Lehrstuhl für Flugsystemdynamik

Novel Control Approaches to Quadrotors Inspired by Dynamic Inversion and Backstepping

Jian Wang

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. Horst Baier	
Prüfer der Dissertation:	1.	Univ.-Prof. Dr.-Ing. Florian Holzapfel
	2.	Assoc. Prof. Tiauw Hiong Yongki Go, Florida Institute of Technology, Melbourne / USA

Die Dissertation wurde am 27.01.2015 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 25.09.2015 angenommen.

To my wife Xu Wei and my baby Aaron without whom
this thesis would have been completed two years earlier

Acknowledgements

This thesis is the result of four years of research within the Institute of Flight System Dynamics (FSD) at Technische Universität München, with the support of the Technische Universität München – Institute for Advanced Study, funded by the German Excellence Initiative. During this period, many people contributed to the realization of this work. I am very grateful to all of these people, but I would like to mention some of them in particular.

First of all, I would like to thank my supervisor Prof. Florian Holzapfel for the immeasurable amount of help and guidance and for his enthusiastic scientific support that has kept me motivated in these past years. It was an honor and a privilege to work with him and develop not only as a student, engineer but also as a person. Moreover, I have the great admiration for his sense of analytical mind, creativity and humor, and I always enjoyed our discussions together. He is probably the most wise, most kind, most funny professor I have ever met, in the area of flight control in Munich.

This research would not have been possible without the efforts of Thomas Raffler and Sebastian Klose for their cooperation and great support in any kind of hard and software problems through the development and application of the work. We have had many inspiring discussions, without these, many ideas would not have been originated.

Then all my thanks goes the FSD faculty, in particular Thomas Bierling, Fubiao Zhang, Zhongjie Wang, Leonhard Höcht, Benjamin Bauer, Stanislav Braun, Miguel Leitão, Florian Peter, Michael Achtelik. I am also grateful to the people at Nanyang Technological University in Singapore, especially to Prof. Yongki Go, Weihua Zhao and Soonhooi Chiew for their support. I will always have many fond memories with all of them along the journey of research.

Last but certainly not least, I am indebted to my family, especially my wife for her love and discontinuous support.

Munich — Jian Wang

Mar 2014

Table of Contents

List of Figures

List of Tables

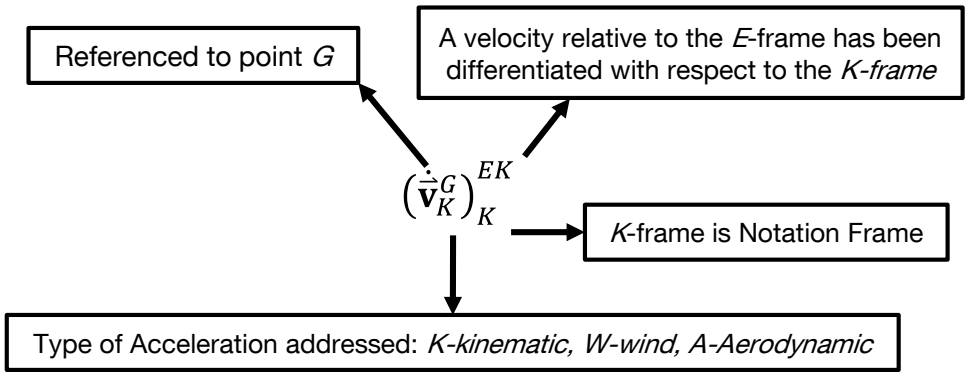
Nomenclature

The following tables contain a survey of the abbreviations and the formula symbols utilized most frequently throughout the thesis. Common symbols as well as indices are written in italics, while matrices and vectors are shown in bold and non-italic. Most matrices are shown with capital letter while vectors and scalars are written with small letters. Furthermore, vectors with a physical meaning in the three-dimensional Euclidean space are marked with an arrow on top of the symbol.

For any Euclidean vector $\vec{\mathbf{x}}$, the following declaration scheme is applied:

$$\left(\vec{\mathbf{x}}^{Reference\ point}_{Type\ of\ motion/Source\ of\ Force/Moment}\right)^{Reference\ Frame}_{Notation\ Frame}$$

For example, the velocity derivative $\left(\dot{\vec{\mathbf{v}}}^{G}_{K}\right)^{EK}_{K}$ means,



Symbols

B	Body-fixed frame
W	World frame, deduced from NED frame by user-defined heading
$\mathbf{M}_{BW}, \mathbf{M}_{WB}$	Transformation matrices from W frame to B frame and B to W
$\Sigma\left(\vec{\mathbf{F}}^{G}\right)_{B}$	$= [F_x \quad F_y \quad F_z]^{\mathrm{T}}$, the total forces on C.G: denoted in B frame,
$\Sigma\left(\vec{\mathbf{M}}^{G}\right)_{B}$	$= [L \quad M \quad N]^{\mathrm{T}}$, the total moments on C.G. denoted in B frame
$(\vec{\boldsymbol{\omega}}^{WB})_{B}$	$= [p \quad q \quad r]^{\mathrm{T}}$, angular rates between W and B frame, denoted in B frame, respectively
$(\vec{\mathbf{r}}^{G})_{W}$	Position vector of C.G., denoted in W frame
$\left(\vec{\mathbf{v}}^{G}_{K}\right)^{W}_{W}$	Kinematic velocity vector of C.G. w.r.t. W frame denoted in W frame
$\left(\vec{\mathbf{a}}^{G}_{K}\right)^{WW}_{W}$	Kinematic acceleration vector of C.G. w.r.t. W frame denoted in W frame
$\mathbf{q}_{WB}$	Unit quaternions to represent the attitude, the same rotation as $\mathbf{M}_{WB}$
$\vec{\mathbf{g}}_{W}$	$= [0 \quad 0 \quad 9.81]^{\mathrm{T}}$, gravitational acceleration vector denoted in W frame
$\vec{\mathbf{g}}_{B}$	$= [g_x \quad g_y \quad g_z]^{\mathrm{T}}$, gravitational acceleration vector denoted in B frame

$\vec{\mathbf{f}}_B$	$= [f_x \quad f_y \quad f_z]^{\mathrm{T}}$, specific force vector denoted in B frame, accelerometer measurements.
T	Total thrust of the quadcopter
$(\mathbf{I}^G)_{BB}$	Moment of inertia of the center of gravity
$\mathbf{n}$	$= [n_1 \quad n_2 \quad n_3 \quad n_4]^{\mathrm{T}}$, normalized rotor rotation speeds of the quadrotor
$\mathbf{I}_3$	$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, unity matrix

Abbreviations

NDI	Nonlinear Dynamic Inversion
BS	Backstepping
LLP	Low Level Processor
HLP	High Level Processor
IMU	Inertial Measurement Unit
SDK	Software Development Kit
AHRS	Attitude Heading Reference System
EKF	Extended Kalman Filter
ESKF	Error State Kalman Filter
CFB	Command Filtered Backstepping

1. Introduction

1.1. Thesis Goal and Research Approach

Many well-established methods and design techniques, such as root locus, Eigen structure assignment, pole placement, LQR, H∞, and etc., exist for linear time-invariant (LTI) systems. However, most of the real systems under control are not LTI systems and those linear theories cannot be applied directly, but they are applied to the linearized systems around the design points, in whose vicinity the plant can be assumed linear. Conservative bounds have to be used due to approximations and when the operation range is large, it is likely to perform very poorly or to be unstable. Nonlinear control methods can explicitly address the specific nonlinear dynamics of the plant and fully or partially linearize the plant, not by any approximation (i.e., Taylor expansion/Jacobian linearization) but by feedback transformations. Then the linear control theory can be applied on the transformed plant. So that full physical and control capabilities of the plant can be exploited.

In addition to the modeled nonlinear dynamics, there is also a wide range of model uncertainties, like un-modeled dynamics, parameter uncertainties, senor errors and even partial plant failures. The adaptive control aims to parameterize those uncertainties and use feedback to learn and estimate these parameters online so that the performance and robustness of the aerial vehicle is improved. Hence, the adaptive control can be augmented to the nonlinear baseline control to account for uncertainties and failures.

The main goal of the thesis is not to develop novel control theory but to investigate the potential of the existing nonlinear adaptive control theories and to bridge the gap between theory and reality by designing novel control architectures in experimental and operational systems. The controlled system shall be able to follow complex, highly curved three-dimensional trajectories with high bandwidth and high robustness. With high bandwidth flight, it becomes very challenging because first the nonlinearities and state couplings will be fully visible and second the influence of model and parameter uncertainties and the actuator dynamics increases.

The multirotor platform serves as an ideal platform to realize and analyze different nonlinear control designs. Given the nonlinear and tightly coupled dynamic nature of this type of vehicles, nonlinear control appears to be a natural choice. Nonlinear Dynamic Inversion (NDI) and Backstepping are two popular nonlinear control methods, which are analyzed and compared in this thesis

1.2. System Overview over Quadrotors

Recent technological progress in low-cost MEMS-based sensors, actuators and energy storage devices enables the development of miniature vertical take-off and landing (VTOL) systems. As one of the most preferred types, the quadrotors have gained increasing interest as a research platform, due to its VTOL and hovering capabilities, easy construction and steering principle, as well as high maneuverability.

1.2.1. Basic steering principle

Figure 1.1 shows one of the quadrotor. As the name suggests it has four rotors. Each rotor produces both a thrust and torque about its center of rotation only by changing the rotation speed of the propellers. If all rotors are spinning at the same angular velocity, with the front and back rotors rotating clockwise and the left and right rotors counterclockwise, the net aerodynamic torque and hence the angular acceleration about the yaw axis is zero. This implies that the yaw stabilizing rotor of conventional helicopter is not needed. Yaw is induced by mismatching the balance in aerodynamic torques, i.e. by offsetting the cumulative thrust commands between the counter rotating blade pairs.

Angular accelerations around the pitch and roll axes can be caused separately without affecting the yaw axis. Each pair of blades rotating in the same direction controls one axis, either roll or pitch. Increasing thrust for one rotor while decreasing thrust for the other will maintain the torque balance needed for yaw stability and induce a net torque around the roll or pitch axes. This way, fixed rotor blades can be used to maneuver the quadrotor in all dimensions. Translational acceleration is achieved by maintaining a non-zero pitch or roll angle.

Figure 1.1 Asctec Quadrotor ‘Hummingbird’

1.2.2. Hardware and software

Two variants of quadrotors are used in the research. The first one is the ‘Hummingbird’ developed by Ascending Technology [1], as shown in Figure 1.1. It has an arm length of 17 cm and a rotor diameter of 8 inches. It includes aluminum mainframe and carbon composite arms, four BLDC Motors with specific motor RPM controllers, and four propellers. As shown in Figure 1.2, the autopilot board consists of two ARM7 60MHz 32 bit microprocessors. The low level processor (LLP) is responsible for all the hardware management and the Asctec default flight control system, which includes sensor data acquisition, preprocessing and fusion, and three different controller: GPS way point control, an attitude control, and a direct motor control for custom controller. The custom code can be freely programmed on the second ARM7, the high level processor (HLP).

Besides the standard hardware from Asctec, we also further developed a Gumstix Overo Fire COM with 720 MHz processor [2] to the ‘Hummingbird’ to allow floating-point computation.

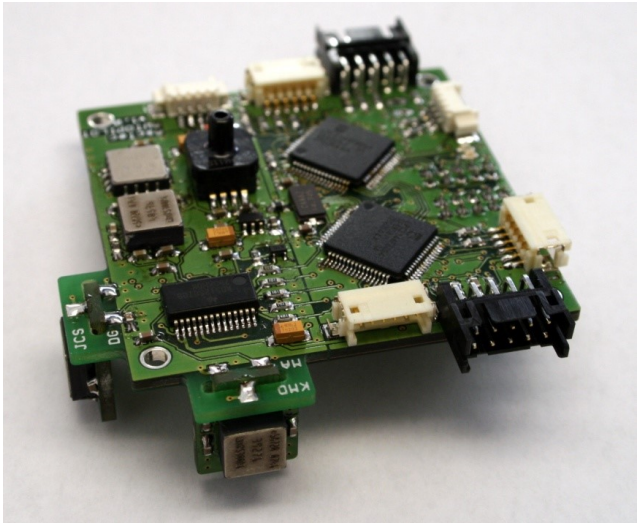


Figure 1.2 Asctec Sensor and Processor Board

The other variant, ‘Pelican’, is a bigger version of ‘Hummingbird’, as shown in the Figure 1.2. It has an arm length of 21cm and a rotor diameter of 10 inches. It is additionally equipped with an Intel atom processor of 1.6GHz to make online processing of image data possible.

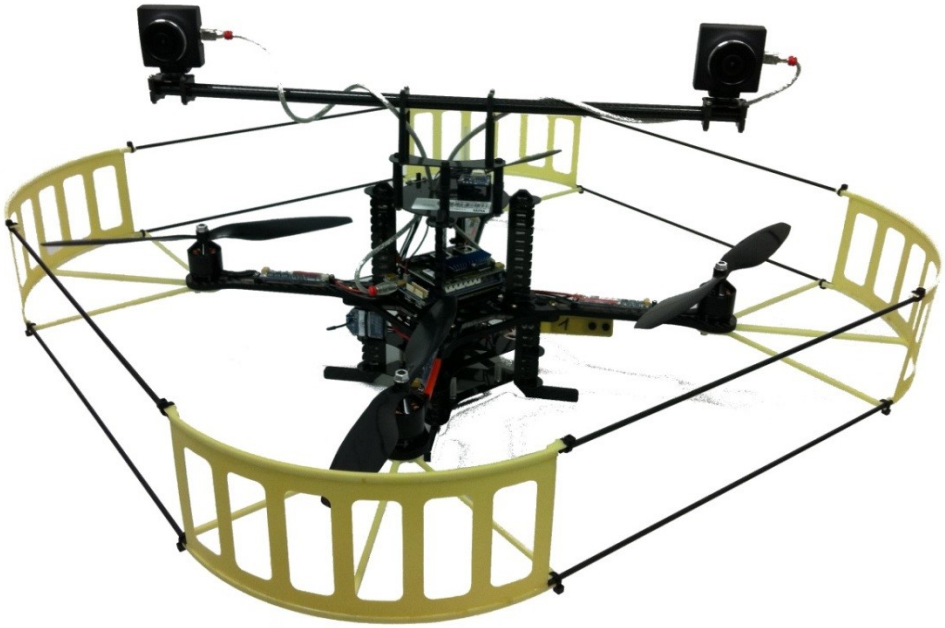


Figure 1.3 Asctec Quadrotor 'Pelican'

The sensors package includes a three-axis gyroscope [3], three-axis accelerometer [4] and three-axis magnetometer, barometer, GPS receiver and vision sensor systems. The technical specifications of the Inertial Measurement Unit (IMU) are attached in the Appendix. There are three different positioning systems used in the experiments, image based vision sensor [5], VICON system [6], and GPS sensor.

The used framework [7] on the HLP is a joint development of the Institute of Flight System Dynamics (FSD) and Ascending Technologies. The HL Software Development Kit (SDK) provides all tools necessary to program custom C-code to flash and debug the code on the processor. The custom C-code can be generated by an automatic process. The control system can be designed in Matlab/Simulink and then translated using the Real-time Workshop Embedded Coder. The developed framework provides all templates for code generation, as well as the code interface between the Simulink controller and the HL-SDK code. Real time LINUX operating system and similar framework as the HLP has also been developed for the Gumstix Overo Board at the Institute of FSD. Based on this work, the flight control system can be designed and implemented without any prior knowledge of the programming language C.

Detailed descriptions of the hardware and software implementations are given in Chapter 8.

1.3. Literature Review – State of the Art

1.3.1. High Bandwidth Quadrotor Control

Due to its mechanical simplicity and robustness, quadrotor or even multirotor received remarkable attention from various research groups, hobbyists and companies. By this time, a tremendous variety of linear and nonlinear as well as adaptive control designs is presented in literature. Highly accurate and mature tracking has been achieved in various publications on the position tracking problem.

Flight tests of position/trajectory controllers presented in Valenti [8] and Bouabdallah [9] were conducted in a controlled lab environment with very feasible, low bandwidth trajectories where linear properties dominate. With the STARMAC project, Hofmann [10] already showed successful outdoor flights with a velocity of up to 2 m/s.

However, the physical capabilities of quadrotors are much beyond the presented autonomous position/trajectory controller. An experienced pilot can fly the quadrotor with highly aggressive maneuvers of speeds more than 10 m/s, aided by the angular rate feedback. In [11] the authors present unprecedented three dimensional and high dynamic manoeuvers that show the physical capabilities of quadrotors. The quadrotor is able to fly aggressive maneuvers that are comparable to the manual flight of an experienced pilot, but more precise in the position tracking. However, these results have been obtained in a controlled lab environment with almost perfect measurements delivered by VICON system [6]. Furthermore, there is no general solution to achieve this level of position tracking performance, but several distinct controllers with a combination of position, attitude and angular rate control for different maneuvers were implemented. This thesis aims to develop generic solutions for autonomous position control aiming maximization of the control bandwidth.

1.3.2. Nonlinear Dynamic Inversion

Nonlinear Dynamic Inversion, also called feedback linearization, became a topic of much research from the 1970s. It is rather a transformation applied to the nonlinear system, which renders a new dynamical system that is LTI. Once a linear system is obtained, a secondary linear control law can be designed to ensure that the overall closed-loop system performs according to the specifications. Two comprehensive textbooks dealing with NDI are [12] [13].

Previous efforts on NDI control include a three-loop design corresponding to inversion of rotational, attitude and path dynamics in separated cascaded loops. A more common design is a two-loop control structure [14], where the outer loop is the position control loop and the inner loop is the attitude control loop. Both control loops have second order dynamics, i.e. the Relative Degree (RD) is two. Two non-cascaded structure designs for quadrotor position control are presented in [15] [16]. However, additional assumptions have to be made in the control designs: In [15] a dynamic input, the second derivative of the thrust, has to be used and in [16] a quasi-static height control has to be assumed for the control of the X- and Y-axes.

1.3.3. Backstepping

Backstepping is a systematic, Lyapunov-based method and is also used to force a nonlinear system behave like a linear one. The name ‘Backstepping’ refers to the recursive nature of the design procedure. The design procedure starts at the output state, which is separated by the largest number of integrations from the control input and ‘step back’ towards the control input. Each step an intermediate control law is calculated and in the last step the real control law is derived. The recommended textbook about Backstepping and Lyapunov theory is [17].

There are several Backstepping approaches on quadrotors as well. A Backstepping control approach that used similar two-loop architectures, i.e. position loop plus attitude loop, was presented in [18] [19]. A non-cascaded position control design using Backstepping was presented in [20]. However, command filters have to be used to estimate the virtual control derivatives and the thrust dynamic is extended to its derivatives. A modification was introduced in [21] that attempts to accounts for the filter estimation errors and it is named ‘Command filtered Backstepping’.

1.3.4. Adaptive Control

The research of adaptive control started in the early 1950s. It has several breakthrough results in the 1970s. Direct and indirect Model Reference Adaptive Control (MRAC) schemes using the Lyapunov based approach were designed and analyzed in [22] [23]. Starting from the mid-1980s, the robust adaptive control designs were proposed and analyzed extensively. Several adaption modifications, like MRAC modification, neural network, $\mathcal{L}_1$ adaptation [24] [25] [26] were introduced to improve the stability and robustness of the adaptive structure, where it is possible to guarantees signal boundedness in the presence of unmodeled dynamics and bounded disturbances as well as performance error bounds that are of the same order as the modeling error. The superior performance of the adaptive control over linear control has been demonstrated in [27] [28], especially in the presence of the parameter uncertainties and structure changes. In the last decade, the research focus is to develop certification strategies for recently developed results of nonlinear adaptive control theory. Hence, the stability and performance of the transient region has also been addressed [29] [30] [31].

1.4. Thesis Organization

Chapter 2 introduces the theoretical background. The standard design procedures are clarified. This chapter is intended as a reference source but not a full coverage of these topics. In Chapter 3 the comparison between NDI and Backstepping is made using a generic 2nd order system.

Starting from Chapter 4 to Chapter 7, the quadrotor control applications of various types are introduced and analyzed. Chapter 4 describes the mathematical model of the quadrotor, including model for simulation, model for data fusion and models for control design. Chapter 5 presents the attitude control designs with conventional Euler angle and a novel state variable. Similar approaches are also implemented on the attitude loop to further validate the comparison results of Chapter 3. Chapter 6 extends the control designs to the position dynamic loop. Again, different architectures are designed for the position loop. Flight test results are presented as well. Chapter 7 applies the $\mathcal{L}_1$ adaptive control to augment the baseline control. Chapter 8 introduces the experimental systems and discusses problems and possible solutions in the practical implementations. In Chapter 9, the conclusion and recommendation for further research are discussed.

1.5. Contribution & Publications

The following aspects are regarded as the main contributions of this thesis to advance the current state of the art in the respective fields:

- **Comparison of NDI and Backstepping designs**
  A throughout comparison of the two methods is made giving a clear overview of the different control approaches. Through the comparison of a simple control system and the attitude control system, the disadvantages and advantages of the control approaches are concluded.
- **Development of a novel state parameter to replace conventional Euler angle**
  The novel state can provide the same functionality as Euler angles and decouple the system control and intermediate control variables so that non-cascaded Backstepping control is made possible. In addition, the dynamic equations/mathematical model are simplified without any approximations, for instance the trigonometric functions are no longer needed, which is beneficial for small UAVs with limited computation power.
- **Development of various NDI control designs for quadrotors, including a novel NDI position control architecture**
  Besides the conventional NDI control structure (Relative Degree 2 position loop + RD2 attitude loop), a novel NDI structure is designed and tested in flight. It is a cascaded loop design with a RD3 position loop and a RD1 rate loop. Due to the faster inner loop, the position loop is able to have a higher control bandwidth compared to the conventional control structure.
- **Development of various Backstepping designs for quadrotors**
  The Lyapunov based Backstepping design with the novel state parameter can provide comparable performance with the NDI designs and less model dependency, i.e. higher robustness. Various Backstepping modifications are analyzed.

Furthermore, the other aspects of the quadrotor control system have also been covered: Implementation of data fusion algorithms for various sensors, and augmentation of the $\mathcal{L}_1$ adaptive control, as well as necessary ground computer algorithms.

Many of the novel designs and applications described in this thesis have been peer reviewed and presented at conferences, while this thesis provides more comprehensive details. The main publications are listed in Bibliography [5] [32] [33] [34] [35] [36] [37] [38] [39], and also listed together in the next page.

- Raffler T., **Wang J.** and Holzapfel F., *Path Generation and Control for Unmanned Multirotor Vehicles Using Nonlinear Dynamic Inversion and Pseudo Control Hedging*, in 19th IFAC Symposium on Automatic Control in Aerospace, Wuerzburg, 2013.
- **Wang J.**, Holzapfel F. et al, *Non-cascaded Dynamic Inversion Design for Quadrotor Position Control with L1 Augmentation*, CEAS EuroGNC, Delft, 2013
- **Wang J.**, Holzapfel F. et al, *Comparison of Dynamic Inversion and Backstepping Control with Application to a Quadrotor*, CEAS EuroGNC, Delft, 2013
- **Wang J.**, Raffler T. et al, *Nonlinear Position Control Approaches for Quadcopters using a Novel State Representation*, AIAA Guidance, Navigation, and Control Conference, Minneapolis, USA, 2012
- **Wang J.**, Bierling T. et al, *Novel Dynamic Inversion Architecture Design for Quadrocopter Control*, in Advances in Aerospace Guidance, Navigation and Control, F. Holzapfel and S. Theil, Eds., pp. 261-272. Springer Berlin Heidelberg, 2011
- Zhao W., **Wang J**. et al, *Leader Follower Quadrotor Formation Flight Control*, CEAS Specialist Conference on Guidance, Navigation & Control, Munich, Germany, 2011
- **Wang J.**, Bierling T. et al, *Attitude Free Position Control of a Quadcopter using Dynamic Inversion*, AIAA Infotech@Aerospace, St. Louis, USA, 2011
- Achtelik M., Bierling T., **Wang, J** et al, *Adaptive Control of a Quadcopter in the Presence of large/complete Parameter Uncertainties*, AIAA Infotech@Aerospace, St. Louis, USA, 2011
- Klose S., **Wang J**. et al, *Markerless Vision Assisted Flight Control of a Quadrocopter*, IEEE RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 2010

2. Theoretical Background

This chapter collects some background material needed throughout the thesis. As the theories are standard and are available in many textbooks, few proofs are offered. The emphasis is to explain the concepts and point out their importance in later applications.

2.1. Mathematical Tools

A few basic mathematical concepts are review in this section.

2.1.1. Diffeomorphism & Coordinate Transformations

A function $\mathbf{f}: \mathcal{D} \subset \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is said to be a Diffeomorphism on $\mathcal{D}$, or a local Diffeomorphism [40], if

- it is continuously differentiable on $\mathcal{D}$, and
- Its inverse $\mathbf{f}^{-1}$ defined by $\mathbf{f}^{-1}\big(\mathbf{f}(\mathbf{x})\big) = \mathbf{x}, \cdot\ \forall \mathbf{x} \in \mathcal{D}$ exists and is continuously differentiable.

The function $\mathbf{f}$ is said to be a global Diffeomorphism if in addtion

- $\mathcal{D} = \mathbb{R}^n$, and
- $\lim_{\mathbf{x}\to\infty} \|\mathbf{f}(\mathbf{x})\| = \infty$

It is often beneficial to perform a coordinate change to transform a state space representation into another. A Diffeomorphism $\mathbf{T}(\mathbf{x})$ can be used to perform a coordinate transformation. Given a nonlinear state space representation of an affine system of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \tag{2.1}$$

Defining $\mathbf{z} = \mathbf{T}(\mathbf{x})$, we have

$$\dot{\mathbf{z}} = \frac{\partial \mathbf{T}}{\partial \mathbf{x}}\dot{\mathbf{x}} = \frac{\partial \mathbf{T}}{\partial \mathbf{x}}[\mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}] \tag{2.2}$$

Given that $\mathbf{T}$ is a Diffeomorphism, we can recover the original state space representation

$$\mathbf{x} = \mathbf{T}^{-1}(\mathbf{z}) \tag{2.3}$$

2.1.2. Relative Degree & Lie Derivative

The relative degree of a scalar dynamic system is defined as the number of times we have to differentiate the output before the input appears.

When deriving the NDI transformation or stability in the sense of Lyapunov, we often need to take a time derivative of a scalar function. This leads to the concept of the *Lie Derivative* [40].

Given a scalar function $h(\mathbf{x})$ and a vector field $\mathbf{f}(\mathbf{x})$, we define the *Lie derivative* $L_{\mathbf{f}}h$, called the derivative of h with respect to $\mathbf{f}$.

$$L_{\mathbf{f}}h(\mathbf{x}) = \frac{\partial h}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}) \tag{2.4}$$

Consider the system of the form as in Eq. (2.1) with output equation $\mathbf{h}(\mathbf{x})$, where $\mathbf{f}$, $\mathbf{G}: \mathcal{D} \subset \mathbb{R}^n \longrightarrow \mathbb{R}^n$ and $\mathbf{h}: \mathcal{D} \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ are sufficiently smooth.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \\ \mathbf{y} &= \mathbf{h}(\mathbf{x}) \end{aligned} \tag{2.5}$$

In order to find the nonlinear state transformation, each output $y_i, i = 1, \dots, p$ has to be differentiated w.r.t. time.

$$\dot{y}_i = \frac{\partial h_i}{\partial \mathbf{x}}\dot{\mathbf{x}} \tag{2.6}$$

Substituting the state dynamic equation and using the definition of Lie derivatives,

$$\begin{aligned} \dot{y}_i &= \frac{\partial h_i}{\partial \mathbf{x}}(\mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}) \\ \dot{y}_i &= L_{\mathbf{f}}h_i(\mathbf{x}) + L_{\mathbf{G}}h_i(\mathbf{x})\mathbf{u} \end{aligned} \tag{2.7}$$

If $L_{\mathbf{G}}h_i(\mathbf{x}) = \mathbf{0}$, the output needs to be differentiated once more, until the control input $\mathbf{u}$ appears, i.e. the coefficient of $\mathbf{u}$ is no longer zero.

$$\ddot{y}_i = L_{\mathbf{f}}^2 h_i(\mathbf{x}) + \underbrace{L_{\mathbf{G}}\left(L_{\mathbf{f}}h_i(\mathbf{x})\right)}_{=\mathbf{0}}\mathbf{u} \tag{2.8}$$

As we can see, the Lie derivative notation is simpler in this case. Thus it is usually preferred whenever higher order derivatives need to be calculated. Let's say in the r_i^{th} derivative the control input appears,

$$y_i^{(r_i)} = L_{\mathbf{f}}^{r_i}h_i(\mathbf{x}) + \underbrace{L_{\mathbf{G}}\left(L_{\mathbf{f}}^{r_i-1}h_i(\mathbf{x})\right)}_{\neq\mathbf{0}}\mathbf{u} \tag{2.9}$$

and r is called *relative degree* of the system with given initial conditions.

2.1.3. System of triangular form

Both NDI and Backstepping are only applicable to a special class of systems having a triangular form. This 'triangular' system, also called pure-feedback system [17], has the following form,

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1) && + g_1(x_1, x_2) \\
\dot{x}_2 &= f_2(x_1, x_2) && + g_2(x_1, x_2, x_3) \\
\dot{x}_3 &= f_3(x_1, x_2, x_3) && + g_3(x_1, x_2, x_3, x_4) \\
&\vdots \\
\dot{x}_n &= f_n(x_1, x_2, x_3, \dots, x_n) + g_n(x_1, x_2, x_3, \dots, x_n, u)
\end{aligned} \tag{2.10}$$

The nonlinearities f_i and g_i depend only on $x_1, \dots, x_i$ and form a lower triangle, thus it is named. The x_{i+1} acts as a pseudo control and appears non-affine in the dynamics of $\dot{x}_i$. In case if the x_{i+1} appears affine in the dynamics of $\dot{x}_i$, the system is called strict-feedback system and has the following form,

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1) && + g_1(x_1) \cdot x_2 \\
\dot{x}_2 &= f_2(x_1, x_2) && + g_2(x_1, x_2) \cdot x_3 \\
\dot{x}_3 &= f_3(x_1, x_2, x_3) && + g_3(x_1, x_2, x_3) \cdot x_4 \\
&\vdots \\
\dot{x}_n &= f_n(x_1, x_2, x_3, \dots, x_n) + g_n(x_1, x_2, x_3, \dots, x_n) \cdot u
\end{aligned} \tag{2.11}$$

An important property of system of strict feedback form is that the nonlinearities are continuously differentiable, so that the Lie derivative may be applied to perform the NDI transformation of the original nonlinear system. A common case for systems of non-triangular form is that the system input u appears not only in the last differential equation, but also in the other nonlinearities f_i and g_i. This invokes the concept of relative degree and makes direct applications of NDI and Backstepping not possible.

2.2. Nonlinear Dynamic Inversion

Nonlinear Dynamic Inversion, also called input-output linearization or feedback linearization, was a topic of much research since 1970s [12] [13]. It has been used successfully to address some practical control problems including helicopter control, high performance aircraft. However, there are also a number of limitations that hinder its use. But even with these shortcomings, NDI is a concept of paramount importance in nonlinear control theory. The intention of this section is to provide a brief introduction to the design procedures by using a second order example system. For a more complete coverage on the theory and detailed derivations, please refer to the two textbooks [12] [13].

2.2.1. The NDI transformation and Companion form

The NDI method is rather a system transformation than a control method. It can be applied to transform the original nonlinear system into the so-called *companion form*, or *controllability canonical form*. Given a nonlinear SISO system as in Eq.(2.12),

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \\ y &= h(\mathbf{x}) \end{aligned} \tag{2.12}$$

Lie Derivative can be applied until the system input explicitly appears in the derivative equation.

$$y^{(r)} = L_{\mathbf{f}}^{r} h(\mathbf{x}) + \underbrace{L_{\mathbf{G}}\left(L_{\mathbf{f}}^{r-1} h(\mathbf{x})\right)}_{\neq \mathbf{0}} u \tag{2.13}$$

$$z_r = y^{(r)} = f_r + g_r u \tag{2.14}$$

The transformed state $\mathbf{T}(\mathbf{x})$ can be defined as $\mathbf{z}$.

$$\mathbf{z} = \mathbf{T}(\mathbf{x}) = \begin{bmatrix} z_1 \\ z_2 \\ . \\ . \\ . \\ z_r \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \\ . \\ . \\ . \\ y^{(r)} \end{bmatrix}$$

The transformed system obtains the *companion form,* represented as below.

$$\begin{aligned} \dot{z}_1 &= z_2 \\ \dot{z}_2 &= z_3 \\ \dot{z}_3 &= z_4 \\ &\vdots \\ \dot{z}_r &= f_r + g_r u \end{aligned} \tag{2.15}$$

In state space representation, it can be written as,

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{b}\nu \tag{2.16}$$

where

$$\nu = f_r + g_r u$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & . & . & 0 \\ 0 & 0 & 1 & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & 1 \\ 0 & 0 & 0 & . & . & 0 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 0 \\ . \\ . \\ . \\ 0 \\ 1 \end{bmatrix}$$

By assuming $\mathbf{G}_n$ is a Diffeomorphism, for systems in the companion form, system nonlinearities can be inverted and the transformed systems can be regarded as a chain of integrators.

$$u = g_n^{-1}[v - f_n] \tag{2.17}$$

$$\dot{z}_r = y^{(r)} = v \tag{2.18}$$

2.2.1.1. Scalar Example

A generic scalar example can be used to illustrate the method and for simplicity full state feedback is assumed. A second order nonlinear state space representation of an affine system is given in Eq.(2.19). This example is extensively used in the remainder of this thesis.

$$\begin{cases} \dot{x}_1 = f_1 + g_1 \cdot x_2 \\ \dot{x}_2 = f_2 + g_2 \cdot u \end{cases} \tag{2.19}$$

The outer loop state x_1 is often the output of interest. To generate a direct relationship between the output x_1 and the input u, let us differentiate the output state,

$$\begin{aligned} \ddot{x}_1 &= \frac{d}{dt}[f_1 + g_1 \cdot x_2] = \dot{f}_1 + \dot{g}_1 \cdot x_2 + g_1 \cdot \dot{x}_2 \\ &= \dot{f}_1 + \dot{g}_1 \cdot [g_1^{-1}(\dot{x}_1 - f_1)] + g_1 \cdot [f_2 + g_2 \cdot u] \\ &= \underbrace{\dot{f}_1 + \dot{g}_1 \cdot [g_1^{-1}(\dot{x}_1 - f_1)] + g_1 \cdot f_2}_{=f_3} + \underbrace{g_1 \cdot g_2}_{=g_3} \cdot u \\ &= f_3 + g_3 \cdot u \end{aligned}$$

The derivatives of the nonlinear functions f_1 and g_1 have to exist and can be expressed by measurable parameters. Assuming the given system is in triangular form (i.e. strictly feedback form), these function can be expressed as in Eq. (2.20),

$$\begin{cases} \frac{d}{dt} f_1(x_1) = \frac{\partial f_1}{\partial x_1}\frac{\partial x_1}{\partial t} = \frac{\partial f_1}{\partial x_1}\dot{x}_1 \\ \frac{d}{dt} g_1(x_1) = \frac{\partial g_1}{\partial x_1}\frac{\partial x_1}{\partial t} = \frac{\partial g_1}{\partial x_1}\dot{x}_1 \end{cases} \tag{2.20}$$

Hence, we can obtain the canonical form as in Eq. (2.15),

$$\begin{cases} \dot{x}_1 = x_3 \\ \dot{x}_3 = f_3 + g_3 \cdot u \end{cases} \tag{2.21}$$

The NDI transformation similar to Eq. (2.17) can be applied to obtain a linear system with a chain of integrators,

$$u = g_3^{-1}[v - f_3]$$

$$v = \dot{x}_3 = \ddot{x}_1$$

If the system is not in triangular form, i.e. $f_1 = f_1(x_1, x_2)$ and $g_1 = g_1(x_1, x_2)$, this will introduce additional states if we try to obtain the canonical form. The relative degree is violated as the system input influence both states of different system order. In this case, a cascaded NDI structure can be the solution. However time scale separation is needed.

$$\begin{cases} \frac{d}{dt} f_1(x_1, x_2) = \frac{\partial f_1}{\partial x_1}\dot{x}_1 + \frac{\partial f_1}{\partial x_2}\dot{x}_2 = \frac{\partial f_1}{\partial x_1}\dot{x}_1 + \frac{\partial f_1}{\partial x_2}(f_2 + g_2 \cdot u) \\ \frac{d}{dt} g_1(x_1, x_2) = \frac{\partial g_1}{\partial x_1}\dot{x}_1 + \frac{\partial g_1}{\partial x_2}\dot{x}_2 = \frac{\partial g_1}{\partial x_1}\dot{x}_1 + \frac{\partial g_1}{\partial x_2}(f_2 + g_2 \cdot u) \end{cases}$$

2.2.2. Linear Controller

After the NDI transformation, the original system becomes a chain of integrators. A linear controller is designed for the transformed linear system. The linear control used includes two parts:

- A Reference model to provide smooth reference signals including reference state derivatives.
- An Error controller to accounts for external disturbances, parameter and modeling uncertainties, and measurement uncertainties.

2.2.2.1. Reference Model

Linear reference models are similar to low pass filters. The differential equation and the transfer function of a first order reference model are

$$\dot{x}_r = \frac{1}{T}(x_C - x_r) \tag{2.22}$$

$$x_r = \frac{1}{Ts + 1} x_C \tag{2.23}$$

where T is the time constant of the transfer function. Additionally the design requirements (i.e. rise time, overshoot) and system constraints can be specified in the reference model. The block diagram and time response to a step command is shown in Figure 2.1 and Figure 2.3. The state derivative $\dot{x}_r$ is limited to 5 as an example.

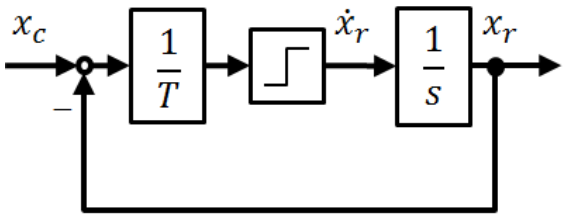


Figure 2.1 Block diagram of first order reference model

The reference model can also be regarded as a simplified version of the controller and plant. For linear reference models, a chain of integrators is simulating the plant and a linear controller

is controlling it. This could generate the reference state signals until the n-th derivatives in the ideal model and then feed forward them to the actual controller, and n is determined by the relative degree of the dynamic model.

To provide the n-th derivative of the reference state, the dynamic order of the reference model should be compliant with the RD of the system model. For the second order example system given in Eq. (2.19), a second order reference model should be used. The differential equation and transfer function for a second order reference model are,

$$v_r = \ddot{x}_r = \omega_0^2(x_C - x_r) + 2\zeta\omega_0\dot{x}_r \tag{2.24}$$

$$x_r = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} x_C \tag{2.25}$$

where ζ is the relative damping and ω_0 is the natural frequency of the reference dynamics. The maximum bandwidth of the reference model can be determined according to the error dynamics. It should not be higher than that of the error dynamics (in the next section). The designed block diagram and the time responses with different relative damping values are shown in Figure 2.2 and Figure 2.4 respectively.

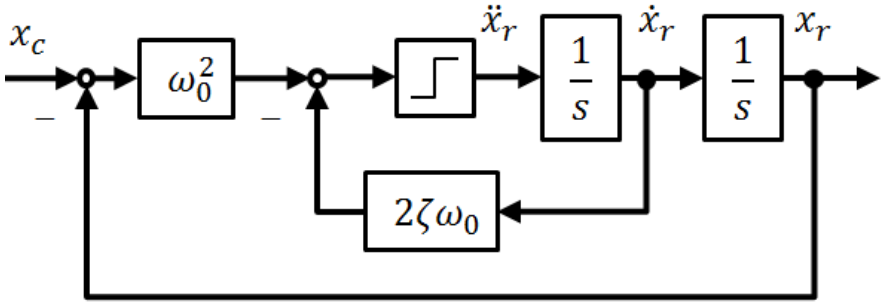


Figure 2.2 Block diagram of second order reference model

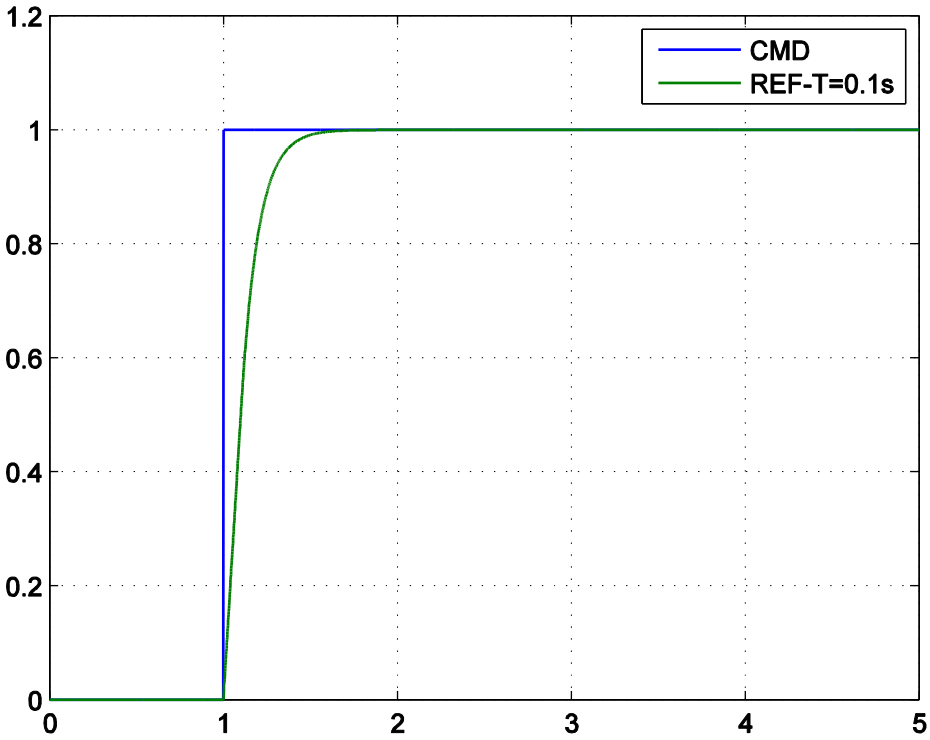


Figure 2.3 Time response of first order reference model

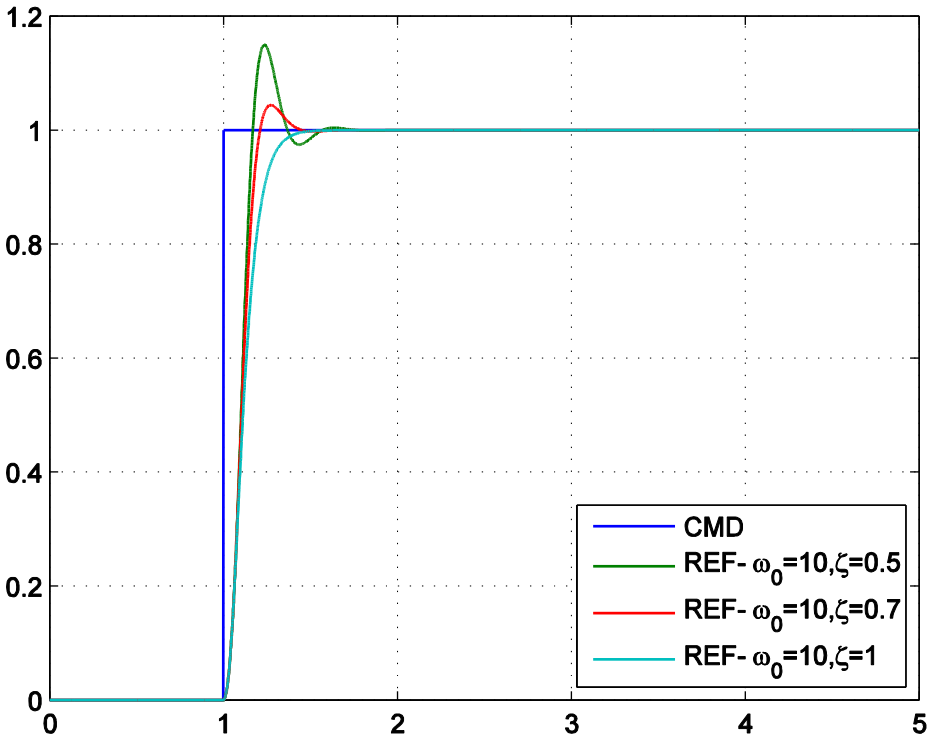


Figure 2.4 Time response of second order reference model

2.2.2.2. Error Controller

In perfect situation, the system states would exactly follow the reference states. However, due to the model uncertainties, sensor uncertainties, actuator constraints and external disturbances, the system state derivative $\ddot{x}$ will differ from the pseudo control v as shown in Eq. (2.26). This will be propagated through the integrations and the system is unstable.

$$\ddot{x} = v + \Delta \tag{2.26}$$

The error controller is designed to account for those deviations. As the name indicated, it controls the errors between the reference signals and the state feedbacks. This moves the poles of the integrator chain into the left half complex plane.

