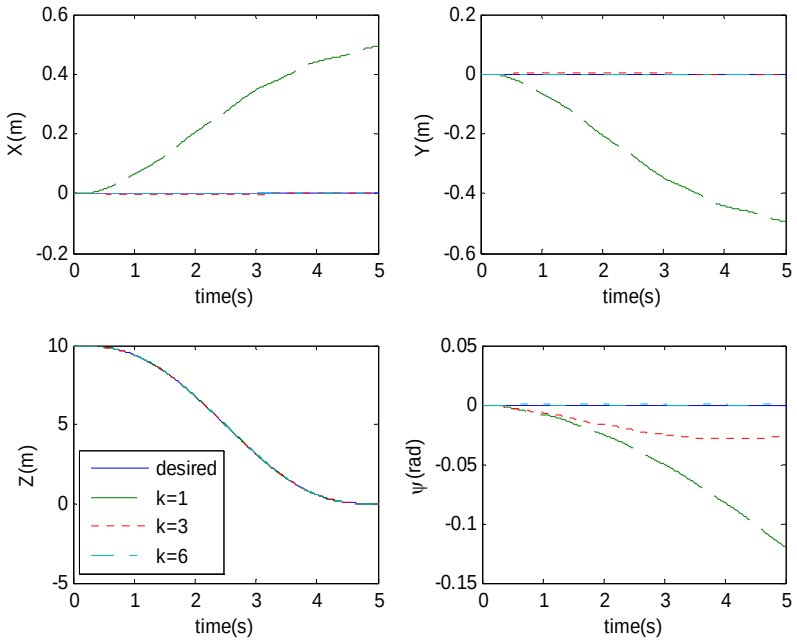

Fig. 5-11 Trajectory tracking performance of SPD ILCs in iteration 1, 3 and 6 for landing

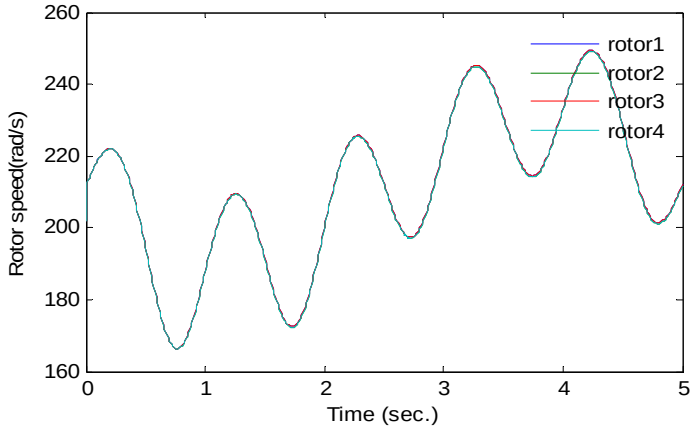

Fig. 5-12 Rotor speed at iteration 10 of PD Online ILCs for landing

5.4 Horizontal Translation

In a similar way to the two previous maneuverings, to demonstrate smooth trajectory tracking performance in X-Y plane translation, the desired trajectories in X and Y direction are chosen as:

$$\begin{cases} X_d(t) = S\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right] \\ Y_d(t) = S\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right] \\ Z_d(t) = 0 \end{cases} \tag{5.5}$$

$$X_d(t) \in [0, S], \qquad Y_d(t) \in [0, S], \qquad t \in [0, T]$$

Two sets of simulations are performed here. First simulation shows short range translation with $S = 10$m and $T = 5$s and the second shows long range translation at higher speed with $S = 100$m and $T = 15$s. For the first set, convergence rates are shown in Fig. 5-13. Here, PD online ILCs tend to take more iterations, as many as 10, to converge to approximate zero errors in X and Y directions while SPD ILCs take 3 iterations and no differences in the Z direction. Trajectory tracking performances for some iteration are shown in Fig. 5-14 and Fig. 5-15, for PD online ILCs and SPD ILCs, respectively. Note that there are relatively large errors in X and Y directions in the first iteration then rapidly converge for both PD online ILCs and SPD ILCs.

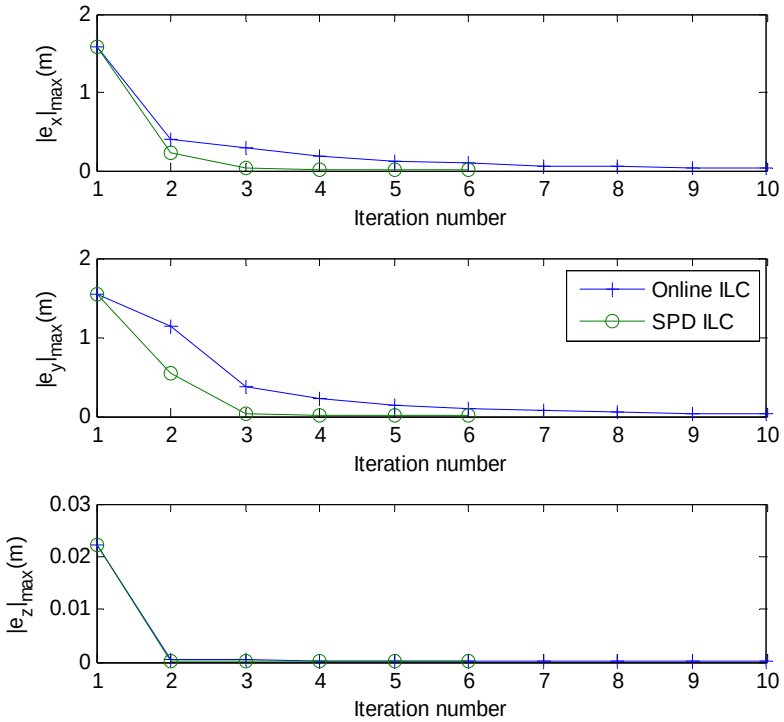

Fig. 5-13 Trajectory tracking performance comparison between PD Online ILCs and SPD ILCs for X-Y translation(S=10m, T=5s)

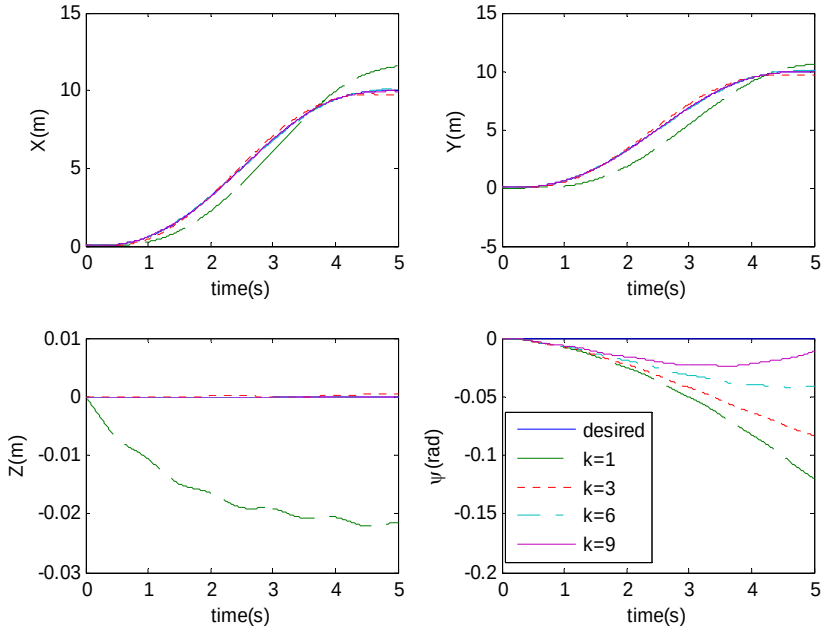

Fig. 5-14 Trajectory tracking performance of PD Online ILCs in iteration 1, 3, 6 and 9 for X-Y translation(S=10m, T=5s)

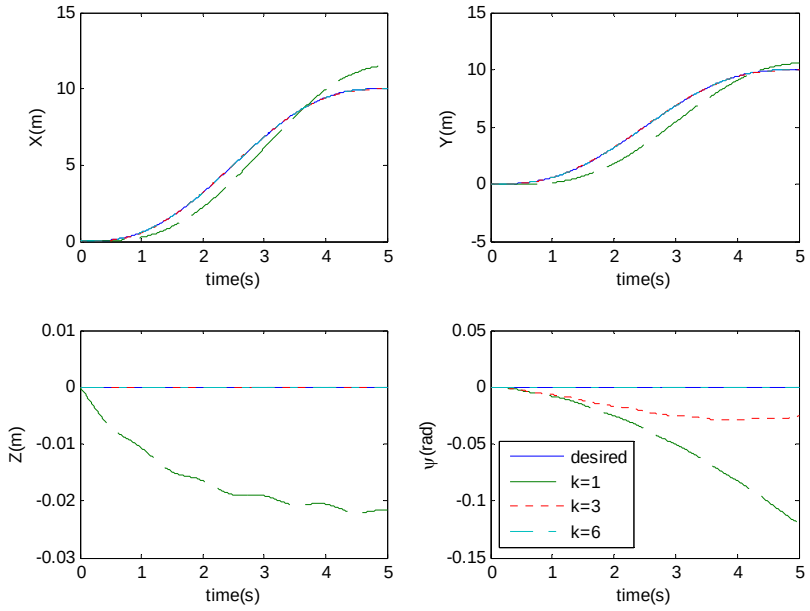

Fig. 5-15 Trajectory tracking performance of SPD ILCs in iteration 1, 3 and 6 for X-Y translation(S=10m, T=5s)