For the second order example system, a PD error feedback is normally used.

$$v = \underbrace{v_r}_{=\ddot{x}_{1r}} + k_d \underbrace{(\dot{x}_{1r} - \dot{x}_1)}_{\dot{e}} + k_p \underbrace{(x_{1r} - x_1)}_{=e} = v_r + k_d\dot{e} + k_p e \tag{2.27}$$

$$\ddot{x}_{1r} - \ddot{x} = -k_d(\dot{x}_{1r} - \dot{x}_1) - k_p(x_{1r} - x_1) - \Delta \tag{2.28}$$

$$\ddot{e} = -k_d\dot{e} - k_p e - \Delta \tag{2.29}$$

The error dynamics can be expressed in state space representation.

$$\begin{bmatrix}\dot{e}\\ \ddot{e}\end{bmatrix} = \begin{bmatrix}0 & 1\\ -k_p & -k_d\end{bmatrix}\begin{bmatrix}e\\ \dot{e}\end{bmatrix} + \begin{bmatrix}0\\ -1\end{bmatrix}\Delta \tag{2.30}$$

In the ideal case where there is no deviation of any kinds, the transfer function from the state output to the reference state is,

$$x = \frac{s^2 + k_d s + k_p}{s^2 + k_d s + k_p} x_r \tag{2.31}$$

The bandwidth of the error dynamics is limited by the actuator dynamics and measurement quality. Enough time scale separation could hide the actuator dynamics. For a second order system, the PD feedback gains are normally set according to the natural frequency and relative damping as in Eq. (2.25).

$$\begin{cases} k_d = 2\zeta\omega_0 \\ k_p = \omega_0^2 \end{cases} \tag{2.32}$$

Together with the NDI transformation, the overall control law is given as follows,

$$\begin{cases} u = g_3^{-1}[v - f_3] \\ v = v_r + 2\zeta\omega_0\dot{e} + \omega_0^2 e \end{cases} \tag{2.33}$$

The overall control structure is shown in Figure 2.5, where G_A is the actuator dynamics.

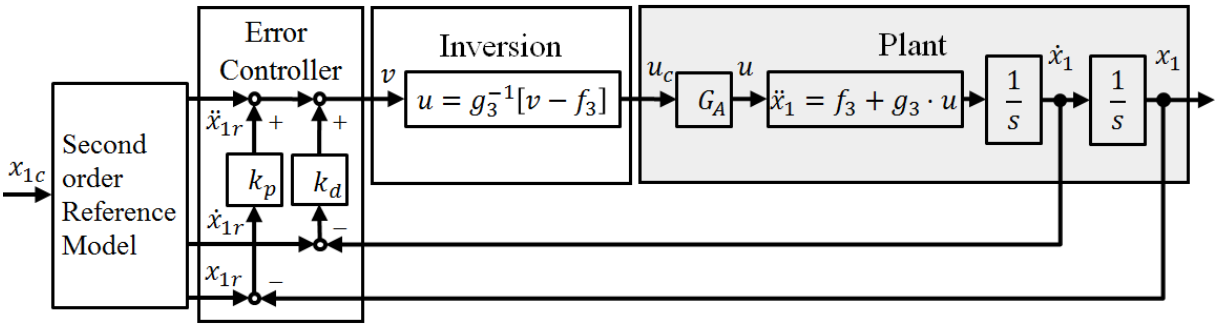


Figure 2.5 Overall NDI design structure for a second order system

2.2.3. Pseudo Control Hedging

One major flaw of the above design is that the actuator dynamics is ignored in the design process. Eric Johnson proposed a modification in [41], called Pseudo-Control Hedging (PCH), to address the actuator problem. PCH could be implemented to NDI control to 'hide' the actuator dynamics from the error dynamics by moving the reference signals in opposite direction by an estimated amount the plant did not move due to system constraints like the actuator saturations. In addition, it can also prevent the integrator wind-up if an integral feedback is used or gradient based adaptation is used, as the actuator will saturate if there is an integrator wind-up.

The overall NDI design structure with PCH is shown in Figure 2.6,

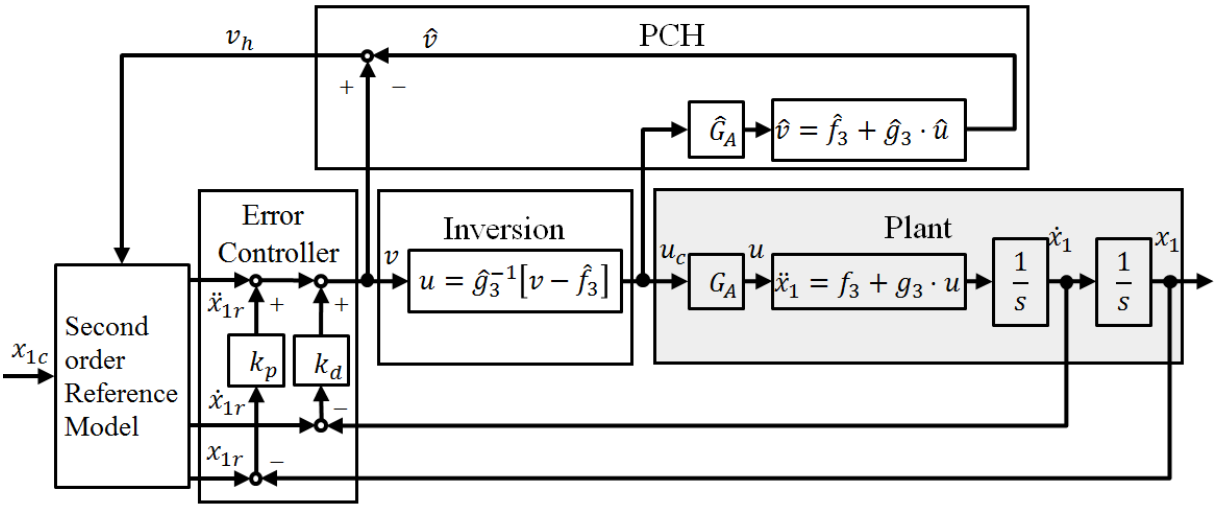


Figure 2.6 Overall NDI design structure with PCH

Due to the actuator, the commanded system input u_c will deviate from the real system input u. The amount the plant did not achieve can be quantified using the pseudo-control signals. For the example system, the pseudo-control signal v is given by the error controller as shown in Eq. (2.27). The hedging signal to the reference model is the expected reaction deficit of the plant. If $\ddot{x}_1$ is measurable, the expected reaction deficit can be directly computed. If $\ddot{x}_1$ is not measurable (in most cases of small UAVs), it has to be estimated using the actuator and system model.

$$\nu_h = \nu - \ddot{x}_1 = \nu - \hat{\nu} \tag{2.34}$$

$$\hat{\nu} = \hat{f}_3 + \hat{g}_3 \cdot \hat{G}_A \cdot u_c \tag{2.35}$$

Recalling Eq. (2.26), the deviation Δ can be splitted into two parts, one part from the modelling errors and external disturbances Δ_e and the other part from the actuator dynamics Δ_{act}. The actuator error is the error caused by the actuator, i.e. the plant reaction with actuator minor the expected plant reaction without the actuator.

$$\ddot{x}_1 = \nu + \Delta_{act} + \Delta_e \tag{2.36}$$

$$\Delta_{act} = \hat{\nu}^* - \nu = (f_3 + g_3 G_A u_c) - (f_3 + g_3 u_c) \tag{2.37}$$

where ν is computed using Eq. (2.27).

After the expected reaction deficit is calculated, the reference model dynamics can be slowed down about this amount. The reference dynamic equation is modified as

$$\ddot{x}_{1r} = \nu_r - \nu_h \tag{2.38}$$

$$\ddot{x}_{1r} = \omega_0^2(x_{1c} - x_{1r}) + 2\zeta\omega_0\dot{x}_{1r} - \nu_h \tag{2.39}$$

The second order reference model with PCH is,

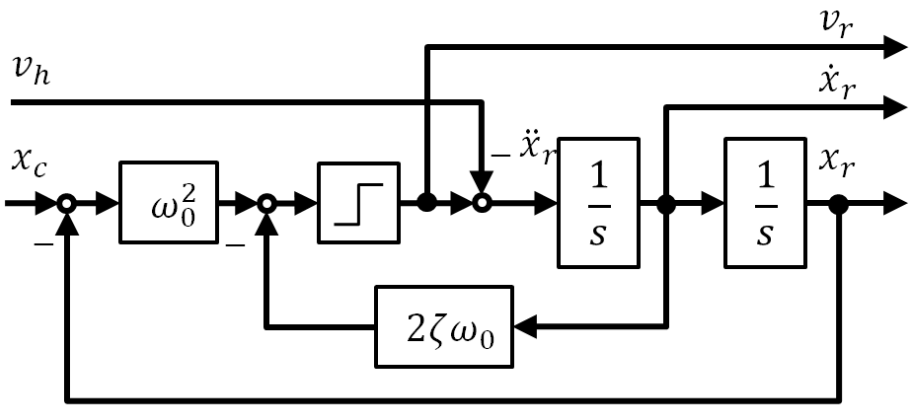


Figure 2.7 Second order reference model with PCH

The overall error dynamics of the plant can be derived to illustrate how the PCH hides the actuator dynamics.

The error feedback part remains the same.

$$\nu = \nu_r + k_d\dot{e} + k_p e \tag{2.40}$$

The closed loop dynamic equation is,

$$\ddot{x}_1 = \nu + \Delta_{act} + \Delta_e \tag{2.41}$$

$$\ddot{x}_1 = \nu_r + k_d\dot{e} + k_p e + \Delta_{act} + \Delta_e \tag{2.42}$$

$$\ddot{x}_1 = \ddot{x}_{1r} + v_h + k_d\dot{e} + k_p e + \Delta_{act} + \Delta_e \tag{2.43}$$

$$\ddot{e} + k_d\dot{e} + k_p e = -v_h - \Delta_{act} - \Delta_e \tag{2.44}$$

If there is no modeling error, or the plant reaction is measurable, the above error terms v_h cancels the deviation caused by actuator.

$$v_h = \nu - \hat{\nu} \approx \nu - \hat{\nu}^* = -\Delta_{act} \tag{2.45}$$

Then the error dynamics of the closed loop system is only excited by the modeling errors, sensor errors and external disturbances. Therefore, it can be shown that the concept of PCH could hide the actuator dynamics from the error dynamics.

$$\ddot{e} + k_d\dot{e} + k_p e = -\Delta_e \tag{2.46}$$

2.3. Backstepping

Backstepping has been developed in the early 1990s by Kristic, Kanellakoulos, and Kokotovic [17]. It is rather a method to construct a Lyapunov function whose derivative can be made negative semi-definite by a wide variety of control laws. Through the Backstepping process, the system nonlinearities can be fully or partially feedback linearized, which is quite similar to NDI design. Detailed comparison of the two methods is given in the next chapter.

The Backstepping design starts from the output and 'steps' back to the input. In each design step a Lyapunov function candidate and the corresponding control law are derived for the current virtual control variable to stabilize the considered subsystem. The Lyapunov function candidate and the control law are complete until the final control input is reached.

2.3.1. Integrator Backstepping

This is the basic form of Backstepping design [40]. Given a scalar system as below,

$$\begin{aligned} \dot{x} &= f(x) + g(x)\xi \\ \dot{\xi} &= u \end{aligned} \tag{2.47}$$

Assumptions: The origin is an equilibrium point of the subsystem $\dot{x} = f(x)$*, i.e.* $f(0) = 0$

Viewing the state variable ξ as an independent 'input' for this subsystem, we assume that there exists a state feedback control law of the form,

$$\xi_{des} = \phi(x), \qquad \phi(0) = 0 \tag{2.48}$$

Where the ξ_{des} is the desired value to stabilize the subsystem represented by the first equation. The given control law will render the derivative of a Lyapunov function candidate $V_1(x)$ negative semi-definite,

$$\dot{V}_1(x) = \frac{\partial V_1}{\partial x}\big(f(x) + g(x)\phi(x)\big) \leq -V_a(x) \leq 0 \tag{2.49}$$

where $V_a(x)$ is a positive semi definite function.

The desired subsystem is given in Eq. (2.50)

$$\dot{x} = f(x) + g(x)\xi_{des} \tag{2.50}$$

However, the ξ in the true system will always have error dynamics. The equivalent true system can be expressed as the desired system plus the error term.

$$\dot{x} = f(x) + g(x)\xi_{des} + g(x)[\xi - \xi_{des}] \tag{2.51}$$

The following error terms are defined to reformulate the system in a new form,

$$\begin{aligned} z &= \xi - \xi_{des} = \xi - \phi(x) \\ \dot{z} &= \dot{\xi} - \dot{\phi}(x) = u - \dot{\phi}(x) \\ \dot{z} &= v \end{aligned} \tag{2.52}$$

Where v is the pseudo input to the transformed but equivalent system now controlling the error. And the derivative $\dot{\phi}$ is given as below.

$$\dot{\phi} = \frac{\partial \phi}{\partial x}\frac{\partial x}{\partial t} = \frac{\partial \phi}{\partial x}\dot{x} = \frac{\partial \phi}{\partial x}[f(x) + g(x)\xi] \tag{2.53}$$

With the state transformation (z is the error variable), the resulting system is

$$\begin{aligned} \dot{x} &= f(x) + g(x)\phi(x) + g(x)z \\ \dot{z} &= v \end{aligned} \tag{2.54}$$

To assess the stability of the system, consider a Lyapunov function candidate of the form

$$V = V_1(x) + \frac{1}{2}z^2 \tag{2.55}$$

Therefore, a stabilizing state feedback law is given by

$$u = \frac{\partial \phi}{\partial x}[f(x) + g(x)\xi] - \frac{\partial V_1}{\partial x}g(x) - k[\xi - \phi(x)] \tag{2.56}$$

Because entering this into the derivative of the Lyapunov function given in Eq. (2.97), you get,

$$\dot{V} = \dot{V}_1(x) + z\dot{z} \tag{2.57}$$

$$\dot{V} = \frac{\partial V_1}{\partial x}(f(x) + g(x)\phi(x) + g(x)z) + z\left(u - \dot{\phi}(x)\right) \tag{2.58}$$

$$\dot{V} = -V_a(x) - kz^2 \tag{2.59}$$

In [40], there is a detailed figure illustrating the design process. Figure a) is the original system. Figure b) shows the modified but equivalent system after introducing the control law $-\phi(x)$. Figure c) is the Backstepping of the $-\phi(x)$ to the front of the integrator. Figure d) is the final system with a new control variable v.

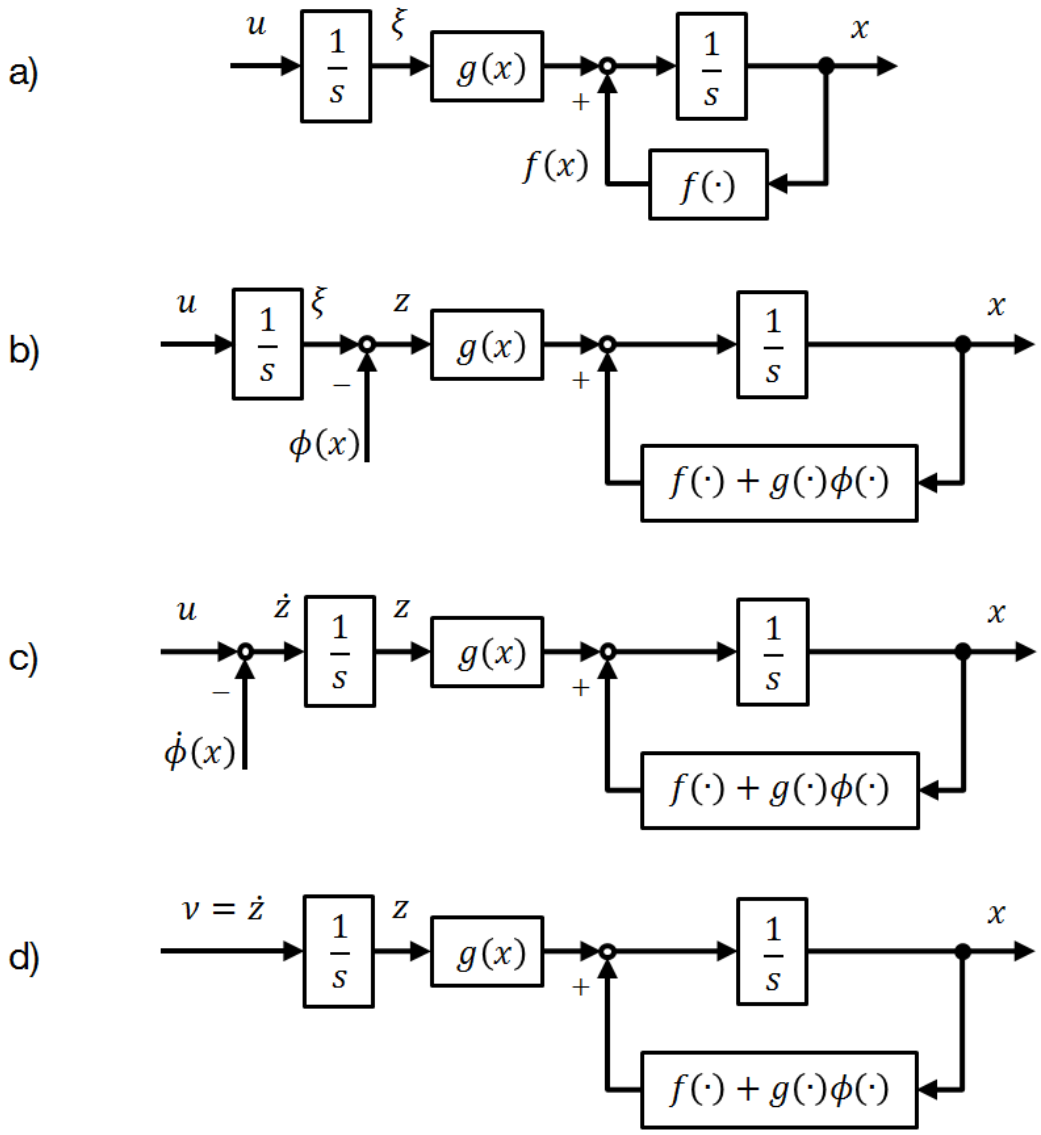


Figure 2.8 Integrator Backstepping Design Procedure ([40],p142)
(a) The original system;
(b) modified system after introducing $-\boldsymbol{\phi}(\boldsymbol{x})$;
(c) **'Backstepping' of** $-\boldsymbol{\phi}(\boldsymbol{x})$**;**
(d) the final system after the change of variables

2.3.2. Block Backstepping Design Procedure

Without loss of generality, the same example can be used to illustrate the Backstepping design procedures. Recall the system from Eq. (2.19),

$$\begin{cases}\dot{x}_1 = f_1 + g_1 \cdot x_2 \\ \dot{x}_2 = f_2 + g_2 \cdot u\end{cases}$$

There are two cascade subsystems in the example. Let's start from the first subsystem and consider only this system now, where x_2 is used to control x_1.

$$\dot{x}_1 = f_1 + g_1 x_{2,des} \tag{2.60}$$

The virtual control $x_{2,des}$ is the desired control law to stabilize the considered subsystem. Define the tracking error z_1 and propose a Lyapunov function,

$$z_1 = x_{1r} - x_1 \tag{2.61}$$

$$V_1 = \frac{1}{2} z_1^2 \tag{2.62}$$

To derive the control law, the Lyapunov function candidate is differentiated,

$$\dot{V}_1 = z_1 \dot{z}_1 = z_1(\dot{x}_{1r} - \dot{x}_1) = z_1\left(\dot{x}_{1r} - f_1 - g_1 x_{2,des}\right) \tag{2.63}$$

The following control law can drive the Lyapunov function derivative to negative semi-definite,

$$x_{2,des} = g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1) \tag{2.64}$$

$$\dot{V}_1 = -k_1 z_1^2 \tag{2.65}$$

That completes the first step. The next step is to track the $x_{2,des}$ as a command. Defining the Backstepping tracking error,

$$z_2 = x_{2,des} - x_2 \tag{2.66}$$

The control law and the error z_2 are substituted into the system equations,

$$\begin{cases}\dot{x}_1 = f_1 + g_1 x_{2,des} - g_1 z_2 \\ \dot{x}_2 = f_2 + g_2 u\end{cases}$$

Augmenting the Lyapunov function in Eq. (2.62) with the new tracking error,

$$V_2 = \frac{1}{2} z_1^2 + \frac{1}{2} z_2^2 \tag{2.67}$$

Differentiate the Lyapunov function to derive the stabilizing control law,

$$\begin{aligned}\dot{V}_2 &= z_1 \dot{z}_1 + z_2 \dot{z}_2 = -k_1 z_1^2 + z_1 g_1 z_2 + z_2\left(\dot{x}_{2,des} - \dot{x}_2\right) \\ &= -k_1 z_1^2 + z_1 g_1 z_2 + z_2\left(\dot{x}_{2,des} - f_2 - g_2 u\right)\end{aligned} \tag{2.68}$$

The control law to render the above Lyapunov function negative semi-definite can be derived,

$$u = g_2^{-1}\left(\dot{x}_{2,des} - f_2 + g_1 z_1 + k_2 z_2\right) \tag{2.69}$$

Together with Eq. (2.64) we derive the control law for the given system. And the stability is guaranteed by the Lyapunov function.

$$\dot{V}_2 = -k_1 z_1^2 - k_2 z_2^2 \tag{2.70}$$

2.3.3. Solutions to the Virtual Control Derivative

As we can see from the control law in Eq. (2.69), the derivative of the virtual control $\dot{x}_{2,des}$ is not available directly and needs to be solved.

There are two ways to solve the derivative, analytically or numerically. Analytical solution means to differentiate Eq. (2.64) and hopefully the parameters exist and are available.

$$\begin{aligned}\dot{x}_{2,des} &= \frac{d}{dt}[g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1)] \\ &= \frac{d}{dt}(x_{2r} + g_1^{-1} k_1 z_1) \\ &= \dot{x}_{2r} + g_1^{-1} k_1 \dot{z}_1 - g_1^{-2} \dot{g}_1 k_1 z_1\end{aligned}$$

Hence the final expression for the control law is,

$$u = g_2^{-1}\left[\dot{x}_{2r} + g_1^{-1} k_1 \dot{z}_1 - g_1^{-2} \dot{g}_1 k_1 z_1 - f_2 + g_1 z_1 + k_2\left(x_{2,des} - x_2\right)\right] \tag{2.71}$$

The derivative of the nonlinear functions $\dot{f}_1$ and $\dot{g}_1$ should be analytically solvable. If they are available, the analytical method provides maximum control bandwidth. However, the analytical method has strong dependence on the model accuracy and sensor availability. In addition, as system order increases, especially when the system order is bigger than three, the analytical solution gets extremely tedious and almost impossible to solve. It gets worse for a system of non-triangular form, where more states (e.g. actuator states) will be introduced when analytically solving the derivatives.

Alternatively, the virtual control derivative can be numerically estimated by means of a low pass filter, which is called ‘command filter’ in this context.

$$\dot{\hat{x}}_{2,des} = k_f\left(x_{2,des} - \hat{x}_{2,des}\right) \tag{2.72}$$

where k_f is the filter gain. It is much easier to implement, but it introduces additional phase lag into the system hence time scale separation is needed. By increasing the bandwidth of the command filter, the performance could be made arbitrarily close to that of the analytical method. However, the bandwidth is limited by the sensor noise and the control bandwidth of the next inner loop.

Theoretically, the analytical method provides the best results. However, in practical implementation, the choice between analytical and numerical methods has to be made on a case-by-case basis. Modelling the real plant is impossible, so the analytical solution also just provides an estimation of the derivative whose accuracy depends on the model accuracy and sensor quality.

2.4. $\mathcal{L}_1$ Adaptive Control

2.4.1. Predictor Based MRAC

The $\mathcal{L}_1$ adaptive control is developed on the basis of the predictor-based MRAC. The predictor based MRAC is given in Figure 2.9.

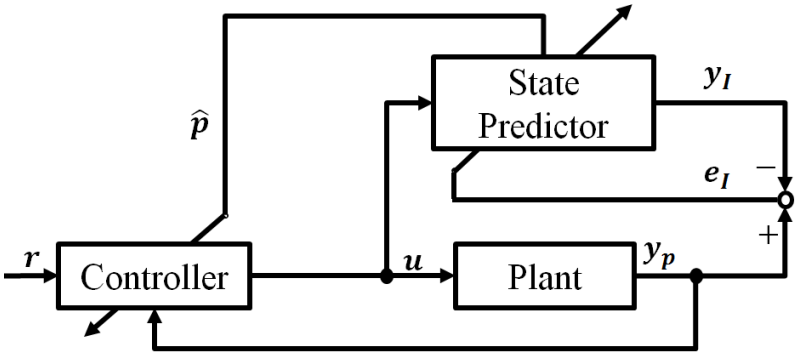


Figure 2.9 Predictor based MRAC

Considering a deterministic, time-continuous and scalar case first order plant with unknown parameters: for instance, the system matrix a and a time varying value f, which e.g. could be a nonlinear function of x but is not limited to that. The input parameter b is assumed to be known.

$$\dot{x} = ax + b(u + f) \tag{2.73}$$

The desired reference dynamics for the above system is,

$$\dot{x}_m = a_m x_m + b_m r \tag{2.74}$$

where a_m is negative. The system in Eq. (2.73) can be written as,

$$\dot{x} = a_m x + (a - a_m)x + b(u + f) \tag{2.75}$$

The state predictor mimics the structure of the parameterized plant (where $\hat{}$ denotes the estimates of the unknown parameters),

$$\dot{\hat{x}} = a_m \hat{x} + (\hat{a} - a_m)x + b\left(u + \hat{f}\right) \tag{2.76}$$

Two adaptive parameters θ_x and θ_f are introduced in the state predictor to estimate the unknown parameters by adaptation (the adaptation law will be derived later).

$$\theta_x = \frac{a_m - \hat{a}}{b}$$
$$\theta_f = -\hat{f} \tag{2.77}$$

The state predictor with the adaptive parameters is,

$$\dot{\hat{x}} = a_m \hat{x} - b\theta_x x + b\left(u - \theta_f\right) \tag{2.78}$$

It can be also written as,

$$\dot{\hat{x}} = a_m\hat{x} + bu - b\underbrace{(\theta_x x + \theta_f)}_{\Delta u} \tag{2.79}$$

The adaptive part $(-\Delta u)$ obtained by adaptation makes the reference model behave like the plant. Then, vice versa, the adaptive part Δu would be the inut difference that would make the plant behaves like the desired reference model in the controller. The control law also consists of a normal feedforward part $k_r r$,

$$u = k_r r + \Delta u = k_r r + \underbrace{\theta_x x + \theta_f}_{\Delta u} \tag{2.80}$$

where,

$$k_r = \frac{b_m}{b} \tag{2.81}$$

The matching conditions can be computed by inserting Eq. (2.80) into Eq. (2.75),

$$\begin{aligned} \dot{x} &= a_m x + (a - a_m)x + b\big(\theta_x x + k_r r + \theta_f + f\big) \\ \dot{x} &= a_m x + (a - a_m + b\theta_x)x + bk_r r + b\big(\theta_f + f\big) \end{aligned} \tag{2.82}$$

Recalling the reference dynamics with Eq.(2.74),

$$\dot{x}_m = a_m x + b_m r \tag{2.83}$$

To render the dynamic equation similar to the reference dynamics, the match conditions should be given as in (2.84), where * denotes the true values of the adaptive parameters,

$$\begin{aligned} \theta_x^* &= \frac{a_m - a}{b} \\ \theta_f^* &= -f \end{aligned} \tag{2.84}$$

The adaptation law can be derived from the predictor error dynamics using the Lyapunov stability theorem [42] [43]. The error terms are defined as,

$$\begin{aligned} \dot{\tilde{x}} &= \dot{\hat{x}} - \dot{x} \\ \tilde{\theta}_x &= \theta_x^* - \theta_x \\ \tilde{\theta}_f &= \theta_f^* - \theta_f \end{aligned} \tag{2.85}$$

The predictor error dynamics can be computed by subtracting Eq. (2.76) by Eq. (2.75),

$$\dot{\tilde{x}} = a_m\tilde{x} + b\left(\underbrace{\theta_x^* - \theta_x}_{\tilde{\theta}_x}\right)x + b\left(\underbrace{\theta_f^* - \theta_f}_{\tilde{\theta}_f}\right) \tag{2.86}$$

Define a Lyapunov function candidate as,

$$V = \frac{1}{2}\tilde{x}^2 + \frac{1}{2\Gamma_x}\tilde{\theta}_x{}^2 + \frac{1}{2\Gamma_f}\tilde{\theta}_f{}^2 \tag{2.87}$$

The Lyapunov function derivative is,

$$\begin{aligned} \dot{V} &= \tilde{x}\dot{\tilde{x}} + \frac{1}{\Gamma_x}\tilde{\theta}_x\dot{\tilde{\theta}}_x + \frac{1}{\Gamma_f}\tilde{\theta}_f\dot{\tilde{\theta}}_f \\ \dot{V} &= \tilde{x}\left(a_m\tilde{x} + b\tilde{\theta}_x x + b\tilde{\theta}_f\right) + \frac{1}{\Gamma_x}\tilde{\theta}_x\dot{\tilde{\theta}}_x + \frac{1}{\Gamma_f}\tilde{\theta}_f\dot{\tilde{\theta}}_f \\ \dot{V} &= a_m\tilde{x}^2 + b\tilde{\theta}_x x\tilde{x} + b\tilde{\theta}_f\tilde{x} + \frac{1}{\Gamma_x}\tilde{\theta}_x\dot{\tilde{\theta}}_x + \frac{1}{\Gamma_f}\tilde{\theta}_f\dot{\tilde{\theta}}_f \end{aligned} \tag{2.88}$$

To render the Lyapunov function derivative negative semi-definite, a possible solution is to drive the second part of the function zero,

$$\dot{V} = a_m\tilde{x}^2 + \underbrace{b\tilde{\theta}_x x\tilde{x} + b\tilde{\theta}_f\tilde{x} + \frac{1}{\Gamma_x}\tilde{\theta}_x\dot{\tilde{\theta}}_x + \frac{1}{\Gamma_f}\tilde{\theta}_f\dot{\tilde{\theta}}_f}_{=0}$$

We can obtain the following adaptation law where Γ_x and Γ_f are adaptation gains,

$$\begin{aligned} \dot{\tilde{\theta}}_x &= -\Gamma_x\, bx\tilde{x} \\ \dot{\tilde{\theta}}_f &= -\Gamma_f\, b\tilde{x} \end{aligned} \tag{2.89}$$

So that,

$$\dot{V} = a_m\tilde{x}^2 \leq 0 \tag{2.90}$$

Using the Barbarlat's Lemma [43] it may be proven that the prediction error tends to zero asymptotically. The overall structure of the scalar example can be seen in Figure 2.10

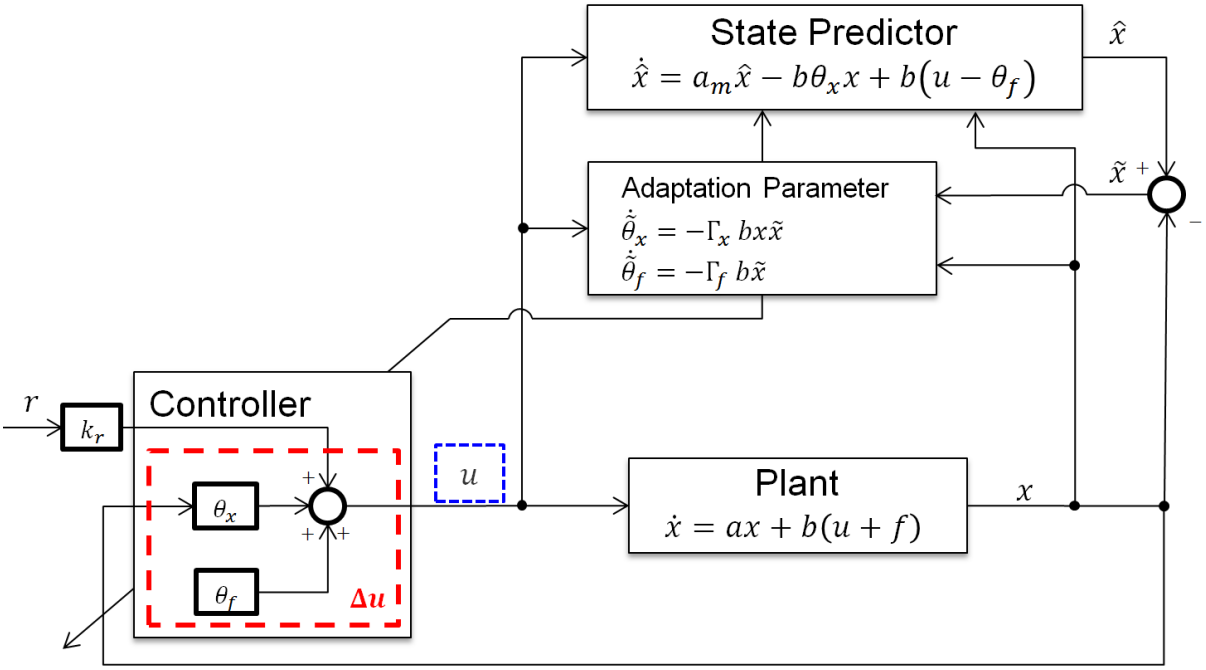


Figure 2.10 Detailed structure of the scalar example using Predictor based MRAC

However, in the presence of uncertainties it may exhibit some adverse characteristics in transient behavior like:

- Control signals of high-frequency or large amplitudes
- Large transient errors
- Slow convergence rate of tracking errors

Different strategies have been investigated to address these characteristics. The $\mathcal{L}_1$ control is one of them.

2.4.2. $\mathcal{L}_1$ Control Architecture

The $\mathcal{L}_1$ adaptive control follows the main structure of the predictor based MRAC but adds a low pass filter $C(s)$ on the control input u indicated by blue in Figure 2.10. By doing so, the state predictor can have fast adapting parameters in the state predictor, but no high frequency signals are fed through the plant. Adaptation is decoupled from robustness [42]. The speed of adaptation is limited only by the hardware limit. A comprehensive textbook with detailed derivation and mathematical proof is [42]. The basic architecture of $\mathcal{L}_1$ adaptive control is shown in Figure 2.11.

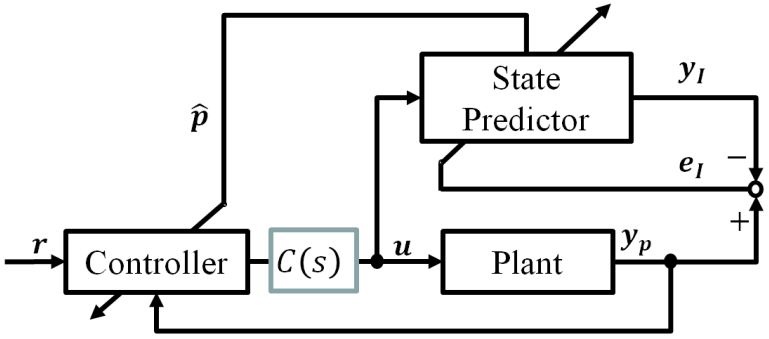


Figure 2.11 $\mathcal{L}_1$ adaptive control

2.4.3. Piecewise Constant Update Laws for $\mathcal{L}_1$ Controller

Having similar performance with the gradient based update laws, piecewise constant update law [31] appears more like linear feedback, which makes it relatively easier to implement and perhaps closer to the certification process. In this thesis, this update law is employed in the augmented $\mathcal{L}_1$ controller for the quadrotors.

For the purpose of analysis, consider the following first order system,

$$\dot{x} = ax + b(u + f) \tag{2.91}$$

where x is the plant state; a is unknown plant parameter but with known bounds; b is known and f is some nonlinear uncertainty. The desired reference dynamics for the above system is,

$$\dot{x}_m = a_m x_m + b_m r \tag{2.92}$$

The system in Eq. (2.91) can be written as,

$$\dot{x} = a_m x + b(u + \sigma) \tag{2.93}$$

where

$$\sigma = \frac{a - a_m}{b} x + f \tag{2.94}$$

The state predictor will take a similar form with the plant but with an adaptive parameter,

$$\dot{\hat{x}} = a_m \hat{x} + b(u + \hat{\sigma}) \tag{2.95}$$

The prediction error dynamics can be computed,

$$\dot{\tilde{x}} = \dot{\hat{x}} - \dot{x} = a_m\tilde{x} + b\hat{\sigma} - b\sigma \tag{2.96}$$

Let's try to solve the error state $\tilde{x}$ in time domain, both sides are multiplied by $e^{-a_m t}$

$$e^{-a_m t}\dot{\tilde{x}} = e^{-a_m t}(a_m\tilde{x} + b\hat{\sigma} - b\sigma)$$

$$e^{-a_m t}\dot{\tilde{x}} - e^{-a_m t}a_m\tilde{x} = (e^{-a_m t}\tilde{x})' = e^{-a_m t}b\hat{\sigma} - e^{-a_m t}b\sigma$$

Integrating both sides by one time step from t_0 to $t_0 + Ts$,

$$e^{-a_m(t_0+Ts)}\tilde{x}(t_0 + Ts) - e^{-a_m t_0}\tilde{x}(t_0) = \int_{t_0}^{t_0+Ts} e^{-a_m t}b\hat{\sigma}\,dt - \int_{t_0}^{t_0+Ts} e^{-a_m t}b\sigma(t)\,dt$$

Note that the adaptive parameter $\hat{\sigma}$ remains constant within one time step; hence the first term on the right hand side can be solved. The second term is the error caused by the uncertainties or disturbance $\sigma(t)$ and thus not analytically solvable.

$$\int_{t_0}^{t_0+Ts} e^{-a_m t}b\hat{\sigma}\,dt - \int_{t_0}^{t_0+Ts} e^{-a_m t}b\sigma(t)dt = -\frac{1}{a_m}\left(e^{-a_m(t_0+Ts)} - e^{-a_m t_0}\right)b\hat{\sigma}(t_0) - \Delta(\sigma)$$

The prediction error can be expressed as follows, assuming $\tilde{x}(t_0) = \tilde{x}_0$,

$$e^{-a_m(t_0+Ts)}\tilde{x}(t_0 + Ts) - e^{-a_m t_0}\tilde{x}_0 = -\frac{1}{a_m}\left(e^{-a_m(t_0+Ts)} - e^{-a_m t_0}\right)b\hat{\sigma}(t_0) - \Delta(\sigma)$$

$$\tilde{x}(t_0 + Ts) = e^{a_m Ts}\tilde{x}_0 - \frac{1}{a_m}(1 - e^{a_m Ts})b\hat{\sigma}(t_0) - \Delta(\sigma) \tag{2.97}$$

There is nothing much we can do about $\Delta(\sigma)$ but it is bounded given bounded uncertainties. By setting the sampling time Ts small enough, one can keep the prediction error small and achieve arbitrary performance improvement.

The piecewise constant update law is derived to render Eq. (2.97) to zero. It will fully compensate the prediction error accumulated on the previous sampling period.

$$\tilde{x}(t_0 + Ts) = e^{a_m Ts}\tilde{x}_0 - \frac{1}{a_m}(1 - e^{a_m Ts})b\hat{\sigma}(t_0) = 0$$

$$\hat{\sigma}(t_0) = \frac{1}{b}(1 - e^{a_m Ts})^{-1}a_m e^{a_m Ts}\tilde{x}_0$$

The sampled update law is,

$$\hat{\sigma} = \hat{\sigma}(iTs) = \frac{1}{b}(1 - e^{a_m Ts})^{-1}a_m e^{a_m Ts}\tilde{x}(iTs), \qquad i = 1,2,3,\ldots$$

The control law is computed as the output of the low pass filter $C(s)$,

$$u(s) = C(s)\left(\frac{b_m}{b}r(s) - \hat{\sigma}(s)\right)$$

2.5. Data fusion

The control system is normally designed in two parts: a 'full-state feedback' control part based on the assumption that all the state variables can be measured; and a data fusion part to estimate the state for purposes of feedback control based on available sensor data. Extended Kalman filter and Luenberger observer are briefly introduced this section, taken from two reference textbooks [44] and [45].

2.5.1. Luenberger Observer

The concept of an observer for a dynamic process was introduced in 1966 by Luenberger [46]. The Kalman filter, which appeared several years earlier, is in fact an important special case of a Luenberger observer, where the observer gain is optimized for the noise present in the input and observation. The Luenberger observer could have constant feedback gain, which is useful in cases that the covariance of the observation is difficult to determine but has to be roughly estimated (e.g. vision sensor). As no covariance update is needed, the computation burden is also heavily reduced. Such an observer is used on the linear portion of the position dynamics of the quadrotor.

Consider the same linear continuous-time dynamic system as in Eq. (2.99), a full-order Luenberger observer has the generic form,

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{K}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}) \tag{2.98}$$

The generic structure is shown as in

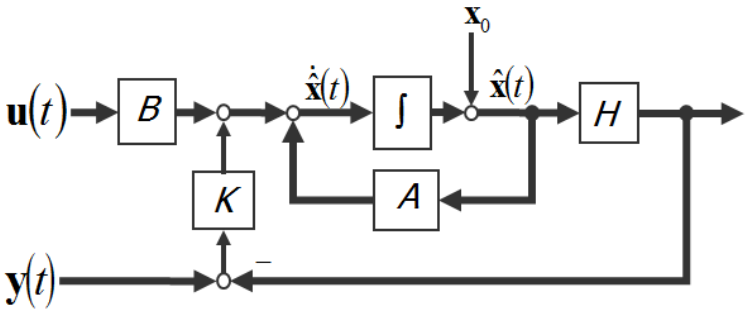


Figure 2.12 Luenberger observer structure

Define the observation error

$$\tilde{\mathbf{x}} = \hat{\mathbf{x}} - \mathbf{x}$$

The error dynamics can be derived by subtracting Eq. (2.98) by Eq. (2.99),

$$\dot{\tilde{\mathbf{x}}} = \mathbf{A}\tilde{\mathbf{x}} + \mathbf{K}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}) = (\mathbf{A} - \mathbf{K}\mathbf{H})\tilde{\mathbf{x}} = \hat{\mathbf{A}}\tilde{\mathbf{x}}$$

The pole placement method can be used to design the observer gain matrix $\mathbf{K}$ to place the poles of the observer, i.e. the eigenvalues of $\hat{\mathbf{A}}$, to appropriate values in the left half of the complex plane. Alternatively, the gain matrix can be chosen as a Kalman filter gain matrix, since the observer given in Eq. (2.98) has the structure of a Kalman filter. The Kalman gain

equation is derived with more details in the next section. In most applications, the steady-state covariance matrix will be used,

$$\mathbf{K} = \mathbf{P}\mathbf{H}^{\mathbf{T}}\mathbf{R}^{-1}$$

where $\mathbf{P}$ is the covariance matrix of the observation error and it satisfies the algebraic Riccati equation.

$$\mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^{\mathbf{T}} - \mathbf{P}\mathbf{H}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{P} + \mathbf{Q} = \mathbf{0}$$

It is rarely possible to determine the exact sensor covariance $\mathbf{Q}$ and $\mathbf{R}$, so they are best treated as design parameters that can be varied to achieve overall system design objectives. There are available algorithms that can be used to solve the Kalman gain in MATLAB, such as 'kalman', 'care', 'dare' [47].

The use of algebraic Riccati equation is usually preferable to design the gain matrix, compared with the pole placement method. When two or more measurements are taken, specification of the eigenvalues of $\widehat{\mathbf{A}}$ does not uniquely specify the gain matrix. In addition to the eigenvalues, most of the eigenstructure needs to be specified.

2.5.2. Kalman Filter

The Kalman filter is one of the fundamental tools for estimation applications. Dan Simon shows in [44] that the Kalman filter is the optimal estimator when the noise is Gaussian, and it is the optimal linear estimator when the noise is not Gaussian. Over the past few decades, this estimation algorithm has found applications in virtually every area of engineering. It is natural to apply it in the quadrotor applications.

A standard Kalman filter has two distinct phases: 'Predict' and 'Correct', or they are also called 'Time update' and 'Measurement update', respectively. The prediction step predicts the state by integrating the high frequency IMU data with strap-down equations. It also propagates the state covariance. At arrival of information other than IMU data (e.g. GPS, vision, barometer), the correction step is performed to correct the predicted state and to update the covariance.

In digital implementation, the continuous state space model is usually transferred to its discrete counterpart. Given a continuous state space model as follows,

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{w}(t) \\ \mathbf{y}(t) = \mathbf{H}\mathbf{x}(t) + \mathbf{v}(t) \end{cases} \tag{2.99}$$

where $\mathbf{w}$ and $\mathbf{v}$ are Gaussian noises with covariance $\mathbf{Q}$ and $\mathbf{R}$ repectively. $\mathbf{w}$ is normally the system process noise representing the model uncertainties and $\mathbf{v}$ is the measurement noise of the sensors. To transfer the system into discrete time, Eq. (2.99) needs to be integrated for one time step. A common factor $e^{-\mathbf{A}t}$ is multiplied on both sides for easier computation.

$$e^{-\mathbf{A}t}\dot{\mathbf{x}} = e^{-\mathbf{A}t}\mathbf{A}\mathbf{x} + e^{-\mathbf{A}t}\mathbf{B}\mathbf{u} + e^{-\mathbf{A}t}\mathbf{w}$$

$$e^{-\mathbf{A}t}\dot{\mathbf{x}} - e^{-\mathbf{A}t}\mathbf{A}\mathbf{x} = \left(e^{-\mathbf{A}t}\mathbf{x}\right)' = e^{-\mathbf{A}t}\mathbf{B}\mathbf{u} + e^{-\mathbf{A}t}\mathbf{w}$$

Integrating both side from step $k-1$ to k, with the sampling time Δt

$$e^{-\mathbf{A}(t+\Delta t)}\mathbf{x}_k - e^{-\mathbf{A}t}\mathbf{x}_{k-1} = \int_t^{t+\Delta t} e^{-\mathbf{A}t}\mathbf{B}\mathbf{u}dt + \int_t^{t+\Delta t} e^{-\mathbf{A}t}\mathbf{w}dt$$

The noise is assumed to be Gaussian and can be ignored and the input $\mathbf{u}$ can be assumed to be constant in the last time step.

$$e^{-\mathbf{A}(t+\Delta t)}\mathbf{x}_k - e^{-\mathbf{A}t}\mathbf{x}_{k-1} \approx \int_t^{t+\Delta t} e^{-\mathbf{A}t}dt\,\mathbf{B}\mathbf{u}_k = -\mathbf{A}^{-1}e^{-\mathbf{A}t}\Big|_t^{t+\Delta t}\mathbf{B}\mathbf{u}_k$$

$$e^{-\mathbf{A}(t+\Delta t)}\mathbf{x}_k - e^{-\mathbf{A}t}\mathbf{x}_{k-1} \approx \left(-\mathbf{A}^{-1}e^{-\mathbf{A}(t+\Delta t)} + \mathbf{A}^{-1}e^{-\mathbf{A}t}\right)\mathbf{B}\mathbf{u}_k$$

$$\mathbf{x}_k = e^{\mathbf{A}\Delta t}\mathbf{x}_{k-1} + \left(-\mathbf{A}^{-1} + \mathbf{A}^{-1}e^{\mathbf{A}\Delta t}\right)\mathbf{B}\mathbf{u}_k$$

The first order Taylor approximation is often used in the discretization process given small time step,

$$e^{\mathbf{A}\Delta t} \approx \mathbf{I} + \mathbf{A}\Delta t$$

Hence,

$$\begin{cases} \mathbf{\Phi} = \mathbf{I} + \mathbf{A}\Delta t \\ \mathbf{\Gamma} = (-\mathbf{A}^{-1} + \mathbf{A}^{-1}e^{\mathbf{A}\Delta t})\,\mathbf{B} = \mathbf{B}\Delta t \end{cases} \tag{2.100}$$

With the approximation, the discrete counterpart of the state space model is,

$$\begin{cases} \mathbf{x}_k = \mathbf{\Phi}\mathbf{x}_{k-1} + \mathbf{\Gamma}\mathbf{u}_{k-1} \\ \mathbf{y}_k = \mathbf{H}\mathbf{x}_k \end{cases} \tag{2.101}$$

With the discrete state space equation, the state can be propagated to the next time step, and the error covariance $\mathbf{P}_k^-$ can be propagated.

$$\begin{cases} \hat{\mathbf{x}}_k = \mathbf{\Phi}\mathbf{x}_{k-1} + \mathbf{\Gamma}\mathbf{u}_{k-1} \\ \mathbf{P}_k^- = \mathbf{\Phi}\mathbf{P}_{k-1}\mathbf{\Phi}^T + \mathbf{\Gamma}\mathbf{Q}\mathbf{\Gamma}^T \end{cases} \tag{2.102}$$

That's the predictor part of the Kalman filter. When a measurement arrives, the correction part can be triggered. Based on the measurement covariance, the optimal Kalman gain is computed and then the states $\hat{\mathbf{x}}_k$ and error covariance $\mathbf{P}_k$ are updated with the computed Kalman gain. The correction equations are as follows,

$$\begin{cases} \mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}^T(\mathbf{H}\mathbf{P}_k^-\mathbf{H}^T + \mathbf{R})^{-1} \\ \hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_k^-) \\ \mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_k^- \end{cases} \tag{2.103}$$

The basic equations and operation cycle of a standard discrete time Kalman filter are given in Figure 2.13.