For the second set, PD online ILCs converge faster and track better than those in the first set while SPD ILCs still perform even better, as seen in Fig. 5-16. Although the errors in the first iteration are a bit larger than those in the first set, they are relatively in small comparing to the distance traveled as seen in Fig. 5-17 and Fig. 5-18. Hence the control gains are reasonable for long range translation but increases might be preferable for a short range one. Fig. 5-19 shows rotor speeds for the long range simulation. For both short and long range translation, it is clearly shown that, after a few iterations, the quadrotor UAV can smoothly and perfectly move from point to point in a straight line.

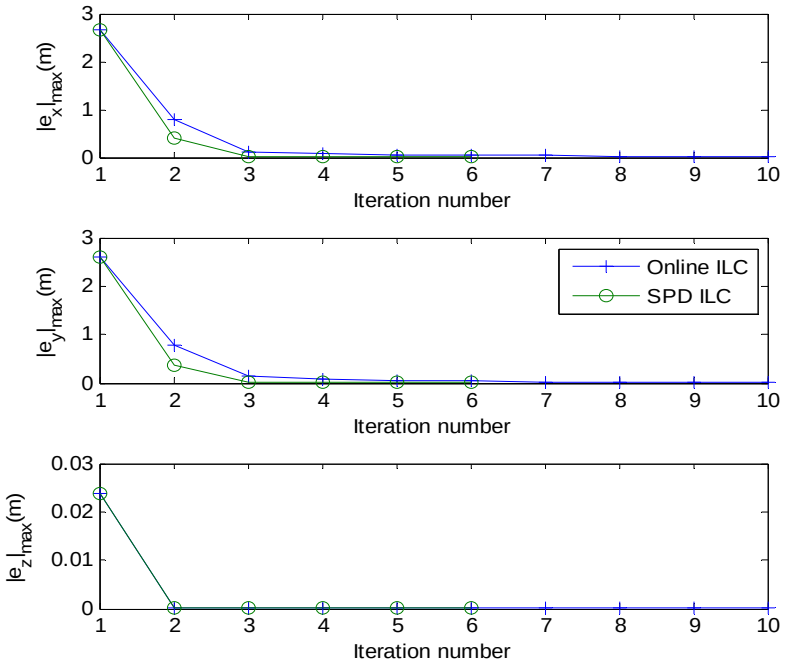

Fig. 5-16 Trajectory tracking performance comparison between PD Online ILCs and SPD ILCs for X-Y translation(S=100m, T=15s)

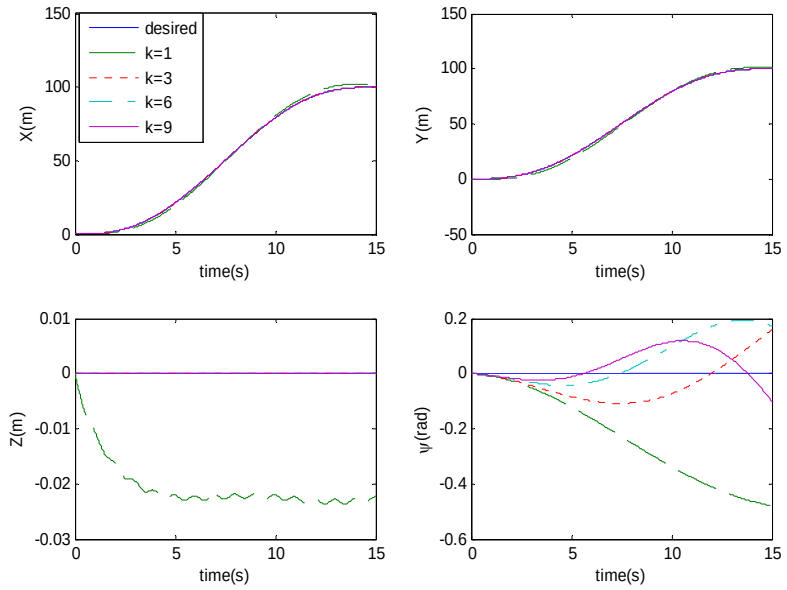

Fig. 5-17 Trajectory tracking performance of PD Online ILCs in iteration 1, 3, 6 and 9 for X-Y translation(S=100m, T=15s)

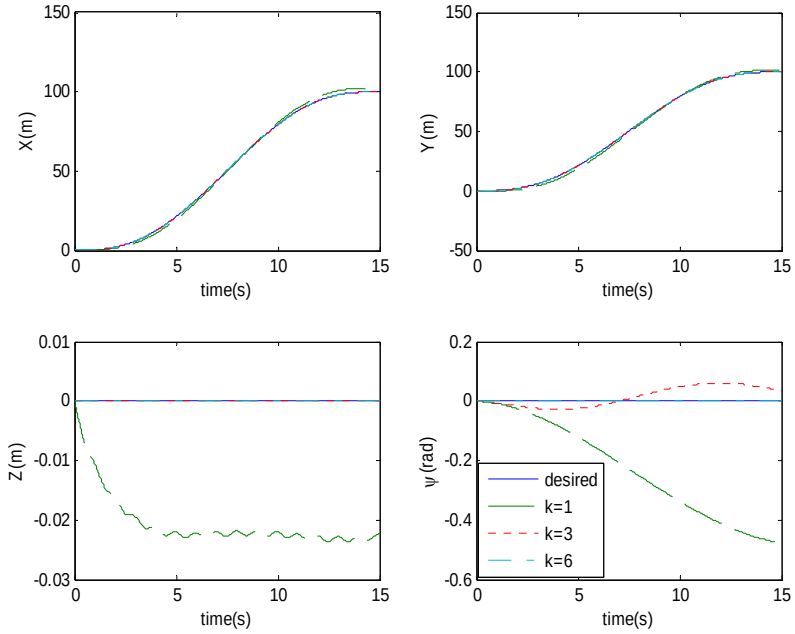

Fig. 5-18 Trajectory tracking performance of SPD ILCs in iteration 1, 3 and 6 for X-Y translation(S=100m, T=15s)

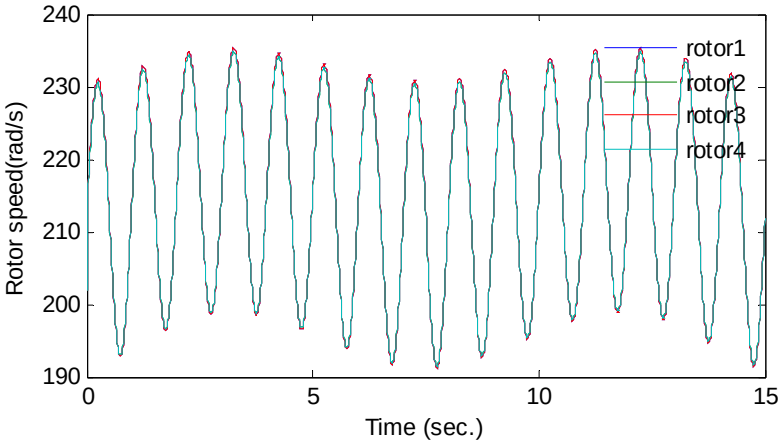

Fig. 5-19 Rotor speed at iteration 6 of SPD ILCs (S=100m, T=15s)

5.5 Horizontal Circular Trajectory

To prove that the quadrotor UAV, controlled by ILCs, can freely maneuver other than a straight line, a circular trajectory is performed. In this mission, the quadrotor revolves around a point in X-Y plane with the radius $R = 20$m within time T =20s, smoothly starting from and ending at the velocity of 0. The desired trajectories are

$$\begin{cases} X_d(t) = Rcos\left(2\pi\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right] + \pi\right) + R \\ Y_d(t) = Rsin\left(2\pi\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right] + \pi\right) \\ Z_d(t) = 0 \end{cases} \tag{5.6}$$

$$X_d(t) \in [0, 2R], \qquad Y_d(t) \in [-R, R], \qquad t \in [0, T]$$

As seen in eq.(5.6), according to the desired trajectories, the desired position starts at 0m and then moves in +X and -Y directions, in the counterclockwise direction with the center of $(x, y) = (R, 0)$ m from the origin while maintaining altitude.