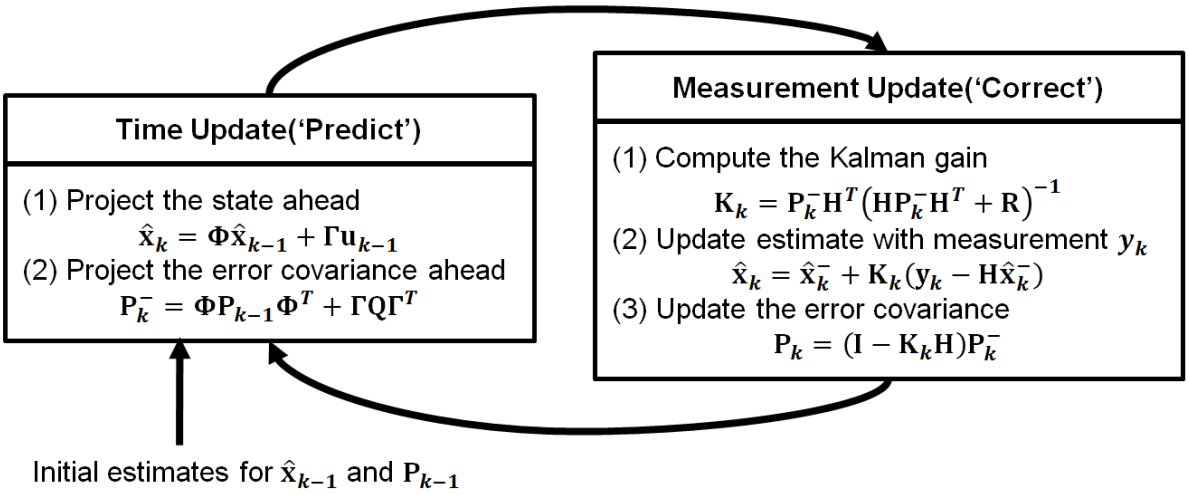


Figure 2.13 Discrete time Kalman filter

2.5.3. Extended Kalman Filter (EKF)

Extended Kalman Filter (EKF) can be applied on nonlinear systems like the quadrotors. The EKF consists of two parts. The first part only integrates the high frequency IMU data and propagates it to a nominal state, which drifts with time due to IMU noises and possible model imperfections. With respect to the nominal state, the nonlinear system can be linearized and an error state space model can be formulated. The second part is an error state Kalman filter applied on the linear error state space model. In the prediction step, only the covariance needs to be propagated and in the correction part, the predicted error state is corrected and the covariance is updated. Both measurement and prediction are considered as error states, computed from the values of the nominal state. The estimated error state is then injected into the nominal state and reset to zero for the next time step. The overall EKF structure is given in Figure 2.14 and the linear error state Kalman filter is given in Figure 2.15.

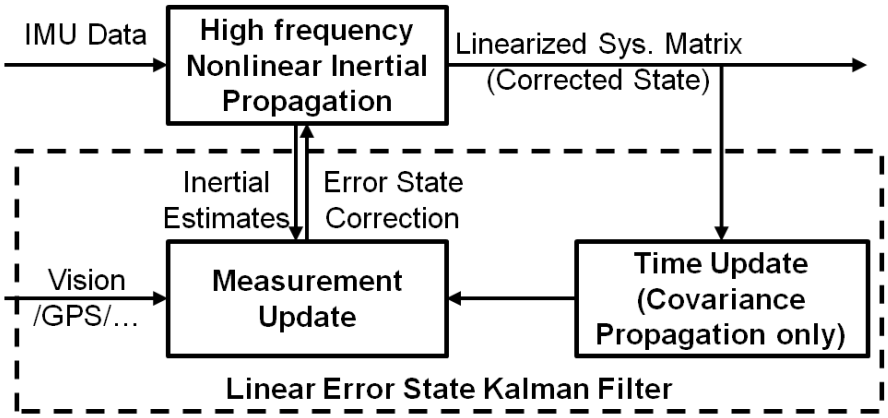


Figure 2.14 EKF structure

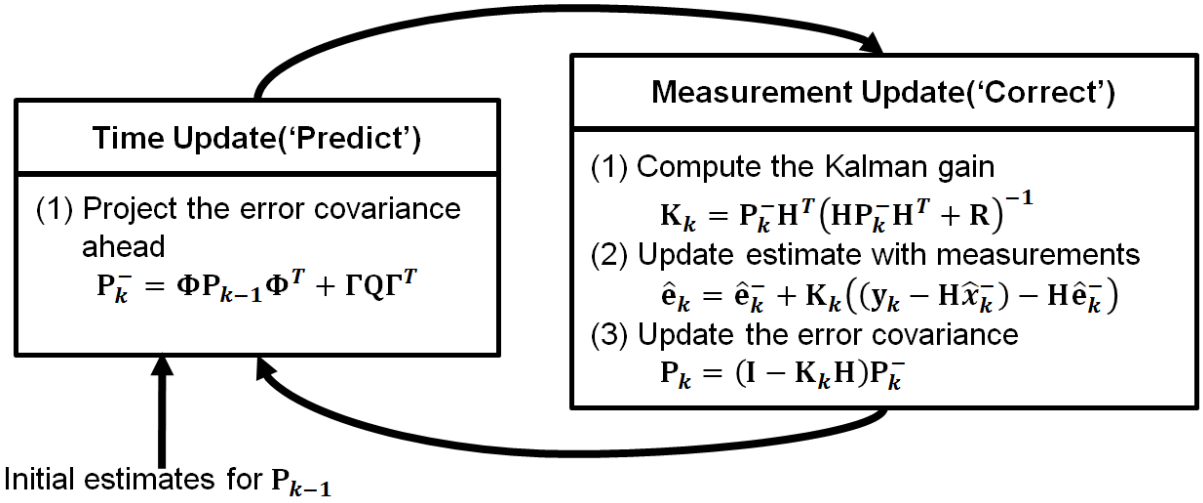


Figure 2.15 Linear error state Kalman filter

The EKF has the following remarkable assets:

- The error-state system is always operating close to the origin, and therefore providing a guarantee that the linearization validity holds at all times.
- As the error state is always small, all second order products are negligible. This makes the computation of the Jacobian relatively easy and fast.
- The error dynamics is slow because all the large-signal dynamics have been integrated in the nominal-state
- The inertial propagation and Kalman filter can be processed with different sampling rate.

3. Generic Comparison of Dynamic Inversion and Backstepping

The purpose of the comparison is to find an optimal control law in the nonlinear control field, so that control accuracy and bandwidth can be maximized yet with adequate stability margin and robustness.

Typically, there are two inputs and one output of the flight control law design as shown in Figure 3.1. The two inputs are the reference signals and feedback signals; the output is the calculated system input. The reference signals are normally generated by the reference model, which can provide the reference signals up to n-th state derivatives (n is the system order excluding the actuator dynamics). Knowledge of the plant dynamics, especially state saturations and constraints, can be included in the reference model so that physically-capable and consistent reference signals can be generated.

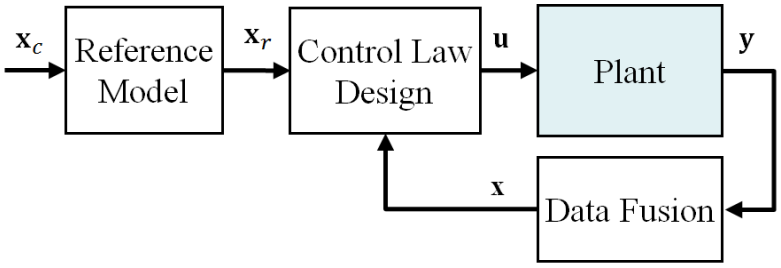


Figure 3.1 Flight Control System Structure

The role of the control design is to extract as much information as possible from both the reference and feedback signals to enable the plant to track a desired trajectory as close as possible in the presence of noise and disturbances despite of uncertainties about the plant. The reference feedforward control tries to track a desired trajectory and the feedback control tries to maintain the plant state near the desired trajectory. The feedforward reference signals give active control, while the feedback control is passive control to regulate errors and reject disturbances. On utilizing the feedback signals, nonlinear control has the advantage over the linear control that the feedback gains are state dependent so that gain scheduling over the flight envelope may not be necessary, or hybrid methods may be used to gain performance. On utilizing the reference signals, it mainly depends on model uncertainties. Feedforward control with signals of high uncertainty deteriorates stability and robustness margin. The difference of different nonlinear control design lies in the feedforward control, while the feedback control of different nonlinear designs can be made arbitrarily close by fine gain tuning.

NDI and Backstepping are widely used and probably the most popular nonlinear control methods applied to multirotors. The design principle and procedures of NDI and Backstepping are different; however, the feedback linearization structure and the final control

law have many similarities. The differences in those control designs could eventually be categorized by the utilization of the feedforward and feedback signals.

The control designs are compared with the same generic example given in Eq. (2.19), to evaluate not only the differences between the two methods but also difference in various designs by the same method. The comparison results are again verified and validated in Chapter 5 and 6 with the quadrotor control applications.

Eight different control designs of the two methods are analyzed and compared. There are three NDI designs a), b) and c);

a) Non-cascaded NDI design
b) Cascaded NDI design
c) Non-cascaded NDI design with original model

and five Backstepping designs, listed as follows,

d) Analytical Backstepping with original model
e) Analytical Backstepping with transformed model
f) Cascaded Backstepping design
g) Backstepping with command filter
h) Command Filtered Backstepping (CFB)

The following issues are addressed through the comparisons,

- The time scale separation analysis in both cascaded and non-cascaded NDI designs
- The difference between the Backstepping error dynamics and NDI error dynamics
- The difference between analytical estimation and numerical estimation of the virtual control derivative in Backstepping designs
- The conditions that the same control law can be obtained between NDI and Backstepping methods.
- The effect of the additional term in Backstepping compared with NDI.
- The improvement of CFB over the Backstepping with command filter.

3.1. Nonlinear Dynamic Inversion Designs

There are two different NDI design structures: the cascaded and non-cascaded design. Cascaded design divide control system into two or more nested loops where the nonlinearities are feedback linearized separately in their respective loops. The NDI transformation can be made much easier compared with a non-cascaded design. However, time scale separation is needed between the cascaded loops for stability consideration. It means that the outer loop is designed to be much slower than the next inner loop so that the inner loop can be assumed to reach the steady state without affecting the outer loop stability. This simplifies the control design but compromises the overall control performance. For stability consideration, the cascaded structure is not able to use the reference inner loop state derivative. Instead, it is commonly estimated by a low pass filter. The non-cascaded design feedback linearized the plant dynamics in one loop so that the reference state derivative can be used without affecting the stability. Therefore no additional time scale separation is needed within the controller.

3.1.1. Non-cascaded NDI design (Design a)

For better comparison, the same example system as in Eq. (2.19) is used to apply all different NDI and Backstepping designs. In addition, the same second order reference model given in Eq. (2.25) is used for all.

The example given in section 2.2 is the non-cascaded NDI design and the control structure is shown in Figure 2.5, though the PCH is not implemented here for comparison purpose. Recall the corresponding control law given in Eq. (2.33),

$$\begin{cases} u = g_3^{-1}[v - f_3] \\ v = v_r + 2\zeta\omega_0\dot{e} + \omega_0^2 e \end{cases}$$

3.1.2. Cascaded NDI design (Design b)

Cascaded NDI design makes use of the time scale separation principle to hide the inner loop dynamics so that quasi-stability in the outer loop can be established. Time scale separation exists in many dynamical systems. It can arise due to small time constants, moments of inertia, actuator dynamics and many other effects. Using the natural separation between ‘fast’ and ‘slow’ variables to reduce the complexity of a dynamical system can greatly simplify the control designs and analysis problem.

The cascaded design is less restrictive than the non-cascaded design and it can be applied to any system of non-triangular form.

The second order system could be separated into two loops. First, the outer loop system is considered,

$$\dot{x}_1 = f_1 + g_1 \cdot x_2 \approx f_1 + g_1 \cdot x_{2,des} \tag{3.1}$$

The virtual control input $x_{2,des}$ is the desired value of x_2 to stabilize the outer loop. Applying the NDI transformation and linear controller, the control law for the outer loop is,

$$x_{2,des} = g_1^{-1}[-f_1 + \dot{x}_{1r} + k_1(x_{1r} - x_1)] \tag{3.2}$$

The desired $x_{2,des}$ is commanded to the inner loop. Usually an inner loop reference model is used to generate smooth signals for the inner loop. In this case, a first order low pass filter is appropriate.

$$\dot{\hat{x}}_{2,des} = k_f\left(x_{2,des} - \hat{x}_{2,des}\right) \tag{3.3}$$

The inner loop control law is designed similarly to the outer loop,

$$u = g_2^{-1}[-f_2 + \dot{\hat{x}}_{2,des} + k_2(\hat{x}_{2,des} - x_2)] \tag{3.4}$$

There are approaches (e.g. [48]) that do not use an inner loop reference model. In certain case they are equivalent. It can be shown that when the filter gain k_f and the feedback gain k_2 are the same, the control law is reduced to the same as without any filter,

$$u = g_2^{-1}[-f_2 + k_2(x_{2,des} - x_2)] \tag{3.5}$$

However, the inner loop reference model gives an additional design freedom and, as shown in later sections, it performs similar tasks with the command filter in the Backstepping design.

Classically for cascaded design, the reference model is designed to have the same system order or relative degree of the nested loop. For instance, if the outer loop has a relative degree of 1, the reference model is then a first order reference model. However, the true system order is larger than the outer loop. It makes sense to extend the dynamic order of the reference model to the true system order. The reference model should have a relative degree of 2 for the second order example. Then, for the cascaded design, the problem is that the 2nd derivative of the reference state is not able to be feedforward to the inner loop due to the separation of control loops. This means, there is model information we could use but we didn't use in the cascaded design. Possible loss of control bandwidth may occur.

After all, the control design structure is given below,

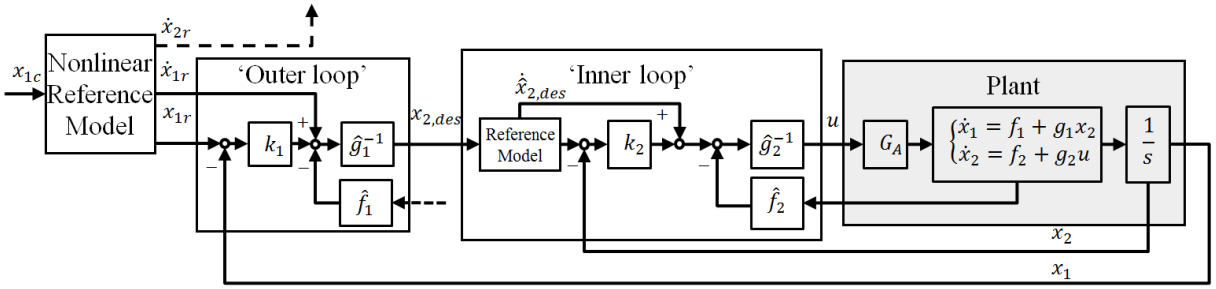


Figure 3.2 Cascaded NDI design

As we can see, the dynamic inversion complexity reduces with the cascaded design, especially when the system order increases. The derivatives of the nonlinear functions are no longer needed. The drawback is that the reference model information is not used completely, i.e. the reference second order derivative cannot be feed through the inner loop, but a numerical estimate $\dot{\hat{x}}_{2,des}$ is used instead. The performance can be drawn arbitrarily close to

the non-cascaded design by increasing the filter bandwidth. However, the filer bandwidth k_f is limited due to sensor noise.

3.1.2.1. Time scale separation

The time constant or control bandwidth of each loop may be influenced by the feedback gains. Take a first order system with a pure integrator as in Eq. (3.6) for instance, the inverse of the feedback gain k is the time constant of the system. And the control bandwidth (dynamic range), is at the value of the feedback gain k. For more complex system, the control bandwidth can be identified from the closed-loop frequency response, where the magnitude decreases more than 3-dB [45].

$$\dot{x} = u = -kx = -\frac{1}{T}x \tag{3.6}$$

The time scale separation normally requires the outer loop to be 3-5 times faster than the inner loop, and the inner loop 3-5 times faster than the actuator dynamics.

The section aims to quantify the time scale separation with a linear example. Consider the following second order linear system,

$$\ddot{x} = u \tag{3.7}$$

A cascaded control loop using simple linear feedback can be designed to stabilize the system,

$$\begin{cases} \dot{x}_c = k_1(x_c - x) \\ u = k_2(\dot{x}_c - \dot{x}) \end{cases} \tag{3.8}$$

Where the feedback gains k_1 and k_2 determine the control bandwidth of each loop. The transfer function of the close loop system can be derived,

$$\begin{aligned} u = \ddot{x} = k_2(\dot{x}_c - \dot{x}) = k_2[k_1(x_c - x) - \dot{x}] \\ \ddot{x} = k_2k_1x_c - k_2k_1x - k_2\dot{x} \\ x = \frac{k_1k_2}{s^2 + k_2s + k_1k_2}x_c \end{aligned} \tag{3.9}$$

This is very similar to the transfer function of the second order reference model given in Eq. (2.25). Hence, we can relate the feedback gains to the relative damping of the system.

$$\begin{cases} k_2 = 2\zeta\omega_0 \\ k_1k_2 = \omega_0^2 \end{cases} \tag{3.10}$$

We have,

$$\begin{cases} k_1 = \dfrac{\omega_0}{2\zeta} \\ k_2 = 2\zeta\omega_0 \end{cases} \tag{3.11}$$

The ratio between the control bandwidth of the two-cascaded loops can be calculated,

$$\frac{k_2}{k_1} = 4\zeta \tag{3.12}$$

We can see that the time scale separation of 4 times corresponds to critical relative damping of 1 for this linear example. Hence, for nonlinear systems, time scale separation between loops is normally required to be 3-5 times.

3.1.3. Non-cascaded NDI design with original model (Design c)

Comparing the above two designs, the cascaded control design has simpler dynamic inversion. However, the reference state derivative cannot be feedforward to the inner loop. Instead, estimation by the inner loop reference model is used.

It is intuitive to propose a new NDI design with original model and a nonlinear reference model, which is able to feed forward the reference commands to the inner loop and consequently eliminate the time scale separation. Due to the feedforward signal, it is a non-cascaded nonlinear control design, though the design structure is rather close to the cascaded NDI design. In [49], such a controller is designed for a missile attitude control system. However, the approach does not fully account for the nonlinearity in the 'outer loop dynamics'. A novel modification term is proposed here to render the tracking error dynamics to a linear one.

The 'outer loop' follows the same design as in Eq. (3.2). For the 'inner loop', [49] suggests to use the reference signal $\dot{x}_{2r}$ in the control law instead of the $\dot{\hat{x}}_{2,des}$ generated by the inner loop reference model. The overall control law as proposed in [49] is summarized,

$$\begin{cases} x_{2,des} = g_1^{-1}[-f_1 + \dot{x}_{1r} + k_1(x_{1r} - x_1)] \\ u = g_2^{-1}[-f_2 + \dot{x}_{2r} + k_2(x_{2,des} - x_2)] \end{cases} \tag{3.13}$$

The design structure is shown in Figure 3.3.

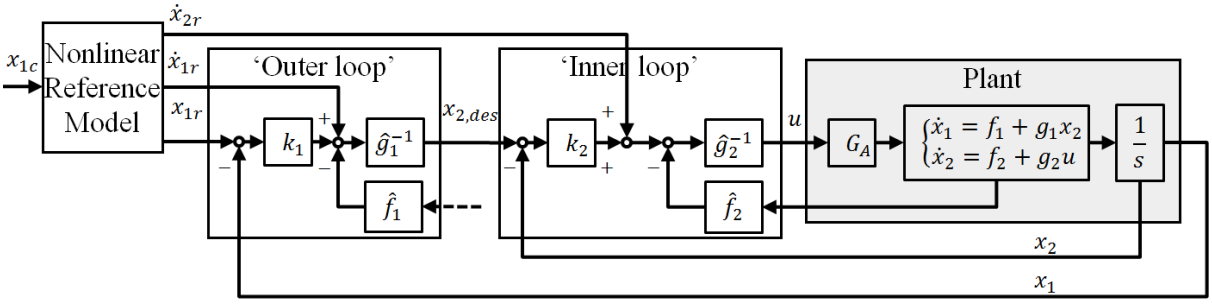


Figure 3.3 Non-cascaded NDI design from [49]

The tracking error dynamics of this control law can be derived by substituting the control law into the system differential equation,

$$\begin{aligned} \ddot{x}_1 = (f_1 + g_1 x_2)' &= \dot{f}_1 + \dot{g}_1 x_2 + g_1 \dot{x}_2 = \dot{f}_1 + \dot{g}_1 x_2 + g_1(f_2 + g_2 u) \\ &= \dot{f}_1 + \dot{g}_1 x_2 + g_1\left(f_2 + g_2 g_2^{-1}[-f_2 + \dot{x}_{2r} + k_2(x_{2,des} - x_2)]\right) \\ &= \dot{f}_1 + \dot{g}_1 x_2 + g_1 \dot{x}_{2r} + k_2 g_1(x_{2,des} - x_2) \end{aligned}$$

The reference signal $\dot{x}_{2r}$ can be calculated from the linear reference signal $\ddot{x}_{1r}$. The reference dynamics use the same dynamic model so the dynamic equation is,

$$\dot{x}_{1r} = f_1 + g_1 x_{2r} \tag{3.14}$$

Differentiating it,

$$\ddot{x}_{1r} = (f_1 + g_1 x_{2r})' = \dot{f}_1 + \dot{g}_1 x_{2r} + g_1 \dot{x}_{2r} \tag{3.15}$$

Rearranging the above equation,

$$\dot{x}_{2r} = g_1^{-1}(\ddot{x}_{1r} - \dot{f}_1 - \dot{g}_1 x_{2r}) \tag{3.16}$$

The $x_{2,des} - x_2$ term can be calculated as follows,

$$x_{2,des} - x_2 = g_1^{-1}[-f_1 + \dot{x}_{1r} + k_1(x_{1r} - x_1)] - g_1^{-1}[-f_1 + \dot{x}_1] = g_1^{-1}[\dot{x}_{1r} - \dot{x}_1 + k_1(x_{1r} - x_1)] \tag{3.17}$$

Substituting them into the system differential equation above,

$$\ddot{x}_1 = \ddot{x}_{1r} - \dot{g}_1(x_{2r} - x_2) + k_2(\dot{x}_{1r} - x_1) + k_2 k_1(x_{1r} - x_1) \tag{3.18}$$

The tracking error dynamic equation is,

$$\ddot{e} + k_2\dot{e} + k_2 k_1 e = \dot{g}_1(x_{2r} - x_2) \tag{3.19}$$

We can see that the proposed control law does not completely cancel the nonlinearities. The dynamics of the tracking error e could not be optimized only by the gains. If the function g_1 is a diffeomorphism, i.e. its inverse and derivative is solvable, further modification can be made to the control law to eliminate the extra term in the tracking error dynamics. The modified control law is,

$$\begin{cases} x_{2,des} = g_1^{-1}[-f_1 + \dot{x}_{1r} + k_1(x_{1r} - x_1)] \\ u = g_2^{-1}\left[-f_2 + \dot{x}_{2r} + k_2\left(x_{2,des} - x_2\right) + g_1^{-1}\dot{g}_1(x_{2r} - x_2)\right] \end{cases} \tag{3.20}$$

Optimal tracking error dynamics can be achieved with the above control law. The resulting tracking error dynamic equation is,

$$\ddot{e} + k_2\dot{e} + k_2 k_1 e = 0 \tag{3.21}$$

The final design structure for the second order system is shown in Figure 3.4.

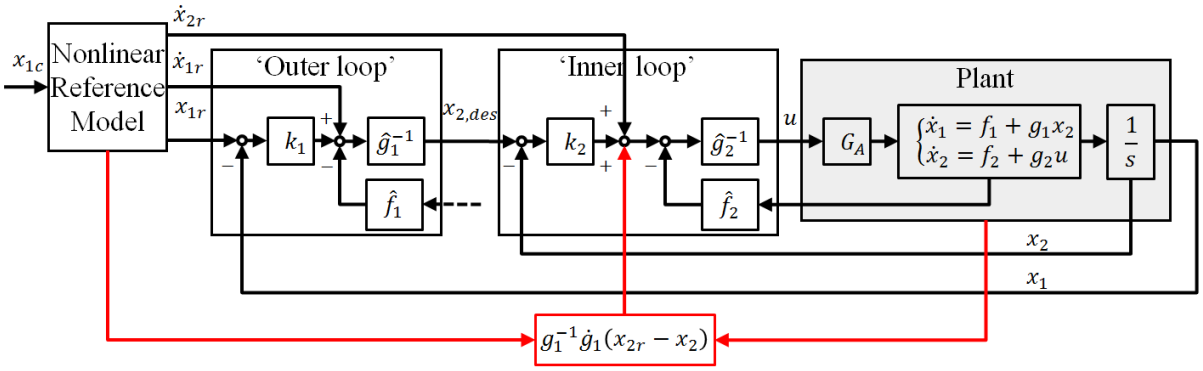


Figure 3.4 Non-cascaded NDI design with original model

We can see that by the novel term (red block in Figure 3.4) the tracking error dynamics is rendered to be the same as the classical case as in Design a), but the design is more intuitive and the inversion is less complicated compared with the classical NDI design a). For systems with relative large parameter uncertainties, such as sensor noise and external disturbances, the effect of the additional term becomes trivial.

3.2. Backstepping Designs

Backstepping designs, derived with the Lyapunov approach, have more design freedoms. For instance, the non-cascaded NDI design can be seen as a special case of Backstepping. The different variations of Backstepping control are derived here for comparison with each other and with the NDI designs.

3.2.1. Analytical Backstepping with original model (Design d)

This is the standard Backstepping approach, whose derivation procedures and results have been given in section 2.3. Recall the control law in Eqs. (2.64), (2.69) and (2.71),

$$\begin{cases} x_{2,des} = g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1) \\ u = g_2^{-1}(\dot{x}_{2,des} - f_2 + g_1 z_1 + k_2 z_2) \\ u = g_2^{-1}[-f_2 + \dot{x}_{2r} + k_2(x_{2,des} - x_2) + g_1^{-1} k_1 \dot{z}_1 - g_1^{-2} \dot{g}_1 k_1 z_1 + g_1 z_1] \end{cases} \tag{3.22}$$

In Figure 3.5, the control structure is shown. The difference to the design c) is shown in the red box.

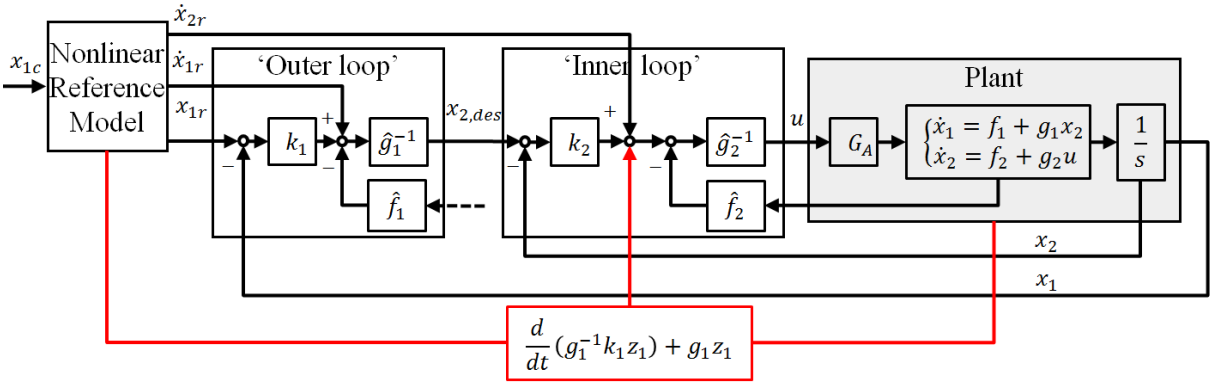


Figure 3.5 Analytical Backstepping with original model

The Backstepping error dynamics can be derived and summarized in matrix form in Eq. (3.23)

$$\begin{aligned} \dot{z}_1 &= \dot{x}_{1r} - \dot{x}_1 = \dot{x}_{1r} - f_1 - g_1 x_{2,des} + g_1 z_2 \\ &= \dot{x}_{1r} - f_1 - g_1 g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1) + g_1 z_2 = -k_1 z_1 + g_1 z_2 \\ \dot{z}_2 &= \dot{x}_{2,des} - \dot{x}_2 = \dot{x}_{2,des} - f_2 - g_2 u \\ &= \dot{x}_{2,des} - f_2 - g_2 g_2^{-1}(\dot{x}_{2,des} - f_2 + g_1 z_1 + k_2 z_2) = -g_1 z_1 - k_2 z_2 \end{aligned}$$

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} -k_1 & g_1 \\ -g_1 & -k_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{3.23}$$

Furthermore, the tracking error dynamics similar to NDI design can be derived, the matrix form is with

$$\begin{aligned} e &= z_1 = x_{1r} - x_1 \\ \dot{e} &= \dot{z}_1 = -k_1 z_1 + g_1 z_2 = -k_1 e + g_1 z_2 \end{aligned}$$

$$\ddot{e} = \ddot{z}_1 = -k_1\dot{e} + \dot{g}_1 z_2 + g_1\dot{z}_2 = -k_1\dot{e} + \dot{g}_1 \underbrace{z_2}_{g_1^{-1}(\dot{e}+k_1 e)} + g_1 \underbrace{\dot{z}_2}_{-g_1 e - k_2 z_2}$$

$$= -k_1\dot{e} + \dot{g}_1 g_1^{-1}(\dot{e} + k_1 e) + g_1\left(-g_1 e - k_2 g_1^{-1}(\dot{e} + k_1 e)\right)$$

$$= -k_1\dot{e} + \dot{g}_1 g_1^{-1}\dot{e} + \dot{g}_1 g_1^{-1} k_1 e - g_1^2 e - k_2\dot{e} - k_2 k_1 e$$

$$= (-k_1 - k_2 + \dot{g}_1 g_1^{-1})\dot{e} + (-k_2 k_1 + \dot{g}_1 g_1^{-1} k_1 - g_1^2)e$$

$$\begin{bmatrix}\dot{e}\\ \ddot{e}\end{bmatrix} = \begin{bmatrix}0 & 1\\ -k_2 k_1 + \dot{g}_1 g_1^{-1} k_1 - g_1^2 & -k_1 - k_2 + \dot{g}_1 g_1^{-1}\end{bmatrix}\begin{bmatrix}e\\ \dot{e}\end{bmatrix} \tag{3.24}$$

The tracking error dynamics is clearly nonlinear as the nonlinear function g_1 is state dependent. The Lyapunov theory only guarantees the stability but nothing regards the performance. Besides the gain design, the tracking performance depends more on the magnitude and rate of change of the function g_1. This is further validated in the later sections.

3.2.2. Analytical Backstepping with transformed model (Design e)

The tracking error dynamics can be rendered linear if the function g_1 is a constant or even unity. The example system is transformed into the companion form as in Eq. (2.21),

$$\begin{cases}\dot{x}_1 = x_3\\ \dot{x}_3 = f_3 + g_3 \cdot u\end{cases}$$

Applying the standard Backstepping procedure (refer to section 2.3.1 for the more detailed procedure):

Step 1, $x_{3,des} = \dot{x}_{1r} + k_1 z_1$

Step 2, $V_3 = \frac{1}{2}z_1^2 + \frac{1}{2}z_3^2$, where $z_3 = x_{3,des} - x_3$

$$u = g_3^{-1}\left(\dot{x}_{3,des} - f_3 + z_1 + k_3 z_3\right)$$

$$\dot{x}_{3,des} = \ddot{x}_{1r} + k_1\dot{z}_1$$

$$\dot{V}_3 = -k_1 z_1^2 - k_2 z_3^2$$

Backstepping error dynamics:

$$\begin{bmatrix}\dot{z}_1\\ \dot{z}_2\end{bmatrix} = \begin{bmatrix}-k_1 & 1\\ -1 & -k_2\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} \tag{3.25}$$

Tracking error dynamics

$$\begin{bmatrix}\dot{e}\\ \ddot{e}\end{bmatrix} = \begin{bmatrix}0 & 1\\ -k_2 k_1 - 1 & -k_1 - k_2\end{bmatrix}\begin{bmatrix}e\\ \dot{e}\end{bmatrix} \tag{3.26}$$

The tracking error dynamics can be exactly the same as the NDI non-cascaded design by assigning proper feedback gains.

$$\begin{cases}k_2 k_1 + 1 = k_p = \omega_0^2\\ k_1 + k_2 = k_d = 2\zeta\omega_0\end{cases} \tag{3.27}$$

The control law is also the same as in Eq. (2.33)

$$\begin{cases} u = g_3^{-1}[v - f_3] \\ v = v_r + 2\zeta\omega_0\dot{e} + \omega_0^2 e \end{cases} \tag{3.28}$$

There are two conditions that make the same control law for Backstepping and NDI,

1. The system is represented in companion form, so that all the nonlinearities can be cancelled out in the most inner loop and both methods transform the system into a chain of integrators
2. The same linear controller is used in the outer lop to provide the desired convergence towards to a reference state.

3.2.3. Cascaded Backstepping design (Design f)

Cascaded Backstepping design of the second order example system is the same as the cascaded NDI design. It is listed here only for the sake of completeness.

However, for more complicated systems, especially with system order bigger than 3, the division of loops needs additional design considerations. This is further illustrated in the position control designs, which have a system order of 4.

3.2.4. Backstepping with command filter (Design g)

For systems not in the strict-feedback form, it could be difficult or even not possible to apply the analytical methods to solve the virtual control derivative $\dot{x}_{2,des}$. Recall the derived Backstepping control law from section 2.3.1,

$$\begin{cases} x_{2,des} = g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1) \\ u = g_2^{-1}(\dot{x}_{2,des} - f_2 + g_1 z_1 + k_2 z_2) \end{cases}$$

As mentioned earlier, it is quite common to implement a low pass filter to estimate $\dot{x}_{2,des}$ as in Eq. (2.72)

$$\dot{\hat{x}}_{2,des} = k_f(x_{2,des} - \hat{x}_{2,des})$$

The final control law is,

$$\begin{cases} x_{2,des} = g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1) \\ u = g_2^{-1}(\dot{\hat{x}}_{2,des} - f_2 + g_1 z_1 + k_2 \hat{z}_2) \\ \hat{z}_2 = \hat{x}_{2,des} - x_2 \end{cases} \tag{3.29}$$

The derived Backstepping control law is very similar to the cascaded NDI control law. The command filter could have the same filter structure as the inner loop reference model in the NDI design. In this case, both are just a first order low pass filter. The only difference is one additional term $g_1 z_1$ in the Backstepping design. The control structure is shown below,

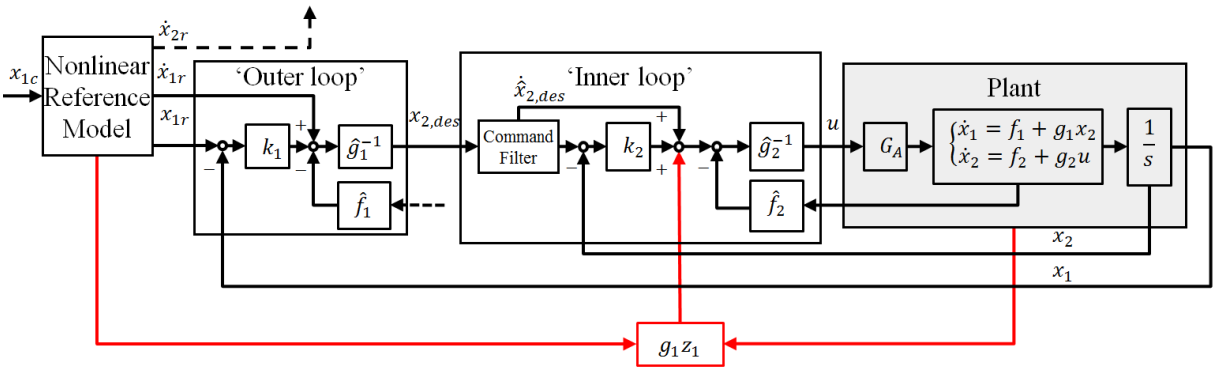


Figure 3.6 Backstepping with command filter

Due to the filter estimation, the Lyapunov stability is violated. Since the original Lyapunov function is not differentiable, a new Lyapunov function is defined,

$$\hat{V}_2 = \frac{1}{2}z_1^2 + \frac{1}{2}\hat{z}_2^2 \tag{3.30}$$

Its derivative can be calculated,

$$\dot{\hat{V}}_2 = z_1\dot{z}_1 + \hat{z}_2\dot{\hat{z}}_2 = -k_1 z_1^2 - k_2\hat{z}_2^2 + z_1 g_1(z_2 - \hat{z}_2) = -k_1 z_1^2 - k_2\hat{z}_2^2 + z_1 g_1\left(x_{2,des} - \hat{x}_{2,des}\right)$$

The estimation error $x_{2,des} - \hat{x}_{2,des}$ violates the negative semi-definite property of the Lyapunov function derivative.

3.2.5. Command Filter Backstepping (Design h)

Farrell [21] introduced the CFB aiming to compensate the unachieved portion of $x_{2,des}$ due to the estimation error. This section presents the derivation of the compensation term from the Lyapunov stability theory, though the original work was presented from a different point of view.

Considering the violated Lyapunov function, the $\hat{z}_2$ is an estimated term, which is difficult to be modified, but the z_1 has certain room for augmentation. A modification term ξ_1, without knowing its formulation, can be introduced to the z_1 term in both the Lyapunov function and the control law,

$$V_{2,CFB} = \frac{1}{2}\boxed{(z_1 + \xi_1)}^2 + \frac{1}{2}\hat{z}_2^2 \tag{3.31}$$

$$\begin{cases} x_{2,des} = g_1^{-1}(\dot{x}_{1r} - f_1 + k_1 z_1) \\ u = g_2^{-1}\left(\dot{\hat{x}}_{2,des} - f_2 + g_1\boxed{(z_1 + \xi_1)} + k_2\hat{z}_2\right) \\ \hat{z}_2 = \hat{x}_{2,des} - x_2 \end{cases}$$

The Lyapunov function derivative is,

$$\begin{aligned} \dot{V}_{2,CFB} &= (z_1 + \xi_1)\left(\dot{z}_1 + \dot{\xi}_1\right) + \hat{z}_2\dot{\hat{z}}_2 \\ &= (z_1 + \xi_1)\left(-k_1 z_1 + g_1 z_2 + \dot{\xi}_1\right) + \hat{z}_2\left(-k_2\hat{z}_2 - g_1(z_1 + \xi_1)\right) \\ &= (z_1 + \xi_1)\left(-k_1 z_1 - k_1\xi_1 + k_1\xi_1 + g_1 z_2 + \dot{\xi}_1\right) + \hat{z}_2\left(-k_2\hat{z}_2 - g_1(z_1 + \xi_1)\right) \end{aligned}$$

$$= (z_1 + \xi_1)\left(-k_1(z_1 + \xi_1) + k_1\xi_1 + g_1 z_2 + \dot{\xi}_1\right) + \hat{z}_2\left(-k_2\hat{z}_2 - g_1(z_1 + \xi_1)\right)$$

$$= -k_1(z_1 + \xi_1)^2 - k_2\hat{z}_2^2 + (z_1 + \xi_1)\left(k_1\xi_1 + g_1 z_2 + \dot{\xi}_1\right) - \hat{z}_2 g_1(z_1 + \xi_1)$$

The optimal result would be to render the Lyapunov function derivative into the following negative semi-definite from,

$$\dot{V}_{2,CFB} = -k_1(z_1 + \xi_1)^2 - k_2\hat{z}_2^2 \tag{3.32}$$

In order to make the Lyapunov function derivative that form, the rest of the equation needs to be zero,

$$(z_1 + \xi_1)\left(k_1\xi_1 + g_1 z_2 + \dot{\xi}_1\right) - \hat{z}_2 g_1(z_1 + \xi_1) = 0 \tag{3.33}$$

The following modification control law can be derived for the ξ_1 to make the derivative negative semi-definite,

$$\dot{\xi}_1 = -k_1\xi_1 + g_1(\hat{z}_2 - z_2) = -k_1\xi_1 + g_1\left(\hat{x}_{2,des} - x_{2,des}\right) \tag{3.34}$$

In Laplace domain, the modification term ξ_1 is the filtered estimation error.

$$\xi_1 = \frac{g_1}{s + k_1}\left(\hat{x}_{2,des} - x_{2,des}\right) \tag{3.35}$$

With this feedback, the Lyapunov function derivative becomes,

$$\dot{V}_{2,CFB} = -k_1(z_1 + \xi_1)^2 - k_2\hat{z}_2^2 \tag{3.36}$$

The control structure is also drawn in similar fashion as before for comparison. As shown below, the modification in the control law is quite small,

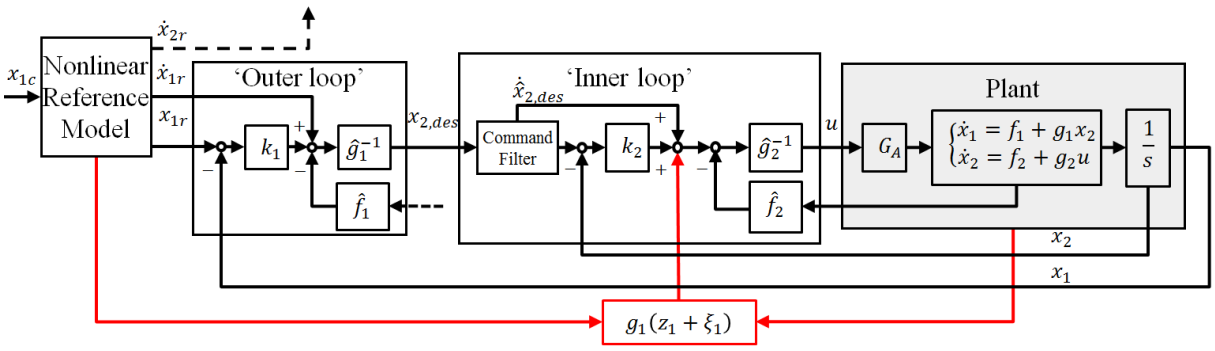


Figure 3.7 Command Filtered Backstepping

Though the Lyapunov stability is satisfied, it should be noticed that the Lyapunov function is not exactly the tracking error (z_1) cost function. Hence, the performance is expected to be compromised.

The modification law for higher order system can be derived similarly. Detailed derivations are given in [21],

$$\dot{\xi}_i = -k_i\xi_i + g_i\left(\hat{x}_{i+1,des} - x_{i+1,des}\right) + g_i\xi_{i+1} \tag{3.37}$$

3.3. Comparison and Simulation Results

3.3.1. Analysis of the different designs

The above eight control designs can be categorized into two groups; one group does not use any low pass filter and the other group does use low pass filters to estimate command derivatives, as there is a distinct performance difference between the two groups. The differences between designs within the same group are comparatively small for the generic example. There are more model dependent differences to be shown in the next chapter.

Control designs a), c), d) and e) belong to the first group. All four designs has non-cascaded structure and the full reference information can be used in the feedforward control, i.e. no filtering is necessary. Design a) and c) are derived from NDI, while d) and e) are derived from Backstepping. Design a) and design e) have exactly the same control laws, which completely invert the plant dynamic in the inner loop so that linear error dynamics can be obtained. Ignoring model uncertainties, they could give the best possible tracking performance with appropriate linear control design. Their control law for the example generic system is,

$$\begin{cases} u = g_3^{-1}[v - f_3] \\ v = v_r + 2\zeta\omega_0\dot{e} + \omega_0^2 e \end{cases}$$

However, not every system could be formulated as system representation 2 in Eq.(2.21), so that control design a) and e) can't be applied. The designs c) and d) could be applied to the original formulation as in Eq. (2.19). The state space representation has nonlinearities in all equations. There plant dynamics has to be inverted separately at different system order. As a result, their tracking error is no longer linear and its performance may not be as good as the clean designs a) and e). In design c) with the additional novel terms, its tracking error dynamics can be rendered to be linear, the same as in design a) and e). All three designs a) c) and e) could has the ideal linear error dynamics,

$$\begin{bmatrix} \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix} \begin{bmatrix} e \\ \dot{e} \end{bmatrix}$$

While in design d), though it has Lyapunov stability proof by Backstepping, the tracking error dynamics is nonlinear and its performance is not deterministic because it depends on certain nonlinear terms in the math formulations. One important point is that the Backstepping method only gives Lyapunov stability guarantee but nothing about the tracking performance. The tracking error dynamics of design d) can be evaluated as in Eq. (3.24),

$$\begin{bmatrix} \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k_2k_1 + \dot{g}_1 g_1^{-1} k_1 e - g_1^2 & -k_1 - k_2 + \dot{g}_1 g_1^{-1} \end{bmatrix} \begin{bmatrix} e \\ \dot{e} \end{bmatrix} \tag{3.38}$$

In many cases, the g_1 related terms are comparatively smaller than the designed gains so its impact is hidden, for example, the linear example demonstrated in this section. It will be shown in the next chapters with the quadrotor example, when the g_1 related terms are comparatively bigger, they can deteriorate the tracking performance.

In the second group, design b), f), g) and h) all used filter to estimate the intermediate control derivatives, which can avoid the tedious computation of the intermediate control derivative especially for complex nonlinear high order systems. The big advantage is that they can be

applied to system of non-triangular form. However, due to the filtered estimation, the reference signal ($\dot{x}_{2r}$ or $\ddot{x}_{1r}$) cannot be used in the feedforward control, which is the main reason causing the performance differences in this linear example.

The control laws from the four designs are listed below,

1. The first part or the 'outer loop' is the same for all four cases. The same filter can be used because its input, output and purpose are the same.

$$x_{2,des} = g_1^{-1}[-f_1 + \dot{x}_{1r} + k_1(x_{1r} - x_1)]$$
$$\dot{\hat{x}}_{2,des} = k_f\left(x_{2,des} - \hat{x}_{2,des}\right)$$

2. The control laws are given below to show the differences,

b)&f) $$u = g_2^{-1}\left[-f_2 + \dot{\hat{x}}_{2,des} + k_2\left(\hat{x}_{2,des} - x_2\right)\right]$$

g) $$u = g_2^{-1}\left[-f_2 + \dot{\hat{x}}_{2,des} + k_2\left(\hat{x}_{2,des} - x_2\right) + g_1(x_{1r} - x_1)\right]$$

h) $$u = g_2^{-1}\left[-f_2 + \dot{\hat{x}}_{2,des} + k_2\left(\hat{x}_{2,des} - x_2\right) + g_1(x_{1r} - x_1 + \xi_1)\right]$$

Design b) and f) are the same. Both are the classical cascaded structure. Enough time scale separation between the control loops is needed for stability, but the control bandwidth is compromised. Design g) has an additional term $g_1(x_{1r} - x_1)$ feedforward to inner loop, which some [50] claimed to work against the time scale separation. However, this feedforward term, at least for this linear example, does not show any effect. It is an error term without any gains, so for systems requiring high gain controller (i.e. high k_2 gain) like quadrotors, this term is rather small comparatively and it doesn't have any impact even in the ideal simulation environment. So its effect in the real system with model uncertainties and sensor errors is questionable. As design g) violates the Lyapunov stability proof by using the filter, design h) introduce an additional term and prove the Lyapunov stability. However, still this additional term is rather small comparatively in the control law. No improvement can be shown in the simulation and in real system (see Chapter 5). The low pass filters used in those designs are the main reason for the performance reduction. None of the modifications gives noticeable improvement in the tracking performance.

The tracking performance of the second group can be rendered arbitrarily close to the first group by increasing the low pass filter bandwidth. However, in real life application the filter bandwidth is often limited by the sensor noise.

3.3.2. Simulation Results

In this preliminary simulation example, a linear system is chosen to compare the different designs. To better observe the fundamental difference, instead of more complex systems, the simplest linear second order system is most representative. No system uncertainties and sensor errors, but only actuator dynamics is considered in the simulation enjoinment in Figure 3.8. The same reference signals are used for all control designs. For the second order system, a second order low pass filter is used to generate smooth reference commands with a relative damping of 1 and natural frequency of 10. The reference signal of a step command is shown in Figure 2.4. The tracking performance in terms of overshoot and rising time is compared.

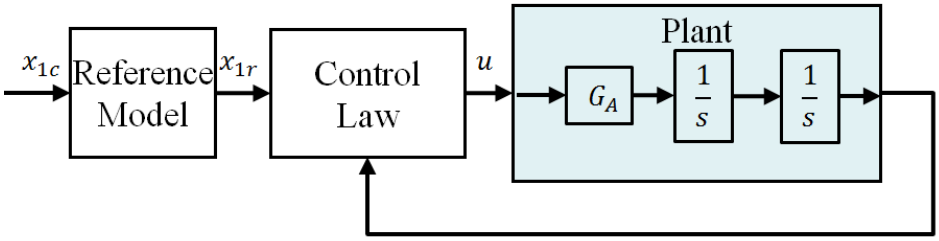


Figure 3.8 Simulation Environment

The system equation of motion is very simple ($g_1 = 1$),

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u \end{cases}$$

The actuator dynamics is modeled as a first order low pass filter with a time constant of 0.04 s, the Laplace transfer function is,

$$G_A(s) = \frac{25}{s + 25}$$

The control law structures have been designed but the gains have to designed, especially in the Backstepping designs. Linear gain design methods like pole placement can be applied to the NDI designs (both cascaded and non-cascaded case). However, as the Backstepping designs (except design e)) often generate non-linear error dynamics, linear methods can't be applied. For the Backstepping design g) and h), different time-scale-gains are used: set I gives TSS between the two states feedbacks, so the gains are same as in those the cascaded design b) & f); set II assumes that the additional Backstepping term removes the TSS, so the gains are the same as those in Design d) & e). By comparing the performance of the two gain sets, the question on whether the additional Backstepping term eliminates the TSS could be answered. The gains are listed in Table 3.1.

Design a)	$k_d = 2\zeta\omega = 20;\ k_p = \omega^2 = 100$
Design c)	$k_1 = 10;\ k_2 = 10;$
Design d) & e)	$k_1 = 9;\ k_2 = 11;$
Design b) & f)	$k_1 = 5;\ k_2 = 20;\ k_f = 20$
Design g)-h) – I	$k_1 = 5;\ k_2 = 20;\ k_f = 20$
Design g)-h) – II	$k_1 = 9;\ k_2 = 11;\ k_f = 20$

Table 3.1 Gain Designs for the linear example

The low pass filter gain for the control designs of the second group is set to be 20 here, though it can be further increased, as there is no sense noise and the actuator model are over-simplified. However, it does not affect the conclusion made. With the above gain designs for the linear example, design a), c), d) and e) could result the same control law. Their differences will be further analyzed in later section with the nonlinear quadrotor example. Design b) and f) are cascaded designs and they have the same control law in any case.

The tracking performance of a step command is shown in Figure 3.9. Besides the reference signal of the step commands (in black), there are three different performing groups, blue, cyan and yellow. The blue signal comes from the non-cascaded designs a), c), d) and e), which give the best performance. Signals from designs b), f), g)-I, and h)-I coincide in the cyan line. They belong to the cascaded designs. The design g) and h) give rather the same performance as the cascaded control designs. The additional Backstepping terms in their control law are negligible compared to the magnitude of the control input u, as shown in Figure 3.11. As a group their performance is not as good as the blue signal, but better than the yellow line, which is given by designs g)-II and h)-II with different gains. These two designs use the non-cascaded design gains and it could have performed better if the TSS was eliminated by the additional Backstepping term. However, the reality is the additional term is too small to have any impact. With the aggressive non-cascaded gain design, the performance gets worse and bigger oscillation and overshoot can be observed. Figure 3.10 shows clearly the differences using error signals.

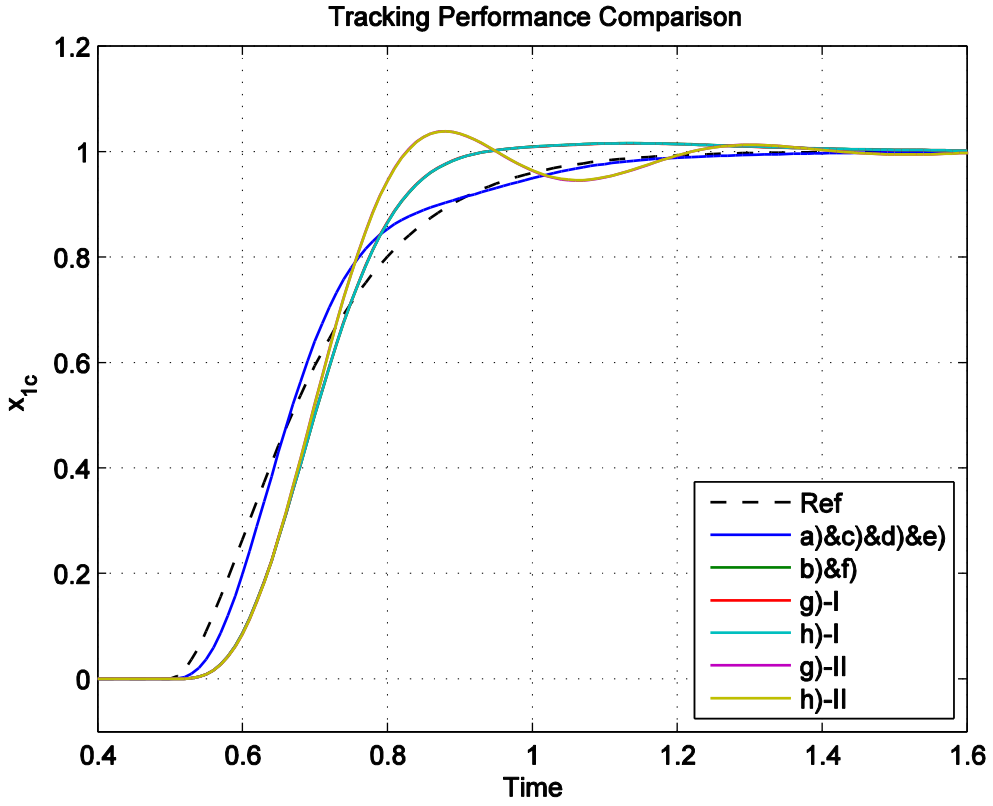


Figure 3.9 Tracking performance of a step command

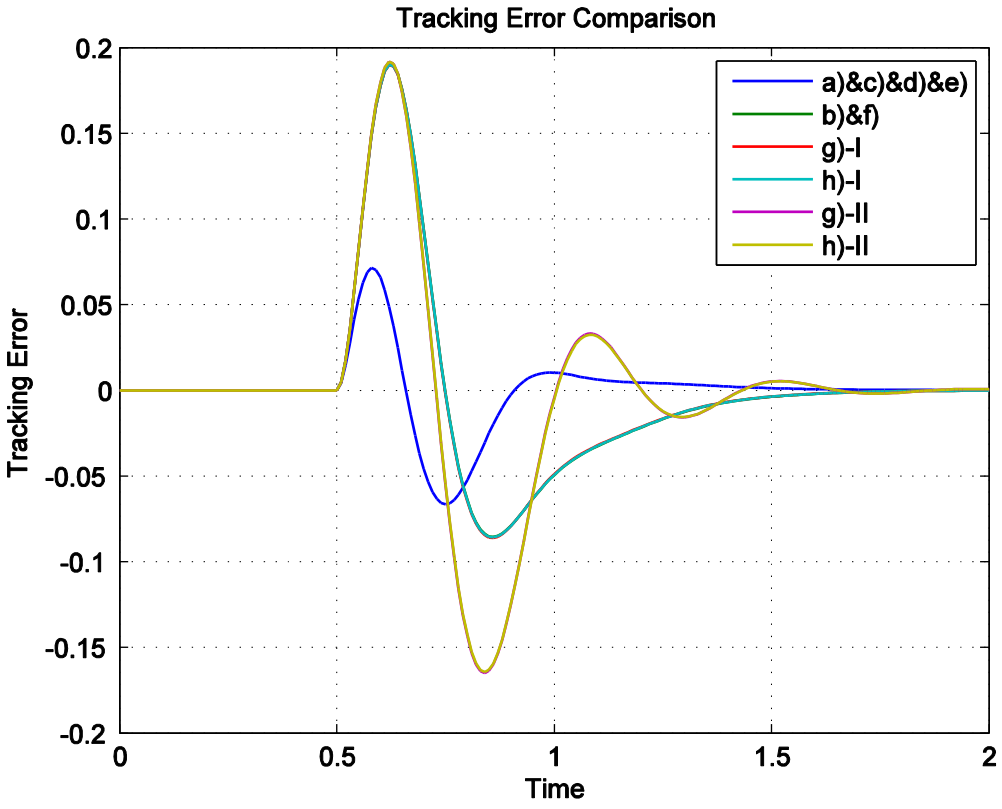


Figure 3.10 Tracking errors of different designs

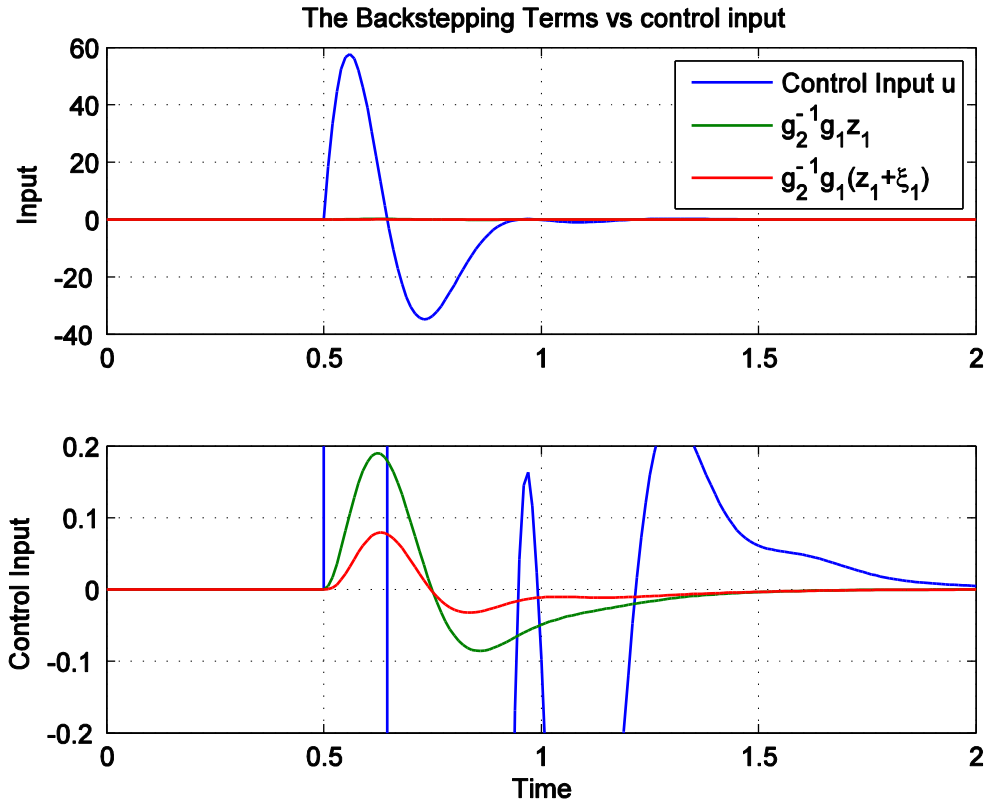


Figure 3.11 The Backstepping Terms in Design g) and h)

3.4. Conclusion

From the above analysis, the non-cascaded designs a), c), d), and e) have obviously better performance over the other designs utilizing a low pass filter. However, the other four designs have more design freedom and could be applied to systems of non-triangular form. Among them, compared to the cascaded structure in design b) and f), the Backstepping approaches g) and h) have additional terms in the control law but still TSS has to be considered in the gain design for better performance. The low pass filter used in the control designs is the reason for the TSS, no matter if it is a NDI or Backstepping design. In this sense, the Backstepping designs utilizing filters should be considered as cascaded control, as their performance is not any better than the cascaded designs.

From the above analysis and simulation, preliminary conclusion can be drawn: though Backstepping method shows more design freedom (even the same control law as NDI can be designed), NDI method gives most simplicity and equivalent best performance as Backstepping. In addition, as NDI is rooted from the physics of the system, the gain design can be more intuitive than the Lyapunov based Backstepping.

To validate the conclusion, the comparisons between NDI and Backstepping are continued with real world example, the quadrotor attitude control and position control systems. The following issues are still open to be addressed,

- The difference between design a), c) and d) can be explored on a nonlinear system: the quadrotor attitude system.
- The filtered designs, b), g) and h) can be further compared on a non-triangular system: the quadrotor position system.

However, design e) and design f) will not be considered as they are analytical the same as the design a) and design b) respectively.

4. Mathematical Models for the Quadrotor

The mathematical formulations are the fundamentals for model-based designs. Deep understanding of the system dynamics is essential for the control design, data fusion and system integrations. Careful choice of math model could simplify the design, reduce implementation effort, and improve the overall performance.

The mathematical models for the simulation, data fusion and control design are all based on state-space model, but the level of fidelity and focuses can be very different. In this chapter, the simulation model is briefly introduced; the models for data fusion are introduced on implementation basis. The models for control designs are analyzed in details and those are the basis for the control designs in later chapters.

4.1. Model for Simulation

The math model for simulation aims to virtually represent the real system in mathematical formulations. A high fidelity simulation model typically consists of the following modules:

- Environment model: includes the atmosphere model, the earth geometry, the earth gravity and magnetic field
- Motion kinematics: computes the kinematic transformations between all relevant frames for positions, velocities, accelerations, angular rates, and angular accelerations.
- Airframe: represents the airframe thus varies from different air vehicles. It is generally divided into two sub–categories, passive systems and active system. The passive system may include avionics, electric system, hydraulic system and control surface actuators; and the active system may include landing gear, control surface deflection, fuel system, propulsion, fluid dynamics, weight and balance.
- Equation of motion: includes the translational and rotational rigid body equation of motion based on the linear and angular momentum equation according to Newton's second law.
- Sensor model: gives the state measurements with realistic sensor errors, like update rate, bias and noises.

For the quadrotor case, the environment consists of an ellipsoidal earth and gravity model conforming to WGS84, the world magnetic model and a static and dynamic atmosphere model. Parts of the airframe subsystem are the aerodynamics of the quadrotor's base unit, gravity force and the propulsion model. As the latter is the crucial part of a multirotor simulation, it comprises a detailed model of the propeller's aerodynamic properties and the electrical drive that consists of a BLDC motor, and a power output stage with RPM control. The electrical system is completed by a lithium-ion battery model. Furthermore, this subsystem contains the ground contact mechanics that are modeled as point contacts on a

2D half-plane with regularized friction laws. This allows conducting realistic takeoff and landing maneuvers in the test procedure.

The equations of motion for the rigid body are formulated for a flat non-rotating earth and w.r.t. the aircraft's reference point (R) which is not equal to the center of gravity. This leads to state propagation equations for rotation, translation, attitude and position. The orientation is described with a quaternion in order to benefit from a singularity free and computationally efficient differential equation. Position is propagated in a local navigation/world frame. The velocity state is denoted in the body fixed coordinate system, which leads to a coupled differential equation for rotation and translation.

$$\begin{bmatrix} m \cdot \mathbf{I}_3 & -m \cdot [(\vec{\mathbf{r}}^{RG}) \times] \\ m \cdot [(\vec{\mathbf{r}}^{RG}) \times] & (\mathbf{I}^R)_{BB} \end{bmatrix} \cdot \begin{bmatrix} \left(\dot{\vec{\mathbf{v}}}^R\right)^{IB} \\ \left(\dot{\vec{\boldsymbol{\omega}}}^{IB}\right)^B \end{bmatrix} = \begin{bmatrix} \sum \vec{\mathbf{F}} - m \cdot \{(\vec{\boldsymbol{\omega}}^{IB}) \times (\vec{\mathbf{v}}^R)^I\} - m \cdot \{(\vec{\boldsymbol{\omega}}^{IB}) \times [(\vec{\boldsymbol{\omega}}^{IB}) \times (\vec{\mathbf{r}}^{RG})]\} \\ \sum \vec{\mathbf{M}} - (\vec{\boldsymbol{\omega}}^{IB}) \times \mathbf{I}^R \cdot (\vec{\boldsymbol{\omega}}^{IB}) - m \cdot (\vec{\mathbf{r}}^{RG}) \times [(\vec{\boldsymbol{\omega}}^{IB}) \times (\vec{\mathbf{v}}^R)^I] \end{bmatrix} \tag{4.1}$$

In order to calculate the state derivatives, the given linear equation system has to be solved in each time step.

In addition to the described physical system, there are models for all sensors and their signal processing in the flight control unit. This includes gyros, accelerometer, magnetometer, static pressure sensor and the optical tracking system. Each sensor model includes various effects like misalignment, scale-factor errors, nonlinearities, temperature dependent bias, limited bandwidth, saturation and noise as well as errors due to quantization and digital sampling.

The described simulation model is implemented in Simulink and integrated in a testbed that allows closed loop testing in real-time. A virtual environment is also built using Simulink 3D animation toolbox as shown Figure 4.1.

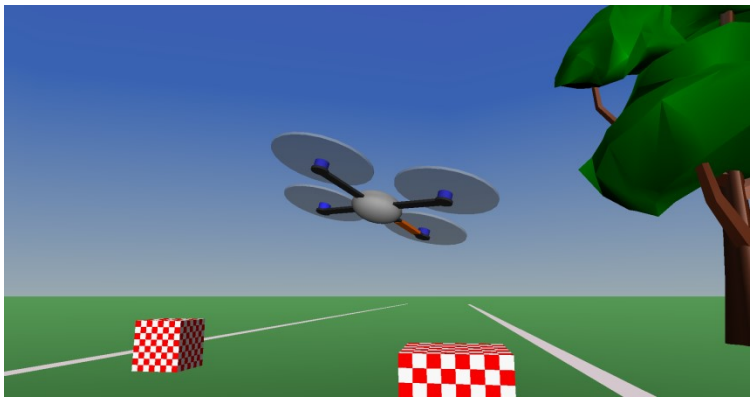


Figure 4.1 Snapshot of the Simulation Environment.

4.2. Models for Data Fusion

The mathematical models for data fusion need to be observable from the sensor outputs to the internal states. The models for data fusion focus on the states to be estimated. Three basic models for data fusion are introduced in this section. They have been implemented and tested on the quadrotor platform. The math models are simple kinematic models and passively depend on sensor measurements.

4.2.1. AHRS system

An Attitude and Heading Reference System (AHRS) provides attitude estimation by integrating gyroscopes and fusing this data with accelerometer data and heading measurement normally from magnetometer. In this case, for the indoor experimental platform, the heading is measured by the vision sensor. Hence, the World frame is used instead of the NED frame.

Typically, the input vector for the AHRS model is the three-axis angular rates measured by the gyroscopes. The output vector for the AHRS model is the gravity vector measured by the accelerometer. The state vector for the AHRS model is the attitude vector, represented by either Euler angle or Quaternions. It is also possible to estimate the gyro bias and scale factor error. However, they are very trivial in the particular case because in the start-up the gyro bias will be calibrated by resetting gyro value to zero and the bias walk during the quadrotor flight phase is negligible. Hence, only the attitude is considered as the state vector.

4.2.1.1. System propagation

The AHRS system is implemented using the EKF structure. As the Euler angle representation of the attitude has singularity problem and more computational intensive, Quaternion representation [51] is used. A unit quaternion has four elements, the last three elements are the 'vector part' of the quaternion and can be thought of as a vector about which rotation should be performed. The first element is the 'scalar part' that specifies the amount of rotation that should be performed about the vector part. Specifically, if θ is the angle of rotation and the vector $[i_x \quad i_y \quad i_z]^T$is a unit vector representing the axis of rotation, then the quaternion elements can be defined as,

$$\bar{\mathbf{q}} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos\frac{1}{2}\theta \\ i_x \sin\frac{1}{2}\theta \\ i_y \sin\frac{1}{2}\theta \\ i_z \sin\frac{1}{2}\theta \end{bmatrix} \tag{4.2}$$

With the quaternion representation, the attitude propagation equation can be derived as Eq. (4.3),

$$\begin{bmatrix}\dot{q}_0\\ \dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{bmatrix}=\begin{bmatrix}-q_1 & -q_2 & -q_3\\ q_0 & -q_3 & q_2\\ q_3 & q_0 & -q_1\\ -q_2 & q_1 & q_0\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{4.3}$$

The system is over-determined as there are four variables but only three are required. Hence additional constraint, the square-sum is one, needs to be enforced. To correct the constraint violations during the propagation, an often used method is “gradient feedback”. This method uses feedback to control the square-sum λ to zero, which is defined as,

$$\lambda = 1-(q_0^2+q_1^2+q_2^2+q_3^2) \tag{4.4}$$

And a constant feedback gain k may be chosen such that $k\cdot Ts<1$ for a fixed integration step size Ts. Then the implemented propagation equation is,

$$\begin{bmatrix}\dot{q}_0\\ \dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{bmatrix}=\begin{bmatrix}-q_1 & -q_2 & -q_3 & q_0\\ q_0 & -q_3 & q_2 & q_1\\ q_3 & q_0 & -q_1 & q_2\\ -q_2 & q_1 & q_0 & q_3\end{bmatrix}\begin{bmatrix}p\\ q\\ r\\ 2k\lambda\end{bmatrix} \tag{4.5}$$

4.2.1.2. System Error State Model

As described in previous section, the EKF consists of a propagation part that propagates the gyro data to nominal states and an error state Kalman filter to fuse the measurement data. Hence, the output equation is an error output equation. In error state, with small angle approximation, the conversion between the Quaternions and Euler angles is,

$$\delta\bar{\mathbf{q}}\approx\begin{bmatrix}1\\ \delta\vec{\boldsymbol{\Psi}}/2\end{bmatrix} \tag{4.6}$$

Where $\delta\vec{\boldsymbol{\Psi}}$ is the attitude of the estimated World frame $\widetilde{W}$, with respect to true World frame,

$$\delta\vec{\boldsymbol{\Psi}}=\begin{bmatrix}\delta\Phi\\ \delta\Theta\\ \delta\Psi\end{bmatrix} \tag{4.7}$$

The rotation of the attitude error vector can be approximated by its skew-symmetric matrix, with small angle assumption, i.e. $\sin\delta\Psi=\delta\Psi, \cos\delta\Psi=1$.

$$\mathbf{M}_{W\widetilde{W}}=\begin{bmatrix}\cos\delta\Theta\cos\delta\Psi & \sin\delta\Phi\sin\delta\Theta\cos\delta\Psi-\cos\delta\Phi\sin\delta\Psi & \cos\delta\Phi\sin\delta\Theta\cos\delta\Psi+\sin\delta\Phi\sin\delta\Psi\\ \cos\delta\Theta\sin\delta\Psi & \sin\delta\Phi\sin\delta\Theta\sin\delta\Psi+\cos\delta\Phi\cos\delta\Psi & \cos\delta\Phi\sin\delta\Theta\sin\delta\Psi-\sin\delta\Phi\cos\delta\Psi\\ -\sin\delta\Theta & \sin\delta\Phi\cos\delta\Theta & \cos\delta\Phi\cos\delta\Theta\end{bmatrix}$$

$$\approx\begin{bmatrix}1 & -\delta\Psi & \delta\Theta\\ \delta\Psi & 1 & -\delta\Phi\\ -\delta\Theta & \delta\Phi & 1\end{bmatrix}=\mathbf{I}+\delta\boldsymbol{\Psi}$$

Where $\delta\boldsymbol{\Psi}$ is a skew-symmetric matrix of $\delta\vec{\boldsymbol{\Psi}}$, defined as,

$$\delta\boldsymbol{\Psi}=\begin{bmatrix}0 & -\delta\Psi & \delta\Theta\\ \delta\Psi & 0 & -\delta\Phi\\ -\delta\Theta & \delta\Phi & 0\end{bmatrix} \tag{4.8}$$

With the approximation, the error propagation equation can be derived. The Euler differentiation rule gives,

$$\dot{\mathbf{M}}_{WB} = \mathbf{M}_{WB}\boldsymbol{\Omega}_{BB}^{WB} \tag{4.9}$$

While the $\boldsymbol{\Omega}_{BB}^{WB}$ is the skew-symmetric matrix of the angular rate,

$$\boldsymbol{\Omega}_{BB}^{WB} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \tag{4.10}$$

And $\mathbf{M}_{WB}$ can be slipped to two parts,

$$\mathbf{M}_{WB} = \mathbf{M}_{W\widetilde{W}}\mathbf{M}_{\widetilde{W}B} = (\mathbf{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\widetilde{W}B} \tag{4.11}$$

Eq. (4.9) can be linearized to derive the error state equation,

$$(\mathbf{I} + \delta\boldsymbol{\Psi})\dot{\mathbf{M}}_{\widetilde{W}B} + \delta\dot{\boldsymbol{\Psi}}\mathbf{M}_{\widetilde{W}B} = (\mathbf{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\widetilde{W}B}\left(\boldsymbol{\Omega}_{BB}^{\widetilde{W}B} + \delta\boldsymbol{\Omega}_{BB}^{WB}\right)$$

While, $(\mathbf{I} + \delta\boldsymbol{\Psi})\dot{\mathbf{M}}_{\widetilde{W}B} = (\mathbf{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\widetilde{W}B}\boldsymbol{\Omega}_{BB}^{\widetilde{W}B}$, to be subtracted,

$$\delta\dot{\boldsymbol{\Psi}}\mathbf{M}_{\widetilde{W}B} = (\boldsymbol{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\widetilde{W}B}\delta\boldsymbol{\Omega}_{BB}^{\widetilde{W}B} = \mathbf{M}_{\widetilde{W}B}\delta\boldsymbol{\Omega}_{BB}^{WB} + \delta\boldsymbol{\Psi}\mathbf{M}_{\widetilde{W}B}\delta\boldsymbol{\Omega}_{BB}^{WB}$$

Ignore the second order error term,

$$\delta\dot{\boldsymbol{\Psi}}\mathbf{M}_{\widetilde{W}B} = \mathbf{M}_{\widetilde{W}B}\delta\boldsymbol{\Omega}_{BB}^{WB} \tag{4.12}$$

Hence,

$$\delta\dot{\boldsymbol{\Psi}} = \mathbf{M}_{\widetilde{W}B}\delta\boldsymbol{\Omega}_{BB}^{WB}\mathbf{M}_{B\widetilde{W}} \tag{4.13}$$

Transform it into vector representation, we can obtained the error propagation equation,

$$\delta\dot{\vec{\boldsymbol{\Psi}}} = \mathbf{M}_{\widetilde{W}B}\delta(\vec{\boldsymbol{\omega}}^{WB})_B \tag{4.14}$$

The output equation from the accelerometer data can be derived by neglecting the translational acceleration or treat it as measurement noise. For the multirotor system, it is a zero-mean state. However, due to this assumption, certain short-term estimation errors may arise during dynamic maneuvers (coordinated turn/circle). There large value accelerations may last for a while. However, in most of the flight envelop, the assumption is practical.

$$\begin{aligned} \vec{\mathbf{a}}_B^{II} &= \vec{\mathbf{f}}_B^{II} + \mathbf{M}_{BW}\vec{\mathbf{g}}_W \approx 0 \\ \vec{\mathbf{f}}_B^{II} &= -\mathbf{M}_{BW}\vec{\mathbf{g}}_W = -\mathbf{M}_{B\widetilde{W}}\mathbf{M}_{\widetilde{W}W}\vec{\mathbf{g}}_W \end{aligned} \tag{4.15}$$

The $\mathbf{M}_{B\widetilde{W}}$ is the estimated transformation matrix computed from the attitude estimation, and $\mathbf{M}_{\widetilde{W}W}$ is,

$$\mathbf{M}_{\widetilde{W}W} = \mathbf{I} - \delta\boldsymbol{\Psi} = \begin{bmatrix} 1 & \delta\Psi & -\delta\Theta \\ -\delta\Psi & 1 & \delta\Phi \\ \delta\Theta & -\delta\Phi & 1 \end{bmatrix} \tag{4.16}$$

Hence,

$$\vec{\mathbf{f}}_B^{II} = -\mathbf{M}_{B\widetilde{W}}(\mathbf{I} - \delta\boldsymbol{\Psi})\vec{\mathbf{g}}_W = -\mathbf{M}_{B\widetilde{W}}\vec{\mathbf{g}}_W + \mathbf{M}_{B\widetilde{W}}\delta\boldsymbol{\Psi}\vec{\mathbf{g}}_W$$

Transforming the above equation into O frame by multiplying both sides with $\mathbf{M}_{\widetilde{W}B}$

$$\vec{\mathbf{f}}_{\widetilde{W}}^{II} = \mathbf{M}_{\widetilde{W}B}\vec{\mathbf{f}}_B^{II} = -\vec{\mathbf{g}}_W + \delta\boldsymbol{\Psi}\vec{\mathbf{g}}_W \tag{4.17}$$

Since $\boldsymbol{\Psi}$ is a skew-symmetric matrix, the following transformation can be made,

$$\delta\boldsymbol{\Psi}\vec{\mathbf{g}}_W = \delta\vec{\boldsymbol{\Psi}} \times \vec{\mathbf{g}}_W = \vec{\mathbf{g}}_W \times \delta\vec{\boldsymbol{\Psi}} = \mathbf{G}_W\delta\vec{\boldsymbol{\Psi}} \tag{4.18}$$

Where,

$$\vec{\mathbf{g}}_W = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

$$\mathbf{G}_W = \begin{bmatrix} 0 & -g & 0 \\ g & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Taking the first two row of the Eq. (4.17), we can derive the output equation for the accelerometer measurement.

$$\begin{bmatrix} f_x \\ f_y \end{bmatrix}_{\widetilde{W}}^{II} = \begin{bmatrix} 0 & g \\ -g & 0 \end{bmatrix}\begin{bmatrix} \delta\Phi \\ \delta\Theta \end{bmatrix} \tag{4.19}$$

The output equation for the vision heading is relative simple

$$\delta\Psi_{\text{vision}} = \delta\Psi \tag{4.20}$$

With the above mathematical model, the standard AHRS [52] can be applied.

4.2.2. Position Observer

The model for the position observer is relatively simple. The input is the accelerometer and the output is the position measured by the vision sensor. The states are the position and velocity in the World frame, which can be regarded as inertial frame for the quadrotors. So Newton's 2nd law can be applied in W frame.

The system equation for this model is linear as a chain of two integrators,

$$\begin{bmatrix} (\dot{\vec{\mathbf{r}}})_W^W \\ (\dot{\vec{\mathbf{v}}})_W^{WW} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} (\vec{\mathbf{r}})_W \\ (\vec{\mathbf{v}})_W^W \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} (\vec{\mathbf{a}})_W^{WW} \tag{4.21}$$

where, the translational acceleration $(\vec{\mathbf{a}})_W^{WW}$ can be calculated using the AHRS attitude estimation and accelerometer data, assuming the World frame is the inertial frame.

$$(\vec{\mathbf{a}})_W^{WW} = \mathbf{M}_{WB}\vec{\mathbf{f}}_B^{II} + \vec{\mathbf{g}}_W \tag{4.22}$$

The measurement equation is,

$$\vec{\mathbf{y}} = \mathbf{H}\vec{\mathbf{x}} \tag{4.23}$$

$$(\vec{\mathbf{r}})_{W,vision} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} (\vec{\mathbf{r}})_W \\ (\vec{\mathbf{v}})_W^W \end{bmatrix} \tag{4.24}$$

Using the fixed-gain-observer structure as given in Eq. (2.98), the gains can be designed based on the properties of the vision sensor and accelerometer. Together with the AHRS system, the full state vector necessary for the position control is available. In the flight tests, the cascaded data fusion structure gives high robustness and adequate performance, though certain attitude estimation error may arise due to the dynamic maneuvers.

4.2.3. Full State Extended Kalman Filter

The full state EKF includes the position, velocity and attitude states. It can correct those errors due to the dynamic maneuvers because the attitude drift is no longer corrected by the acceleration measurement, but by the position measurement, either from the GPS or the vision sensor.

The EKF for the complete state vector of position dynamics has same structure as the AHRS system, but extend the states to position. The propagation part propagates IMU measurement to all states. It combines the propagations in the AHRS and position observer as in Eqs. (4.3) and (4.21).

The error state model needs to be derived. In addition to the cascaded structure, the accelerometer bias can be observed from the velocity state in the full state model (it is not observable in hovering state where the velocity is zero). In the implemented EKF, there are 12 error states: position, velocity, attitude, and accelerometer bias. The error attitude dynamics has been derived in Eq. (4.14). The error velocity dynamics is to be derived. Recall the full state translational dynamics equation,

$$\left(\dot{\mathbf{v}}\right)_W^{WW} = \mathbf{M}_{WB}\vec{\mathbf{f}}_B^{II} + \vec{\mathbf{g}}_W \tag{4.25}$$

Break the true state into estimated state and error state,

$$\left(\dot{\mathbf{v}}\right)_W^{WW} = \left(\dot{\tilde{\mathbf{v}}}\right)_W^{WW} + \delta\left(\dot{\mathbf{v}}\right)_W^{WW} \tag{4.26}$$

$$\mathbf{M}_{WB} = (\mathbf{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\tilde{W}B} \tag{4.27}$$

$$\vec{\mathbf{f}}_B^{II} = \left(\tilde{\vec{\mathbf{f}}}_B^{II} + \delta\vec{\mathbf{f}}_B^{II}\right) \tag{4.28}$$

Substitute them into the dynamic equation,

$$\left(\dot{\tilde{\mathbf{v}}}\right)_W^{WW} + \delta\left(\dot{\mathbf{v}}\right)_W^{WW} = (\mathbf{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\tilde{W}B}\left(\tilde{\vec{\mathbf{f}}}_B^{II} + \delta\vec{\mathbf{f}}_B^{II}\right) + \vec{\mathbf{g}}_W \tag{4.29}$$

While the estimated state dynamics is,

$$\left(\dot{\tilde{\mathbf{v}}}\right)_W^{WW} = \mathbf{M}_{\tilde{W}B}\tilde{\vec{\mathbf{f}}}_B^{II} + \vec{\mathbf{g}}_W \tag{4.30}$$

The remaining error part can be obtained by subtracting the above two equations,

$$\delta\left(\dot{\mathbf{v}}\right)_W^{WW} = (\mathbf{I} + \delta\boldsymbol{\Psi})\mathbf{M}_{\tilde{W}B}\left(\tilde{\vec{\mathbf{f}}}_B^{II} + \delta\vec{\mathbf{f}}_B^{II}\right) - \mathbf{M}_{\tilde{W}B}\tilde{\vec{\mathbf{f}}}_B^{II} \tag{4.31}$$

Ignoring the second order error term,

$$\delta\left(\dot{\mathbf{v}}\right)_W^{WW} = \mathbf{M}_{\tilde{W}B}\delta\vec{\mathbf{f}}_B^{II} + \delta\boldsymbol{\Psi}\mathbf{M}_{\tilde{W}B}\tilde{\vec{\mathbf{f}}}_B^{II} \tag{4.32}$$

$$\delta\left(\dot{\vec{\mathbf{v}}}\right)_{W}^{WW} = \mathbf{M}_{\tilde{W}B}\delta\vec{\mathbf{f}}_{B}^{II} + \delta\boldsymbol{\Psi}\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \tag{4.33}$$

Where,

$$\delta\boldsymbol{\Psi}\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} = \delta\vec{\boldsymbol{\Psi}} \times \tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} = -\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \times \delta\vec{\boldsymbol{\Psi}} = -\left(\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \times\right)\delta\vec{\boldsymbol{\Psi}}$$

The $\left(\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \times\right)$ indicate the skew symmetric matrix form of the vector $\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II}$

$$\left(\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \times\right) = \begin{bmatrix} 0 & -f_z & f_y \\ f_z & 0 & -f_x \\ -f_y & f_x & 0 \end{bmatrix} \tag{4.34}$$

In addition, the error accelerometer consists of noise and accelerometer bias,

$$\delta\vec{\mathbf{f}}_{B}^{II} = \delta\tilde{\vec{\mathbf{f}}}_{B}^{II} + \vec{\mathbf{b}}_{a} \tag{4.35}$$

Hence, the error state equation for velocity derivative is,

$$\delta\left(\dot{\vec{\mathbf{v}}}\right)_{W}^{WW} = \mathbf{M}_{\tilde{W}B}\left(\delta\tilde{\vec{\mathbf{f}}}_{B}^{II} + \vec{\mathbf{b}}_{a}\right) - \left(\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \times\right)\delta\vec{\boldsymbol{\Psi}} \tag{4.36}$$

To sum the error state equation up in matrix form,

$$\begin{bmatrix} \delta\left(\dot{\vec{\mathbf{r}}}\right)_{W}^{W} \\ \delta\left(\dot{\vec{\mathbf{v}}}\right)_{W}^{WW} \\ \delta\dot{\vec{\boldsymbol{\Psi}}} \\ \dot{\vec{\mathbf{b}}}_{a} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\left(\tilde{\vec{\mathbf{f}}}_{\tilde{W}}^{II} \times\right) & \mathbf{M}_{\tilde{W}B} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \delta(\vec{\mathbf{r}})_{W} \\ \delta(\vec{\mathbf{v}})_{W}^{W} \\ \delta\vec{\boldsymbol{\Psi}} \\ \vec{\mathbf{b}}_{a} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{\tilde{W}B} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{\tilde{W}B} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \delta\tilde{\vec{\mathbf{f}}}_{B}^{II} \\ \delta(\vec{\boldsymbol{\omega}}^{WB})_{B} \\ \delta\vec{\mathbf{b}}_{a} \end{bmatrix} \tag{4.37}$$

With the above mathematical model, the standard Kalman filter technique can be applied. In the quadrotor implementation, the EKF has a couple of disadvantages compared to the cascaded structure of AHRS and position observer. The main reason is that position measurement covariance for the vision sensor is very difficult to quantify and the estimated covariance could be far away from the true value. In the full state EKF, the accuracy and integrity of the position measurement, especially the confidence level, could propagate to the attitude estimation. Good estimate could improve the estimation quality, but false estimate of the confidence level will reduce the robustness of the attitude estimation because of the extra dependence on position measurement. In the case of bad or wrong covariance information given by the implemented vision algorithm [5], wrong corrections will be applied to attitude estimation. In addition, if position signals are lost for a while and then recovered later (typical GPS behavior), EKF would response with large corrections to recover the states as fast as possible. However, this will, on one hand, effect the attitude estimation with big correction step; and on the other hand, couple the control system due to the large state correction. In these aspects, the full state EKF is not as robust as the cascaded data fusion structure, though it has slightly better estimation performance. The choice of full state EKF and cascaded data fusion structure shall subject to applications. In our quadrotor application, the cascaded structure is favored in the implementations.

4.3. Models for Control Design

The mathematical models for the control designs are developed so that model-based control designs like NDI and Backstepping can be applied. It is often much simpler than the simulation model, but it should capture the most distinct feature of the nonlinearities of the plant dynamics. In the following section, the mathematical models used for the control design are introduced, starting from the system input, the motor commands, until the system output, position.

The signal flow diagram is shown in Figure 4.2. The new variable $\mathbf{g}_{xy}$ which serves as an alternative of the pitch and bank angles, will be introduced in the following section. The four integrators indicated the position system has at least system order of four, if the actuator dynamics is not considered.

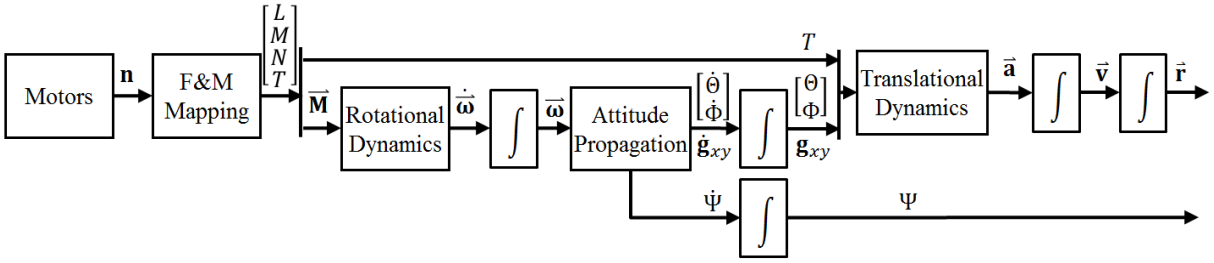


Figure 4.2 Signal flow diagram of the quadrotor position dynamics.

4.3.1. The Force and Moment Mapping

The force and moment mapping refers to the mapping from the system input to the actuated forces and moments. The actuator dynamics, which is the motor dynamics in this case, is not included in the models for control design. Its dynamics is normally regarded as first order lag element dynamics.

The control input for the experimental quadrotor is the RPM commands. There is an internal motor controller regulates the propeller rotation speed with feedback control. In the normal region of operation, a quadratic relationship is normally assumed between the RPM command and the force/torque generated by the propeller. However, the propeller efficiency drops with at high RPM and the power index of the mapping gets smaller.

$$F_{prop,i} = k_n \cdot n_i^p; \quad M_{prop,i} = k_m \cdot F_{prop,i} \tag{4.38}$$

Where k_n and k_m are the propeller thrust and moment constant, and the power index p is equal or smaller than two. They can be estimated in experiments and flight tests. Certain errors can be expected in the estimation.