Fig. 5-20, Fig. 5-21 and Fig. 5-22 show convergence rates and tracking performances of PD online ILCs and SPD ILCs for the circular trajectory. It can be observed that both ILCs can

track the trajectory well through iterations, SPD ILCs converge to approximate zero errors at iteration 3 while PD online ILCs take 6 iterations and SPDs perform much better in yaw angle.

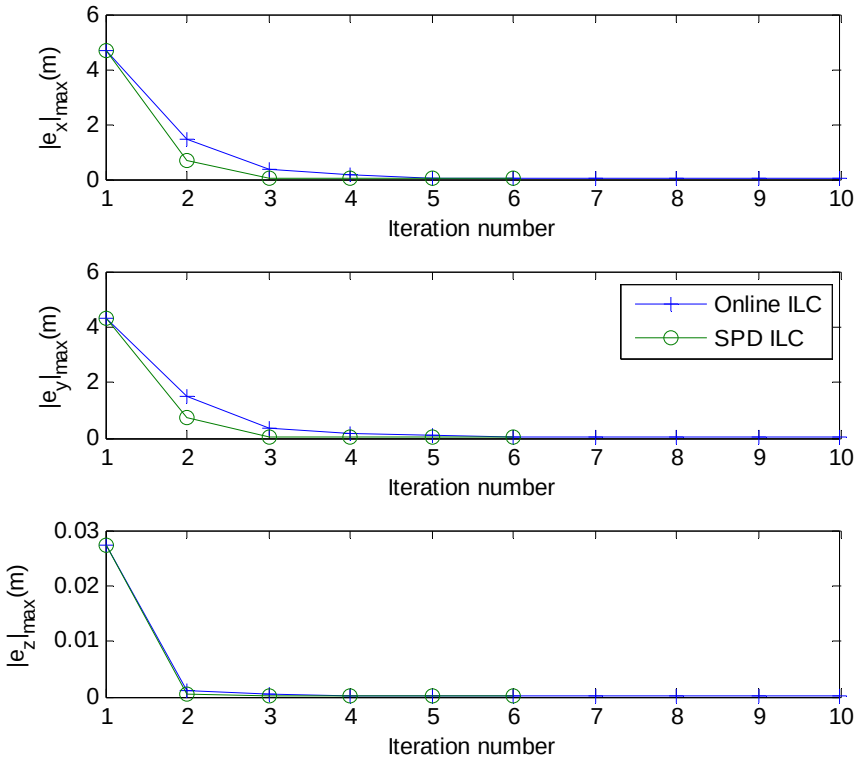

Fig. 5-20 Trajectory tracking performance comparisons between PD Online ILCs and SPD ILCs for horizontal circular trajectory

A similar simulation of circular trajectory was performed in [***58***]. The paper adopted different model parameter and utilized backstepping control for trajectory tracking. But the trajectory was a function of exponential of time rather than polynomial. In the simulation circular path was in X-Y plane with the radius of 1m and period of 6s while increasing altitude and rotating in the yaw angle. It was proved to track the trajectory very well but no disturbances were taken into account. Furthermore, the radius was very small comparing to 20m in this thesis. Thus, again, implies better performance and robustness of online ILCs over backstepping method used in that paper.

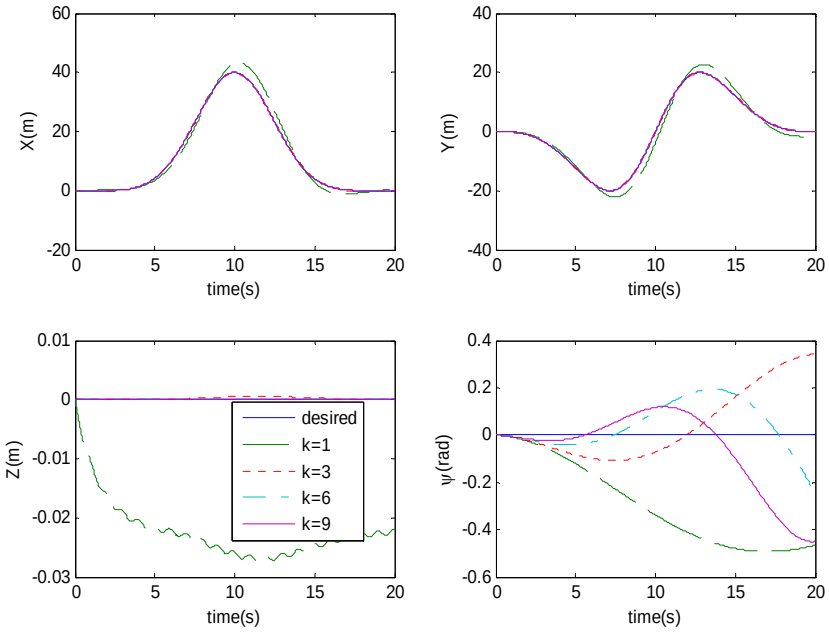

Fig. 5-21 Trajectory tracking performance of PD Online ILCs in iteration 1, 3, 6 and 9 for horizontal circular trajectory

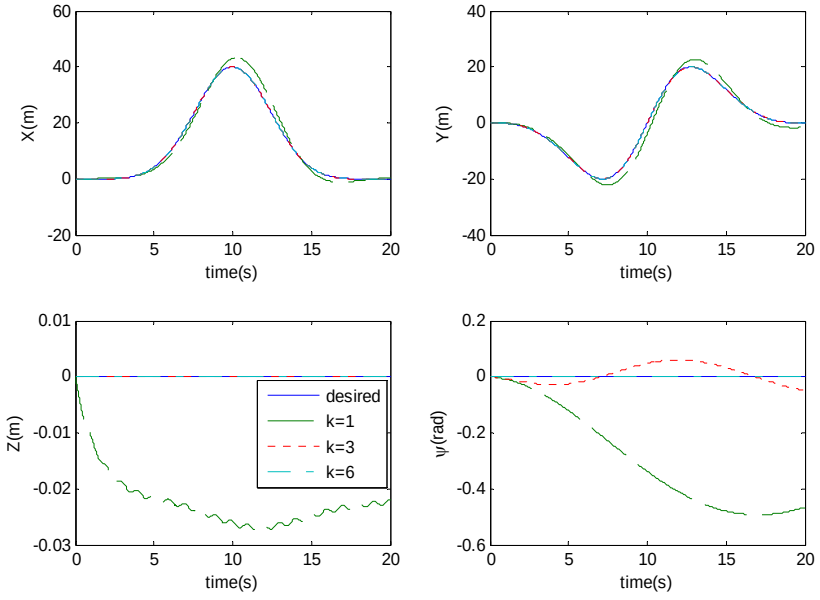

Fig. 5-22 Trajectory tracking performance of SPD ILCs in iteration 1, 3 and 6 for horizontal circular trajectory

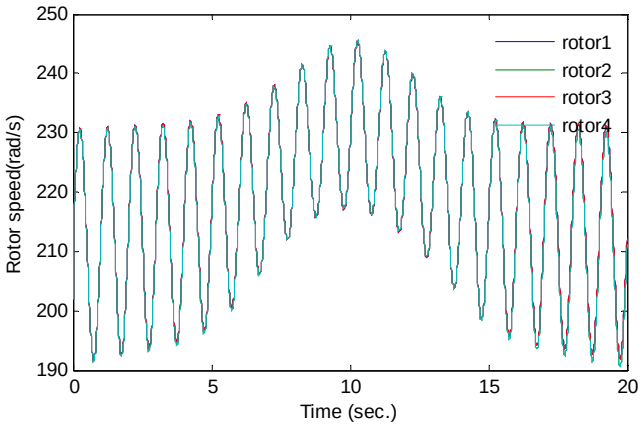

Fig. 5-23 Rotor speed at iteration 6 of SPD ILCs for horizontal circular trajectory

5.6 Spatial Circular Trajectory

Ultimately in this mission, the quadrotor UAV is commanded to perform a circular motion in 3D, assumingly from hovering state. This is the same as horizontal circular trajectory except a sinusoidal trajectory is added in the Z direction instead of being set to 0. Hence the desired trajectories in all three directions can be written as:

$$\begin{cases} X_d(t) = Rcos\left(2\pi\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right] + \pi\right) + R \\ Y_d(t) = Rsin\left(2\pi\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right] + \pi\right) \\ Z_d(t) = \frac{R}{2}cos\left(2\pi\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right]\right) + 1.5\frac{R}{2} \end{cases} \quad (5.7)$$

$$X_d(t) \in [0, 2R], \qquad Y_d(t) \in [-R, R]$$
$$Z_d(t) \in [0.5R, 2.5R], \qquad t \in [0, T]$$

Note that, from eq. (5.7), the radius of translation in the Z direction is half of those in X and Y directions in order to keep the rotor speeds within limit constrained by the model parameter while proving the capability of this mission. Furthermore, since it is assumed that the

quadrotor UAV perform this mission starting from hover, the lower bound of altitude is chosen as $0.5R$ to keep the quadrotor off the ground.