With the single propeller force and moment, the total forces and moments can be calculated as follows,

$$\begin{bmatrix} L \\ M \\ N \\ T \end{bmatrix} = \begin{bmatrix} 0 & -R & 0 & R \\ R & 0 & -R & 0 \\ -k_m & k_m & -k_m & k_m \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} F_{prop,1} \\ F_{prop,2} \\ F_{prop,3} \\ F_{prop,4} \end{bmatrix} \tag{4.39}$$

The force and moment models are typically utilized to compute a unique solution from the desired force and moments to the actuator commands. In the control system, this is called control allocation. For over-actuated system, e.g. hex-copter, the control allocation problems can be formulated as optimization problems so that all available actuations can be utilized and secondary objectives may be achieved. However, the quadrotor is an under-actuated system, i.e. the force actuation is only acting on the Body-fixed z axis so that it cannot accelerate in a direction perpendicular to the motor axis, although it can be indirectly controlled. For such under-actuated system, the controls can be allocated by simple inverse of the mapping equation (4.39).

4.3.2. Rotational Dynamics

Given the total moments from the motor inputs, the rotational dynamics can be derived using Newton's second law,

$$\left(\dot{\vec{\boldsymbol{\omega}}}^{WB}\right)_B^B = (\mathbf{I}^G)_{BB}^{-1} \cdot \Sigma\left(\vec{\mathbf{M}}^G\right)_B - (\mathbf{I}^G)_{BB}^{-1} \cdot (\vec{\boldsymbol{\omega}}^{WB})_B \times \{(\mathbf{I}^G)_{BB} \cdot (\vec{\boldsymbol{\omega}}^{WB})_B\} \tag{4.40}$$

In a simplified form without the notations,

$$\dot{\vec{\boldsymbol{\omega}}} = \mathbf{I}^{-1} \cdot \vec{\mathbf{M}} - \mathbf{I}^{-1} \cdot \vec{\boldsymbol{\omega}} \times \mathbf{I}\vec{\boldsymbol{\omega}} \tag{4.41}$$

This rotational dynamic equation is the same for most of the aerial vehicles. This nonlinear dynamics is often feedback linearized by so-called inertial decoupling. The moment of inertia of the quadrotor can be calculated from the CAD model or measured experimentally.

4.3.3. Attitude propagation by Euler Angles

After the rotational dynamics, the attitude propagation is only kinematic, i.e. there is no uncertainty in the equation. Simply propagating the angular rate, the attitude represented by Euler angles can be derived.

$$\begin{bmatrix} \dot{\Phi} \\ \dot{\Theta} \\ \dot{\Psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\Phi\tan\Theta & \cos\Phi\tan\Theta \\ 0 & \cos\Phi & -\sin\Phi \\ 0 & \dfrac{\sin\Phi}{\cos\Theta} & \dfrac{\cos\Phi}{\cos\Theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.42}$$

However, the Euler angle representation of the attitude leads to a singularity in the above equation, at pitch angle of 90°. In addition, the trigonometric functions can't be directly implemented with fixed-point computing environment. Look-up tables could be used to compute them, but a compromise needs to be made between the accuracy and computational power.

The attitude could also be propagated using the quaternions representation. The propagation equations have been introduced in section 4.2.1.1. The singularity at pitch angle of 90° can be eliminated as well as computational power is reduced, however, the quaternions introduce extra complexities in the control design, which is not covered in the scope of this thesis.

4.3.4. Attitude propagation using a novel parameterization

A novel parameter was first introduced in the author's paper [36] to represent the pitch and bank angles. In the multirotor platform, the pitch and bank angles are commanded to change the direction of the total thrust vector. However, imaging the force dynamics in the Body-fixed frame, the thrust vector is always acting on the z-axis, the only 'force' acting on the x- and y-axis is the gravitational force. By pitch or bank rotations, the gravitational force distribution on the x- and y-axis changes from 0 to 1g.

The idea is to use the gravitational acceleration components acting on the x- and y-axis of the Body-fixed frame to represent the pitch and bank angle. The acceleration representation naturally fits better in the dynamic model constructed based on the Newton's law, rather than the human-orientated Euler angle definition. The gravitational acceleration in the Body-fixed frame is,

$$\vec{\mathbf{g}}_B = \mathbf{M}_{BW} \cdot \vec{\mathbf{g}}_W \tag{4.43}$$

Where

$$\mathbf{M}_{BW} = \begin{bmatrix} \cos\Psi\cos\Theta & \sin\Psi\cos\Theta & -\sin\Theta \\ \cos\Psi\sin\Theta\sin\Phi - \sin\Psi\cos\Phi & \sin\Psi\sin\Theta\sin\Phi + \cos\Psi\cos\Phi & \cos\Theta\sin\Phi \\ \cos\Psi\sin\Theta\cos\Phi + \sin\Psi\sin\Phi & \sin\Psi\sin\Theta\cos\Phi - \cos\Psi\sin\Phi & \cos\Theta\cos\Phi \end{bmatrix}$$

$$\vec{\mathbf{g}}_W = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

To insert the transformation matrix calculated from the data fusion either quaternion or Euler angle formulation,

$$\begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = g \cdot \begin{bmatrix} 2(q_1q_3 - q_0q_2) \\ 2(q_2q_3 + q_0q_1) \\ q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} = g \cdot \begin{bmatrix} -\sin\Theta \\ \cos\Theta\sin\Phi \\ \cos\Theta\cos\Phi \end{bmatrix} \tag{4.44}$$

Hence, two variables can be extracted from above,

$$\begin{cases} g_x = g \cdot -\sin\Theta \\ g_y = g \cdot \cos\Theta\sin\Phi \end{cases} \tag{4.45}$$

Mathematically, we can see from Eq. (4.45) that the two variables g_x and g_y contain the same information as pitch and bank angles. Physically, thinking in the Body-fixed frame, the rotation of the attitude does not change the direction of thrust vector (z-component), but the direction of the gravitational vector (x-y-components).

The propagation equation from the angular rate to the new variables can be derived from the Euler differentiation rule,

$$\left(\dot{\vec{\mathbf{g}}}\right)_B^B = \underbrace{\left(\dot{\vec{\mathbf{g}}}\right)_B^W}_{0} - \left(\vec{\boldsymbol{\omega}}^{WB}\right)_B \times \vec{\mathbf{g}}_B = \vec{\mathbf{g}}_B \times \left(\vec{\boldsymbol{\omega}}^{WB}\right)_B \tag{4.46}$$

In scalar form,

$$\begin{aligned} \dot{g}_x &= -g_z \cdot q + g_y \cdot r \\ \dot{g}_y &= g_z \cdot p - g_x \cdot r \\ \dot{g}_z &= -g_y \cdot p + g_x \cdot q \end{aligned}$$

The first two equations give the propagation from pitch and roll rate to the g_x and g_y. Compared to the conventional attitude parameterization, the heading is excluded, however considering it is not related to the translational dynamic, a separate heading propagation equation works fine as well.

$$\dot{\Psi} = \frac{\sin\Phi}{\cos\Theta} q + \frac{\cos\Phi}{\cos\Theta} r \tag{4.47}$$

$$\dot{\Psi} = \frac{g}{g^2 - g_x^2}\left(g_y q + g_z r\right) \tag{4.48}$$

Together with the novel parameterization, the equation of motion can be collected in matrix form,

$$\begin{bmatrix} \dot{g}_x \\ \dot{g}_y \\ \dot{\Psi} \end{bmatrix} = \begin{bmatrix} 0 & -g_z & g_y \\ g_z & 0 & -g_x \\ 0 & \frac{g g_y}{g^2 - g_x^2} & \frac{g g_z}{g^2 - g_x^2} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.49}$$

4.3.5. Translational dynamics

The translational dynamics can be derived in an inertial frame where Newton's second law is applicable, in our case, the World frame.

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} = \frac{\Sigma\left(\vec{\mathbf{F}}^G\right)_W}{m} = \frac{\mathbf{M}_{WB}}{m} \cdot \left(\vec{\mathbf{F}}_{prop,B} + \vec{\mathbf{F}}_{aero,B}\right) + \vec{\mathbf{g}}_W \tag{4.50}$$

where the transformation matrix $\mathbf{M}_{WB}$,

$$\mathbf{M}_{WB} = \mathbf{M}_{BW}^T$$

$$\mathbf{M}_{WB} = \begin{bmatrix} \cos\Psi\cos\Theta & \cos\Psi\sin\Theta\sin\Phi - \sin\Psi\cos\Phi & \cos\Psi\sin\Theta\cos\Phi + \sin\Psi\sin\Phi \\ \sin\Psi\cos\Theta & \sin\Psi\sin\Theta\sin\Phi + \cos\Psi\cos\Phi & \sin\Psi\sin\Theta\cos\Phi - \cos\Psi\sin\Phi \\ -\sin\Theta & \cos\Theta\sin\Phi & \cos\Theta\cos\Phi \end{bmatrix} \tag{4.51}$$

where the propeller force $\vec{\mathbf{F}}_{prop,B}$ is the total thrust force vector. It is aligned with the Body-fixed z-axis of the vehicle.

$$\vec{\mathbf{F}}_{prop,B} = \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} \tag{4.52}$$

The aerodynamic force $\vec{\mathbf{F}}_{aero,B}$ mainly consists of aerodynamic drag and disturbances. In conventional formulation using Euler angles, there are often ignored in order to solve the control commands of attitude angles. So that Eq. (4.50) can be simplified as,

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} \approx \mathbf{M}_{WB} \cdot \begin{bmatrix} 0 \\ 0 \\ -\frac{T}{m} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = -\frac{T}{m} \cdot \begin{bmatrix} \cos\Psi \sin\Theta \cos\Phi + \sin\Psi \sin\Phi \\ \sin\Psi \sin\Theta \cos\Phi - \cos\Psi \sin\Phi \\ \cos\Theta \cos\Phi \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{4.53}$$

Based on the dynamic equation, proper model based control design computing the desired thrust, pitch and bank angles can be developed [5].

With the new parameterization introduced, the translational dynamics can be formulated in a much simpler form,

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} = \mathbf{M}_{WB} \cdot \vec{\mathbf{f}}_B + \vec{\mathbf{g}}_W = \mathbf{M}_{WB} \cdot \left(\vec{\mathbf{f}}_B + \vec{\mathbf{g}}_B\right) \tag{4.54}$$

The specific force $\vec{\boldsymbol{f}}_B$ is the sum of propeller force and aerodynamic force (i.e. all non-gravitational force) normalized by the mass. It is also what a three-axis accelerometer measures. The pseudo control can be grouped together,

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} = \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}_B + \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix}_B \right) = \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_x \\ g_y \\ f_z \end{bmatrix}_B \right) \tag{4.55}$$

The specific force on the Body-fixed z-axis f_z is composed of two parts, propeller thrust and aerodynamic force. Ignoring the aerodynamic drag and disturbances on the z-axis, the translational dynamic equation of motion is,

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} = \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_x \\ g_y \\ \Delta f_z - \frac{T}{m} \end{bmatrix}_B \right)$$

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} \approx \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_x \\ g_y \\ -\frac{T}{m} \end{bmatrix}_B \right) \tag{4.56}$$

4.3.6. Position Propagation

The position propagation in the World frame is linear and straightforward,

$$\left(\dot{\vec{\mathbf{r}}}_k^G\right)_W^W = \left(\vec{\mathbf{v}}_k^G\right)_W^W \tag{4.57}$$

For the normal operation of the multirotors, NED or a leveled local frame is often used. GPS measurements can be linearized and transformed in the local coordinate. Hence, the position propagation is with no complexity for multirotors.

5. Attitude Control Designs

Nowadays IMU is almost a standard sensor package integrated on quadrotors. It can provide accuracy and robust attitude estimations for the control task. The heading angle could also be measured by onboard magnetometers. Therefore, the attitude control is one of the most common and basic control tasks for quadrotors.

In this section, first the control allocation is introduced, because it is the same for all control designs. Next, the attitude control is designed based on the conventional Euler angle representation. Then the novel parameterization introduced in above section is used for the attitude control design.

The comparison of different control design structures is further illustrated with the attitude example. From the mathematical model of the attitude dynamics, we can see it is a system of triangular form. Therefore direct NDI/Backstepping can be applied, which are gives better performance to the other designs with filter involved. In the direct NDI/Backstepping designs, there are four variations: a), c), d) and e) in section 3. Designs a) and e) are actually the same. Hence, in total three different designs [a), c) & d)] will be introduced in the following sections.

The final comparison results in the last section are taken from simulation and flight tests. Detailed experimental setups are described in Chapter 8.

5.1. Control Allocation

The control allocation unit allocates the desired forces and moments to the system input, in this case, the RPM of each motor. For the experimental vehicle used in the scope of the thesis, the motor controller has dedicated RPM feedback control, which ensures the stability of the internal actuator dynamics. Hence the nonlinear control approaches don't have to deal with unstable internal dynamics with this class of system.

The allocation equation can be obtained by inversing the force and moment mapping equations in Eqs. (4.38) and (4.39),

$$\begin{bmatrix} F_{prop,1} \\ F_{prop,2} \\ F_{prop,3} \\ F_{prop,4} \end{bmatrix} = \frac{1}{4} \cdot \begin{bmatrix} 0 & 2R^{-1} & -k_m^{-1} & 1 \\ -2R^{-1} & 0 & k_m^{-1} & 1 \\ 0 & -2R^{-1} & -k_m^{-1} & 1 \\ 2R^{-1} & 0 & k_m^{-1} & 1 \end{bmatrix} \begin{bmatrix} L \\ M \\ N \\ T \end{bmatrix}, \qquad n_i = \left(\frac{F_{prop,i}}{k_n}\right)^p \tag{5.1}$$

Using the above equations, the allocation works well in the nominal range. Hardware limits on the total thrust and moments can be specified beforehand. However, if any of the saturations get into conflicts, priorities need to be assigned. Optimization is not necessary for quadrotor, as there is no redundancy in the system.

The priorities can be assigned depending on the application. In this case, to ensure rotational stability, the pitching and rolling moments have the top priorities, followed by the total thrust, and the yawing moment is with lowest priority.

Step 1, for the pitching and rolling moments only the hardware limit is imposed,

$$\begin{cases} -L_{max} = L_{min} < L < L_{max} \\ -M_{max} = M_{min} < M < M_{max} \end{cases} \tag{5.2}$$

The maximum pitching and rolling moments can be computed as,

$$L_{max} = M_{max} = \left(\frac{T_{max}}{4} - \frac{T_{min}}{4}\right) \cdot R$$

Step 2, for the total thrust, not only the hardware limit is imposed, but also the pitching and rolling moments limits are considered. By imposing the limits, the total thrust is dynamically limited by the pitching and rolling moments.

$$\begin{gathered} 2N = T_{min} < T < T_{max} = 14N \\ max\left(\left|\frac{2L}{R}\right|, \left|\frac{2M}{R}\right|\right) < T < T_{max} - max\left(\left|\frac{2L}{R}\right|, \left|\frac{2M}{R}\right|\right) \end{gathered} \tag{5.3}$$

Step 3, for the yawing moments, the limits comes from the physical limits and the other three control inputs

$$\begin{gathered} N_{min} < N < N_{max} \\ max\left(\left|\frac{2L}{R}\right| - T,\ T + \left|\frac{2M}{R}\right| - T_{max}\right) < \frac{N}{k_m} < min\left(T - \left|\frac{2M}{R}\right|, T_{max} - T - \left|\frac{2L}{R}\right|\right) \end{gathered} \tag{5.4}$$

5.2. Attitude Control using Euler Angles

The Euler angles are probably the most intuitive way to represent the attitude from pilot point of view. Hence, it is natural to use them as control state. If the quadrotor is only controlled near the hovering attitude, linear control using the Euler angles is sufficient. However, in the large angle commands or aggressive maneuver region, the dynamics gets highly nonlinear and tightly coupled. There nonlinear control is a better choice to provide consistent control behavior without gain scheduling (at least with less dense grid).

5.2.1. Classical NDI Control using Euler Angles

Classical NDI control is designed using Euler angles in this section, i.e. NDI design a). As shown in the attitude propagation by Euler angle in Eq. (4.42) and the rotational dynamics, the attitude control has system order of 2, which corresponds to NDI relative degree 2.

5.2.1.1. Dynamic Inversion

The considered system output is the Euler angles and the control input is the desired moments. The nonlinear control loop(s) calculates the desired moments, which is then allocated to the motor RPM in the control allocation unit.

As the desired moments do not appear in the attitude propagation equation in Eq. (4.42), the attitude derivative needs to be further differentiated,

$$\ddot{\vec{\Psi}} = \begin{bmatrix} \ddot{\Phi} \\ \ddot{\Theta} \\ \ddot{\Psi} \end{bmatrix} = \begin{bmatrix} \dot{\Theta}\dot{\Phi}\tan\Theta + \dfrac{\dot{\Theta}\dot{\Psi}}{\cos\Theta} \\ -\dot{\Psi}\dot{\Phi}\cos\Theta \\ \dfrac{\dot{\Theta}\dot{\Phi}}{\cos\Theta} + \dot{\Theta}\dot{\Psi}\tan\Theta \end{bmatrix} + \begin{bmatrix} 1 & \sin\Phi\tan\Theta & \cos\Phi\tan\Theta \\ 0 & \cos\Phi & -\sin\Phi \\ 0 & \dfrac{\sin\Phi}{\cos\Theta} & \dfrac{\cos\Phi}{\cos\Theta} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \tag{5.5}$$

Inverting the above equation, the desired angular accelerations can be calculated from the pseudo control $\ddot{\vec{\Psi}}_v$, i.e., the 2nd derivative of the output vector,

$$\dot{\vec{\omega}}_d = \begin{bmatrix} \ddot{\Phi}_v - \ddot{\Psi}_v \sin\Theta - \dot{\Theta}\dot{\Psi}\cos\Theta \\ \ddot{\Theta}_v \cos\Phi + \ddot{\Psi}_v \sin\Phi\cos\Theta + \dot{\Psi}\dot{\Phi}\cos\Phi\cos\Theta - \dot{\Theta}\dot{\Phi}\sin\Phi - \dot{\Theta}\dot{\Psi}\sin\Phi\sin\Theta \\ -\ddot{\Theta}_v \sin\Phi + \ddot{\Psi}_v \cos\Phi\cos\Theta - \dot{\Psi}\dot{\Phi}\sin\Phi\cos\Theta - \dot{\Theta}\dot{\Phi}\cos\Phi - \dot{\Theta}\dot{\Psi}\cos\Phi\sin\Theta \end{bmatrix} \tag{5.6}$$

Next, from the angular accelerations, the desired moment can be calculated by inverting the rotational dynamic equation in Eq. (4.40).

$$\vec{\mathbf{M}}_d = \mathbf{I}\dot{\vec{\omega}}_d + \vec{\omega} \times \mathbf{I}\vec{\omega} \tag{5.7}$$

5.2.1.2. Reference Model and Error Controller

With the above feedback linearization and the control allocation unit, the attitude system is transformed into a chain of two integrators, from the 2nd derivatives of the Euler angles to the Euler angles. The corresponding linear reference model and controller can be designed easily as in section 2.2.2.

The reference model gets direct commands attitude commands $\vec{\boldsymbol{\Psi}}_c$,

$$\ddot{\vec{\boldsymbol{\Psi}}}_r = 2\boldsymbol{\zeta}\boldsymbol{\omega}_0\dot{\vec{\boldsymbol{\Psi}}}_r + \boldsymbol{\omega}_0^2\left(\vec{\boldsymbol{\Psi}}_c - \vec{\boldsymbol{\Psi}}_r\right) \tag{5.8}$$

Normally the relative damping ζ is set to be 1, and the natural frequency $\boldsymbol{\omega}_0$ can be replaced by a diagonal matrix here so that different frequencies can be assigned to different axis. Typically for quadrotor, the yaw dynamics is much slower than the pitch or roll dynamics. Hence, smaller natural frequency shall be assigned to the azimuth angle. The same applies to the feedback gain design.

A PD linear controller is designed as follows,

$$\ddot{\vec{\boldsymbol{\Psi}}}_v = \ddot{\vec{\boldsymbol{\Psi}}}_r + \mathbf{K}_d\left(\dot{\vec{\boldsymbol{\Psi}}}_r - \dot{\vec{\boldsymbol{\Psi}}}\right) + \mathbf{K}_p\left(\vec{\boldsymbol{\Psi}}_r - \vec{\boldsymbol{\Psi}}\right) \tag{5.9}$$

The attitude control structure is illustrated in Figure 5.1. In the data fusion block, a AHRS is implemented for attitude estimation.

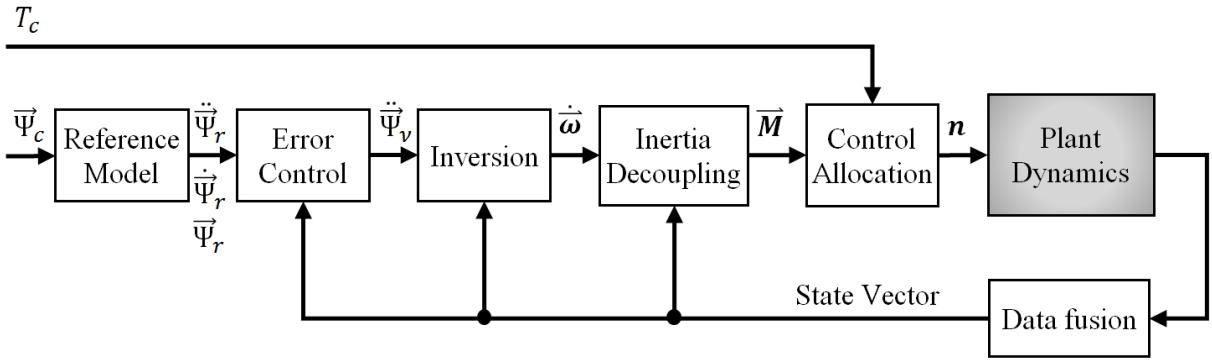


Figure 5.1 Attitude Control Structure using the Euler Angles

5.3. Attitude Control using the Novel Parameterization

The dynamic equations using the novel parameterization have been stated in section 4.3.4. As it is a system of triangular form (strict-feedback form), the non-filtered control designs a), c) and d) can be applied on the attitude control system.

Unlike the Euler angle approach, the pitch and roll dynamics in this case are separated from the yaw dynamics with the novel parameterization. In fact, it is not bad to separate them: firstly, the yaw dynamics is much slower than the pitch or roll, secondly, the yaw dynamics is inherited decoupled from position control due to the quadrotor symmetry. Therefore, the yaw control can be separately designed.

The implementation of the dynamic inversion in Eq. (5.6) is quite tedious, especially in fixed-point environment due to the trigonometric functions. This can be simplified with the new attitude parameterization in the next section.

In the following control designs, the control allocation, inertial decoupling and reference model stay the same as those in the Euler angle application, the main difference lies in the feedback control formulations of the $\mathbf{g}_{xy}$ control block. This will be explained in detailed in the following sections. The general control structure is shown in Figure 5.2,

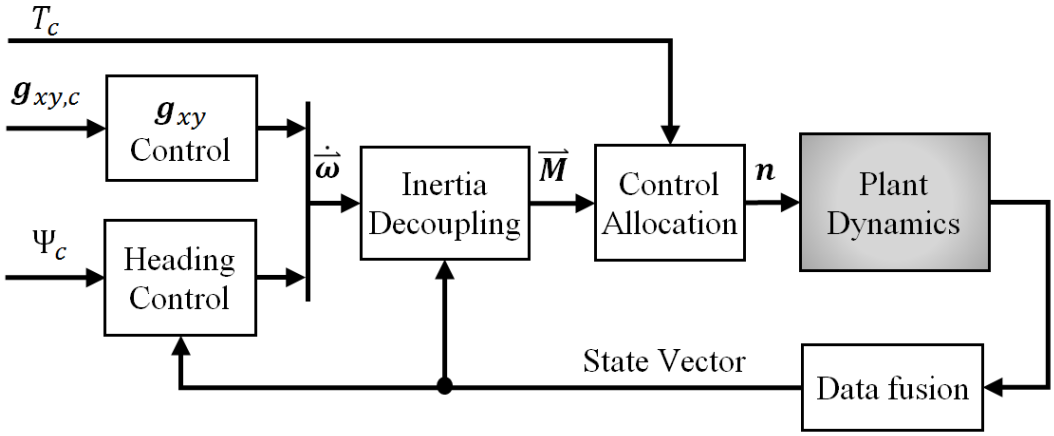


Figure 5.2 Attitude Control Structure using the Novel Parameterization

5.3.1. Classical NDI (Design a)

Recall the attitude propagation equation,

$$\left(\dot{\vec{\mathbf{g}}}\right)_B^B = \vec{\mathbf{g}}_B \times \left(\vec{\boldsymbol{\omega}}^{WB}\right)_B \tag{5.10}$$

The desired angular accelerations do not appear directly; hence, further differentiation is needed,

$$\left(\ddot{\vec{\mathbf{g}}}\right)_B^{BB} = \left(\dot{\vec{\mathbf{g}}}\right)_B^B \times \left(\vec{\boldsymbol{\omega}}^{WB}\right)_B + \vec{\mathbf{g}}_B \times \left(\dot{\vec{\boldsymbol{\omega}}}^{WB}\right)_B^B \tag{5.11}$$

Extracting the first two variables in scalar form,

$$\begin{bmatrix}\ddot{g}_x\\ \ddot{g}_y\end{bmatrix}_B^{BB} = g_z\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}\begin{bmatrix}\dot{p}\\ \dot{q}\end{bmatrix} + \begin{bmatrix}g_y\\ -g_x\end{bmatrix}_B \dot{r} - \begin{bmatrix}0 & -r & q\\ r & 0 & -p\end{bmatrix}\left(\dot{\boldsymbol{g}}\right)_B^B \tag{5.12}$$

The desired angular accelerations can be computed from the dynamic Eq. (5.12). The plant nonlinearities are grouped and separated from the controls. The $\left(\dot{\boldsymbol{g}}\right)_B^B$ can be computed by Eq. (5.10). The $\dot{r}$ term is yaw acceleration without measurement, but can be approximated by the desired yaw acceleration $\dot{r}_d$. A simple PI controller could be used for yaw rate control.

$$\dot{r}_d = \dot{r}_r + k_p(r_r - r) + k_I\int(r_r - r)\,dt \tag{5.13}$$

The dynamic inversion equation from the pseudo control $\ddot{\mathbf{g}}_{xy}{}_{B,v}^{BB}$ to the desired angular accelerations $[\dot{p} \quad \dot{q}]_d$ is,

$$\begin{bmatrix}\dot{p}\\ \dot{q}\end{bmatrix}_d = g_z^{-1}\cdot\left(\begin{bmatrix}\ddot{g}_y\\ -\ddot{g}_x\end{bmatrix}_{B,v}^{BB} + \begin{bmatrix}g_x\\ g_y\end{bmatrix}_B \dot{r}_d + \begin{bmatrix}r\cdot\dot{g}_x - p\cdot\dot{g}_z\\ r\cdot\dot{g}_y - q\cdot\dot{g}_z\end{bmatrix}\right) \tag{5.14}$$

We can see the inversion is much simpler and easier to be implemented. Similar PD error controller is designed to calculated the pseudo control,

$$\ddot{\mathbf{g}}_{xy}{}_{B,v}^{BB} = \left(\ddot{\mathbf{g}}_{xy}\right)_{B,r}^{BB} + \mathbf{K}_d\left[\left(\dot{\mathbf{g}}_{xy}\right)_{B,r}^{B} - \left(\dot{\mathbf{g}}_{xy}\right)_B^B\right] + \mathbf{K}_p\left[\left(\mathbf{g}_{xy}\right)_{B,r} - \left(\mathbf{g}_{xy}\right)_B\right] \tag{5.15}$$

The desired angular accelerations (together with the yaw acceleration from the yaw control) can be used to calculated the desired moment via the inertia decoupling, and then the system input motor RPM can be calculated in the control allocation.

The control structure in Figure 2.5 is also representative here except the PCH block. The inversion block is a sum of the Eqs. (5.14), (5.7) and (5.1).

The Backstepping design e) will result the same control law as the classical NDI, though the derivation is different.

5.3.2. NDI with original model (Design c)

Recall the dynamic equations from the section 4.3.4,

$$\left(\dot{\mathbf{g}}_{xy}\right)_B^B = \begin{bmatrix}\dot{g}_x\\ \dot{g}_y\end{bmatrix}_B^B = \begin{bmatrix}g_y\\ -g_x\end{bmatrix}_B r + g_z\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}\begin{bmatrix}p\\ q\end{bmatrix} \tag{5.16}$$

The inertia decoupling and control allocation will be the same as the previous cases, so the angular acceleration can be regarded as virtual system input.

$$\begin{bmatrix}\dot{p}\\ \dot{q}\end{bmatrix} = \mathbf{u}$$

Applying the design c) general form, we have the following matches,

$$\begin{cases}\dot{\mathbf{x}}_1 = \mathbf{f}_1 + \mathbf{G}_1\boldsymbol{x}_2\\ \dot{\mathbf{x}}_2 = \mathbf{u}\end{cases}$$

$$\begin{cases}\mathbf{x}_1 = (\mathbf{g}_{xy})_B \\ \mathbf{x}_2 = \begin{bmatrix} p \\ q \end{bmatrix} \\ \mathbf{u} = \begin{bmatrix} \dot{p} \\ \dot{q} \end{bmatrix}\end{cases}$$

$$\begin{cases}\mathbf{f}_1 = \begin{bmatrix} g_y \\ -g_x \end{bmatrix}_B r \\ \mathbf{G}_1 = g_z \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \\ \mathbf{f}_2 = 0 \\ \mathbf{G}_2 = 1\end{cases}$$

The control law can be easily derived from the scalar solution in Eq. (3.20),

$$\begin{cases} \mathbf{x}_{2,des} = \mathbf{G}_1^{-1}[-\mathbf{f}_1 + \dot{\mathbf{x}}_{1r} + \mathbf{K}_1(\mathbf{x}_{1r} - \mathbf{x}_1)] \\ \mathbf{u} = \dot{\mathbf{x}}_{2r} + \mathbf{K}_2(\mathbf{x}_{2,des} - \mathbf{x}_2) + \mathbf{G}_1^{-1}\dot{\mathbf{G}}_1(\mathbf{x}_{2r} - \mathbf{x}_2) \end{cases}$$

Where,

$$\mathbf{G}_1^{-1} = g_z^{-1} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

$$\dot{\mathbf{G}}_1 = \dot{g}_z \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} = (-g_y \cdot p + g_x \cdot q) \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

In addition, the nonlinear reference signals in this implementation are calculated from the linear reference signals as in Eq. (3.16).

$$\mathbf{x}_{2r} = \mathbf{G}_1^{-1}(\dot{\mathbf{x}}_{1r} - \mathbf{f}_1)$$

$$\dot{\mathbf{x}}_{2r} = \mathbf{G}_1^{-1}(\ddot{\mathbf{x}}_{1r} - \dot{\mathbf{f}}_1 - \dot{\mathbf{G}}_1\mathbf{x}_{2r})$$

$$\dot{\mathbf{x}}_{2r} = \mathbf{G}_1^{-1}(\ddot{\mathbf{x}}_{1r} - \dot{\mathbf{f}}_1 - \dot{\mathbf{G}}_1\mathbf{G}_1^{-1}(\dot{\mathbf{x}}_{1r} - \mathbf{f}_1))$$

Substituting the physical parameters,

$$\begin{bmatrix} \dot{p} \\ \dot{q} \end{bmatrix}_{\boldsymbol{r}} = g_z^{-1}\left(\begin{bmatrix} \ddot{g}_y \\ -\ddot{g}_x \end{bmatrix}_{B,r}^{BB} + (\mathbf{g}_{xy})_B \cdot \dot{r} + (\dot{\mathbf{g}}_{xy})_B^B \cdot r\right) - g_z^{-2}\dot{g}_z\left((\mathbf{g}_{xy})_B \cdot r + \begin{bmatrix} \dot{g}_y \\ -\dot{g}_x \end{bmatrix}_{B,r}^{B}\right) \tag{5.17}$$

Similarly, the control law can be calculated,

$$\begin{aligned}\begin{bmatrix} p \\ q \end{bmatrix}_{\boldsymbol{d}} &= g_z^{-1}\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\left[-\begin{bmatrix} g_y \\ -g_x \end{bmatrix}_B r + (\dot{\mathbf{g}}_{xy})_{B,r}^B + \mathbf{K}_1\left((\mathbf{g}_{xy})_{B,r} - (\mathbf{g}_{xy})_B\right)\right] \\ &= g_z^{-1}\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\left[\begin{bmatrix} -g_y \\ g_x \end{bmatrix}_B r + (\dot{\mathbf{g}}_{xy})_{B,r}^B + \mathbf{K}_1\left((\mathbf{g}_{xy})_{B,r} - (\mathbf{g}_{xy})_B\right)\right] \\ &= g_z^{-1}\left(\begin{bmatrix} g_x \\ g_y \end{bmatrix}_B r + \begin{bmatrix} \dot{g}_y \\ -\dot{g}_x \end{bmatrix}_{B,r}^B + \mathbf{K}_1\begin{bmatrix} g_{y,r} - g_y \\ g_x - g_{x,r} \end{bmatrix}_B\right)\end{aligned}$$

$$\begin{aligned}\begin{bmatrix} \dot{p} \\ \dot{q} \end{bmatrix}_{\boldsymbol{d}} &= \begin{bmatrix} \dot{p} \\ \dot{q} \end{bmatrix}_{\boldsymbol{r}} + \mathbf{K}_2\left(\begin{bmatrix} p \\ q \end{bmatrix}_{\boldsymbol{d}} - \begin{bmatrix} p \\ q \end{bmatrix}\right) + g_z^{-1}\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\dot{g}_z\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\left(\begin{bmatrix} p \\ q \end{bmatrix}_{\boldsymbol{r}} - \begin{bmatrix} p \\ q \end{bmatrix}\right) \\ &= \begin{bmatrix} \dot{p} \\ \dot{q} \end{bmatrix}_{\boldsymbol{r}} + \mathbf{K}_2\left(\begin{bmatrix} p \\ q \end{bmatrix}_{\boldsymbol{d}} - \begin{bmatrix} p \\ q \end{bmatrix}\right) + g_z^{-1}\dot{g}_z\left(\begin{bmatrix} p \\ q \end{bmatrix}_{\boldsymbol{r}} - \begin{bmatrix} p \\ q \end{bmatrix}\right)\end{aligned}$$

With the above equations, the output tracking error dynamics can be rendered linear as in Eq. (3.21),

$$\left(\ddot{\mathbf{g}}_{xy,r}-\ddot{\mathbf{g}}_{xy}\right)_B^{BB}+\mathbf{K}_2\left(\dot{\mathbf{g}}_{xy,r}-\dot{\mathbf{g}}_{xy}\right)_B^B+\mathbf{K}_2\mathbf{K}_1\left(\mathbf{g}_{xy,r}-\mathbf{g}_{xy}\right)_B=0 \tag{5.18}$$

For the control structure, please refer to Figure 3.4. Compared with the previous control design, the same linear error dynamic can be obtained but the error feedback is less model-dependent than the classical NDI, i.e. the feedback state is the angular rates instead of $\left(\dot{\mathbf{g}}_{xy}\right)_B^B$. More model information, i.e. the nonlinearities, is shifted to the nonlinear reference model.

5.3.3. Analytical Backstepping with Original model (Design d)

The Backstepping control law can be derived for the attitude system similarly with the scalar example. The general form of the control law is given below similar to Eq. (3.22),

$$\begin{cases}\mathbf{x}_{2,des}=\mathbf{G}_1^{-1}(\dot{\mathbf{x}}_{1r}-\mathbf{f}_1+\mathbf{K}_1\mathbf{z}_1)\\ \mathbf{u}=\dot{\mathbf{x}}_{2r}+\mathbf{K}_2\left(\mathbf{x}_{2,des}-\mathbf{x}_2\right)+\mathbf{G}_1^{-1}\mathbf{K}_1\dot{\mathbf{z}}_1-\mathbf{G}_1^{-2}\dot{\mathbf{G}}_1\mathbf{K}_1\mathbf{z}_1+\mathbf{G}_1^T\mathbf{z}_1\end{cases}$$

Substituting the physical parameters into the above control law,

$$\begin{bmatrix}p\\q\end{bmatrix}_{\boldsymbol{d}}=g_z^{-1}\begin{bmatrix}0&1\\-1&0\end{bmatrix}\left[\begin{bmatrix}-g_y\\g_x\end{bmatrix}_B r+\left(\dot{\mathbf{g}}_{xy}\right)_{B,r}^B+\mathbf{K}_1\left[\left(\mathbf{g}_{xy}\right)_{B,r}-\left(\mathbf{g}_{xy}\right)_B\right]\right]$$

$$\begin{aligned}\begin{bmatrix}\dot{p}\\\dot{q}\end{bmatrix}_{\boldsymbol{d}}=&\begin{bmatrix}\dot{p}\\\dot{q}\end{bmatrix}_{\boldsymbol{r}}+\mathbf{K}_2\left(\begin{bmatrix}p\\q\end{bmatrix}_{\boldsymbol{d}}-\begin{bmatrix}p\\q\end{bmatrix}\right)+g_z^{-1}\begin{bmatrix}0&1\\-1&0\end{bmatrix}\mathbf{K}_1\left[\left(\dot{\mathbf{g}}_{xy}\right)_{B,r}^B-\left(\dot{\mathbf{g}}_{xy}\right)_B^B\right]\\&-g_z^{-1}\begin{bmatrix}0&1\\-1&0\end{bmatrix}g_z^{-1}\begin{bmatrix}0&1\\-1&0\end{bmatrix}\dot{g}_z\begin{bmatrix}0&-1\\1&0\end{bmatrix}\mathbf{K}_1\left[\left(\mathbf{g}_{xy}\right)_{B,r}-\left(\mathbf{g}_{xy}\right)_B\right]\\&+g_z\begin{bmatrix}0&1\\-1&0\end{bmatrix}\left[\left(\mathbf{g}_{xy}\right)_{B,r}-\left(\mathbf{g}_{xy}\right)_B\right]\end{aligned}$$

After rearrangement, the control law to be implemented is,

$$\begin{bmatrix}p\\q\end{bmatrix}_{\boldsymbol{d}}=g_z^{-1}\left(\begin{bmatrix}g_x\\g_y\end{bmatrix}_B r+\begin{bmatrix}\dot{g}_y\\-\dot{g}_x\end{bmatrix}_{B,r}^B+\mathbf{K}_1\begin{bmatrix}g_{y,r}-g_y\\g_x-g_{x,r}\end{bmatrix}_B\right) \tag{5.19}$$

$$\begin{bmatrix}\dot{p}\\\dot{q}\end{bmatrix}_{\boldsymbol{d}}=\begin{bmatrix}\dot{p}\\\dot{q}\end{bmatrix}_{\boldsymbol{r}}+\mathbf{K}_2\left(\begin{bmatrix}p\\q\end{bmatrix}_{\boldsymbol{d}}-\begin{bmatrix}p\\q\end{bmatrix}\right)+g_z^{-1}\mathbf{K}_1\begin{bmatrix}\dot{g}_{y,r}-\dot{g}_y\\\dot{g}_x-\dot{g}_{x,r}\end{bmatrix}_B^B+\left[g_z-g_z^{-2}\dot{g}_z\mathbf{K}_1\right]\begin{bmatrix}g_{y,r}-g_y\\g_x-g_{x,r}\end{bmatrix} \tag{5.20}$$

The resulting output tracking error dynamics is in nonlinear form similar to Eq. (3.24),

$$\begin{bmatrix}\dot{\mathbf{e}}\\\ddot{\mathbf{e}}\end{bmatrix}=\begin{bmatrix}0&1\\-\mathbf{K}_2\mathbf{K}_1+\dot{\mathbf{G}}_1\mathbf{G}_1^{-1}\mathbf{K}_1-\mathbf{G}_1^T\mathbf{G}_1&-\mathbf{K}_1-\mathbf{K}_2+\dot{\mathbf{G}}_1\mathbf{G}_1^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{e}\\\dot{\mathbf{e}}\end{bmatrix} \tag{5.21}$$

Where,

$$\mathbf{e}=\left(\mathbf{g}_{xy}\right)_{B,r}-\left(\mathbf{g}_{xy}\right)_B$$

The derivative gain and the proportional gain are state dependent,

$$\begin{aligned}\mathbf{K}_d&=-\mathbf{K}_1-\mathbf{K}_2+\dot{g}_z\begin{bmatrix}0&-1\\1&0\end{bmatrix}g_z^{-1}\begin{bmatrix}0&1\\-1&0\end{bmatrix}\\&=-\mathbf{K}_1-\mathbf{K}_2+\dot{g}_z g_z^{-1}\cdot\mathbf{I}\\\mathbf{K}_p&=-\mathbf{K}_2\mathbf{K}_1+\dot{\mathbf{G}}_1\mathbf{G}_1^{-1}\mathbf{K}_1-\mathbf{G}_1^T\mathbf{G}_1\end{aligned}$$

$$= -\mathbf{K}_2\mathbf{K}_1 + \dot{g}_z \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} g_z^{-1} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{K}_1 - g_z \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} g_z \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$= -\mathbf{K}_2\mathbf{K}_1 + \dot{g}_z g_z^{-1}\mathbf{K}_1 - g_z^2 \cdot \mathbf{I}$$

The error dynamic equation becomes,

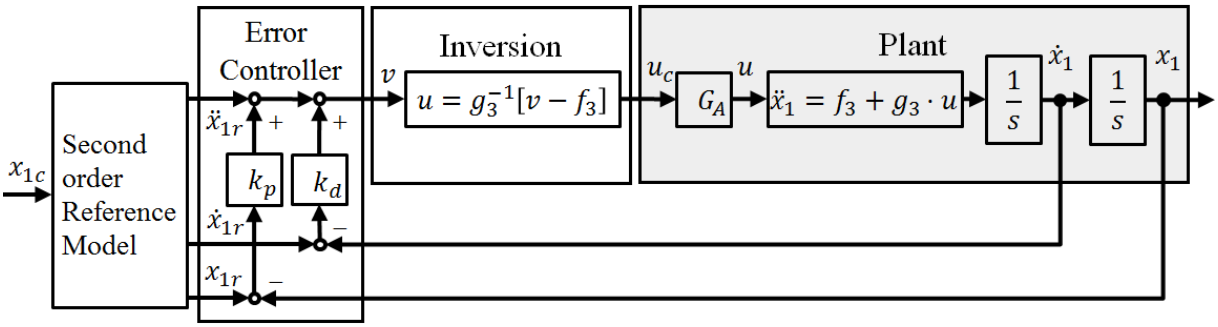
$$\begin{bmatrix} \dot{\mathbf{e}} \\ \ddot{\mathbf{e}} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\mathbf{K}_2\mathbf{K}_1 + \dot{g}_z g_z^{-1}\mathbf{K}_1 - g_z^2 \cdot \mathbf{I} & -\mathbf{K}_1 - \mathbf{K}_2 + \dot{g}_z g_z^{-1} \cdot \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{e} \\ \dot{\mathbf{e}} \end{bmatrix} \tag{5.22}$$

With the above equation, the control performance may only be addressed locally using the linear theory. Near hovering condition, the magnitude of the parameter g_z is almost 9.81. The term g_z^2 adds big influence (ranging from 0~96) to the proportional gains, the resulting relative damping can be rather small depending on the ratio of g_z^2 over the constant gain $\mathbf{K}_2\mathbf{K}_1$. Therefore, oscillations may be observed compared with the first two designs if the same constant gain setting is used. What's more, the numerical influence (0~96) on the proportional gain could have dominate effect with slow actuators, where normal gain is rather small. Fortunately, the quadrotor has rather fast actuators, so no severe stability problem occurs to the quadrotor despite of performance degradation. The simulation and experimental results are shown in the next sections.

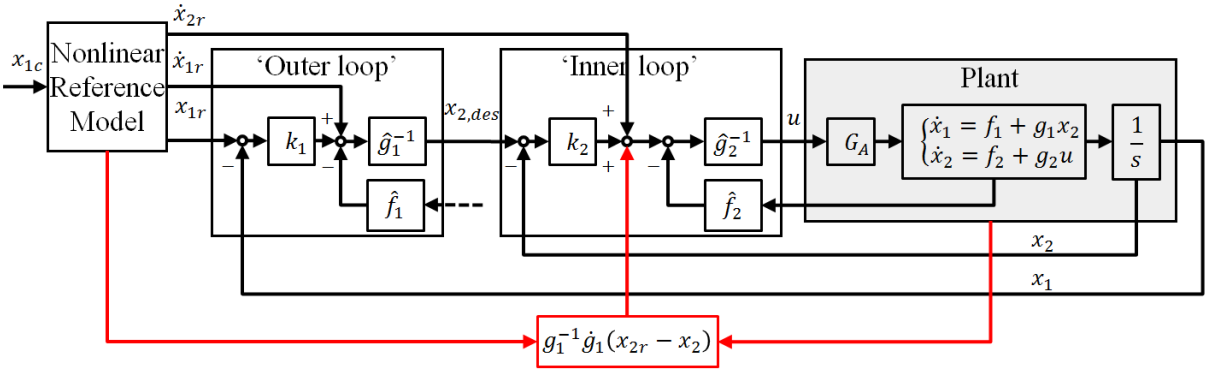
5.3.4. Comparison and Results

Three different non-cascaded designs have been introduced in the above section. As discussed in section 3.3, design a) and c) results the same linear error dynamics as in Eq. (3.21), and their performance are expected to be the same. The design d) has different error dynamics as shown in Eq. (5.22), though the performance is the same for the linear example in section 3. We can see more differences with the nonlinear attitude system. Here the results in simulation and experiments are directly presented while the implementation details are given in Chapter 8.

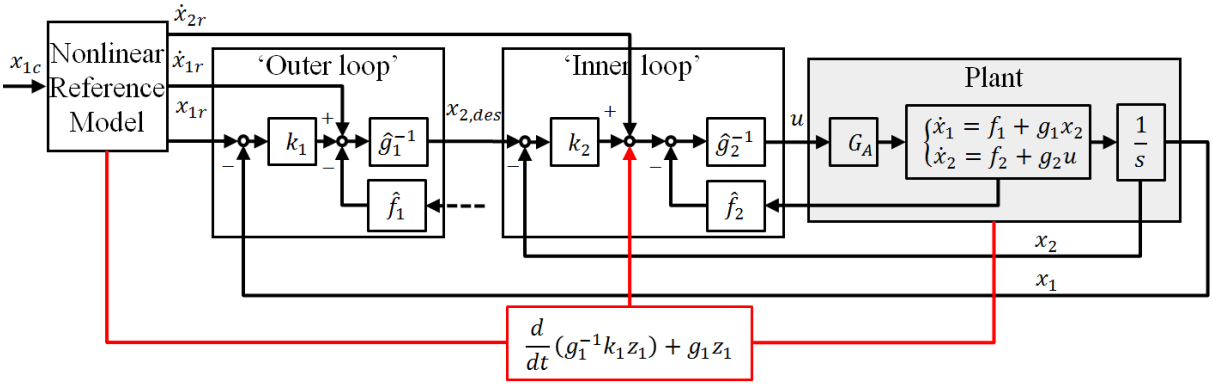
The implementations structures are collected below for better visualizations.



a)



c)



d)

Figure 5.3 The comparison of non-cascaded control designs for the attitude system

Theoretically, there is no doubt that the design a) should give the best performance. In the simulation, the natural frequency $\boldsymbol{\omega_0}$ of the reference signals can be tuned up to 16, beyond which noticeable oscillations can be observed in angular rate, i.e. not enough robustness margin left. This threshold value is used in reference model for all control designs, so that the differences can be observed under the same reference. The design c) could have exactly the same error dynamics as the design a) so as the tracking performance. The tracking performance of both designs in response to a 3-2-1-1 step commands is shown in Figure 5.4

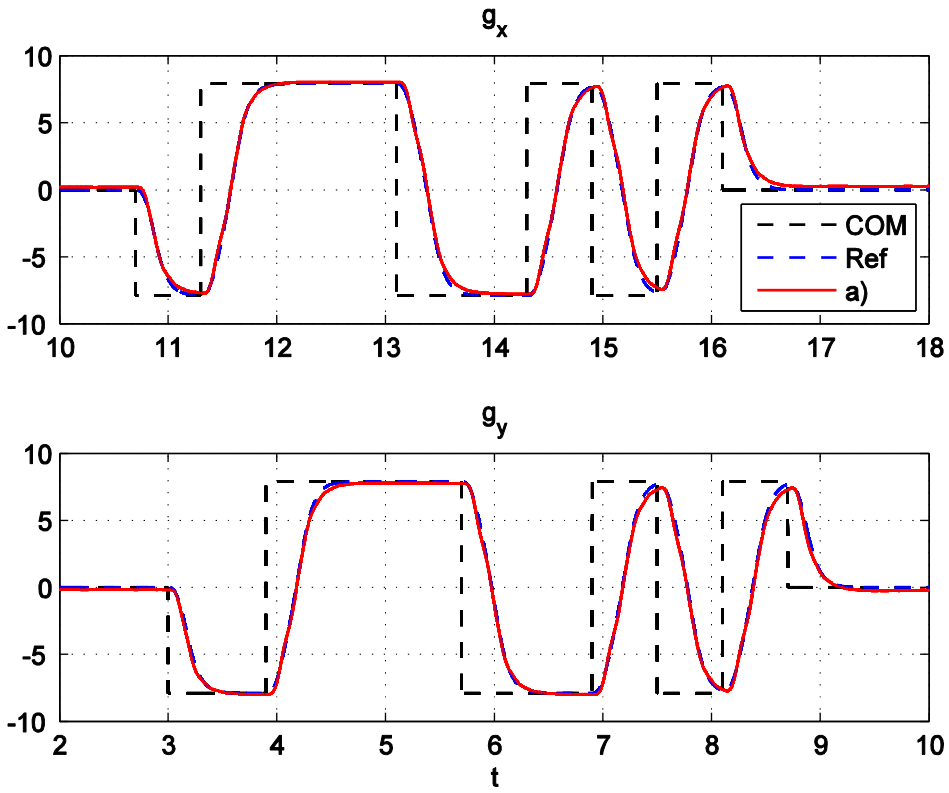


Figure 5.4 Tracking Performance of Design a) in Simulation

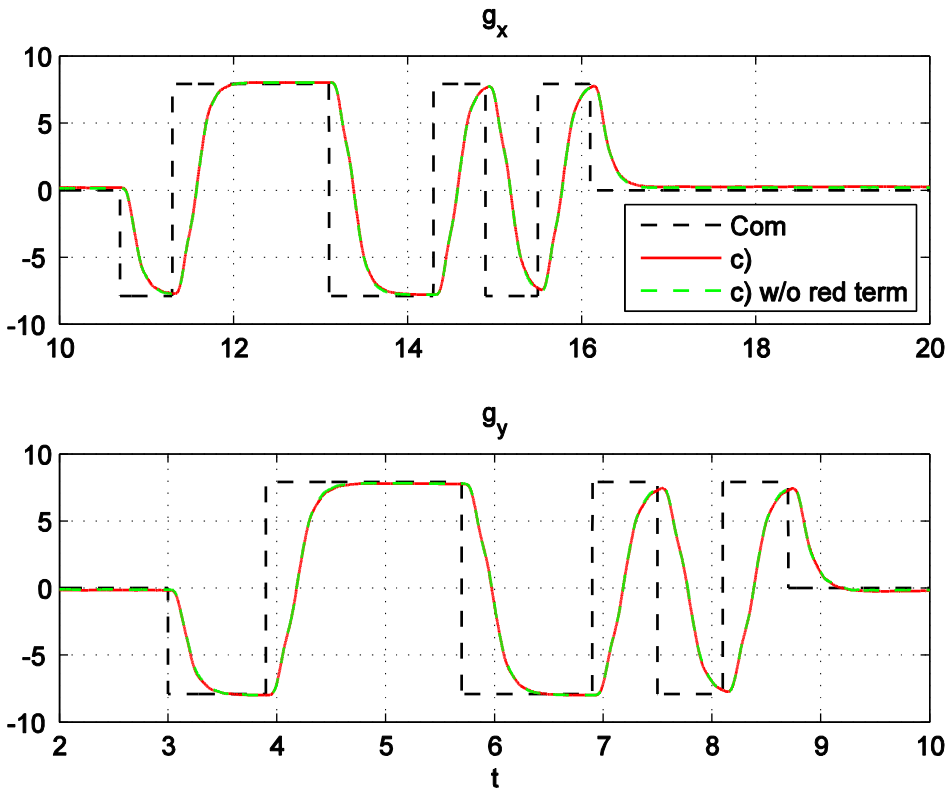


Figure 5.5 Tracking Performance of two variations in Design c) in Simulation

There is a variation of design c), which is the control structure without the modification term (red block in Figure 5.3-c). This gives more design freedom to systems where the modification term is not available, i.e. mainly the derivative term $\dot{\mathbf{G}}_\mathbf{1}$. In the simulation, the performance difference is almost negligible as shown in Figure 5.5. The modification term could render the error dynamics to be linear in theory; however, in actual implementation, the effect is overwhelmed by the sensor errors and parameter uncertainties.

The design d) here is no longer the same as the previous designs. In Figure 5.6, the commands (black dotted line), the tracking performance of design c) (blue line) and design d) (red line) are compared. Since the resulting error dynamics shows a big addition to the proportional feedback gains, oscillations in the attitude can be observed in Figure 5.6 as expected. Of course if the natural frequency (or the feedback gains) is set smaller, then the oscillation could get smaller and even disappear. This means the controller can only follow lower bandwidth reference commands, without any benefit.

The above controllers are also implemented on the Hummingbird for flight tests. The natural frequency ω_0 of the reference model needs to be decreased to 15, so as the feedback gains. In addition, integral gains are turned off for better comparison. The attitude commands are given from the remote control by the pilot. The flight test results verify the analytical and simulation results. The tracking performance of design a) and c) is given in Figure 5.7.

Using the same gains, the design d) produces oscillations from beginning of the flight as shown in Figure 5.8. Reducing the gain setting to natural frequency ω_0 of 13, the oscillations get smaller but it is able to fly. There are still occasional oscillations shown in Figure 5.9.

From the above comparisons and those in chapter 3, the design a) and c) both give the best tracing performance for system of triangular form. The Backstepping design guarantees the stability but does not address the performance. Due to its resulting nonlinear error dynamics, its performance depends on the mathematical model and the gain tuning needs more effort than the NDI designs.

The attitude control system gives good validation on the non-cascaded / analytical control designs. The latter four control designs, which utilize low-pass filters, will be applied and compared on the position control of the quadrotor.

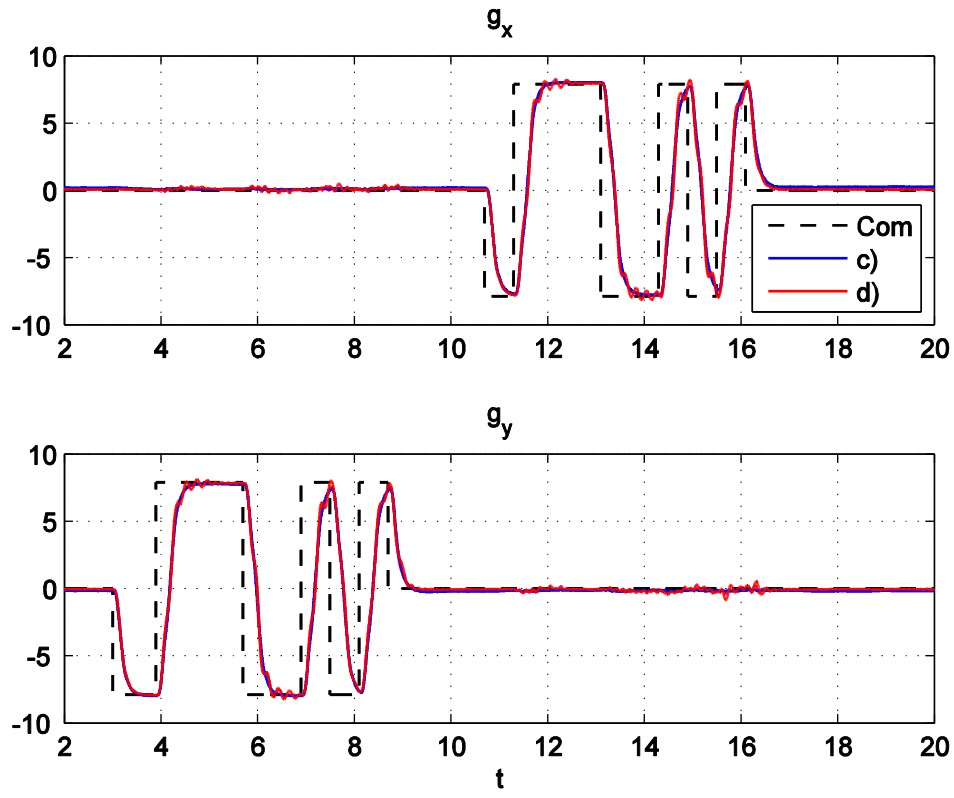


Figure 5.6 Tracking Performance of Design d) in Simulation

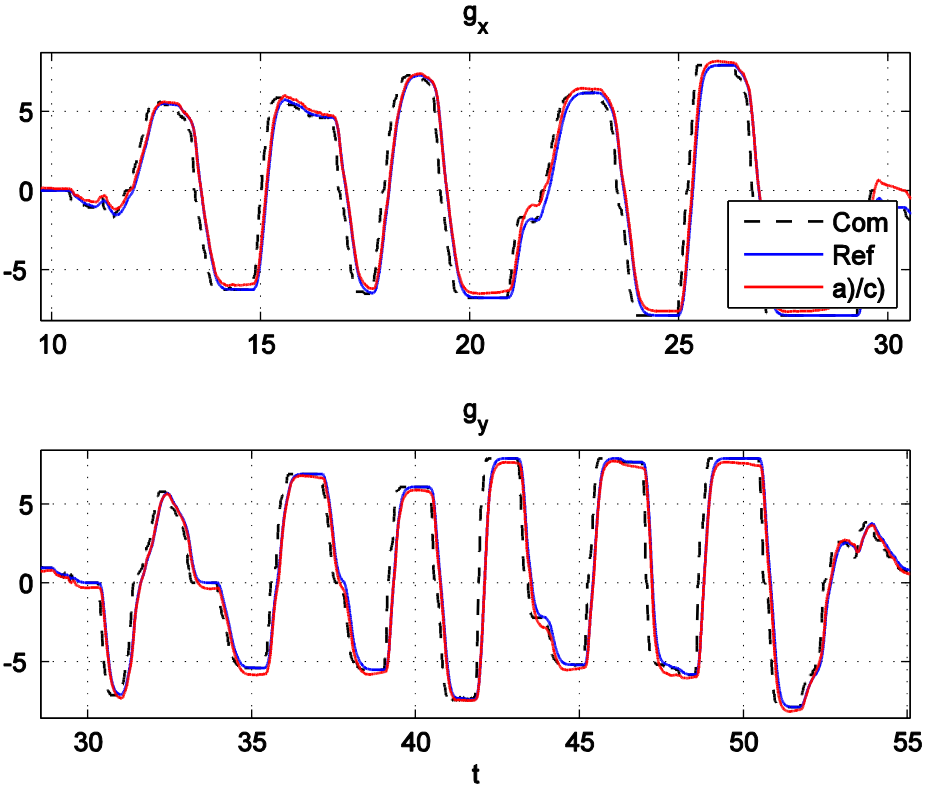


Figure 5.7 Tracking Performance of Design a)/c) in Flight

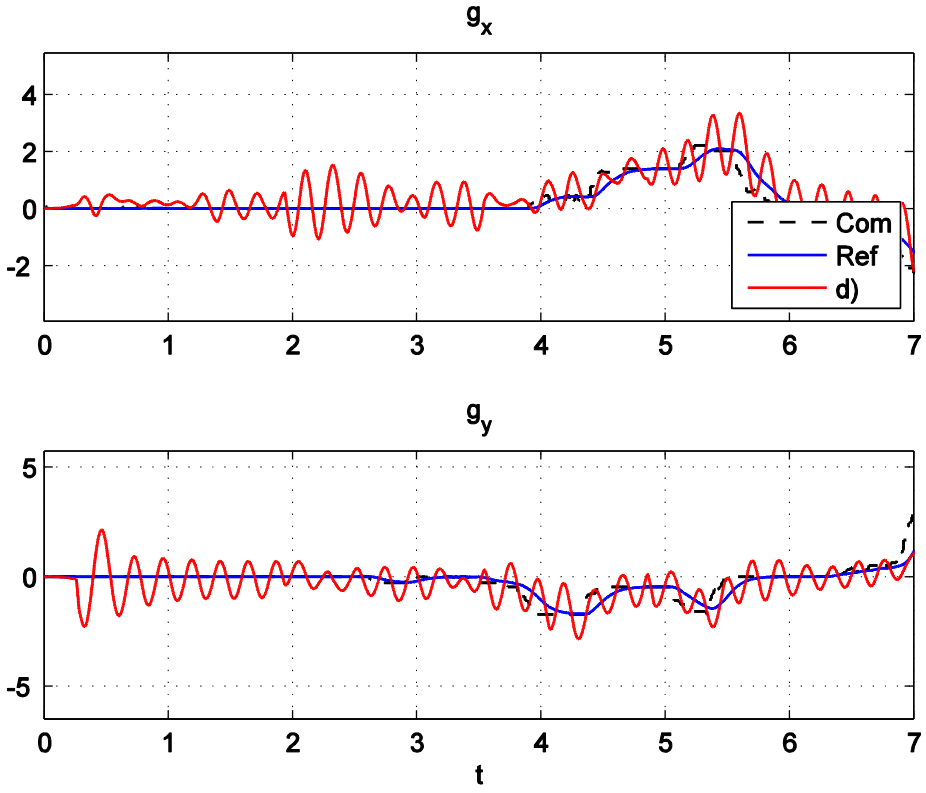


Figure 5.8 Tracking Performance of Design d) at $\omega_0 = 15$ in Flight

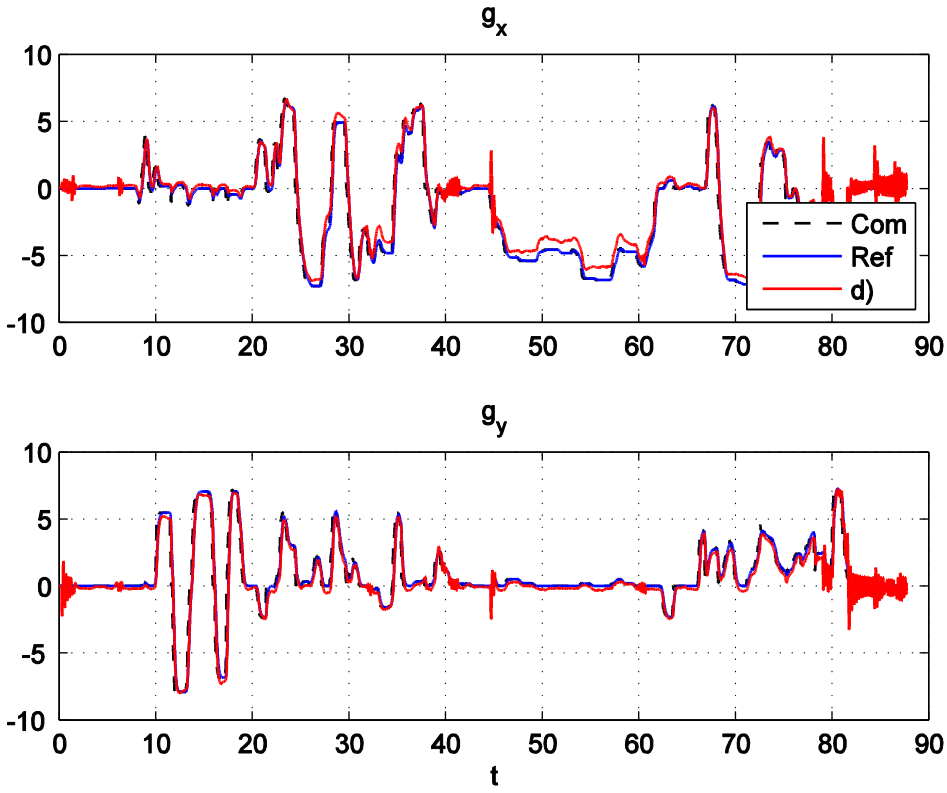


Figure 5.9 Tracking Performance of Design d) at $\omega_0 = 13$ in Flight

6. Position Control Designs

The attitude control takes care of the rotation dynamics. To control the position of the quadrotor, the translational dynamics needs to be considered. As introduced in section 4, the position dynamics has a system order of four (excluding actuator dynamics). The design freedom/complexity increases with the system order. Moreover, the position dynamics is a system of non-triangular form. Hence, the non-cascaded control designs could not be applied, unless with the dynamic extension (will be introduced later). In the following NDI and Backstepping control designs, the control concepts in design a), b), f) and g) are implemented and compared.

In general, there are two considerations in the control designs: on one hand, to improve the overall control bandwidth, not only the feedback gain shall be maximized but also the reference signals shall be utilized in the feedforward control as much as possible; on the other hand, certain stability and robustness has to be guaranteed.

In the first two sections, the NDI and Backstepping position control designs are introduced. In the third section, the reference model for smooth reference position generation is introduced, which is used for all the position controllers.

The comparison results are taken from the simulation and flight tests. Detailed experimental setups are described in Chapter 8.

6.1. NDI Position Control Designs

Three different position controllers are designed using NDI. The first two controllers use the cascaded structure as in design f). The third controller uses the non-cascaded structure as in design a).

6.1.1. RD2 Position Loop + RD2 Attitude Loop (Design b1)

The most intuitive control structure is to divide the control loops by the dynamic nature: a position output loop of relative degree 2 governing the translational motion and an attitude inner loop of relative degree 2 governing the rotational motion. This structure also has good robustness from the navigation point of view. There is a clean separation of sensors between the loops: the outer loop uses the vision sensor or GPS measurement (which is less reliable), while the inner loop is independent of those and only uses the onboard IMU. In case of any failure in the outer loop due to the vision sensor or GPS, the stability of inner loop attitude control is not affected.

The choice of control loop interface could result different implementations, though the tracking performance of the two designs below can be rather similar. In the first design, the interface between the outer loop and inner loop is the Euler angles while the second design uses the newly defined parameter $\mathbf{g}_{xy}$ as interface.

6.1.1.1. Euler angle approach

Normally, the control state vector for the outer loop is the position and velocity defined in the World frame as shown in Eq. (4.57).

$$\mathbf{x}_{outer} = \begin{bmatrix} (\vec{\mathbf{r}}_k^G)_W \\ (\vec{\mathbf{v}}_k^G)_W^W \end{bmatrix} = \begin{bmatrix} \vec{\mathbf{r}} \\ \vec{\mathbf{v}} \end{bmatrix} \tag{6.1}$$

The control state for the inner loop is the Euler angle and Euler angle derivative,

$$\mathbf{x}_{inner} = \begin{bmatrix} \vec{\boldsymbol{\Psi}} \\ \dot{\vec{\boldsymbol{\Psi}}} \end{bmatrix} \tag{6.2}$$

where,

$$\vec{\boldsymbol{\Psi}} = \begin{bmatrix} \Phi \\ \Theta \\ \Psi \end{bmatrix}$$

The inner loop control design is the same as the design in section 5.1. The outer loop design, which is also published in [5], is explained here.

After two differentiations of the output vector $(\vec{\mathbf{r}}_k^G)_W$, the 2nd derivative of the position vector can be mapped to the inner loop commands and total thrust using the Newton's Second Law as already stated in Eq. (4.53). The aerodynamic drag and disturbances are ignored in the model assumption.

$$\left(\ddot{\vec{\mathbf{r}}}_k^G\right)_W^{WW} = \vec{\mathbf{a}} \approx -\frac{T}{m} \cdot \begin{bmatrix} \cos\Psi \sin\Theta \cos\Phi + \sin\Psi \sin\Phi \\ \sin\Psi \sin\Theta \cos\Phi - \cos\Psi \sin\Phi \\ \cos\Theta \cos\Phi \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{6.3}$$

We can see that, as the heading angle also appears, the inversion is not so straightforward. In classical helicopter control design, a rotated NED frame $\bar{O}$ was introduced, which is still leveled but rotated by the heading angle Ψ only, so that the x-axis of the coordinate also points to the Body-fixed x-axis and in this way the heading angle can be decoupled from the translational dynamics.

The additional assumption for the derivation is that the rotated NED frame can be used as inertia frame, where the Newton's second law can be applied. This can only be fulfilled if the heading dynamics is much slower than the position dynamics, i.e. the yaw rate is kept small so that the Coriolis terms can be ignored. In the rotated NED frame, the dynamic equation for the acceleration in $\bar{O}$ frame is derived below.

$$\left(\ddot{\vec{\mathbf{r}}}_k^G\right)_{\bar{O}}^{\bar{O}\bar{O}} = \vec{\mathbf{a}}_{\bar{O}} \approx -\frac{T}{m} \cdot \begin{bmatrix} \sin\Theta \cos\Phi \\ -\sin\Phi \\ \cos\Theta \cos\Phi \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{6.4}$$

In the control design, the desired acceleration $\vec{\mathbf{a}}_{des}$ in W frame (it can also be defined in O frame depend on the heading measurement) is the pseudo control $\boldsymbol{\nu}$, which can be calculated by the linear feedback control and with appropriate reference signals. For the second order system, a second order reference model and a PD control is sufficient,

$$\boldsymbol{\nu} = \vec{\mathbf{a}}_{des} = \vec{\mathbf{a}}_r + \mathbf{K}_d(\vec{\mathbf{v}}_r - \vec{\mathbf{v}}) + \mathbf{K}_p(\vec{\mathbf{r}}_r - \vec{\mathbf{r}}) \tag{6.5}$$

The inner loop commands and total thrust can be then solved by inverting the dynamic equation given in Eq. (6.4), however the pseudo control needs to be rotated into the $\bar{O}$ frame. Rewrite it for the control equation and rearrange it,

$$\begin{bmatrix} a_{x,\nu} \\ a_{y,\nu} \\ a_{z,\nu} \end{bmatrix}_{\bar{O}} = \mathbf{M}_{\bar{O}W}\boldsymbol{\nu} \approx -\frac{T}{m} \cdot \begin{bmatrix} \sin\Theta \cos\Phi \\ -\sin\Phi \\ \cos\Theta \cos\Phi \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{6.6}$$

$$\begin{bmatrix} a_{x,\nu} \\ a_{y,\nu} \\ a_{z,\nu} \end{bmatrix}_{\bar{O}} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \approx -\frac{T}{m} \cdot \begin{bmatrix} \sin\Theta \cos\Phi \\ -\sin\Phi \\ \cos\Theta \cos\Phi \end{bmatrix} \tag{6.7}$$

First, the desired total thrust can be solved by taking the norm of both sides,

$$T_d \approx m\sqrt{a_{x,\nu}^2 + a_{y,\nu}^2 + \left(a_{z,\nu} - g\right)^2} \tag{6.8}$$

Second, the desired pitch and bank angle can be solved,

$$\Theta_{\mathrm{d}} \approx \arctan\left(\frac{a_{x,\nu}}{a_{z,\nu} - \mathrm{g}}\right) \tag{6.9}$$

$$\Phi_{\mathrm{d}} \approx \arcsin\left(\frac{m \cdot a_{y,v}}{T_d}\right) = \arcsin\left(\frac{a_{y,v}}{\sqrt{a_{x,v}^2 + a_{y,v}^2 + \left(a_{z,v} - g\right)^2}}\right) \tag{6.10}$$

Together with the azimuth angle commands, the inner loop commands are complete.

The overall control structure is shown in Figure 6.1. The data fusion block consists of an AHRS and a position observer.

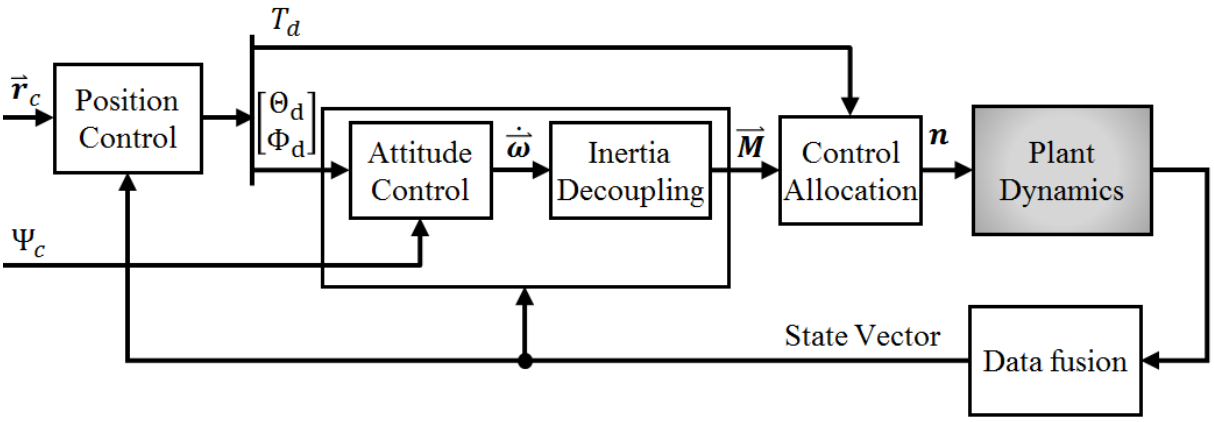


Figure 6.1 Conventional Position Control Structure with Euler Angles

6.1.1.2. Gravitational vector approach

In this approach, the new parameter $\mathbf{g}_{xy}$ is used for the inner loop control state, so the inner loop controller is the same as in section 5.3.1 or 5.3.2. The difference lies in the dynamic inversion from the outer loop pseudo control $\boldsymbol{\nu}$ to the inner loop commands.

Rewrite the translational dynamic equation in Eq. (4.56) for the control design,

$$\boldsymbol{\nu} = \begin{bmatrix} a_{x,v} \\ a_{y,v} \\ a_{z,v} \end{bmatrix}_W = \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_{x,d} \\ g_{y,d} \\ -T_d/m \end{bmatrix}_B \right) \tag{6.11}$$

Solve the control variable by inverting the equation above,

$$\mathbf{M}_{BW} \cdot \begin{bmatrix} a_{x,v} \\ a_{y,v} \\ a_{z,v} \end{bmatrix}_W = \mathbf{M}_{BW} \cdot \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_{x,d} \\ g_{y,d} \\ -T_d/m \end{bmatrix}_B \right) \tag{6.12}$$

$$\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_{x,d} \\ g_{y,d} \\ -T_d/m \end{bmatrix}_B = \mathbf{M}_{BW} \cdot \begin{bmatrix} a_{x,v} \\ a_{y,v} \\ a_{z,v} \end{bmatrix}_W \tag{6.13}$$

$$\begin{bmatrix} g_{x,d} \\ g_{y,d} \\ -T_d/m \end{bmatrix}_B = \mathbf{M}_{BW} \cdot \begin{bmatrix} a_{x,v} \\ a_{y,v} \\ a_{z,v} \end{bmatrix}_W - \begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B \tag{6.14}$$

The inversion equation is much simpler than the conventional inversion of Euler angles. The heading dynamic coupling with the translational dynamics is decoupled by the rotation matrix $\mathbf{M}_{BW}$.

Moreover, the accelerator measurement in the Body-fixed x-y- axis can be used. Besides measurement and structural noise, the specific force f_x and f_y are the normalized aerodynamic disturbance and drag. For instance, during fast forward flight with constant speed, to work against the constant aerodynamic drag so a constant pitch/bank angle is expected to compensate that. In the conventional design, the desired pitch/bank angle is driven by the reference acceleration and the position error feedback. So a constant pitch/bank angle command can be generated only by a constant position error. The compensation is slow and induces steady state errors. In the new design, the drag in the Body-fixed x-y-components is measured and used in the inversion equation, so a constant $g_{x,d}$ or $g_{y,d}$ will be commanded without additional error terms in position and velocity. As shown in the specific force equilibrium diagram in Figure 6.2, the specific drag force f_x in red in the Body-fixed x-axis can be used in the controller and the corresponding attitude $g_{x,d}$ can be commanded to the inner loop.

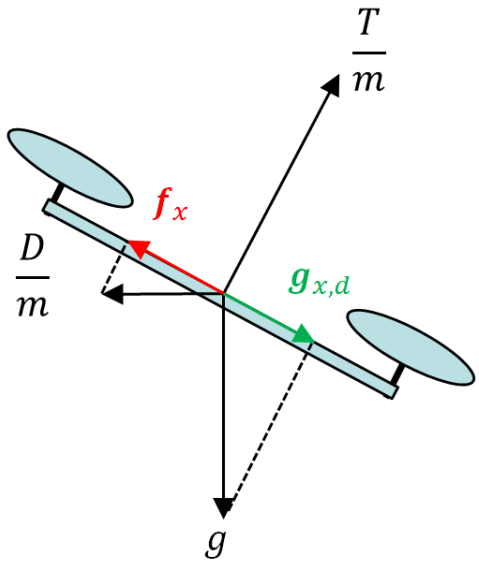


Figure 6.2 Specific force diagram in forward flight with constant speed

The overall position control structure is shown in Figure 6.3.

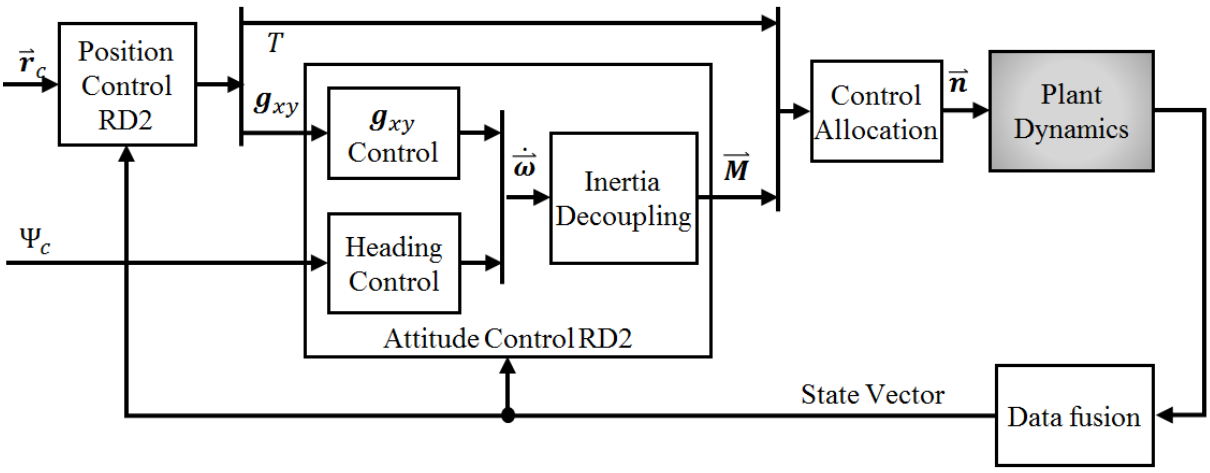


Figure 6.3 Conventional Position Control Structure with New Parameterization

6.1.2. RD3 Position Loop + RD1 Rate Loop (Design b2)

The above designed control structure is very robust in the physics as well as in the sensor separation. The inner loop state feedback is the attitude and angular rate, which are quite accurate and robust; the outer loop state feedback is the position and velocity, which are not so accurate and robust. However, due to the RD2 inner loop, the reference signals can only be used up to the second derivative of the position vector, i.e. reference acceleration.

In this new control structure design, the position loop is extended to RD 3, which enables the feedforward of the 3rd derivative of the position vector. The linear state vector for the RD 3 position loop is,

$$\mathbf{X} = [\vec{\mathbf{r}} \quad \vec{\mathbf{v}} \quad \vec{\mathbf{a}}]^T \tag{6.15}$$

The differential equation can be derived starting from the translational dynamic equation in Eq. (4.53),

$$\left(\ddot{\vec{\mathbf{r}}}\right)_W^{WW} \approx \mathbf{M}_{WB} \cdot \begin{bmatrix} 0 \\ 0 \\ -T/m \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

Taking one more differentiation,

$$\left(\dddot{\vec{\mathbf{r}}}\right)_W^{WWW} \approx \dot{\mathbf{M}}_{WB} \cdot \begin{bmatrix} 0 \\ 0 \\ -\frac{T}{m} \end{bmatrix} + \mathbf{M}_{WB} \cdot \begin{bmatrix} 0 \\ 0 \\ -\frac{\dot{T}}{m} \end{bmatrix}$$

$$\approx \mathbf{M}_{WB} \cdot \boldsymbol{\Omega}_{WB}^{BB} \cdot \begin{bmatrix} 0 \\ 0 \\ -T/m \end{bmatrix} + \mathbf{M}_{WB} \cdot \begin{bmatrix} 0 \\ 0 \\ -\dot{T}/m \end{bmatrix}$$

where $\boldsymbol{\Omega}_{WB}^{BB}$ is the skew-symmetric matrix of the angular rate vector,

$$\boldsymbol{\Omega}_{WB}^{BB} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \tag{6.16}$$

Hence, the dynamic equation can be rearrange as follows,

$$\left(\dddot{\mathbf{r}}\right)_W^{WWW} \approx \mathbf{M}_{WB} \cdot \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -\frac{T}{m} \end{bmatrix} + \mathbf{M}_{WB} \cdot \begin{bmatrix} 0 \\ 0 \\ -\frac{\dot{T}}{m} \end{bmatrix} \tag{6.17}$$

$$\left(\dddot{\mathbf{r}}\right)_W^{WWW} \approx \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -qT \\ pT \\ -\dot{T} \end{bmatrix} \tag{6.18}$$

The final expression of the differential equation is,

$$\left(\dddot{\mathbf{r}}\right)_W^{WWW} \approx \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} 0 & -T & 0 \\ T & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} p \\ q \\ \dot{T} \end{bmatrix} \tag{6.19}$$

A rather simple dynamic equation is obtained from the differentiation and the pitch and roll rates directly appear in the equation, which can be connected to the rate control inner loop. The only problem here is that, the total thrust, which can be originally obtained by the 2nd derivative of the position, is now further differentiated to thrust derivative. A dynamic extension was proposed in [15] to overcome this kind of problems. In this case, the thrust derivative is assumed the virtual command and the dynamic extension is an integrator to generate the thrust command by integrating the thrust derivative.

With this modification, a proper RD 3 NDI controller can be designed in the outer loop, while the inner loop is RD 1 angular rate control. The advantage over the RD 2 structure is that the reference signal can be used up to the 3rd derivative of the position vector, i.e. the acceleration derivative can be feedforward in the error controller, hence more model based information can be used and higher control bandwidth can be reached.

Given the derived differential equation, i.e. math model for control design, the NDI equation can be obtained by inverting Eq. (6.19). The pseudo control $\boldsymbol{\nu}$ is the position third derivative $\left(\dddot{\mathbf{r}}\right)_W^{WWW}$ for the RD 3 outer loop.

$$\begin{bmatrix} p \\ q \\ \dot{T} \end{bmatrix}_d = \begin{bmatrix} 0 & T^{-1} & 0 \\ -T^{-1} & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \mathbf{M}_{BW} m \cdot \boldsymbol{\nu} \tag{6.20}$$

The nonlinear position dynamics plus the dynamic inversion together can be regarded as a linear system with three integrators. The linear reference model and linear controller for the equivalent linear third order system can be easily designed the transformed system. Similar to the second order system, the third order linear error controller is as follows,

$$\mathbf{v} = \dot{\vec{\mathbf{a}}}_d = \dot{\vec{\mathbf{a}}}_r + \mathbf{K}_2(\vec{\mathbf{a}}_r - \vec{\mathbf{a}}) + \mathbf{K}_1(\vec{\mathbf{v}}_r - \vec{\mathbf{v}}) + \mathbf{K}_0(\vec{\mathbf{r}}_r - \vec{\mathbf{r}}) \tag{6.21}$$

The gains can be designed by the pole placement method and finally tuned in flight test. The rate inner loop is quite simple and the standard NDI structure applies. The reference model, named inner loop reference model, takes the desired angular rate from the outer loop and heading control,

The inner loop reference model is,

$$\dot{\vec{\omega}}_f = \mathbf{K_f}(\vec{\omega}_d - \vec{\omega}_f) \tag{6.22}$$

And the error dynamics has the feed-forwarded $\dot{\vec{\omega}}_f$ and a proportional feedback control.

$$\dot{\vec{\omega}}_d = \dot{\vec{\omega}}_f + \mathbf{K}_\omega(\vec{\omega}_f - \vec{\omega}) \tag{6.23}$$

If the inner loop reference model has the same time constant as the error dynamics, we have simply the feedback control.

$$\dot{\vec{\omega}}_d = \mathbf{K}_\omega(\vec{\omega}_d - \vec{\omega}) \tag{6.24}$$

Desired moment can be computed afterward by Eq. (5.7).

The overall control structure can be designed as in Figure 6.4.

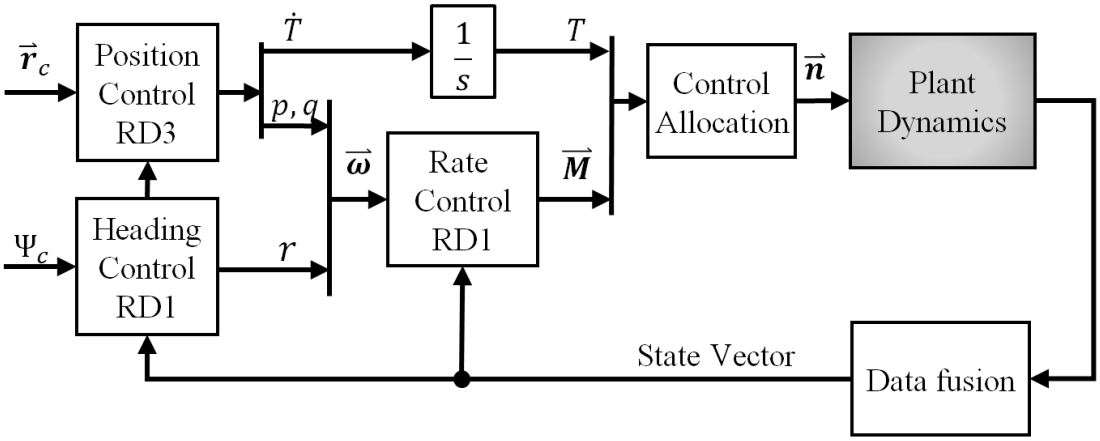


Figure 6.4 Novel RD3 Position Control Structure

6.1.3. Non-cascaded RD4 Position Control (Design a)

After extending the thrust dynamic to form a RD 3 position loop, we may even extend it further to RD 4 position loop. The differential equation for the 4th order position derivative can be obtained by further differentiating Eq. (6.19),

$$\left(\dddot{\vec{\mathbf{r}}}_k^G\right)_W^{WWW} = \left(\dddot{\vec{\mathbf{a}}}_k^G\right)_W^{WWW} = \dot{\mathbf{M}}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -qT \\ pT \\ -\dot{T} \end{bmatrix} + \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -qT \\ pT \\ -\dot{T} \end{bmatrix}'$$

$$= \mathbf{M}_{WB} \cdot \boldsymbol{\Omega}_{WB}^{BB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -qT \\ pT \\ -\dot{T} \end{bmatrix} + \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -\dot{q}T - q\dot{T} \\ \dot{p}T + p\dot{T} \\ -\ddot{T} \end{bmatrix}$$

$$= \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{bmatrix} -qT \\ pT \\ -\dot{T} \end{bmatrix} + \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -\dot{q}T - q\dot{T} \\ \dot{p}T + p\dot{T} \\ -\ddot{T} \end{bmatrix}$$

$$= \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -rpT - q\dot{T} \\ -rqT + p\dot{T} \\ q^2T + p^2T \end{bmatrix} + \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -\dot{q}T - q\dot{T} \\ \dot{p}T + p\dot{T} \\ -\ddot{T} \end{bmatrix}$$

$$= \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -rpT - 2q\dot{T} \\ -rqT + 2p\dot{T} \\ q^2T + p^2T \end{bmatrix} + \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} -\dot{q}T \\ \dot{p}T \\ -\ddot{T} \end{bmatrix}$$

The final expression of the differential equation can be written as,

$$\left(\ddot{\vec{\mathbf{a}}}_k^G\right)_W^{WWW} = \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \left(\begin{bmatrix} -rpT - 2q\dot{T} \\ -rqT + 2p\dot{T} \\ q^2T + p^2T \end{bmatrix} + \begin{bmatrix} 0 & -T & 0 \\ T & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \ddot{T} \end{bmatrix} \right) \tag{6.25}$$

As we can see, not only the angular acceleration appears explicitly, but also the thrust has been differentiated twice. That is the analytical reason why we need to extend the thrust dynamics to its second derivative. The pseudo control is the translational acceleration second derivative while the virtual controls are the pitch & roll accelerations and thrust second derivative.

With the derived differential equation, the NDI equation can be derived from the pseudo control to the virtual control,

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \ddot{T} \end{bmatrix}_d = \begin{bmatrix} 0 & T^{-1} & 0 \\ -T^{-1} & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \left(\mathbf{M}_{BW} m \cdot \boldsymbol{\nu} - \begin{bmatrix} -Trp - 2\dot{T}q \\ -Trq + 2\dot{T}p \\ Tp^2 + Tq^2 \end{bmatrix} \right) \tag{6.26}$$

The desired angular accelerations can be used to compute the desired moment and the thrust second derivative needs two integrators, i.e. dynamic extension, to compute the desired thrust.

The linear error controller for the transformed plant is fourth order. The control equation is,

$$\boldsymbol{\nu} = \ddot{\vec{\mathbf{a}}}_d = \ddot{\vec{\mathbf{a}}}_r + \mathbf{K}_3\left(\dot{\vec{\mathbf{a}}}_r - \dot{\vec{\mathbf{a}}}\right) + \mathbf{K}_2(\vec{\mathbf{a}}_r - \vec{\mathbf{a}}) + \mathbf{K}_1(\vec{\mathbf{v}}_r - \vec{\mathbf{v}}) + \mathbf{K}_0(\vec{\mathbf{r}}_r - \vec{\mathbf{r}}) \tag{6.27}$$

Together with RD 2 heading control, we have the following control structure,

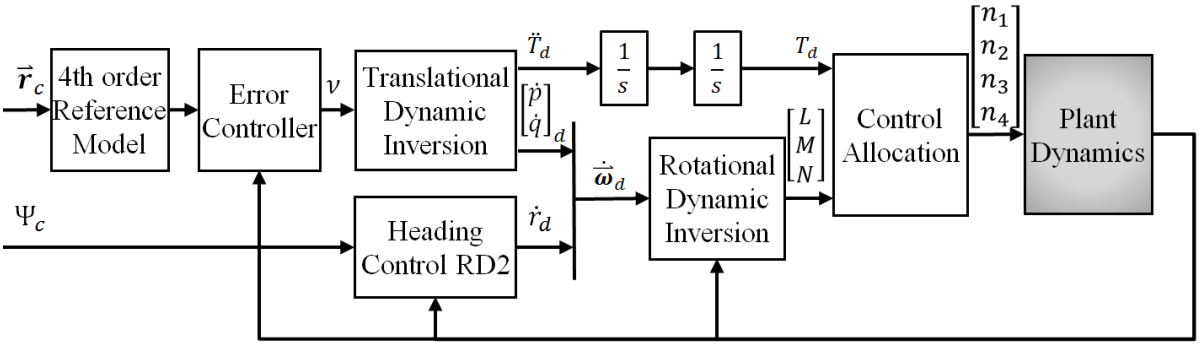


Figure 6.5 Non-cascaded RD4 Position Control Structure

In total, there are three different control structures, ‘RD2 + RD2’, ‘RD3 + RD1’, and ‘RD4’. Their position control performance is compared in later section, as well as three Backstepping position control designs derived in next section.

6.1.4. Heading Control

From the position dynamic equation, be it RD2, RD3 or RD4, the common thing is that the position control is not directly affected by the heading dynamics, just the state information is needed in the inversion equation. Therefore the heading control can be considered secondary goal compared to the position control.

The focus of this heading control is to stabilize the heading in a robust yet efficient way so that the actuation level can be kept low without interference with the position control. The control bandwidth and accuracy can be compromised. The control power shall be reserved for the position or pitch/roll control but not occupied by the heading control.