Comparison between trajectory tracking errors of PD online ILCs and SPD ILCs for the spatial circular trajectory are shown in Fig. 5-24. In the same way as in the horizontal circular trajectory, maximum errors in X and Y directions in the first iteration are relatively large and then converge quickly from iteration 2. Again, SPD ILCs show a faster convergence rate than that of PD online ILCs. Tracking errors in the Z direction are very small, which are similar to those of previous mission despite the circular motion in the Z direction.

Fig. 5-25 and Fig. 5-26 show trajectory tracking for at some iterations for PD online ILCs and SPD ILCs, respectively. The results of both ILCs are very similar in X and Y directions and are almost identical in the Z direction. However, as in previous mission, SPD ILCs outperform PD online ILCs in terms of convergence rates in the yaw direction. Proof of operation within rotor speed limit is shown in Fig. 5-27.

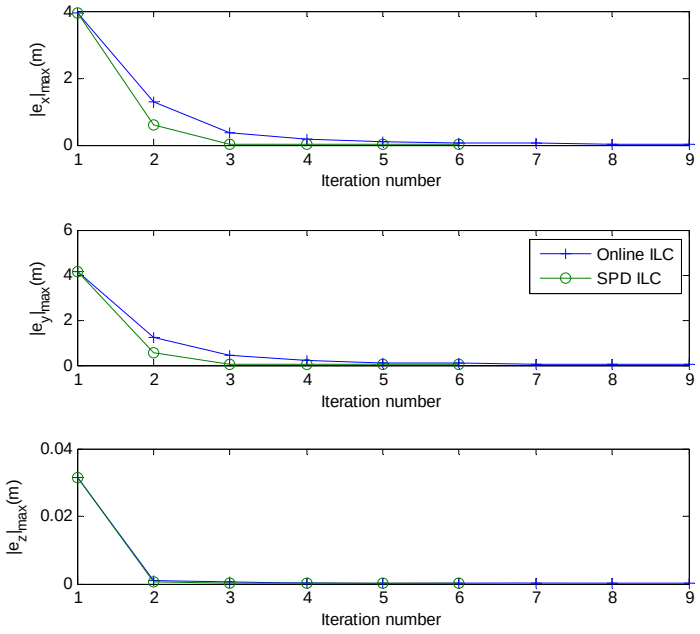

Fig. 5-24 Trajectory tracking performance comparisons between PD Online ILCs and SPD ILCs for spatial circular trajectory

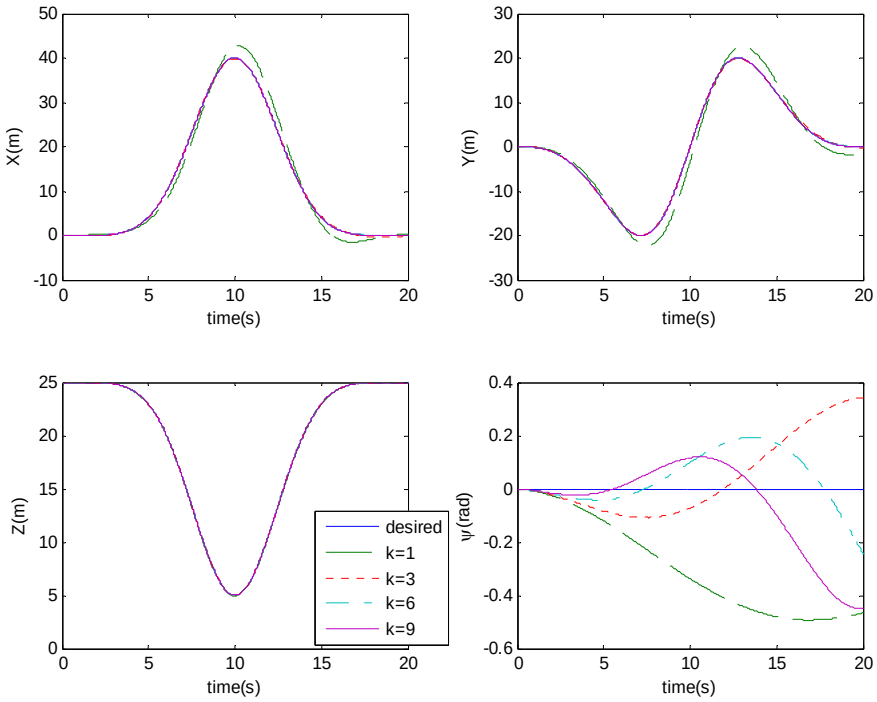

Fig. 5-25 Trajectory tracking performance of PD Online ILCs in iteration 1, 3, 6 and 9 for spatial circular trajectory

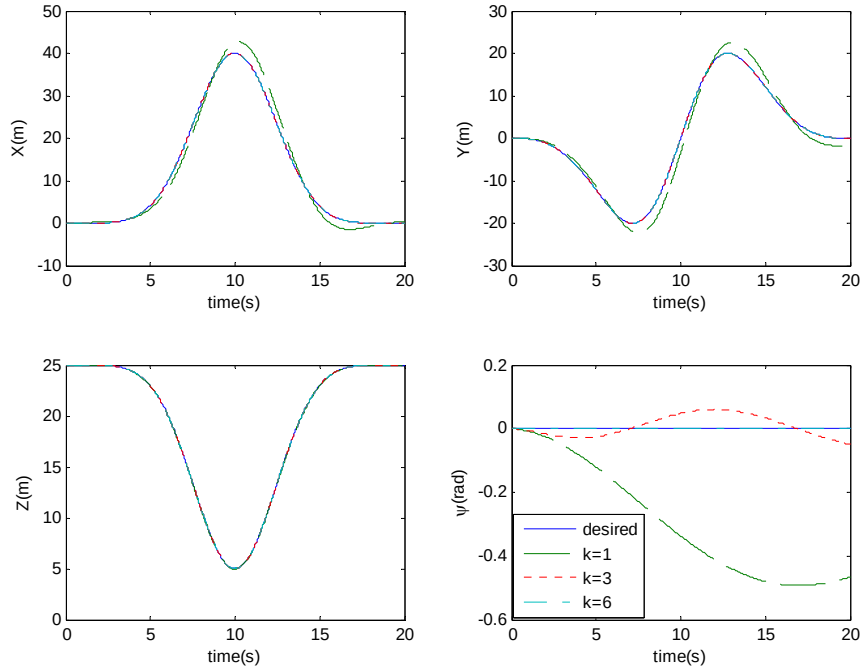

Fig. 5-26 Trajectory tracking performance of SPD ILCs in iteration 1, 3 and 6 for spatial circular trajectory

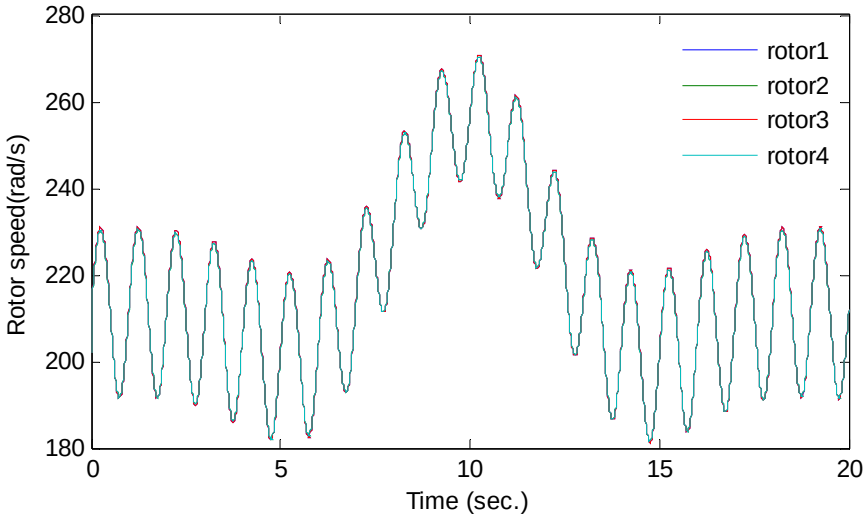

Fig. 5-27 Rotor speed at iteration 6 of SPD ILCs for spatial circular trajectory

5.7 Concluding Remarks

In this chapter, simulation results were presented. Firstly, disturbances used in the simulations were introduced and discussed. Sub-optimal control gains obtained via trial and error were proposed and adopted for every simulation. The simulation results showed that the quadrotor UAV model was capable of performing various missions including take-off, landing, X-Y plane translation, horizontal and spatial circular trajectory tracking in the presence of relatively large sinusoidal disturbances in many directions. For take-off, variations in each control gain in both P and D parts were experimented and shown, clarifying the effect of each gain on the tracking performance. The simulation of take-off also implied better performance of online ILCs over classic PID control and LQR control while circular trajectory simulations showed better robustness over the backstepping method. In general, simulation results showed that the online ILCs could perform the aforementioned maneuverings well in a timely manner and were robust against fairly large disturbances with decreasing tracking errors over iterations, while operating within the limitations of actuators used in the real application from the referred paper. SPD ILCs were shown to converge much faster than PD online ILCs.

Chapter 6 Conclusion & Discussion

This chapter simply generalizes and summarized concepts, studies and the obtained simulation results. This chapter is organized as beginning with general review of this thesis in Section 6.1, followed by main contributions of the work done in Section 6.2 and finally future development, which could possibly be further investigated, in Section 6.3.

6.1 General Review

Since advancement in technology decades ago, numerous applications became available. Significant development in UAVs occurred and led to exploitation of studies in VTOL UAVs including quadrotors which are known to be very agile and simple and therefore capable of various missions. However, their agility result in low stability and under-actuated system and hence stabilization and control are needed.

This thesis aims to propose a quadrotor UAV trajectory tracking control using online learning approach for its simplicity and robustness against uncertainties and disturbances then verified the developed control system simulation based on a nonlinear dynamic model. In Chapter 2, literature reviews are presented. This includes general concepts of UAVs, introduction of quadrotors, investigation in past studies on quadrotor UAVs, control design of UAVs, and development of ILCs.

Chapter 3 presents quadrotor modeling and its detailed analysis along with fundamental assumptions. An appropriate rotation matrix is adopted to simplify the dynamic model. Analysis on quadrotor motions implies coupling behaviors of the model. The dynamic model is then derived using the Newton-Euler method with basic aerodynamic and gyroscopic effects considered. The model parameters are based on a well-designed real quadrotor UAV.

In Chapter 4, detailed concepts, utilization and analyses are established for ILCs. Various types of ILCs are introduced along with convergence analysis. P-type, D-type, PD-type, offline, online, online-offline and SPD ILCs comparisons are also shown. PD online and SPD ILCs are chosen to be implemented to the quadrotor model. The control gains correspond to

six control laws for three translational and three rotational direction motions. A 5^{th} order polynomial function is used to generate trajectory for all maneuver missions.

Chapter 5 presents simulation results and cites some simulations performed in other papers. Disturbances and standard control gains derived from trial and error are used. Four missions are tested which includes take-off, landing, horizontal translations in straight line and circle and spatial circular trajectory. Variation in control gains showing effect on each control gain is also presented. The simulation results showed the robustness of trajectory tracking performance for both PD online and SPD ILCs. Incredibly fast convergence rates can be obtained using SPD ILCs. Both types show better performance and robustness, through learning, over some control methods tested in other studies.

6.2 Main Contributions

Quadrotor UAV controlling has been tested and tried with various methods in literatures, successfully and unsuccessfully. However, most studies did not address disturbance rejection and only a few dealt with full control of the system. So far this thesis is the first to utilize the ILC method for trajectory tracking control for a quadrotor UAV. Although unexposed to such an application, ILCs have been widely known for their simplicities and robustness and fit well in repeating tasks such as robotic manipulations. Main contribution of this work can be summarized as

- opening doors for further development in ILC-implemented quadrotor UAV control
- proposing a simple alternative method of quadrotor UAV control where model uncertainties or lack of complicated aerodynamic effects consideration are primary concern
- proving robustness and fast convergence rates of online ILCs in harsh conditions
- increasing versatilities of ILCs, i.e., not limited only to traditional applications

6.3 Future Development

The work done in this thesis is considered a preliminary step toward complete implementation on a real application for quadrotor UAVs. It shows the possibility of controlling a quadrotor UAV using the simple and robust online ILCs and proved to be promising via simulations based on parameters of a real well-designed application. Yet, some issues for future development can be summed up as follows:

- Since ILCs deal with trajectory tracking problems, they are not very efficient when it comes to point control. That means ILCs are not a good alternative to stabilize a quadrotor and to maintain position (although they can but with some errors). The solution could be applying readily studied stabilization methods investigated in many researches such as state-feedback control or LQR control, etc. This could be a good compensation of each other's advantages and drawbacks.

- The simulation results here were done based on certain trajectories, which means the quadrotor can perform various, but limited, missions. Adding more trajectory tracking might increase the capability of more missions. However, utilizing online optimal trajectory generation seems to be the best way to achieve full autonomy that the quadrotor can choose its own path of traveling. This has been already done in some researches.

- SPD ILCs were proved to yield very fast convergence rates and small errors. However, as they iterate to a certain cycle where errors are very small, noises are then easily detected and largely amplified, resulting in instability. This can be easily solved by shutting down switching behavior when errors are very small and turn it on again when disturbances are vast.

Appendix: MATLAB Code for Simulation

In this thesis, there are two separate MATLAB codes for simulation. One is the main program that calls another program for calculating errors, choosing simulation modes, entering and calling current iteration data needed in next iteration. The other one is a function that mainly contains dynamic model and disturbances. The two programs are shown subsequently.