A possible solution for the heading control is to design a rate-control attitude-hold controller, i.e. PI yaw rate control. The integral gain will control the heading angle indirectly with minimum impact on the other two axes. The heading drift due to sensor noise and bias is also negligible in one-battery flight time (the gyro drift rate is 400°/hour). Recall the PI yaw-rate control law given in Eq. (5.13),

$$\dot{r}_d = \dot{r}_r + k_p(r_r - r) + k_I \int (r_r - r)\, dt \tag{6.28}$$

If the heading needs to be controlled more precisely, both NDI and linear control should work fine. For the NDI case, the dynamic equation in Eq. (4.48) needs to be inverted, and there is a singularity at bank angle of 90°. For instant, the RD1 inversion equation from the azimuth angle derivative to the angular rate is shown below,

$$r_d = \frac{\cos\Theta}{\cos\Phi}\dot{\Psi}_v - \tan\Phi\, q \tag{6.29}$$

$$r_d = \frac{1}{g_z}\left(g\dot{\Psi}_v - g_y q\right) \tag{6.30}$$

Alternatively, normal linear PD control is also sufficient the heading control as the heading dynamics is much slower than the pitch and roll motions for the quadrotor configuration. The control law is given below,

$$\dot{r}_d = k_r\left[\underbrace{k_\Psi(\Psi_c - \Psi)}_{r_d} - r\right] \tag{6.31}$$

6.2. Backstepping Position Control Designs

As illustrated earlier, for the position dynamics of the quadrotors, the Backstepping method could not be applied directly. Cascaded structure or command filter can be augmented to apply the Backstepping method. The cascaded Backstepping designs, which could result the same control law as the NDI, will not be introduced again. The design f) Backstepping using command filter is used.

For high order system, there are also many variations in the design. Each system order will generate a Backstepping virtual control and its derivative can be solved either numerically or analytically. The objective here is to find the analytical solution as far as possible, which means the more reference signals can be feedforward to the plant. In some cases, the analytical solution could be difficult to be solved, which are highly dependent on the mathematical model to be used. A suitable math model could ease the analytical derivation while if the wrong math model is used, the virtual control derivative may not be solvable analytically.

In this section, two Backstepping designs with command filter are introduced. The first one uses the novel parameterization in the math model. The second one uses the NDI-RD3 math model.

6.2.1. Backstepping with the novel parameterization (Design g1)

In previous Backstepping position designs [18] [19] [20], the analytical solution only went to the second step, i.e. the attitude level. Then command filter has to be applied to estimate the attitude derivatives. The reason is that the analytical solutions encounter difficulties with the Euler angle and thrust coupling.

Using the math model with the gravity parameterization introduced in section 4.3.4, the analytical solution can be solved to the angular rate. So that the reference feedforward signals can also be used in the angular rate level, which give additional model information into the controller and thus increase the control bandwidth, similarly with the NDI designs.

The following state vectors and control inputs are defined for the position model,

$$\mathbf{x}_1 = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_W \quad \mathbf{x}_2 = \begin{bmatrix} u \\ v \\ w \end{bmatrix}_W^W \quad \mathbf{x}_3 = \begin{bmatrix} g_x \\ g_y \end{bmatrix}_B \quad \mathbf{x}_4 = \begin{bmatrix} p \\ q \\ r \end{bmatrix}_B \quad u_1 = -\frac{T}{m} \quad \mathbf{u}_2 = \begin{bmatrix} L \\ M \\ N \end{bmatrix}_B \tag{6.32}$$

The math equations, collected from section 4.3, can be rewritten in the following form for better clarifications,

$$\begin{aligned} \dot{\mathbf{x}}_1 &= \mathbf{f}_1 + \mathbf{G}_1 \mathbf{x}_2 \\ \dot{\mathbf{x}}_2 &= \mathbf{f}_2 + \mathbf{G}_2 \begin{bmatrix} \mathbf{x}_3 \\ u_1 \end{bmatrix} \\ \dot{\mathbf{x}}_3 &= \mathbf{f}_3 + \mathbf{G}_3 \mathbf{x}_4(1{:}2) \\ \dot{\mathbf{x}}_4 &= \mathbf{f}_4 + \mathbf{G}_4 \mathbf{u}_2 \end{aligned} \tag{6.33}$$

Where the nonlinear functions $\mathbf{f}_i$ are

$$\begin{cases} \mathbf{f}_1 = [0 \quad 0 \quad 0]^T \\ \mathbf{f}_2 = \mathbf{M}_{WB} \cdot [f_x \quad f_y \quad g_z]_B^T \\ \mathbf{f}_3 = [g_y \quad -g_x]^T \cdot r \\ \mathbf{f}_4 = -(\mathbf{I}^G)_{BB}^{-1} \cdot (\vec{\boldsymbol{\omega}}^{WB})_B \times \{(\mathbf{I}^G)_{BB} \cdot (\vec{\boldsymbol{\omega}}^{WB})_B\} \end{cases} \tag{6.34}$$

The input matrices $\mathbf{G}_i$ and their inverse are,

$$\begin{cases} \mathbf{G}_1 = \mathbf{I}_3 \\ \mathbf{G}_2 = \mathbf{M}_{WB} \\ \mathbf{G}_3 = g_z \cdot \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \\ \mathbf{G}_4 = (\mathbf{I}^G)_{BB}^{-1} \end{cases} \& \begin{cases} \mathbf{G}_1^{-1} = \mathbf{I}_3 \\ \mathbf{G}_2^{-1} = \mathbf{M}_{BW} \\ \mathbf{G}_3^{-1} = g_z^{-1} \cdot \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \\ \mathbf{G}_4^{-1} = (\mathbf{I}^G)_{BB} \end{cases} \tag{6.35}$$

It can be already noticed that there is a singularity at $g_z = 0$, which is when the quadrotor is at pitch/bank angle of 90°. It needs to be bounded away from zero in implementation for this method. This singularity would also occur if the Euler angles are used.

The recursive derivation structure of the Backstepping position control is shown in Figure 6.6 as follows,

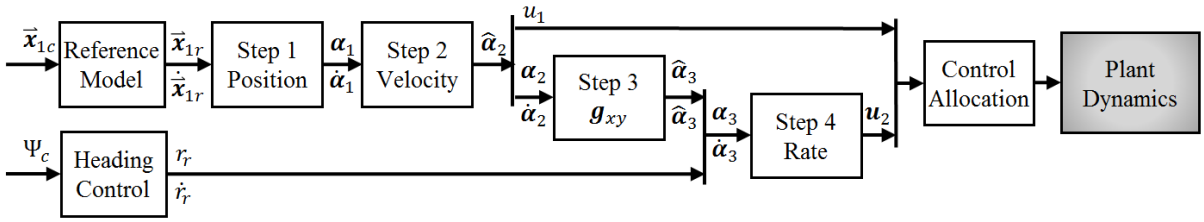


Figure 6.6 Derivation Structure of Backstepping with the New Parameterization

Step 1 Position Control

Consider the following virtual system for the position propagation,

$$\dot{\mathbf{x}}_1 = \boldsymbol{\alpha}_1 \tag{6.36}$$

Define the tracking error for the position and a Lyapunov function candidate,

$$\begin{aligned} \mathbf{z}_1 &= \mathbf{x}_{1r} - \boldsymbol{x}_1 \\ V_1 &= \frac{1}{2}\mathbf{z}_1^T\mathbf{z}_1 \end{aligned} \tag{6.37}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\boldsymbol{\alpha}_1$,

$$\begin{aligned} \dot{V}_1 &= \mathbf{z}_1^T\dot{\mathbf{z}}_1 = \mathbf{z}_1^T(\dot{\mathbf{x}}_{1r} - \boldsymbol{\alpha}_1) \\ \boldsymbol{\alpha}_1 &= \dot{\mathbf{x}}_{1r} + \mathbf{A}_1\boldsymbol{z}_1 \end{aligned} \tag{6.38}$$

Where $\mathbf{A}_1$ is a positive definite matrix and the Lyapunov function derivative can be rendered negative semi-definite.

$$\dot{V}_1 = -\mathbf{z}_1^T\mathbf{A}_1\mathbf{z}_1 \leq 0 \tag{6.39}$$

Step 2 Velocity Control

Consider the following virtual system for the translational dynamics,

$$\dot{\mathbf{x}}_2 = \mathbf{f}_2 + \mathbf{G}_2\hat{\boldsymbol{\alpha}}_2 \tag{6.40}$$

Define the tracking error $\mathbf{z}_2$ for the velocity,

$$\mathbf{z}_2 = \boldsymbol{\alpha}_1 - \mathbf{x}_2 \tag{6.41}$$

Due to this error, the dynamic equation in step 1 is no longer valid and we need to recover the true system dynamic equation in step 1,

$$\dot{\mathbf{x}}_1 = \boldsymbol{\alpha}_1 - \mathbf{z}_2 \tag{6.42}$$

The augmented Lyapunov function candidate is,

$$V_2 = \frac{1}{2}\mathbf{z}_1^T\mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T\mathbf{z}_2 \tag{6.43}$$

Compute the position tracking error derivative,

$$\dot{\mathbf{z}}_1 = \dot{\mathbf{x}}_{1r} - \dot{\mathbf{x}}_1 = \dot{\mathbf{x}}_{1r} - \boldsymbol{\alpha}_1 + \mathbf{z}_2 = -\mathbf{A}_1\mathbf{z}_1 + \mathbf{z}_2 \tag{6.44}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\hat{\boldsymbol{\alpha}}_2$:

$$\dot{V}_1 = \mathbf{z}_1^T\dot{\mathbf{z}}_1 + \mathbf{z}_2^T\dot{\mathbf{z}}_2 = \mathbf{z}_1^T(-\mathbf{A}_1\mathbf{z}_1 + \mathbf{z}_2) + \mathbf{z}_2^T(\dot{\boldsymbol{\alpha}}_1 - \mathbf{f}_2 - \mathbf{G}_2\hat{\boldsymbol{\alpha}}_2)$$

$$\hat{\boldsymbol{\alpha}}_2 = \mathbf{G}_2^{-1}(\dot{\boldsymbol{\alpha}}_1 - \mathbf{f}_2 + \mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2)$$

Where $\mathbf{A}_2$ is a positive definite matrix, and it yields,

$$\dot{V}_2 = -\sum_{i=1}^{2}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i \leq 0 \tag{6.45}$$

The augmented control law gives asymptotic stability for $\mathbf{x}_1$ and $\mathbf{x}_2$ states. The virtual control input $\hat{\boldsymbol{\alpha}}_2$ consists of two parts, real control input T and virtual control input $\mathbf{g}_{\text{xy}}$. As explained earlier, due to the new parameterization, they are decoupled in this case and the virtual input can be used in the next step.

$$\begin{cases}\mathbf{u}_1 = \hat{\boldsymbol{\alpha}}_2(3) \\ \boldsymbol{\alpha}_2 = \hat{\boldsymbol{\alpha}}_2(1{:}2)\end{cases} \tag{6.46}$$

We could also look at the derivation, if not the new parameter $\mathbf{g}_{\text{xy}}$ but the normal Euler angles are used, recall the dynamic equation for step 2,

$$\dot{\mathbf{x}}_2 = -\frac{T}{m}\cdot\begin{bmatrix}\cos\Psi\sin\Theta\cos\Phi + \sin\Psi\sin\Phi \\ \sin\Psi\sin\Theta\cos\Phi - \cos\Psi\sin\Phi \\ \cos\Theta\cos\Phi\end{bmatrix} + \begin{bmatrix}0\\0\\g\end{bmatrix} \tag{6.47}$$

The real control input T and virtual control input and Θ & Φ are coupled in the dynamic equation. It is difficult to separate them in order to proceed with the normal analytical Backstepping derivation. Filtered estimation has to be used in this step.

Step 3 g_{xy} Control

Consider the following virtual system,

$$\dot{\mathbf{x}}_3 = \mathbf{f}_3 + \mathbf{G}_3\hat{\boldsymbol{\alpha}}_3 \tag{6.48}$$

Define the tracking error for $\mathbf{g}_{xy}$ and recover the true system dynamic equation in step 2

$$\begin{cases} \mathbf{z}_3 = \boldsymbol{\alpha}_2 - \mathbf{x}_3 \\ \dot{\mathbf{x}}_2 = \mathbf{f}_2 + \mathbf{G}_2\hat{\boldsymbol{\alpha}}_2 - \mathbf{G}_2\begin{bmatrix}\mathbf{z}_3 \\ 0\end{bmatrix} \end{cases} \tag{6.49}$$

The augmented Lyapunov function candidate is

$$V_3 = \frac{1}{2}\mathbf{z}_1^T\mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T\mathbf{z}_2 + \frac{1}{2}\mathbf{z}_3^T\mathbf{z}_3 \tag{6.50}$$

Compute the velocity error derivative,

$$\dot{\mathbf{z}}_2 = \dot{\boldsymbol{\alpha}}_1 - \dot{\mathbf{x}}_2 = -\mathbf{z}_1 - \mathbf{A}_2\mathbf{z}_2 + \mathbf{G}_2\begin{bmatrix}\mathbf{z}_3 \\ 0\end{bmatrix} \tag{6.51}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\hat{\boldsymbol{\alpha}}_2$:

$$\begin{aligned} \dot{V}_3 &= \mathbf{z}_1^T\dot{\mathbf{z}}_1 + \mathbf{z}_2^T\dot{\mathbf{z}}_2 + \mathbf{z}_3^T\dot{\mathbf{z}}_3 \\ &= \mathbf{z}_1^T(-\mathbf{A}_1\mathbf{z}_1 + \mathbf{z}_2) + \mathbf{z}_2^T\left(-\mathbf{z}_1 - \mathbf{A}_2\mathbf{z}_2 + \mathbf{G}_2\begin{bmatrix}\mathbf{z}_3 \\ 0\end{bmatrix}\right) + \mathbf{z}_3^T(\dot{\boldsymbol{\alpha}}_2 - \mathbf{f}_3 - \mathbf{G}_3\hat{\boldsymbol{\alpha}}_3) \end{aligned}$$

$$\hat{\boldsymbol{\alpha}}_3 = \mathbf{G}_3^{-1}(\dot{\boldsymbol{\alpha}}_2 - \mathbf{f}_3 + \mathbf{G}_2^T(1{:}2,:)\mathbf{z}_2 + \mathbf{A}_3\mathbf{z}_3)$$

Where $\mathbf{G}_2^T(1{:}2,:)$ is the first two rows of the matrix $\mathbf{G}_2^T$ and $\mathbf{A}_3$ is a positive definite matrix and it yields,

$$\dot{V}_3 = -\sum_{i=1}^{3}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i \leq 0 \tag{6.52}$$

The virtual control input $\hat{\boldsymbol{\alpha}}_3$ is the desired pitch and roll rate. Together with the desired yaw rate, we have the command to step one more integrator back.

$$\boldsymbol{\alpha}_3 = \begin{bmatrix}\hat{\boldsymbol{\alpha}}_3 \\ r_d\end{bmatrix} \tag{6.53}$$

Step 4 Rate Control

Last step inverses the rotational dynamics and the control input, moments appear in the equation. Rewrite the equation for rotational dynamics for completeness,

$$\dot{\mathbf{x}}_4 = \mathbf{f}_4 + \mathbf{G}_4\mathbf{u}_2 \tag{6.54}$$

Define the tracking error for the angular rates and rewrite the true system dynamic equation for step 3,

$$\begin{cases}\mathbf{z}_4 = \boldsymbol{\alpha}_3 - \mathbf{x}_4 \\ \dot{\mathbf{x}}_3 = \mathbf{f}_3 + \mathbf{G}_3\hat{\boldsymbol{\alpha}}_3 - \mathbf{G}_3\mathbf{z}_4(1:2)\end{cases} \tag{6.55}$$

Where the notation $\mathbf{z}_4(1:2)$ means it is the first two elements of the vector $\mathbf{z}_4$.

The final augmented Lyapunov function candidate is

$$V_4 = \frac{1}{2}\mathbf{z}_1^T\mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T\mathbf{z}_2 + \frac{1}{2}\mathbf{z}_3^T\mathbf{z}_3 + \frac{1}{2}\mathbf{z}_4^T\mathbf{z}_4 \tag{6.56}$$

Compute the $\mathbf{g}_{xy}$ error derivative,

$$\dot{\mathbf{z}}_3 = \dot{\boldsymbol{\alpha}}_2 - \dot{\mathbf{x}}_3 = -\mathbf{G}_2^T(1:2,:)\mathbf{z}_2 - \mathbf{A}_3\mathbf{z}_3 + \mathbf{G}_3\mathbf{z}_4(1:2) \tag{6.57}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\boldsymbol{\alpha}_3$:

$$\dot{V}_4 = -\mathbf{z}_1^T\mathbf{A}_1\mathbf{z}_1 - \mathbf{z}_2^T\mathbf{A}_2\mathbf{z}_2 - \mathbf{z}_3^T\mathbf{A}_3\mathbf{z}_3 + \mathbf{z}_3^T\mathbf{G}_3\mathbf{z}_4(1:2) + \mathbf{z}_4^T(\dot{\boldsymbol{\alpha}}_3 - \mathbf{f}_4 - \mathbf{G}_4\mathbf{u}_2)$$

$$\mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\boldsymbol{\alpha}}_3 - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T \\ \mathbf{0}_{1\times 2}\end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\mathbf{z}_4\right)$$

where $\mathbf{A}_4$ is a positive definite matrix and the Lyapunov function derivative can be rendered negative semi-definite.

$$\dot{V}_4 = -\sum_{i=1}^{4}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i \leq 0 \tag{6.58}$$

The equations of the Backstepping control law are summarized as follows,

$$\begin{cases}\boldsymbol{\alpha}_1 = \dot{\mathbf{x}}_{1r} + \mathbf{A}_1\mathbf{z}_1 \\ \hat{\boldsymbol{\alpha}}_2 = \mathbf{G}_2^{-1}(\dot{\boldsymbol{\alpha}}_1 - \mathbf{f}_2 + \mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2) \\ u_1 = \hat{\boldsymbol{\alpha}}_2(3) \\ \boldsymbol{\alpha}_2 = \hat{\boldsymbol{\alpha}}_2(1:2) \\ \hat{\boldsymbol{\alpha}}_3 = \mathbf{G}_3^{-1}(\dot{\boldsymbol{\alpha}}_2 - \mathbf{f}_3 + \mathbf{G}_2^T(1:2,:)\mathbf{z}_2 + \mathbf{A}_3\mathbf{z}_3) \\ \boldsymbol{\alpha}_3 = \begin{bmatrix}\hat{\boldsymbol{\alpha}}_3 \\ r_d\end{bmatrix} \\ \mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\boldsymbol{\alpha}}_3 - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T \\ \mathbf{0}_{1\times 2}\end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\mathbf{z}_4\right)\end{cases}$$

Step 5 Virtual Control Derivatives

The Backstepping control law still consists of virtual control derivatives, which are not directly available. As stated earlier for this design, the virtual control derivatives can be solved up to the third step, i.e. the virtual control derivatives $\dot{\boldsymbol{\alpha}}_1$ and $\dot{\boldsymbol{\alpha}}_2$ will be solve analytically.

For $\dot{\boldsymbol{\alpha}}_1$, it is straightforward,

$$\dot{\boldsymbol{\alpha}}_1 = \ddot{\mathbf{x}}_{1r} + \mathbf{A}_1(\dot{\mathbf{x}}_{1r} - \mathbf{x}_2) \tag{6.59}$$

For $\dot{\boldsymbol{\alpha}}_2$, the analytical solution get a little longer but still solvable. First, rewrite the virtual control $\hat{\boldsymbol{\alpha}}_2$ itself,

$$\hat{\boldsymbol{\alpha}}_2 = \mathbf{G}_2^{-1}(\dot{\boldsymbol{\alpha}}_1 - \mathbf{f}_2 + \mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2) = \mathbf{M}_{BW}\left(\underbrace{\dot{\boldsymbol{\alpha}}_1 + \mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2}_{\boldsymbol{\nu}_a}\right) - \begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B \tag{6.60}$$

$$\hat{\boldsymbol{\alpha}}_2 = \mathbf{M}_{BW}\boldsymbol{\nu}_a - \begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B \tag{6.61}$$

Where $\boldsymbol{\nu}_a$ physically is the desired acceleration in W frame. It can be solved, as well as its derivative.

$$\boldsymbol{\nu}_a = \ddot{\mathbf{x}}_{1r} + (\mathbf{A}_1 + \mathbf{A}_2)(\dot{\mathbf{x}}_{1r} - \mathbf{x}_2) + (\mathbf{A}_1\mathbf{A}_2 + \mathbf{I}_3)(\mathbf{x}_{1r} - \mathbf{x}_1)$$

$$\dot{\boldsymbol{\nu}}_a = \dddot{\mathbf{x}}_{1r} + (\mathbf{A}_1 + \mathbf{A}_2)\left(\ddot{\mathbf{x}}_{1r} - \mathbf{M}_{WB}(\vec{\mathbf{f}}_B + \vec{\mathbf{g}}_B)\right) + (\mathbf{A}_1\mathbf{A}_2 + \mathbf{I}_3)(\dot{\mathbf{x}}_{1r} - \mathbf{x}_2)$$

Taking the derivative of Eq. (6.61),

$$\dot{\hat{\boldsymbol{\alpha}}}_2 = \mathbf{M}_{BW}\dot{\boldsymbol{\nu}}_a + \mathbf{M}_{BW}\boldsymbol{\Omega}_{BW}^{BB}\boldsymbol{\nu}_a - \begin{bmatrix} \dot{f}_x \\ \dot{f}_y \\ \dot{g}_z \end{bmatrix}_B \tag{6.62}$$

The virtual control derivative $\dot{\boldsymbol{\alpha}}_2$ only takes the first two components of $\dot{\hat{\boldsymbol{\alpha}}}_2$,

$$\dot{\boldsymbol{\alpha}}_2 = \mathbf{M}_{BW}(1{:}2,:)\dot{\boldsymbol{\nu}}_a + \mathbf{M}_{BW}(1{:}2,:)\boldsymbol{\Omega}_{BW}^{BB}\boldsymbol{\nu}_a - \begin{bmatrix} \dot{f}_x \\ \dot{f}_y \end{bmatrix}_B \tag{6.63}$$

The third term in the above equation consists of mainly the disturbance force derivatives, which can be ignored and is ignored in other designs as well.

Lastly in step 4, another virtual control derivative $\dot{\boldsymbol{\alpha}}_3$ needs to be solved. Analytical solution is no longer possible, or at least many assumptions needed. Command filter is not a bad idea here to estimate the derivative. A first order low pass filter can fulfill the task,

$$\dot{\boldsymbol{\alpha}}_{3f}(s) = \frac{s}{Ts + 1}\boldsymbol{\alpha}_3(s) \tag{6.64}$$

The control law for $\mathbf{u}_2$, the desired moments, can be computed,

$$\mathbf{u}_2 = \mathbf{G}_4^{-1}(\dot{\boldsymbol{\alpha}}_{3f} - \mathbf{f}_4 + \begin{bmatrix} \mathbf{G}_3^T \\ \mathbf{0}_{1\times 2} \end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\mathbf{z}_4) \tag{6.65}$$

All together, the Backstepping control law is derived completely and can be implemented.

In term of the control bandwidth, this design is comparable with the NDI RD3+RD1 structure. Both utilize the same reference information, up to the third derivative of the position reference. The command filter is used on the same signals as the NDI RD3+RD1 structure, i.e. the angular rate. As compared before, this control law is almost the same as

cascaded Backstepping with RD3+RD1 structure. The only difference is the $\begin{bmatrix}\mathbf{G}_3^T \\ \mathbf{0}_{1\times 2}\end{bmatrix}\boldsymbol{z}_3$ term in Eq. (6.65). Once again, this term is rather negligible, compared to the filter estimation error, sensor noise etc. Without this term, this design is the same as cascaded Backstepping design, whose outer loop controls the position and commands thrust and angular rate to the inner loop.

The advantage over the NDI-RD3 design is that the direct thrust command is possible, due to the decoupling of the thrust and the virtual controls. The disadvantage is the difficult gain tuning due to the resulting nonlinear error dynamics, at least no so straightforward as in the NDI designs.

6.2.2. Backstepping with NDI-RD3 model (Design g2)

The position control design 2) used NDI method on a novel mathematical model, named NDI-RD3 model here. As would be shown later in the result, this design gives the best overall performance over the other design. It is worthwhile to apply Backstepping on the same math model for direct comparison.

The concept of dynamic extension to augment the system of non-triangular form into a triangular form could be applied to Backstepping method as well. In the NDI-RD3 design, the position outer loop is extended to a third order system of triangular form (or strictly feedback system). Naturally, the Backstepping method could also derive a similar design structure. Its performance can be predicted to be very close to the NDI-RD3 controller. For comparison and the sake of completeness, the derivation process is listed below.

The following state vectors and control inputs are defined for the position model,

$$\mathbf{x}_1 = \vec{\mathbf{r}}_W \quad \mathbf{x}_2 = \vec{\mathbf{v}}_W^W \quad \mathbf{x}_3 = \vec{\mathbf{a}}_W^{WW} \quad \mathbf{x}_4 = \vec{\boldsymbol{\omega}}_B \quad u_1 = \dot{T} \quad \mathbf{u}_2 = \begin{bmatrix} L \\ M \\ N \end{bmatrix}_B \tag{6.66}$$

The math equations, collected from section 4.3, can be rewritten in the following form for better clarifications,

$$\begin{aligned} \dot{\mathbf{x}}_1 &= \mathbf{x}_2 \\ \dot{\mathbf{x}}_2 &= \mathbf{x}_3 \\ \dot{\mathbf{x}}_3 &= \mathbf{G}_3 \begin{bmatrix} \mathbf{x}_4(1:2) \\ u_1 \end{bmatrix} \\ \dot{\mathbf{x}}_4 &= \mathbf{f}_4 + \mathbf{G}_4 \mathbf{u}_2 \end{aligned} \tag{6.67}$$

Where the nonlinear functions $\mathbf{f_4}$ is the same, recall it here,

$$\mathbf{f}_4 = -(\mathbf{I}^G)_{BB}^{-1} \cdot (\vec{\boldsymbol{\omega}}^{WB})_B \times \{(\mathbf{I}^G)_{BB} \cdot (\vec{\boldsymbol{\omega}}^{WB})_B\} \tag{6.68}$$

The input matrices $\mathbf{G_3}$ can be found in Eq. (6.19) and $\mathbf{G}_4$ remains the same,

$$\begin{cases} \mathbf{G}_3 = \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} 0 & -T & 0 \\ T & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \\ \mathbf{G}_4 = (\mathbf{I}^G)_{BB}^{-1} \end{cases} and \begin{cases} \mathbf{G}_3^{-1} = \begin{bmatrix} 0 & T^{-1} & 0 \\ -T^{-1} & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \mathbf{M}_{BW} m \\ \mathbf{G}_4^{-1} = (\mathbf{I}^G)_{BB} \end{cases} \tag{6.69}$$

Step 1 Position Control

The Lyapunov function candidate,

$$\begin{aligned} \mathbf{z}_1 &= \mathbf{x}_{1r} - \mathbf{x}_1 \\ V_1 &= \frac{1}{2}\mathbf{z}_1^T \mathbf{z}_1 \end{aligned} \tag{6.70}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\boldsymbol{\alpha}_1$,

$$\begin{aligned} \dot{V}_1 &= \mathbf{z}_1^T \dot{\mathbf{z}}_1 = \mathbf{z}_1^T (\dot{\mathbf{x}}_{1r} - \boldsymbol{\alpha}_1) \\ \boldsymbol{\alpha}_1 &= \dot{\mathbf{x}}_{1r} + \mathbf{A}_1 \mathbf{z}_1 \end{aligned} \tag{6.71}$$

Where $\mathbf{A}_1$ is a positive definite matrix and the Lyapunov function derivative can be rendered negative semi-definite.

$$\dot{V}_1 = -\mathbf{z}_1^T \mathbf{A}_1 \mathbf{z}_1 \leq 0 \tag{6.72}$$

Step 2 Velocity Control

Consider the following virtual system for the translational dynamics,

$$\dot{\mathbf{x}}_2 = \boldsymbol{\alpha}_2 \tag{6.73}$$

Define the tracking error $\mathbf{z}_2$ for the velocity, due to which we need to recover the true system dynamic equation in step 1,

$$\begin{aligned} \mathbf{z}_2 &= \boldsymbol{\alpha}_1 - \mathbf{x}_2 \\ \dot{\mathbf{x}}_1 &= \boldsymbol{\alpha}_1 - \mathbf{z}_2 \end{aligned} \tag{6.74}$$

The augmented Lyapunov function candidate is,

$$V_2 = \frac{1}{2}\mathbf{z}_1^T \mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T \mathbf{z}_2 \tag{6.75}$$

Compute the position tracking error derivative,

$$\dot{\mathbf{z}}_1 = \dot{\mathbf{x}}_{1r} - \dot{\mathbf{x}}_1 = -\mathbf{A}_1 \mathbf{z}_1 + \mathbf{z}_2 \tag{6.76}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\hat{\boldsymbol{\alpha}}_2$:

$$\begin{aligned} \dot{V}_1 &= \mathbf{z}_1^T \dot{\mathbf{z}}_1 + \mathbf{z}_2^T \dot{\mathbf{z}}_2 = \mathbf{z}_1^T (-\mathbf{A}_1 \mathbf{z}_1 + \mathbf{z}_2) + \mathbf{z}_2^T (\dot{\boldsymbol{\alpha}}_1 - \mathbf{f}_2 - \mathbf{G}_2 \hat{\boldsymbol{\alpha}}_2) \\ \boldsymbol{\alpha}_2 &= \dot{\boldsymbol{\alpha}}_1 + \mathbf{z}_1 + \mathbf{A}_2 \mathbf{z}_2 \end{aligned} \tag{6.77}$$

Where $\mathbf{A}_2$ is a positive definite matrix and it yields,

$$\dot{V}_2 = -\sum_{i=1}^{2} \mathbf{z}_i^T \mathbf{A}_i \mathbf{z}_i \leq 0 \tag{6.78}$$

Step 3 Acceleration Control

Consider the following virtual system,

$$\dot{\mathbf{x}}_3 = \mathbf{G}_3 \boldsymbol{\alpha}_3 \tag{6.79}$$

Define the tracking error for the acceleration and recover the true system dynamic equation in step 2,

$$\begin{cases} \mathbf{z}_3 = \boldsymbol{\alpha}_2 - \mathbf{x}_3 \\ \dot{\mathbf{x}}_2 = \boldsymbol{\alpha}_2 - \mathbf{z}_3 \end{cases} \tag{6.80}$$

The augmented Lyapunov function candidate is,

$$V_3 = \frac{1}{2}\mathbf{z}_1^T \mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T \mathbf{z}_2 + \frac{1}{2}\mathbf{z}_3^T \mathbf{z}_3 \tag{6.81}$$

Compute the velocity error derivative,

$$\dot{\mathbf{z}}_2 = \dot{\boldsymbol{\alpha}}_1 - \dot{\mathbf{x}}_2 = -\mathbf{z}_1 - \mathbf{A}_2 \mathbf{z}_2 + \mathbf{z}_3 \tag{6.82}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\widehat{\boldsymbol{\alpha}}_2$:

$$\begin{aligned} \dot{V}_3 &= \mathbf{z}_1^T \dot{\mathbf{z}}_1 + \mathbf{z}_2^T \dot{\mathbf{z}}_2 + \mathbf{z}_3^T \dot{\mathbf{z}}_3 \\ &= \mathbf{z}_1^T(-\mathbf{A}_1 \mathbf{z}_1 + \mathbf{z}_2) + \mathbf{z}_2^T(-\mathbf{z}_1 - \mathbf{A}_2 \mathbf{z}_2 + \mathbf{z}_3) + \mathbf{z}_3^T(\dot{\boldsymbol{\alpha}}_2 - \mathbf{G}_3 \boldsymbol{\alpha}_3) \end{aligned}$$

$$\boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}(\dot{\boldsymbol{\alpha}}_2 + \mathbf{z}_2 + \mathbf{A}_3 \mathbf{z}_3)$$

Where $\mathbf{A}_3$ is a positive definite matrix and it yields,

$$\dot{V}_3 = -\sum_{i=1}^{3} \mathbf{z}_i^T \mathbf{A}_i \mathbf{z}_i \leq 0 \tag{6.83}$$

The virtual control input $\boldsymbol{\alpha}_3$ is the desired pitch & roll rate and thrust derivative.

The resulting control law at this step can be summarized as follows,

$$\begin{cases} \boldsymbol{\alpha}_1 = \dot{\mathbf{x}}_{1r} + \mathbf{A}_1 \mathbf{z}_1 \\ \boldsymbol{\alpha}_2 = \dot{\boldsymbol{\alpha}}_1 + \mathbf{z}_1 + \mathbf{A}_2 \mathbf{z}_2 \\ \boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}(\dot{\boldsymbol{\alpha}}_2 + \mathbf{z}_2 + \mathbf{A}_3 \mathbf{z}_3) \end{cases} \tag{6.84}$$

So far, all the controlled states are linear and the virtual control derivatives are quite easy to solve analytically. For the first virtual control derivative,

$$\dot{\boldsymbol{\alpha}}_1 = \ddot{\mathbf{x}}_{1r} + \mathbf{A}_1 \dot{\mathbf{z}}_1 \tag{6.85}$$

The Backstepping error also needs to be converted to the position error $\mathbf{z}_1$ and its derivatives, because $\mathbf{z}_1$ is the known error between reference position and true position. For $\mathbf{z}_2$ we have,

$$\mathbf{z}_2 = \boldsymbol{\alpha}_1 - \mathbf{x}_2 = \dot{\mathbf{x}}_{1r} + \mathbf{A}_1\mathbf{z}_1 - \dot{\mathbf{x}}_1 = \dot{\mathbf{z}}_1 + \mathbf{A}_1\mathbf{z}_1 \tag{6.86}$$

Hence, the second virtual control can be rewritten,

$$\boldsymbol{\alpha}_2 = \ddot{\mathbf{x}}_{1r} + \mathbf{A}_1\dot{\mathbf{z}}_1 + \mathbf{z}_1 + \mathbf{A}_2(\dot{\mathbf{z}}_1 + \mathbf{A}_1\mathbf{z}_1) = \ddot{\mathbf{x}}_{1r} + (\mathbf{A}_1 + \mathbf{A}_2)\dot{\mathbf{z}}_1 + (\mathbf{A}_1\mathbf{A}_2 + 1)\mathbf{z}_1 \tag{6.87}$$

Its derivative can be solved,

$$\dot{\boldsymbol{\alpha}}_2 = \dddot{\mathbf{x}}_{1r} + (\mathbf{A}_1 + \mathbf{A}_2)\ddot{\mathbf{z}}_1 + (\mathbf{A}_1\mathbf{A}_2 + 1)\dot{\mathbf{z}}_1 \tag{6.88}$$

For $\mathbf{z}_3$ we have,

$$\mathbf{z}_3 = \boldsymbol{\alpha}_2 - \mathbf{x}_3 = \ddot{\mathbf{x}}_{1r} + (\mathbf{A}_1 + \mathbf{A}_2)\dot{\mathbf{z}}_1 + (\mathbf{A}_1\mathbf{A}_2 + 1)\mathbf{z}_1 - \ddot{\mathbf{x}}_1$$

$$\mathbf{z}_3 = \ddot{\mathbf{z}}_1 + (\mathbf{A}_1 + \mathbf{A}_2)\dot{\mathbf{z}}_1 + (\mathbf{A}_1\mathbf{A}_2 + 1)\mathbf{z}_1$$

Hence, the control law for $\boldsymbol{\alpha}_3$ is,

$$\boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}\{\dddot{\mathbf{x}}_{1r} + (\mathbf{A}_1 + \mathbf{A}_2)\ddot{\mathbf{z}}_1 + (\mathbf{A}_1\mathbf{A}_2 + 1)\dot{\mathbf{z}}_1 + \dot{\mathbf{z}}_1 + \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_3[\ddot{\mathbf{z}}_1 + (\mathbf{A}_1 + \mathbf{A}_2)\dot{\mathbf{z}}_1 + (\mathbf{A}_1\mathbf{A}_2 + 1)\mathbf{z}_1]\}$$

$$\boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}\Big\{\dddot{\mathbf{x}}_{1r} + \underbrace{(\mathbf{A}_1 + \mathbf{A}_2 + \mathbf{A}_3)}_{\mathbf{K}_a}\ddot{\mathbf{z}}_1 + \underbrace{(\mathbf{A}_1\mathbf{A}_2 + \mathbf{A}_1\mathbf{A}_3 + \mathbf{A}_3\mathbf{A}_2 + 2)}_{\mathbf{K}_v}\dot{\mathbf{z}}_1 + \underbrace{(\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3 + \mathbf{A}_1 + \mathbf{A}_3)}_{\mathbf{K}_r}\mathbf{z}_1\Big\}$$

$$\boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}\{\dddot{\mathbf{x}}_{1r} + \mathbf{K}_a\ddot{\mathbf{z}}_1 + \mathbf{K}_v\dot{\mathbf{z}}_1 + \mathbf{K}_r\mathbf{z}_1\}$$

It is the same as the NDI-RD3 outer loop controller if the feedback gains are set to be the same. This makes sense because the math model is the same and all the nonlinearities are inverted in the last step.

To proceed to the next loop, the desired thrust derivative is the control input and the desired angular rates will be commanded to the rate loop.

$$\begin{aligned} u_1 &= \boldsymbol{\alpha}_3(3) = \dot{T}_d \\ \hat{\boldsymbol{\alpha}}_3 &= \begin{bmatrix} \boldsymbol{\alpha}_3(1{:}\,2) \\ r_d \end{bmatrix} = \begin{bmatrix} p_d \\ q_d \\ r_d \end{bmatrix} \end{aligned} \tag{6.89}$$

Step 4 Rate Control

Last step remains more or less the same as in previous Backstepping design. Rewrite the equation for rotational dynamics for completeness,

$$\dot{\mathbf{x}}_4 = \mathbf{f}_4 + \mathbf{G}_4\mathbf{u}_2 \tag{6.90}$$

Define the tracking error for the angular rates and rewrite the true system dynamic equation for step 3,

$$\begin{cases}\mathbf{z}_4 = \hat{\boldsymbol{\alpha}}_3 - \mathbf{x}_4 \\ \dot{\mathbf{x}}_3 = \mathbf{G}_3\boldsymbol{\alpha}_3 - \mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix}\end{cases} \tag{6.91}$$

The final augmented Lyapunov function candidate is

$$V_4 = \frac{1}{2}\mathbf{z}_1^T\mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T\mathbf{z}_2 + \frac{1}{2}\mathbf{z}_3^T\mathbf{z}_3 + \frac{1}{2}\mathbf{z}_4^T\mathbf{z}_4 \tag{6.92}$$

Compute the acceleration error derivative,

$$\dot{\mathbf{z}}_3 = \dot{\boldsymbol{\alpha}}_2 - \dot{\mathbf{x}}_3 = -\mathbf{z}_2 - \mathbf{A}_3\mathbf{z}_3 + \mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix} \tag{6.93}$$

Take the time derivative of the Lyapunov function to derive the stabilization control law the virtual control input $\hat{\boldsymbol{\alpha}}_3$:

$$\dot{V}_4 = -\mathbf{z}_1^T\mathbf{A}_1\mathbf{z}_1 - \mathbf{z}_2^T\mathbf{A}_2\mathbf{z}_2 - \mathbf{z}_3^T\mathbf{A}_3\mathbf{z}_3 + \mathbf{z}_3^T\mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix} + \mathbf{z}_4^T\left(\dot{\hat{\boldsymbol{\alpha}}}_3 - \mathbf{f}_4 - \mathbf{G}_4\mathbf{u}_2\right)$$

$$\mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\hat{\boldsymbol{\alpha}}}_3 - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\mathbf{z}_4\right)$$

where $\mathbf{A}_4$ is a positive definite matrix and the Lyapunov function derivative can be rendered negative semi-definite.

$$\dot{V}_4 = -\mathbf{z}_1^T\mathbf{A}_1\mathbf{z}_1 - \mathbf{z}_2^T\mathbf{A}_2\mathbf{z}_2 - \mathbf{z}_3^T\mathbf{A}_3\mathbf{z}_3 + \mathbf{z}_3^T\mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix} + \mathbf{z}_4^T\left(-\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3 - \mathbf{A}_4\mathbf{z}_4\right)$$

$$\dot{V}_4 = -\sum_{i=1}^{4}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i + \mathbf{z}_3^T\mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix} - \mathbf{z}_4^T\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3$$

$$\dot{V}_4 = -\sum_{i=1}^{4}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i + \mathbf{z}_3^T\mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix} - \left(\mathbf{z}_4^T\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3\right)^T$$

$$\dot{V}_4 = -\sum_{i=1}^{4}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i + \mathbf{z}_3^T\mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\ 0\end{bmatrix} - \mathbf{z}_3^T[\mathbf{G}_3(:,1:2) \quad \mathbf{0}_{3\times 1}]\mathbf{z}_4$$

$$\dot{V}_4 = -\sum_{i=1}^{4}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i + \mathbf{z}_3^T\mathbf{G}_3(:,1:2)\mathbf{z}_4(1:2) - \mathbf{z}_3^T\mathbf{G}_3(:,1:2)\mathbf{z}_4(1:2)$$

$$\dot{V}_4 = -\sum_{i=1}^{4}\mathbf{z}_i^T\mathbf{A}_i\mathbf{z}_i \le 0$$

The equations of the Backstepping control law are summarized as follows,

$$\begin{cases}\boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}\{\ddot{\mathbf{x}}_{1r} + \mathbf{K}_a\ddot{\mathbf{z}}_1 + \mathbf{K}_v\dot{\mathbf{z}}_1 + \mathbf{K}_r\mathbf{z}_1\} \\ u_1 = \boldsymbol{\alpha}_3(3) \\ \hat{\boldsymbol{\alpha}}_3 = \begin{bmatrix}\boldsymbol{\alpha}_3(1:2)\\ r_d\end{bmatrix} \\ \mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\hat{\boldsymbol{\alpha}}}_3 - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\mathbf{z}_4\right)\end{cases} \tag{6.94}$$

Where the third virtual control derivative is estimated by command filter,

$$\dot{\boldsymbol{\alpha}}_{3f}(s) = \frac{s}{Ts+1}\boldsymbol{\alpha}_3(s) \tag{6.95}$$

The new control law with the command filter is,

$$\mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\hat{\boldsymbol{\alpha}}}_{3f} - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\hat{\mathbf{z}}_4\right) \tag{6.96}$$

The difference between this design and the NDI-RD3 design lies in the additional term $\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3$. However, as stated earlier it is rather a negligible term, especially in the presence of parameter uncertainties such as modeling errors, sensor noises and external disturbances. There is no difference at all in terms of performance.

For the sake of completeness, the Command Filtered Backstepping is also derived and implemented in the next section to further validate the comparison results.

6.2.3. CFB with NDI-RD3 Model (Design h)

The CFB control design follows the control law from the above design. The filter estimation error violates the Lyapunov stability proof, so an additional modification is introduced to compensate the estimation error.

The original Lyapunov function from Eq. (6.92) needs to be modified when using the command filter,

$$\hat{V}_4 = \frac{1}{2}\mathbf{z}_1^T\mathbf{z}_1 + \frac{1}{2}\mathbf{z}_2^T\mathbf{z}_2 + \frac{1}{2}\mathbf{z}_3^T\mathbf{z}_3 + \frac{1}{2}\hat{\mathbf{z}}_4^T\hat{\mathbf{z}}_4$$

Where,

$$\hat{\mathbf{z}}_4 = \hat{\boldsymbol{\alpha}}_{3f} - \mathbf{x}_4$$

To compensate the estimation error in $\hat{\mathbf{z}}_4$, a modification term $\boldsymbol{\xi}_3$ in $\mathbf{z}_3$ needs to be added in the Lyapunov function. Because all the error terms $\mathbf{z}_i$ have cross-coupled terms to be compensated, an additional term $\boldsymbol{\xi}_2$ needs to be added in $\mathbf{z}_2$ to compensate $\boldsymbol{\xi}_3$ and again $\boldsymbol{\xi}_1$ to be added in $\mathbf{z}_1$ to compensate $\boldsymbol{\xi}_2$.