A1. Main Program

```
clear all
global kp
global kd
global torque
global iter
global noise_level
global g
global a1 a2 a3 a4 a5 a11 a22 a33 m b
global Om_r x0 zd k R w
global duration
global t_mode d_mode

%% MODEL PARAMETERS
par_store = [0.232 0.52 7.5e-7 3.13e-5 6.228e-3 6.228e-3 1.121e-2 6e-5];
%sets of parameters

par = par_store(1,:); %select parameter set
l = par(1);
m = par(2);
d = par(3);
b = par(4);
Ix = par(5);
Iy = par(6);
Iz = par(7);
Jr = par(8);

a1 = (Iy-Iz)/Ix;
a2 = Jr/Ix;
a3 = (Iz-Ix)/Iy;
a4 = Jr/Iy;
a5 = (Ix-Iy)/Iz;
a11 = l/Ix;
a22 = l/Iy;
a33 = d/Iz;

C = [1 1 1 1;0 -1 0 1;1 0 -1 0;1 -1 1 -1];
invC = inv(C);
g = 9.81;
%% INITIAL SETTINGS
```

```
s_mode = [0 1 0]; %simulation mode [offline online spd] : 0 = OFF, 1 = ON

t_mode = 2; %trajectory mode : 1 = X-Y, 2 = take-off, 3 = landing, 5 = 3D
circular, 6 = 2D circular

d_mode = 1; %disturbance mode : 0 = no noise, 1 = repetitive noise, 2 =
random noise

noise_level = 1;% 0 = no noise, 1 = normal noise

k_off = [0.05 0.05 5 0.5 0.5 0];
kp_off = 0.001*s_mode(1)*k_off;
kd_off = 0.005*s_mode(1)*k_off;

k_on = [0.1 0.1 35 15 2 75];%online learning gain : ux uy u1...u4
kp = s_mode(2)*k_on;
kd = 1.5*kp;
K = 1;
kp = K*kp;
kd = K*kd;

iter_no = 9; %number of iterations
duration = 5;
samp = 0.01; %sampling time

x3i = 0;
x7i = 0;

k = 100; %for t_mode =1
R = 20; %radius for t_mode =5 and 6
w = 2*pi; %omega for t_mode =5 and 6

if t_mode == 2
    zd = 10; %desired height
elseif t_mode == 3
    x3i = 10;
elseif t_mode == 4
    x7i = -12.5;
elseif t_mode == 5
    x3i = 2.5*R/2;
end

x0 = [0 0 x3i 0 0 0 0 x7i 0 0 0 0]; %initial states
Om_r = 0; %initial omega_r

%% SIMULATION COMMENCING

tp = 0:samp:duration; %simulation time
num = length(tp);
torque = zeros(num,9);
torque(:,1)=tp';
x = zeros(num,12);
xd = x;
dx7 = zeros(num,1);
dx8 = dx7;
dx7d = dx7;
dx8d = dx7;

file3=fopen('max_QR','w');
```

```
file4=fopen('x_xd_QR','w');

for iter=1:iter_no
    if iter == 1
        torque(1,2) = g*m/b;
        kp = K*k_on;
        kd = K*1.5*kp;
    else
        if s_mode(1) == 1
            kp = s_mode(2)*kp;
            kd = s_mode(2)*kd;
        end
        if s_mode(3) == 1
            s = iter*[1 1]; %switching gain
            kp = s(1)*kp;
            kd = s(2)*kd;
        end
        file1=fopen('t_QR','r');% read the torque data from the data file
        aa=fscanf(file1,'%f %e %e %e %e %e %e %e %e',[9,inf]);
        fclose(file1);
        torque=aa';
    end

    options=odeset('RelTol', 1e-5, 'AbsTol', 1e-5*[1 1 1 1 1 1 1 1 1 1
1]);
    [tp,x]=ode23tb('ilc_QR_model',tp,x0,options);

    for i=1:num
        t=tp(i);
%% TRAJECTORIES

        if t_mode == 1
            tr = t/duration;
            xd(i,1) = k*(10*tr^3-15*tr^4+6*tr^5); %Xd
            xd(i,2) = k*(10*tr^3-15*tr^4+6*tr^5); %Yd
            xd(i,3) = 0; %Zd
            xd(i,7) = k*(30*tr^2-60*tr^3+30*tr^4)/duration; %dXd
            xd(i,8) = k*(30*tr^2-60*tr^3+30*tr^4)/duration; %dYd
            xd(i,9) = 0; %dZd
            dx7d(i) = k*(60*tr-180*tr^2+120*tr^3)/duration^2;  %ddXd;
            dx8d(i) = k*(60*tr-180*tr^2+120*tr^3)/duration^2;  %ddYd;
        elseif t_mode == 2
            tr = t/duration;
            xd(i,1) = 0;
            xd(i,2) = 0;
            xd(i,3) = zd*(10*tr^3-15*tr^4+6*tr^5);
            xd(i,7) = 0;
            xd(i,8) = 0;
            xd(i,9) = zd*(30*tr^2-60*tr^3+30*tr^4)/duration;
            dx7d(i) = 0;
            dx8d(i) = 0;
        elseif t_mode == 3
            z0 = x0(3);
            tr = t/duration;
            xd(i,1) = 0;
            xd(i,2) = 0;
            xd(i,3) = z0*(1-(10*tr^3-15*tr^4+6*tr^5));
            xd(i,7) = 0;
            xd(i,8) = 0;
            xd(i,9) = -z0*(30*tr^2-60*tr^3+30*tr^4)/duration;
```

```
            dx7d(i) = 0;
            dx8d(i) = 0;
        elseif t_mode == 5
            tr = t/duration;
            k = 10*tr^3-15*tr^4+6*tr^5;
            kdot = (30*tr^2-60*tr^3+30*tr^4)/duration;
            kddot = (60*tr-180*tr^2+120*tr^3)/duration^2;
            ang = w*k+pi;
            xd(i,1) = R*cos(ang)+R;
            xd(i,2) = R*sin(ang);
            xd(i,3) = R/2*cos(w*k)+1.5*R/2;
            xd(i,7) = -w*R*sin(ang)*kdot;
            xd(i,8) = w*R*cos(ang)*kdot;
            xd(i,9) = -w*R/2*sin(w*k)*kdot;
            dx7d(i) = -R*((w*kdot)^2*cos(ang)+w*kddot*sin(ang));  %ddXd;
            dx8d(i) = -R*((w*kdot)^2*sin(ang)-w*kddot*cos(ang));  %ddYd;
        elseif t_mode == 6
            tr = t/duration;
            k = 10*tr^3-15*tr^4+6*tr^5;
            kdot = (30*tr^2-60*tr^3+30*tr^4)/duration;
            kddot = (60*tr-180*tr^2+120*tr^3)/duration^2;
            ang = w*k+pi;
            xd(i,1) = R*cos(ang)+R;
            xd(i,2) = R*sin(ang);
            xd(i,3) = 0; %Zd
            xd(i,7) = -w*R*sin(ang)*kdot;
            xd(i,8) = w*R*cos(ang)*kdot;
            xd(i,9) = 0; %dZd
            dx7d(i) = -R*((w*kdot)^2*cos(ang)+w*kddot*sin(ang));  %ddXd;
            dx8d(i) = -R*((w*kdot)^2*sin(ang)-w*kddot*cos(ang));  %ddYd;
        end
    %% CONTROL LAWS ASSIGNING

        u1 = torque(i,2)+g*m/b*(kp(3)*(xd(i,3)-x(i,3))+kd(3)*(xd(i,9)-
    x(i,9))); %u1
        dx7(i) = -sin(x(i,5))*cos(x(i,4))*b/m*u1; %x_dd
        dx8(i) = sin(x(i,4))*b/m*u1;  %y_dd

        ux = torque(i,6)+kp(1)*(xd(i,1)-x(i,1))+kd(1)*(xd(i,7)-x(i,7));
        uy = torque(i,7)+kp(2)*(xd(i,2)-x(i,2))+kd(2)*(xd(i,8)-x(i,8));
        dux = torque(i,8)+kp(1)*(xd(i,7)-x(i,7))+kd(1)*(dx7d(i)-dx7(i));
        duy = torque(i,9)+kp(2)*(xd(i,8)-x(i,8))+kd(2)*(dx8d(i)-dx8(i));

        lim_uy = 0.95;
        if uy > lim_uy
            uy = lim_uy;
        elseif uy < -lim_uy
            uy = -lim_uy;
        end
        Cx4 = cos(x(i,4));
        Tx4 = tan(x(i,4));
        uxCx4 = ux/Cx4;

        lim_uxCx4 = 0.95;
        if uxCx4 > lim_uxCx4
            uxCx4 = lim_uxCx4;
        elseif uxCx4 <-lim_uxCx4
            uxCx4 = -lim_uxCx4;
        end
        xd(i,4) = asin(uy); %desired phi
```

```
        xd(i,5) = -asin(uxCx4);%desired theta
        xd(i,6) = 0; %desired psi
        xd(i,10) = 1/sqrt(1-uy^2)*duy; %desired phi_dot
        xd(i,11) = -1/sqrt(1-uxCx4^2)*(dux+Tx4*x(i,10)*ux)/Cx4;%desired
theta_dot
        xd(i,12) = 0; %desired psi_dot

        torque(i,2)= u1+g*m/b*(kp_off(3)*(xd(i,3)-
x(i,3))+kd_off(3)*(xd(i,9)-x(i,9)));
        u(1,1) = u1;
        for j = 3:5
            torque(i,j)=((kp(j+1)+kp_off(j+1))*(xd(i,j+1)-
x(i,j+1))+(kd(j+1)+kd_off(j+1))*(xd(i,j+7)-x(i,j+7)))+torque(i,j); %u1...u4
            u(j-1,1) = torque(i,j);
        end
        torque(i,6) = ux+kp_off(1)*(xd(i,1)-x(i,1))+kd_off(1)*(xd(i,7)-
x(i,7));
        torque(i,7) = uy+kp_off(2)*(xd(i,2)-x(i,2))+kd_off(2)*(xd(i,8)-
x(i,8));
        torque(i,8) = dux+kp_off(1)*(xd(i,7)-x(i,7))+kd_off(1)*(dx7d(i)-
dx7(i));
        torque(i,9) = duy+kp_off(2)*(xd(i,8)-x(i,8))+kd_off(2)*(dx8d(i)-
dx8(i));
        Om(:,i) = sqrt(inv(C)*u);
        Om_r = Om(1,i)-Om(2,i)+Om(3,i)-Om(4,i);
    end
    z(:,1)=tp';
    for i = 2:4
        z(:,i) = xd(:,i-1)-x(:,i-1); %ex ey ez
        z(:,i+3) = x(:,i+2)*180/pi; %phi theta psi (deg)
    end
    for i = 1:4
        z(:,i+7) = Om(i,:); %rotor speed 1-4
    end

    max_par=max(abs(z));

    file2=fopen('t_QR','w');
    for i = 1:9
        tor1(i,:)=torque(:,i)';
    end
    fprintf(file2,'%7.4f %7.4e %7.4e %7.4e %7.4e %7.4e %7.4e %7.4e
%7.4e\n',tor1);
    fclose(file2);
    fprintf(file3,'%5d %7.4f %7.4e %7.4e %7.4e %7.4e %7.4e %7.4e %7.4e
%7.4e %7.4e %7.4e\n',iter, max_par);
    zz = [iter*ones(num,1),z(:,1),x(:,1:6),xd(:,1:6)];
    if iter ==1 || mod(iter,3) ==0
        fprintf(file4,'%5d %7.2f %7.4e %7.4e %7.4e %7.4e %7.4e %7.4e %7.4e
%7.4e %7.4e %7.4e %7.4e %7.4e\n',zz');
    end

    iter
end
```

A2. Called Function

```
function dx=ilc_QR_model(t,x)

global kp
global kd
global torque
global g
global a1 a2 a3 a4 a5 a11 a22 a33 m b
global Om_r x0 zd k R w
global duration
global noise_level
global t_mode d_mode

dx=zeros(12,1);
xd = dx;
du = zeros(4,1);
nt=size(torque,1);
for i=1:nt
    if torque(i,1)>=t
        break
    end
end

if d_mode == 0
    d1 = 0;
    d2 = 0;
    d3 = 0;
    d4 = 0;
elseif d_mode == 1
    f = 2*pi;
    D = noise_level*(10+20*sin(f*t));
    d1 = 0.1*D;
    d2 = D;
    d3 = d2;
    d4 = 0.001*D;
else
    d1 = 10e-9*2*noise_level*(rand-0.5);
    d2 = 0;
    d3 = 0;
    d4 = 0;
end

for j =1:8
    du(j)=torque(i,j+1); %u1...u4,ux,uy,dux,duy
end

if t_mode == 1
    tr = t/duration;
    xd(1) = k*(10*tr^3-15*tr^4+6*tr^5); %Xd
    xd(2) = k*(10*tr^3-15*tr^4+6*tr^5); %Yd
    xd(3) = 0; %Zd
    xd(7) = k*(30*tr^2-60*tr^3+30*tr^4)/duration; %dXd
    xd(8) = k*(30*tr^2-60*tr^3+30*tr^4)/duration; %dYd
    xd(9) = 0; %dZd
    dx7d = k*(60*tr-180*tr^2+120*tr^3)/duration^2;  %ddXd;
    dx8d = k*(60*tr-180*tr^2+120*tr^3)/duration^2;  %ddYd;
elseif t_mode == 2
```

```
    tr = t/duration;
    xd(1) = 0;
    xd(2) = 0;
    xd(3) = zd*(10*tr^3-15*tr^4+6*tr^5);
    xd(7) = 0;
    xd(8) = 0;
    xd(9) = zd*(30*tr^2-60*tr^3+30*tr^4)/duration;
    dx7d = 0;
    dx8d = 0;
elseif t_mode == 3
    z0 = x0(3);
    tr = t/duration;
    xd(1) = 0;
    xd(2) = 0;
    xd(3) = z0*(1-(10*tr^3-15*tr^4+6*tr^5));
    xd(7) = 0;
    xd(8) = 0;
    xd(9) = -z0*(30*tr^2-60*tr^3+30*tr^4)/duration;
    dx7d = 0;
    dx8d = 0;
elseif t_mode == 5
    tr = t/duration;
    k = 10*tr^3-15*tr^4+6*tr^5;
    kdot = (30*tr^2-60*tr^3+30*tr^4)/duration;
    kddot = (60*tr-180*tr^2+120*tr^3)/duration^2;
    ang = w*k+pi;
    xd(1) = R*cos(ang)+R;
    xd(2) = R*sin(ang);
    xd(3) = R/2*cos(w*k)+1.5*R/2;
    xd(7) = -w*R*sin(ang)*kdot;
    xd(8) = w*R*cos(ang)*kdot;
    xd(9) = -w*R/2*sin(w*k)*kdot;
    dx7d = -R*((w*kdot)^2*cos(ang)+w*kddot*sin(ang));  %ddXd;
    dx8d =  -R*((w*kdot)^2*sin(ang)-w*kddot*cos(ang));  %ddYd;
elseif t_mode == 6
    tr = t/duration;
    k = 10*tr^3-15*tr^4+6*tr^5;
    kdot = (30*tr^2-60*tr^3+30*tr^4)/duration;
    kddot = (60*tr-180*tr^2+120*tr^3)/duration^2;
    ang = w*k+pi;
    xd(1) = R*cos(ang)+R;
    xd(2) = R*sin(ang);
    xd(3) = 0; %Zd
    xd(7) = -w*R*sin(ang)*kdot;
    xd(8) = w*R*cos(ang)*kdot;
    xd(9) = 0; %dZd
    dx7d = -R*((w*kdot)^2*cos(ang)+w*kddot*sin(ang));  %ddXd;
    dx8d =  -R*((w*kdot)^2*sin(ang)-w*kddot*cos(ang));  %ddYd;
end

u1 = du(1)+g*m/b*(kp(3)*(xd(3)-x(3))+kd(3)*(xd(9)-x(9)));
dx(7) = -sin(x(5))*cos(x(4))*b/m*u1; %x_dd
dx(8) = sin(x(4))*b/m*u1;  %y_dd
dx(9) = -d1-g+cos(x(5))*cos(x(4))*b/m*u1; %z_dd

ux = du(5)+kp(1)*(xd(1)-x(1))+kd(1)*(xd(7)-x(7));
uy = du(6)+kp(2)*(xd(2)-x(2))+kd(2)*(xd(8)-x(8));
dux = du(7)+kp(1)*(xd(7)-x(7))+kd(1)*(dx7d-dx(7));
duy = du(8)+kp(2)*(xd(8)-x(8))+kd(2)*(dx8d-dx(8));

lim_uy = 0.95;
```

```
if uy > lim_uy
    uy = lim_uy;
elseif uy < -lim_uy
    uy = -lim_uy;
end

Cx4 = cos(x(4));
Tx4 = tan(x(4));
uxCx4 = ux/Cx4;
lim_uxCx4 = 0.95;
if uxCx4 > lim_uxCx4
    uxCx4 = lim_uxCx4;
elseif uxCx4 <-lim_uxCx4
    uxCx4 = -lim_uxCx4;
end

xd(4) = asin(uy); %desired phi
xd(5) = -asin(uxCx4);%desired theta
xd(6) = 0; %desired psi
xd(10) = 1/sqrt(1-uy^2)*duy; %desired phi_dot
xd(11) = -1/sqrt(1-(uxCx4)^2)*(dux+Tx4*x(10)*ux)/Cx4;%desired theta_dot
xd(12) = 0; %desired psi_dot

for j = 1:6
    dx(j) = x(j+6); %dx1...dx6
end
dx(10) = -d2+a1*x(11)*x(12)-a2*Om_r*x(11)+a11*(du(2)+kp(4)*(xd(4)-
x(4))+kd(4)*(xd(10)-x(10))); %phi_dd
dx(11) = -d3+a3*x(10)*x(12)+a4*Om_r*x(10)+a22*(du(3)+kp(5)*(xd(5)-
x(5))+kd(5)*(xd(11)-x(11))); %theta_dd
dx(12) = -d4+a5*x(10)*x(11)+a33*(du(4)+kp(6)*(xd(6)-x(6))+kd(6)*(xd(12)-
x(12))); %psi_dd
```

Bibliography

[1] J. Gordon Leishman. (2000) A History of Helicopter Flight. [Online]. http://terpconnect.umd.edu/~leishman/Aero/history.html

[2] Samir Bouabdallah, Roland Siegwart, and Gilles Caprari, "Design and Control of an Indoor Coaxial Helicopter," in *International Conference on Intelligent Robots and Systems*, Beijing, 2006, pp. 2930 - 2935.

[3] (2011, January) Dragonfly Pictures Inc. [Online]. http://www.dragonflypictures.com/

[4] Patrick C. O'Brien, "Using a Robotic Helicopter to Fuel Interest in and Augment the Human Exploration of the Planet Mars," in *AIAA Space 2003 Conference and Exposition*, Long Beach, CA, 2003.

[5] Early Helicopter History. [Online]. http://www.aerospaceweb.org/design/helicopter/history.shtml

[6] Convertawings Model A. [Online]. http://www.aviastar.org/helicopters_eng/convertawings.php

[7] Ilan Kroo and Fritz Prinz, "The Mesicopter: A Miniature Rotorcraft Concept," Stanford University, PhD Thesis 2001.

[8] Erdinc Altug, James P. Ostrowski, and Robert Mahony, "Control of a Quadrotor Helicopter Using Visual Feedback," in *IEEE International Conference on Robotics & Automation*, Washington DC, 2002.

[9] P. Pounds, R. Mahony, P. Hynes, and J. Roberts, "Design of a Four-Rotor Aerial Robot," in *Australasian Conference on Robotics and Automation*, Auckland, 2002.

[10] Paul Pounds, Robert Mahony, and Peter Corke, "Modelling and Control of a Quad-Rotor Robot," in *Australasian Conference on Robotics and Automation 2006*, 2006.

[11] Ming Chen and Mihai Huzmezan, "A Combined MBPC/ 2 DOF $H\infty$ Controller For a Quadrotor UAV," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Austin, Texas, 2003.

[12] Eryk Bryan Nice, "Design of a Four Rotor Hovering Vehicle," Cornell University, MASc Thesis 2004.

[13] Gabe Hoffmann et al., "The Stanford Testbed Of Autonomous Rotorcraft For Multi Agent Control (STARMAC)," in *In Proceedings of the 23rd Digital Avionics Systems Conference*, 2004.

[14] Gabriel M. Hoffmann, Haomiao Huang, Steven L. Waslander, and Claire J. Tomlin, "Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Hilton Head, 2007.

[15] Samir Bouabdallah and Roland Siegwart, "Backstepping and Sliding-mode Techniques Applied to an Indoor Micro Quadrotor," in *IEEE*, 2005.

[16] Samir Bouabdallah and Roland Siegwart, "Towards Intelligent Miniature Flying Robots," in *Advances in Telerobotics*., 2006, pp. 429-440.

[17] Cosmin Coza and C.J.B. Macnab, "A New Robust Adaptive-Fuzzy Control Method Applied to Quadrotor Helicopter Stabilization," in *IEEE*, 2006.

[18] Ian D. Cowling, Oleg A. Yakimenko, James F. Whidborne, and Alastair K. Cooke, "A Prototype of an Autonomous Controller for a Quadrotor UAV," in *European Control Conference*, 2007.

[19] Gabriel M. Hoffmann and Steven L. Waslander, "Quadrotor Helicopter Trajectory Tracking Control," in *AIAA Guidance, Navigation, and Control Conference*, Honolulu,HI,USA, 2008.

[20] Samir Bouabdallah and Roland Siegwart, "Full Control of a Quadrotor," in *International Conference on Intelligent Robots and Systems*, San Diego,CA,USA, 2007.

[21] Patrick Adigbli, Christophe Grand, Jean-Baptiste Mouret, and Stephane Doncieux, "Nonlinear Attitude and Position Control of a Micro Quadrotor using Sliding Mode and Backstepping Techniques," in *European Micro Air Vehicle Conference and Flight Competition*, Toulouse, France, 2007.

[22] Holger Voos, "Nonlinear State-Dependent Riccati Equation Control of a Quadrotor UAV," in *International Conference on Control Applications*, 2006.

[23] Yiting Wu, "Development and Implementation of a Control System for a quadrotor UAV(thesis)," , 2009.

[24] Jack F. Shephard III, A Hierarchical Neuro-Evolutionary Approach to Small Quadrotor Control(thesis), 2009.

[25] C. Nicol, C.J.B. Macnab, and A. Ramiraz-Serrano, "Robust Neural Network Control of a Quadrotor Helicopter," in *IEEE*, 2008.

[26] Travis Diersk and S Jagannathan, "Neural Network Output Feedback Control of a Quadrotor UAV," in *IEEE Conference on Decision and Control*, 2008.

[27] Travis Diersk and S Jagannathan, "Output Feedback Control of a Quadrotor UAV Using Neural Networks," *IEEE Transactions on Neural Networks*, vol. 21, No1, January 2010.

[28] Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki, "Bettering Operation of Robots by Learning," *Journal of Robotic Systems*, vol. 1, pp. 123-140, 1984.

[29] P.R. Ouyang and Pong-in P., "Iterative Learning Control: A Comparison Study," , Vancouver, 2010.

[30] Zhao Qing Song, Jian Qin Mao, and Shao Wu Dai, "First-order D-Type Iterative Learning Control for Nonlinear Systems with Unknown Relative Degree," *July*, vol. 31, no. 4, 2005.

[31] Samer S. Saab, "Stochastic P-type/D-type Iterative Learning Control Algorithms," *International Journal Control*, vol. 76, No.2, 2003.

[32] Kevin L. Moore, "An Observation About Monotonic Convergence in Discrete-Time, P-Type Iterative Learning Control," in *International Symposium on Intelligent Control*, Mexico City, 2001.

[33] James D. Ratcliffe et al., "P-type Iterative Learning Control for Systems that Contain Resonance," *International Journal Of Adaptive Control and Signal Processing*, vol. 19, pp. 769–796, 2005.

[34] Bin Zhang, Danwei Wang, and YongQiang Ye, "Two-Mode Iterative Learning Control Using P-Type and Pseudo-Downsampled Learning," in *American Control Conference*, Portland, OR, 2005.

[35] Chi Keng Chen and James Hwang, "PD- Type Iterative Learning Control for the Trajectory Tracking of a Pneumatic X-Y Table with Disturbances," vol. 49, no. Serires C, No. 2, 2006.

[36] Yang Quan Chen and Kevin L. Moore, "An Optimal Design of PD-type Iterative Learning Control with Monotonic Convergence," in *International Symposium on Intelligent Control*, Vancouver, Canada, 2002.

[37] Danwei Wang, "On D-type and P-type ILC Designs and Anticipatory Approach," vol. 73, no. 10, pp 890-901, 2000.

[38] Kwang Hyun Park, "an Average Operator-Based PD-Type Iterative Learning Control for Variable Initial State Error," *IEEE Transactions on Automatic Control*, vol. 50, No 6, 2005.

[39] Yangquan Chen and Kevin L. Moore, "PI-Type Iterative LEarning Control Revisited," in *Proceedings of the American Control Conference*, Anchorage,AK, 2002.

[40] Kwang Hyun Park, Bien Zeungham, and Dong Hwan Hwang, "A Study on the Robustness of a PID-Type Iterative Learning Controller Against Initial State Error," *International Journal of Systems Science*, vol. 30, No. 1, pp. 49-59, 1999.

[41] Ali Madady, "PID Type Iterative Learning Control with Optimal Gains," *International Journal of Control, Automation, and Systems*, vol. 6, No.2, pp. 194-203, 2008.

[42] P.R. Ouyang, "PD-PD Type Learning Control For Uncertain Nonlinear Systems," in *Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, San Diego, CA, 2009.

[43] Chiang-Ju Chien and Jing-Sin Liu, "A P-Type Learning Controller for Robustness Output Tracking of Nonlinear Time-Varying Systems," , Baltimore, Maryland, 1994.

[44] X.G. Yan, I.M. Chen, and J. Lam, "D-Type Learning Control for Nonlinear Time-Varying Systems with Unknown Initial States and Inputs," *Transactions of the Institute of Measurement and Control*, vol. 23, No.2, pp. 69–82, 2001.

[45] P.R. Ouyang, W.J. Zhang, and M.M. Gupta, "PD-Type On-line Learning Control For Systems With State Uncertainties and Measurement Disturbances," *Control and Intelligent Systems*, vol. 35, No. 4, 2007.

[46] P.R. Ouyang, W.J. Zhang, and M.M. Gupta, "An Adaptive Switching Learning Control Method for Trajectory Tracking of Robot Manipulators," 2006.

[47] Shaojaun Yu, Shibo Xiong, and Jingcai Bai, "Design of Iterative Learning Controller Combined with Feedback Control for Electrohydraulic Servo System," in *Fourth International Conference on Natural Computation*, 2008.

[48] Jianxia Shou, Daoying Pi, and Wenhai Wang, "Sufficient Conditions for the Convergence of Open-closed-loop PID-type Iterative Learning Control for Nonlinear Time-varying Systems," in *IEEE*, 2003.

[49] Shao Juan Yu, Ju Hua Wu, and Xue Wen Yan, "A PD-Type Open-Closed-Loop Iterative Laerning Control and Its Convergence For Discrete Systems," in *Proceedings of the First International Conference on MAchine Learning and Cybernetics*, Beijing, 2002.

[50] P.R. Ouyang, B.A. Petz, and F.F. Xi, "Iterative Learning Control with Switching Gain PD Feedback For Nonlinear Systems".

[51] Richard M. Murray and Zexiang Li, *A Mathematical Introduction to Robotic Manipulation*.: CRC Press, 1994.

[52] P.R. Ouyang and Pong-in P., "Iterative Learning Control: A Comparison Study," in *Proceedings of ASME International Mechanical Engineering Congress & Exposition*, Vancouver, 2010.

[53] Bin Zhang, Danwei Wang, and YongQiang Ye, "Two-Mode Iterative Learning Control Using P-Type and Pseudo-Downsampled Learning," in *American Control Conference*, Portland, OR, 2005, pp. 190-195.

[54] P.R. Ouyang, W.J. Zhang, and M.M. Gupta, "PD-Type On-line Learning Control For Systems With State Uncertainties and Measurement Disturbances," *Control and Intelligent Systems*, vol. 35, No. 4, pp. 351-359, 2007.

[55] "Iterative Learning Control for Uncertain Nonlinear Discrete-time Systems using Current Iteration Tracking Error," in *Iterative Learning Control*.: Springer Berlin / Heidelberg, 1999, vol. 248/1999, ch. 4.

[56] P.R. Ouyang, B.A. Petz, and F.F. Xi, "Iterative Learning Control with Switching Gain PD Feedback For Nonlinear Systems," in *Proceedings of IEEE International Conference on Science and Technology for Humanity (TIC-STH)* , Toronto, 2009, pp. 875 - 880.

[57] Etienne Dombre and Wisama Khalil, "Trajectory Generation," in *Modeling, Performance Analysis and Control of Robot Manipulators*.: ISTE Ltd., 2007, p. 191.

[58] Mu Huang, Bin Xian, Chen Diao, Kaiyan Yang, and Yu Feng, "Adaptive Tracking Control of Underactuated Quadrotor Unmanned Aerial Vehicles via Backstepping," in *American Control Conference*, Baltimore, 2010, pp. 2076-2081.