In total three modification terms $\boldsymbol{\xi}_1$, $\boldsymbol{\xi}_2$ and $\boldsymbol{\xi}_3$ are introduced to form a new CFB Lyapunov function,

$$V_{CFB} = \frac{1}{2}\hat{\mathbf{z}}_1^T\hat{\mathbf{z}}_1 + \frac{1}{2}\hat{\mathbf{z}}_2^T\hat{\mathbf{z}}_2 + \frac{1}{2}\hat{\mathbf{z}}_3^T\hat{\mathbf{z}}_3 + \frac{1}{2}\hat{\mathbf{z}}_4^T\hat{\mathbf{z}}_4 \tag{6.97}$$

Where

$$\hat{\mathbf{z}}_1 = \mathbf{z}_1 + \boldsymbol{\xi}_1$$

$$\hat{\mathbf{z}}_2 = \mathbf{z}_2 + \boldsymbol{\xi}_2$$

$$\hat{\mathbf{z}}_3 = \mathbf{z}_3 + \boldsymbol{\xi}_3$$

Next, similar to the derivation in section 3.2.5, we could derive the control law for the introduced terms $\boldsymbol{\xi}_i$ so that the new CFB Lyapunov function derivative can be rendered negative semi-definite. The resulting general form has been already given in Eq. (3.37).

$$\dot{\boldsymbol{\xi}}_i = -\mathbf{A}_i\boldsymbol{\xi}_i + \mathbf{G}_i\left(\boldsymbol{\alpha}_{if} - \boldsymbol{\alpha}_i\right) + \mathbf{G}_i\boldsymbol{\xi}_{i+1} \tag{6.98}$$

The first two terms $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ are relatively simpler, as the filter estimation error has not involved yet,

$$\begin{aligned}\dot{\boldsymbol{\xi}}_1 &= -\mathbf{A}_1\boldsymbol{\xi}_1 + \boldsymbol{\xi}_2 \\ \dot{\boldsymbol{\xi}}_2 &= -\mathbf{A}_2\boldsymbol{\xi}_2 + \boldsymbol{\xi}_3 \\ \dot{\boldsymbol{\xi}}_3 &= -\mathbf{A}_3\boldsymbol{\xi}_3 + \mathbf{G}_3\begin{bmatrix}\hat{\boldsymbol{\alpha}}_{3f} - \hat{\boldsymbol{\alpha}}_3 \\ \mathbf{0}\end{bmatrix}\end{aligned} \tag{6.99}$$

The normal control law needs small modifications (marked in red) on the error feedforward terms accordingly,

$$\begin{cases}\boldsymbol{\alpha}_1 = \dot{\mathbf{x}}_{1\mathrm{r}} + \mathbf{A}_1\mathbf{z}_1 \\ \boldsymbol{\alpha}_2 = \dot{\boldsymbol{\alpha}}_1 + \mathbf{z}_1 + \boldsymbol{\xi}_1 + \mathbf{A}_2\mathbf{z}_2 \\ \boldsymbol{\alpha}_3 = \mathbf{G}_3^{-1}(\dot{\boldsymbol{\alpha}}_2 + \mathbf{z}_2 + \boldsymbol{\xi}_2 + \mathbf{A}_3\mathbf{z}_3) \\ \mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\hat{\boldsymbol{\alpha}}}_{3f} - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T(1:2,:) \\ \mathbf{0}_{1\times 3}\end{bmatrix}(\mathbf{z}_3 + \boldsymbol{\xi}_3) + \mathbf{A}_4\hat{\mathbf{z}}_4\right)\end{cases} \tag{6.100}$$

The control law is transformed below for simpler implementation,

$$\begin{cases}\boldsymbol{\alpha}_{3,CFB} = \mathbf{G}_3^{-1}\{\ddot{\mathbf{x}}_{1r} + \mathbf{K}_a\ddot{\mathbf{z}}_1 + \mathbf{K}_v\dot{\mathbf{z}}_1 + \mathbf{K}_r\mathbf{z}_1 + \dot{\boldsymbol{\xi}}_1 + \boldsymbol{\xi}_2\} \\ u_1 = \boldsymbol{\alpha}_{3,CFB}(3) \\ \hat{\boldsymbol{\alpha}}_{3,CFB} = \begin{bmatrix}\boldsymbol{\alpha}_{3,CFB}(1:2) \\ r_d\end{bmatrix} \\ \mathbf{u}_2 = \mathbf{G}_4^{-1}\left(\dot{\hat{\boldsymbol{\alpha}}}_{3f,CFB} - \mathbf{f}_4 + \begin{bmatrix}\mathbf{G}_3^T(1:2,:) \\ \mathbf{0}_{1\times 3}\end{bmatrix}(\mathbf{z}_3 + \boldsymbol{\xi}_3) + \mathbf{A}_4\hat{\mathbf{z}}_4\right)\end{cases} \tag{6.101}$$

6.2.3.1. Lyapunov stability proof of the derived control law

The Lyapunov stability can be proved by substituting the control laws into the CFB Lyapunov function derivative,

$$\dot{V}_{CFB} = \hat{\mathbf{z}}_1^T\dot{\hat{\mathbf{z}}}_1 + \hat{\mathbf{z}}_2^T\dot{\hat{\mathbf{z}}}_2 + \hat{\mathbf{z}}_3^T\dot{\hat{\mathbf{z}}}_3 + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = (\mathbf{z}_1 + \boldsymbol{\xi}_1)^T(\dot{\mathbf{z}}_1 + \dot{\boldsymbol{\xi}}_1) + (\mathbf{z}_2 + \boldsymbol{\xi}_2)^T(\dot{\mathbf{z}}_2 + \dot{\boldsymbol{\xi}}_2) + \hat{\mathbf{z}}_3^T\dot{\hat{\mathbf{z}}}_3 + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = \hat{\mathbf{z}}_1^T(-\mathbf{A}_1\mathbf{z}_1 + \mathbf{z}_2 + \dot{\boldsymbol{\xi}}_1) + \hat{\mathbf{z}}_2^T(-\hat{\mathbf{z}}_1 - \mathbf{A}_2\mathbf{z}_2 + \mathbf{z}_3 + \dot{\boldsymbol{\xi}}_2) + \hat{\mathbf{z}}_3^T\dot{\hat{\mathbf{z}}}_3 + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = \hat{\mathbf{z}}_1^T(-\mathbf{A}_1\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{A}_1\boldsymbol{\xi}_1 + \boldsymbol{\xi}_2) + \hat{\mathbf{z}}_2^T(-\hat{\mathbf{z}}_1 - \mathbf{A}_2\mathbf{z}_2 + \mathbf{z}_3 - \mathbf{A}_2\boldsymbol{\xi}_2 + \boldsymbol{\xi}_3) + \hat{\mathbf{z}}_3^T\dot{\hat{\mathbf{z}}}_3 + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = \hat{\mathbf{z}}_1^T(-\mathbf{A}_1\hat{\mathbf{z}}_1 + \hat{\mathbf{z}}_2) + \hat{\mathbf{z}}_2^T(-\hat{\mathbf{z}}_1 - \mathbf{A}_2\hat{\mathbf{z}}_2 + \hat{\mathbf{z}}_3) + \hat{\mathbf{z}}_3^T\dot{\hat{\mathbf{z}}}_3 + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 + \hat{\mathbf{z}}_2^T\hat{\mathbf{z}}_3 + \hat{\mathbf{z}}_3^T\dot{\hat{\mathbf{z}}}_3 + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 + \hat{\mathbf{z}}_2^T\hat{\mathbf{z}}_3 + \hat{\mathbf{z}}_3^T\left(-\hat{\mathbf{z}}_2 - \mathbf{A}_3\mathbf{z}_3 + \mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\0\end{bmatrix} + \dot{\boldsymbol{\xi}}_3\right) + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 + \hat{\mathbf{z}}_3^T\left(-\mathbf{A}_3\mathbf{z}_3 + \mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\0\end{bmatrix} + \dot{\boldsymbol{\xi}}_3\right) + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 + \hat{\mathbf{z}}_3^T\left(-\mathbf{A}_3\mathbf{z}_3 + \mathbf{G}_3\begin{bmatrix}\mathbf{z}_4(1:2)\\0\end{bmatrix} - \mathbf{A}_3\boldsymbol{\xi}_3 + \mathbf{G}_3\begin{bmatrix}\hat{\boldsymbol{\alpha}}_{3f} - \hat{\boldsymbol{\alpha}}_3\\\mathbf{0}\end{bmatrix}\right) + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 + \hat{\mathbf{z}}_3^T\left(-\mathbf{A}_3\hat{\mathbf{z}}_3 + \mathbf{G}_3\begin{bmatrix}\hat{\mathbf{z}}_4(1:2)\\0\end{bmatrix}\right) + \hat{\mathbf{z}}_4^T\dot{\hat{\mathbf{z}}}_4$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 - \hat{\mathbf{z}}_3^T\mathbf{A}_3\hat{\mathbf{z}}_3 + \hat{\mathbf{z}}_3^T\mathbf{G}_3\begin{bmatrix}\hat{\mathbf{z}}_4(1:2)\\0\end{bmatrix} + \hat{\mathbf{z}}_4^T\left(-\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\\mathbf{0}_{1\times3}\end{bmatrix}\hat{\mathbf{z}}_3 - \mathbf{A}_4\hat{\mathbf{z}}_4\right)$$

$$\dot{V}_{CFB} = -\hat{\mathbf{z}}_1^T\mathbf{A}_1\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2^T\mathbf{A}_2\hat{\mathbf{z}}_2 - \hat{\mathbf{z}}_3^T\mathbf{A}_3\hat{\mathbf{z}}_3 + \hat{\mathbf{z}}_3^T\mathbf{G}_3\begin{bmatrix}\hat{\mathbf{z}}_4(1:2)\\0\end{bmatrix} + \hat{\mathbf{z}}_4^T\left(-\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\\mathbf{0}_{1\times3}\end{bmatrix}\hat{\mathbf{z}}_3 - \mathbf{A}_4\hat{\mathbf{z}}_4\right)$$

$$\dot{V}_{CFB} = -\sum_{i=1}^{4}\hat{\mathbf{z}}_i^T\mathbf{A}_i\hat{\mathbf{z}}_i + \hat{\mathbf{z}}_3^T\mathbf{G}_3\begin{bmatrix}\hat{\mathbf{z}}_4(1:2)\\0\end{bmatrix} - \hat{\mathbf{z}}_4^T\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\\mathbf{0}_{1\times3}\end{bmatrix}\hat{\mathbf{z}}_3$$

$$\dot{V}_{CFB} = -\sum_{i=1}^{4}\hat{\mathbf{z}}_i^T\mathbf{A}_i\hat{\mathbf{z}}_i + \hat{\mathbf{z}}_3^T\mathbf{G}_3\begin{bmatrix}\hat{\mathbf{z}}_4(1:2)\\0\end{bmatrix} - \left(\hat{\mathbf{z}}_4^T\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\\mathbf{0}_{1\times3}\end{bmatrix}\hat{\mathbf{z}}_3\right)^T$$

$$\dot{V}_{CFB} = -\sum_{i=1}^{4}\hat{\mathbf{z}}_i^T\mathbf{A}_i\hat{\mathbf{z}}_i + \hat{\mathbf{z}}_3^T\mathbf{G}_3\begin{bmatrix}\hat{\mathbf{z}}_4(1:2)\\0\end{bmatrix} - \hat{\mathbf{z}}_3^T[\mathbf{G}_3(:,1:2)\quad\mathbf{0}_{3\times1}]\hat{\mathbf{z}}_4$$

$$\dot{V}_{CFB} = -\sum_{i=1}^{4}\hat{\mathbf{z}}_i^T\mathbf{A}_i\hat{\mathbf{z}}_i \leq 0$$

Therefore, the modification terms successfully render the new CFB Lyapunov function derivative negative semi-definite, and stability can be guaranteed. However, one flaw is that the Lyapunov function is no longer to minimize the tracking error but the compensated tracking error, due to the additional terms added in each error term of the original Lyapunov function candidate.

6.3. High Order Reference Model

All the introduced control designs require reference model of different system orders. In this section, a common high order reference model is introduced to provide common basis for the controller comparison. For the position dynamics, a fourth order reference model is used.

The reference model has two main functions: 1. Generate smooth enough signal for the plant to follow, or generate the reference dynamics based on given requirement if there is any; 2. consider the physical plant constraints and limit the reference commands.

6.3.1. State inconsistency due to state limits

For first order reference model, there is only one integrator so actuator limit can be easily incorporated as shown in Figure 2.1. The state limit can be incorporated on the command x_c as there is no overshoot with the first order low pass filter.

For the second order system, the linear reference model can be formulated similarly with a second order low pass filter in Figure 2.2. When system order increases, the complexity increases if the actuator and state limits need to be included, esp. for highly dynamic system. A simple and common way to incorporate state limits is to put the limits in the integrators. For instance, the second order low pass filter with state limits is shown below,

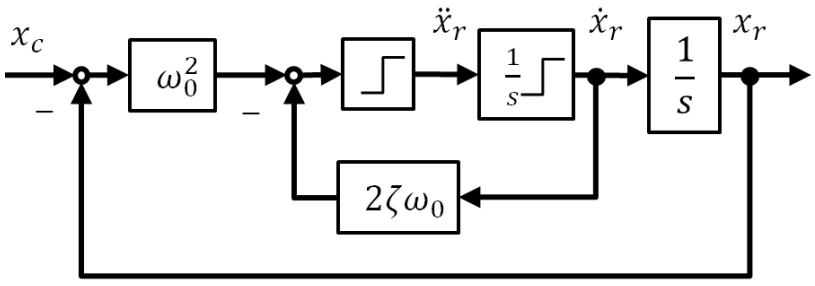


Figure 6.7 Block diagram of second order reference model with direct integrator limits

If the state limits are not reached for $\dot{x}_r$ or inactive, then it is fine as before. However, when the control bandwidth drives up, highly dynamic states are demanding and the limits will be reached. Once that happen, this simple limitation in the integrators will cause inconsistency between states. An example is given in Figure 6.9 to show the inconsistency between the states. At t≈1.05 s, the first derivative $\dot{x}_r$ reaches the pre-set limits of ±5. To be consistent, the second derivative $\ddot{x}_r$ shall be zero but it is not zero. Afterwards, due to the actuator limit (±100) before $\ddot{x}_r$ it takes a while for the controller to finally drive the second derivatives $\ddot{x}_r$ to zero. In other words, the same x_r cannot be obtained if you integrating the second derivative $\ddot{x}_r$ twice, due to inconsistency caused by the integrator limits in between.

To avoid inconsistency between the states, one idea is to consider the reference model as a complete close loop system with an ideal 'plant' and a corresponding controller. On one hand, the ideal plant can be formulated based on the true plant dynamics and the integrators there should not be -limited so that the states are consistent with each other. It can be either linear or nonlinear depending what reference signals are required later. On the other hand, a reference controller can be designed to control the reference plant. Because of the perfect

state feedback, the controller is also super effective so the physical limits can be easily implemented there. No out-of-limit commands goes to the ideal plant. For the second order example, the proposed solution is,

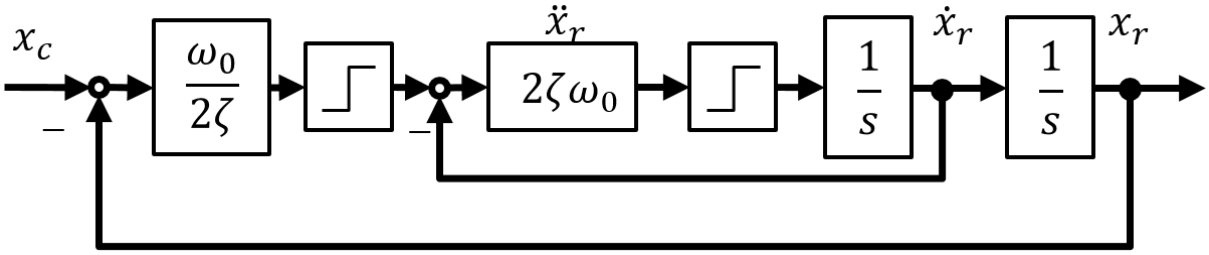


Figure 6.8 Block diagram of second order reference model with state limits

The resulting reference signals have no inconsistency. They are compared in the Figure 6.10, with the same state limits and parameters. The new signals (in red) also could reach the $\dot{x}_r$ limit of ±5 but this time with consistent $\ddot{x}_r$ signals. This implementation will be further illustrated in the following position reference model.

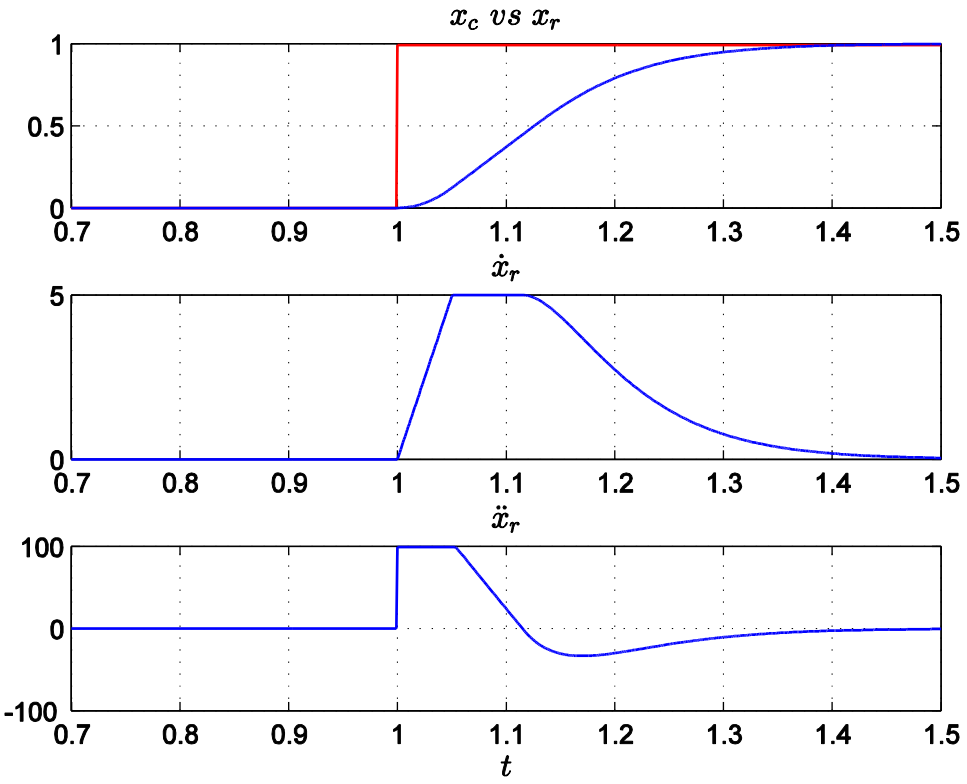


Figure 6.9 Possible state inconsistency of the second order reference model

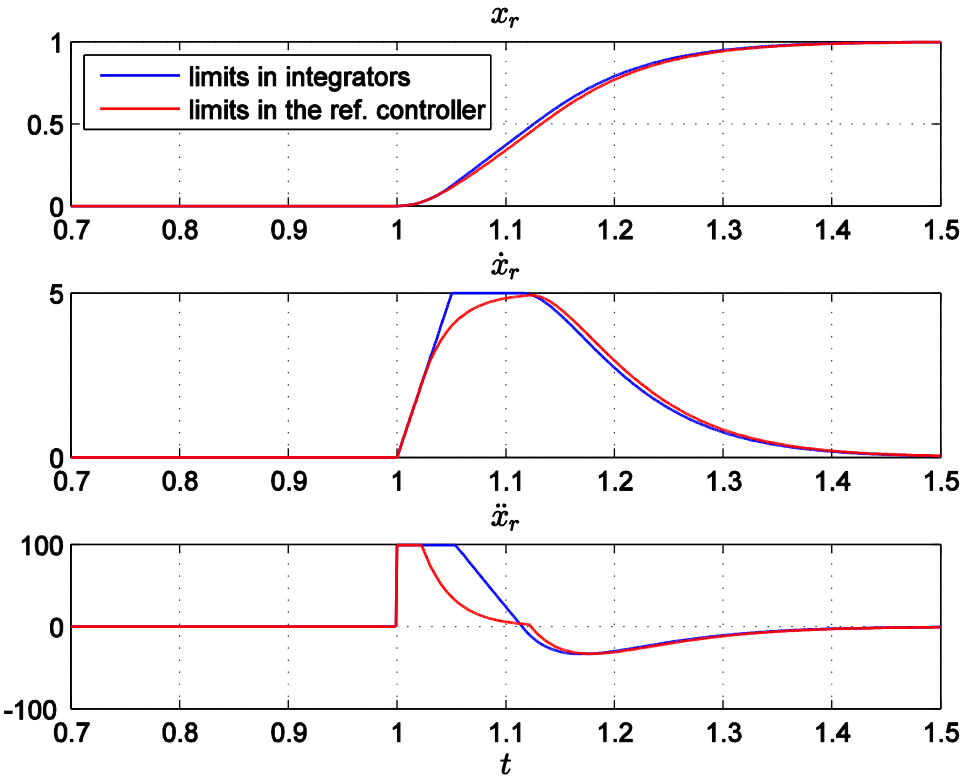


Figure 6.10 Proposed implementation of the second order reference model

6.3.2. Position reference model

For the position dynamics of the quadrotor, at least a fourth order reference model, better fifth order, is needed to generate smooth enough position signals, which are differentiable until the actuator input (4 integrators in between). With high system order, it is rather difficult to incorporate the physical limits. Two challenges arise: 1. the reference state inconsistency easily arise if the state limits are hard imposed in the integrators, even worse if just after the integrators; 2. the physical limits are normally on the nonlinear states, so it is not accurate to impose them directly on the linear reference states.

The inconsistency could be more problematic here due to the high system order. It is quite common that the state reach saturation in the integrators, but the derivative, i.e. input to the integrators could not reach zero immediately due to the high order system dynamics.

6.3.2.1. Linear fourth order reference model

Similar approach as for the second order reference model is applied here. There are two choices for the 'ideal plant', one is the ideal nonlinear dynamic plant, and the other is a linear plant as a chain of integrators. In [49], the ideal nonlinear plant is used in the reference model and a linear controller is designed to control it. A first thought from that reference model is why not to use nonlinear control like NDI for the ideal nonlinear plant. However, the perfect inversion in the NDI control plus the nonlinear ideal plant will result a chain of integrator, which is the same as in the linear low pass filter. In this reference model design, the ideal plant is chosen to be a chain of integrator. Nonlinear reference signals can be afterwards computed from the linear ones.

Therefore, the reference plant is chosen to be a chain of integrators, i.e. four integrators for a fourth order reference model. The transfer function of a fourth order filter from the position command $\vec{\mathbf{r}}_c$ to the reference position $\vec{\mathbf{r}}_r$ is straightforward,

$$\vec{\mathbf{r}}_r = \frac{k_0}{s^4 + k_3 s^3 + k_2 s^2 + k_1 s + k_0} \vec{\mathbf{r}}_c \tag{6.102}$$

From the Laplace transfer function, we can derive the reference control law in time domain,

$$\ddddot{\vec{\mathbf{r}}}_r = k_0(\vec{\mathbf{r}}_c - \vec{\mathbf{r}}_r) - k_1 \dot{\vec{\mathbf{r}}}_r - k_2 \ddot{\vec{\mathbf{r}}}_r - k_3 \dddot{\vec{\mathbf{r}}}_r \tag{6.103}$$

The control law can be rearranged to the following controller structure,

$$\ddddot{\vec{\mathbf{r}}}_r = k_3 \Bigg\{ \underbrace{\frac{k_2}{k_3} \Bigg\{ \underbrace{\frac{k_1}{k_2} \Bigg[\underbrace{\frac{k_0}{k_1}(\vec{\mathbf{r}}_c - \vec{\mathbf{r}}_r)}_{\vec{\mathbf{v}}_d} - \dot{\vec{\mathbf{r}}}_r \Bigg]}_{\dot{\vec{\mathbf{a}}}_d} - \ddot{\vec{\mathbf{r}}}_r \Bigg\}}_{\ddot{\vec{\mathbf{a}}}_d} - \dddot{\vec{\mathbf{r}}}_r \Bigg\} \tag{6.104}$$

Where the states $\vec{\mathbf{v}}$, $\vec{\mathbf{a}}$ and $\dot{\vec{\mathbf{a}}}$ are linear derivative state of position $\vec{\mathbf{r}}$ denoted in the World frame. This can be also represented in a cascaded controller structure as follows,

$$\begin{cases} \vec{\mathbf{v}}_{\boldsymbol{d}} = \dfrac{k_0}{k_1}(\vec{\mathbf{r}}_c - \vec{\mathbf{r}}_r) \\ \vec{\mathbf{a}}_{\boldsymbol{d}} = \dfrac{k_1}{k_2}(\vec{\mathbf{v}}_{\boldsymbol{d}} - \vec{\mathbf{v}}_r) \\ \dot{\vec{\mathbf{a}}}_{\boldsymbol{d}} = \dfrac{k_2}{k_3}(\vec{\mathbf{a}}_{\boldsymbol{d}} - \vec{\mathbf{a}}_r) \\ \ddot{\vec{\mathbf{a}}}_{\boldsymbol{d}} = k_3\left(\dot{\vec{\mathbf{a}}}_{\boldsymbol{d}} - \dot{\vec{\mathbf{a}}}_r\right) \approx \ddot{\vec{\mathbf{a}}}_{\boldsymbol{r}} = \dddot{\vec{\mathbf{r}}}_r \end{cases} \tag{6.105}$$

With the above transformation, the linear state limits can be incorporated on the state commands, meaning only the maximum allowable state is commanded to the next loop. In this way, we end up with a clean integrator chain without any forced state limits. The implementation structure is,

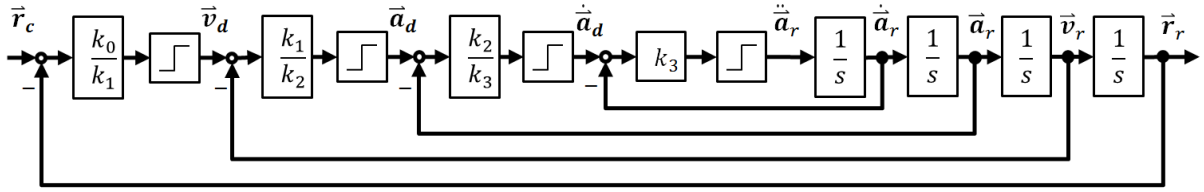


Figure 6.11 Fourth order reference model with linear state limits

The gains can be determined by pole placement method. All four poles are assigned to the same value '-ω', so that the reference error dynamics is critically damped. In addition, the gain tuning is reduced to one parameter (ω) tuning. The gains can be higher than that in the real error controller, due to the perfect state feedback. And the generated reference signal will fulfil the physical constraints due to the state limiters. The gains can be assigned according to the characteristic equation in the Laplace domain.

$$s^4 + k_3 s^3 + k_2 s^2 + k_1 s + k_0 = s^4 + 4\omega s^3 + 6\omega^2 s^2 + 4\omega^3 s + \omega^4 \tag{6.106}$$

So the gains in the above structure can be represented by ω,

$$\begin{cases} k_3 = 4\omega \\ \dfrac{k_2}{k_3} = \dfrac{3}{2}\omega \\ \dfrac{k_1}{k_2} = \dfrac{2}{3}\omega \\ \dfrac{k_0}{k_1} = \dfrac{1}{4}\omega \end{cases} \tag{6.107}$$

The linear reference signals generated by the above structure are shown in Figure 6.12. The step command for the position is 20 m at time equal to 1 second. The intermediate state limits are set as, $\vec{\mathbf{v}}_r$10 m/s, $\vec{\mathbf{a}}_r$ 15 m/s^2, $\dot{\vec{\mathbf{a}}}_r$ 60 m/s^3 and $\ddot{\vec{\mathbf{a}}}_r$ 400 m/s^4. Without exceeding the bounds, smooth reference signals can be generated using this simple control idea.

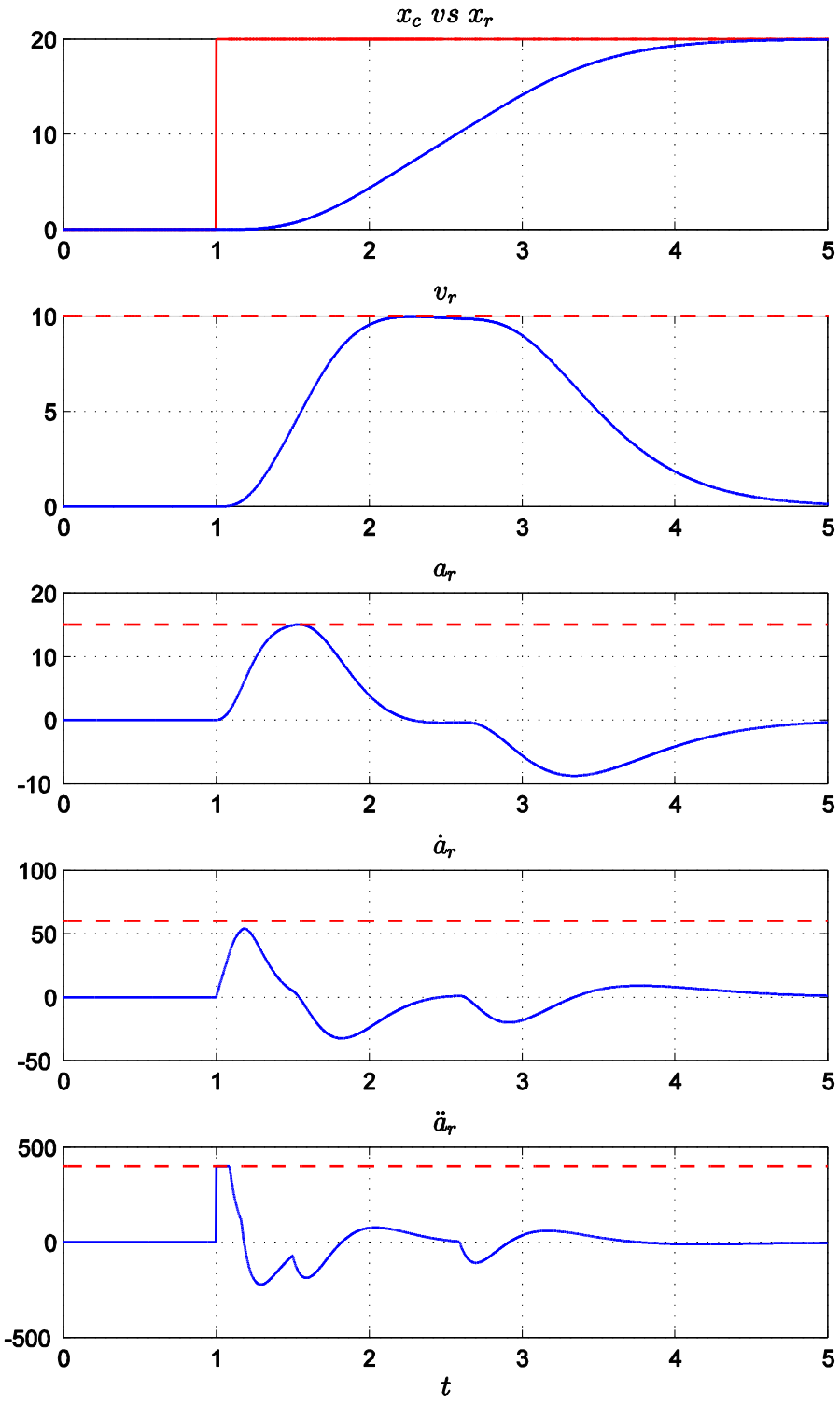


Figure 6.12 Fourth order reference model with trajectory commands

6.3.2.2. Position reference model with trajectory commands

In addition, this reference model design could also receive trajectory commands including velocity $\vec{\mathbf{v}}_c$ and acceleration $\vec{\mathbf{a}}_c$, which can be feedforward to the reference 'controller'. The new control equation is,

$$\dddot{\vec{\mathbf{r}}}_r = k_3 \left\{ \underbrace{\frac{k_2}{k_3} \left\{ \underbrace{\frac{k_1}{k_2} \left[\underbrace{\frac{k_0}{k_1}(\vec{\mathbf{r}}_c - \vec{\mathbf{r}}_r) + \vec{\mathbf{v}}_c}_{\vec{\mathbf{v}}_d} - \dot{\vec{\mathbf{r}}}_r \right]}_{\vec{\mathbf{a}}_d} + \vec{\mathbf{a}}_c - \ddot{\vec{\mathbf{r}}}_r \right\}}_{\dot{\vec{\mathbf{a}}}_d} - \ddot{\vec{\mathbf{r}}}_r \right\}$$

And the implemented structure is,

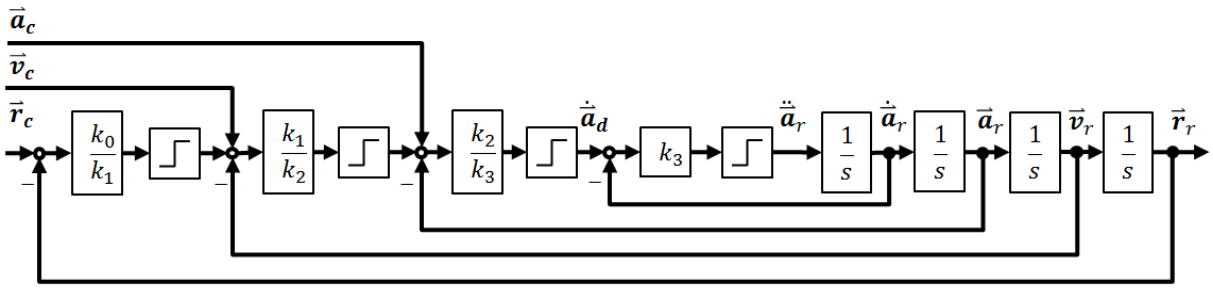


Figure 6.13 Fourth order reference model with trajectory commands

6.3.2.2.1. Implementation of NDI RD4 error controller

The NDI control design in section 6.1.3 has very similar error controller structure as the reference controller in the reference model. The implementation of the error controller can be made similarly, just with more feedforward reference signals.

$$\ddot{\vec{\mathbf{a}}}_d = \ddot{\vec{\mathbf{a}}}_r + k_3 \left\{ \underbrace{\frac{k_2}{k_3} \left\{ \underbrace{\frac{k_1}{k_2} \left[\underbrace{\frac{k_0}{k_1}(\vec{\mathbf{r}}_r - \vec{\mathbf{r}}) + \vec{\mathbf{v}}_r}_{\vec{\mathbf{v}}_d} - \vec{\mathbf{v}} \right]}_{\vec{\mathbf{a}}_d} + \vec{\mathbf{a}}_r - \vec{\mathbf{a}} \right\}}_{\dot{\vec{\mathbf{a}}}_d} + \dot{\vec{\mathbf{a}}}_r - \dot{\vec{\mathbf{a}}} \right\} \tag{6.108}$$

$$\begin{cases} \vec{\mathbf{v}}_d = \frac{k_0}{k_1}(\vec{\mathbf{r}}_r - \vec{\mathbf{r}}) + \vec{\mathbf{v}}_r \\ \vec{\mathbf{a}}_d = \frac{k_1}{k_2}(\vec{\mathbf{v}}_d - \vec{\mathbf{v}}) + \vec{\mathbf{a}}_r \\ \dot{\vec{\mathbf{a}}}_d = \frac{k_2}{k_3}(\vec{\mathbf{a}}_d - \vec{\mathbf{a}}) + \dot{\vec{\mathbf{a}}}_r \\ \ddot{\vec{\mathbf{a}}}_d = k_3(\dot{\vec{\mathbf{a}}}_d - \dot{\vec{\mathbf{a}}}) + \ddot{\vec{\mathbf{a}}}_r \end{cases} \tag{6.109}$$

Even though the dynamic inversion is non-cascaded, the linear control can be still implemented like the cascaded structure but with additional feedforward reference signals. In this way, the desired states going to the next level can be limited.

The feedback gains can initially designed based on the linear transfer function from the reference position $\vec{\mathbf{r}}_r$ to the position $\vec{\mathbf{r}}$ as shown in Eq. (6.110). However, it needs to be fine-tuned in flight test.

$$\vec{\mathbf{r}} = \frac{s^4 + k_3 s^3 + k_2 s^2 + k_1 s + k_0}{s^4 + k_3 s^3 + k_2 s^2 + k_1 s + k_0} \vec{\mathbf{r}}_r \tag{6.110}$$

$$\begin{cases} k_3 = 4\omega \\ \dfrac{k_2}{k_3} = \dfrac{3}{2}\omega \\ \dfrac{k_1}{k_2} = \dfrac{2}{3}\omega \\ \dfrac{k_0}{k_1} = \dfrac{1}{4}\omega \end{cases} \tag{6.111}$$

Similar structure can be also used in the reference model to take trajectory commands, whose velocity and acceleration commands can be feedforward in the reference controller. High frequency (small sampling time) reference signal with all necessary derivatives can be generated from the trajectory commands, which may have larger time step or even gaps due to signal transmission.

6.3.2.3. Nonlinear limits in reference model and error controllers

The above design only builds up the structure to incorporate linear state limits; however, the quadrotor has the physical limits on the nonlinear states:

- The maximum speed, about 10 to 15 m/s depending on the propeller types.
- The maximum total thrust, depending on the propellers and motors. For the 'Hummingbird', it is around 14N.
- The maximum angular rate, limited by the gyroscope sensor range. It is 400°/s for the 'Hummingbird'.
- The maximum angular accelerations, limited by the maximum moments. For the 'Hummingbird', the pitching and rolling moments are about 0.6 Nm, while the yawing moment is 0.1Nm.

The nonlinear dynamic inversion idea can be applied here in the reference model to incorporate the nonlinear state limits, as well as in the actual error controllers. In fact, this technique is widely used in the classical control of fixed wing aircraft. The limitation block of each 'control loop' can be modified as such, first the nonlinear state (which has physical limits) is computed from the linear one and set the limits, then invert the limited nonlinear state back the linear state.

The position and velocity limits can be directly limited on $\vec{\mathbf{r}}_c$ and $\vec{\mathbf{v}}_d$ without any conversions. The force limits need to convert the desired acceleration $\vec{\mathbf{a}}_d$ to the thrust vector. The nonlinear conversion could be done via Eq. (6.14):

$$\mathbf{f}_2^{-1}: \begin{bmatrix} g_{x,d} \\ g_{y,d} \\ -T_d/m \end{bmatrix}_B = \mathbf{M}_{BW} \cdot \begin{bmatrix} a_{x,d} \\ a_{y,d} \\ a_{z,d} \end{bmatrix}_W - \begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B \tag{6.112}$$

The Eq. (6.8) could also be used to compute the thrust, however to convert the thrust back to the acceleration after limiting the thrust could be tricky here. A proportional scaling on the acceleration might end up reducing the total thrust and attitude together. Using the above equation, the thrust and the gravitational components could be limited separately. The maximum thrust limit applies only on the z-axis of the Body-fixed frame but does not affect the attitude, which is represented by the gravitational components and might be separately limited if necessary. It could be converted back to the desired accelerations by Eq. (4.56) after imposing the limits,

$$\mathbf{f}_2\text{:}\ \ \vec{\mathbf{a}}_d \approx \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_x \\ g_y \\ -T/m \end{bmatrix}_B \right) \tag{6.113}$$

Next, the desired acceleration derivatives $\dot{\vec{\boldsymbol{a}}}_{\boldsymbol{d}}$ can be converted to the angular rate using Eq. (6.20).

$$\mathbf{f}_3^{-1}\text{:}\ \ \begin{bmatrix} p \\ q \\ \dot{T} \end{bmatrix}_d = \begin{bmatrix} 0 & T^{-1} & 0 \\ -T^{-1} & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \mathbf{M}_{BW} m \cdot \dot{\vec{\mathbf{a}}}_{\boldsymbol{d}} \tag{6.114}$$

The pitch and roll rate can be limited accordingly then converted back using Eq. (6.19).

$$\mathbf{f}_3\text{:}\ \ \dot{\vec{\mathbf{a}}}_{\boldsymbol{d}} \approx \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \begin{bmatrix} 0 & -T & 0 \\ T & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} p \\ q \\ \dot{T} \end{bmatrix} \tag{6.115}$$

The desired acceleration 2nd derivatives $\ddot{\vec{\mathbf{a}}}_{\boldsymbol{d}}$ can be converted to the angular acceleration using Eq. (6.26). After limiting, it needs to be converted back by using Eq. (6.25).

$$\mathbf{f}_4^{-1}\text{:}\ \ \begin{bmatrix} \dot{p} \\ \dot{q} \\ \ddot{T} \end{bmatrix}_d = \begin{bmatrix} 0 & T^{-1} & 0 \\ -T^{-1} & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \left(\mathbf{M}_{BW} m \cdot \ddot{\vec{\mathbf{a}}}_{\boldsymbol{d}} - \begin{bmatrix} -Trp - 2\dot{T}q \\ -Trq + 2\dot{T}p \\ Tp^2 + Tq^2 \end{bmatrix} \right) \tag{6.116}$$

$$\mathbf{f}_4\text{:}\ \ \ddot{\vec{\mathbf{a}}}_{\boldsymbol{d}} = \mathbf{M}_{WB} \cdot \frac{1}{m} \cdot \left(\begin{bmatrix} -rpT - 2q\dot{T} \\ -rqT + 2p\dot{T} \\ q^2T + p^2T \end{bmatrix} + \begin{bmatrix} 0 & -T & 0 \\ T & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \ddot{T} \end{bmatrix} \right) \tag{6.117}$$

In those equations, all the necessary plant parameters and states can be computed from the linear reference state (from the integrator chain). The transformation matrix $\mathbf{M}_{WB}$ is calculated from the reference acceleration,

$$\mathbf{M}_{WB,r} = \begin{bmatrix} \frac{a_z - g}{a_{xz}} & \frac{a_x a_y}{a_{xz} f_z} & \frac{a_x}{f_z} \\ 0 & \frac{a_{xz}}{f_z} & \frac{a_y}{f_z} \\ -\frac{a_x}{a_{xz}} & \frac{a_x a_y}{a_{xz} f_z} & \frac{a_z - g}{f_z} \end{bmatrix} \tag{6.118}$$

Where,

$$\begin{cases} f_z = -\sqrt{a_x^2 + a_y^2 + (a_z - g)^2} \\ a_{xz} = -\sqrt{a_x^2 + (a_z - g)^2} \end{cases}$$

To illustrate the idea, the nonlinear conversions are denoted by a nonlinear function blocks $\mathbf{f}_i$ as shown in Figure 6.14.

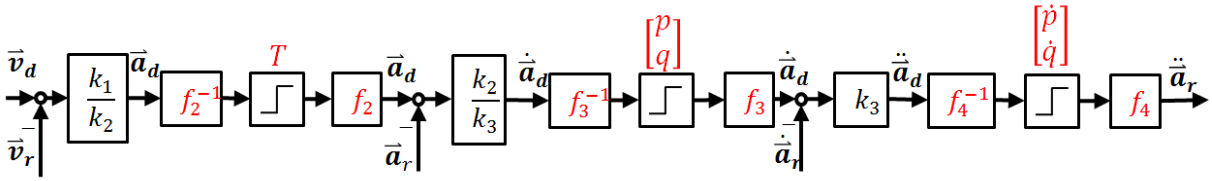


Figure 6.14 Nonlinear state conversion and limits

With the above modification, smooth, physically capable and more importantly consistent reference signals can be generated for the controllers. The stability of the reference model is well preserved because the nonlinear modification is done on the limit blocks of the reference controller.

In addition, nonlinear reference state can be also computed from the linear ones and provided to the controllers if necessary.

6.4. Comparison and Results

There are in total six different position control designs: three NDI designs and three Backstepping designs. Five of the six need to use filter in the controllers because the quadrotor position system is of non-triangular form. The only non-cascaded control design given in section 6.1.3 used a dynamic extension to transform the system into triangular form. In the five filtered control designs, design b1) and b2) use cascaded NDI structure; design g1) and g2) use Backstepping with filter and design h) is CFB based on design g2).

The comparisons are categorized into three groups. The first group compares the two variations in Design b1): the Euler angle approach and the gravitational vector approach. The second group compares the three NDI approaches: NDI-RD2 design, NDI-RD3 design, and NDI-RD4 design. The third group compares the Backstepping designs g1), g2) and h) with the best NDI controller, the RD3 design.

The high-fidelity simulation results are more often used for the comparison for the better clarity. In addition, the second comparison has additional VICON test data. Unfortunately, it is not available for other control designs. The experimental data from the Webcam system is not used for comparison because of the speed and position limitation. With low bandwidth maneuvers, the control performance is rather the same for all controllers. The GPS flight data is also not suitable because of the big position measurement errors. Nevertheless, the experimental data will be presented in Chapter 8.

6.4.1. Euler angle VS gravitational vector

Design b1) is the most intuitive design structure when it comes to the quadrotor position control. The inner loop governs the attitude, while the output loop governs the point-mass translational motion. There are two versions with different control states, one with the conventional Euler angle parameterizations, and the other with the gravitational vector parameterization. The gravitational vector has obvious advantages over the Euler angle,

- **Fast disturbance rejection and no steady-state errors** due to the aerodynamic drag in high speed flight. External disturbances can be actively compensated by the control feedback of the accelerator measurements. In other words, the gravitational vector can compensate the disturbances at acceleration level, which of course is fast and aggressive. While the Euler angle approach can only react after the disturbances have been propagated to velocity and position. Therefore, the gravitational vector approach has better position tracking, while the Euler angle approach has more stable and robust attitude control.
  Figure 6.15 shows the result in the simulation test of a constant 5 m/s speed flight. The inner loop command $g_{x,cmd}$ at steady state is the negative value of the measured acceleration f_x on the same axis. In comparison, the Euler angle approach will generate equivalent angle commands from constant position errors, which is relatively slow.
  The disturbance rejection performance is also improved as shown in Figure 6.16. Both approaches are subject to the same amount of disturbances. The gravitational vector

approach (in blue) reacts much faster than the Euler angle approach (in red), hence smaller velocity and position errors arise under the same level of disturbances.

- **Simplification and efficiency**. Limited payload or limited computation power is a normal problem for small UAVs like the quadrotors. The hummingbird is originally equipped with ARM 7 microprocessor, which is only capable of fixed-point computation. The trigonometric functions in the Euler angle approach are slow and need a lot of resources to run on embedded systems. The new parameterization embeds the attitude information in the transformation matrix, which can be efficiently calculated using Quaternions in the data fusion. In my implementation on the ARM 7, the new approach save about 20%-40% of computation power than the Euler angle approach. Once again, the detailed implementations will be introduced in Chapter 8.
- **Intuitive interpretation and better nonlinear mapping**. From Newton's second law, the translational motion is only actuated by force, not by Euler angles or anything else. There are two dominate forces on the quadrotors: the thrust force and the gravitational force. The thrust can be easily controlled by the motors. After rotating the gravitational vector in the Body-fixed frame, the gravitational force is no longer constant but a control variable. With the Euler angle as control variable, the error dynamics is linear over the angle ranges. However, with gravitational vector, the mapping is more reasonable: at low angle range, the control of gravitational force is more effective; while at high angle range, the control of gravitational force is much less effective. This is due to the sine mapping as shown in Eq.(6.119). With this mapping, the constant gain set on the gravitational error is more aggressive at small angles but reluctant at high angles, which is good for the robustness considerations.

$$\begin{bmatrix} g_x \\ g_y \end{bmatrix} = g \cdot \begin{bmatrix} \sin\Theta\cos\Phi \\ -\sin\Phi \end{bmatrix} \tag{6.119}$$

- An additional advantage of the gravitational vector parameterization is that the Backstepping design g1) is made possible by the new parameterization. The intermediate control variables and thrust command is decoupled and the analytical solution to the next inner state is thus possible.

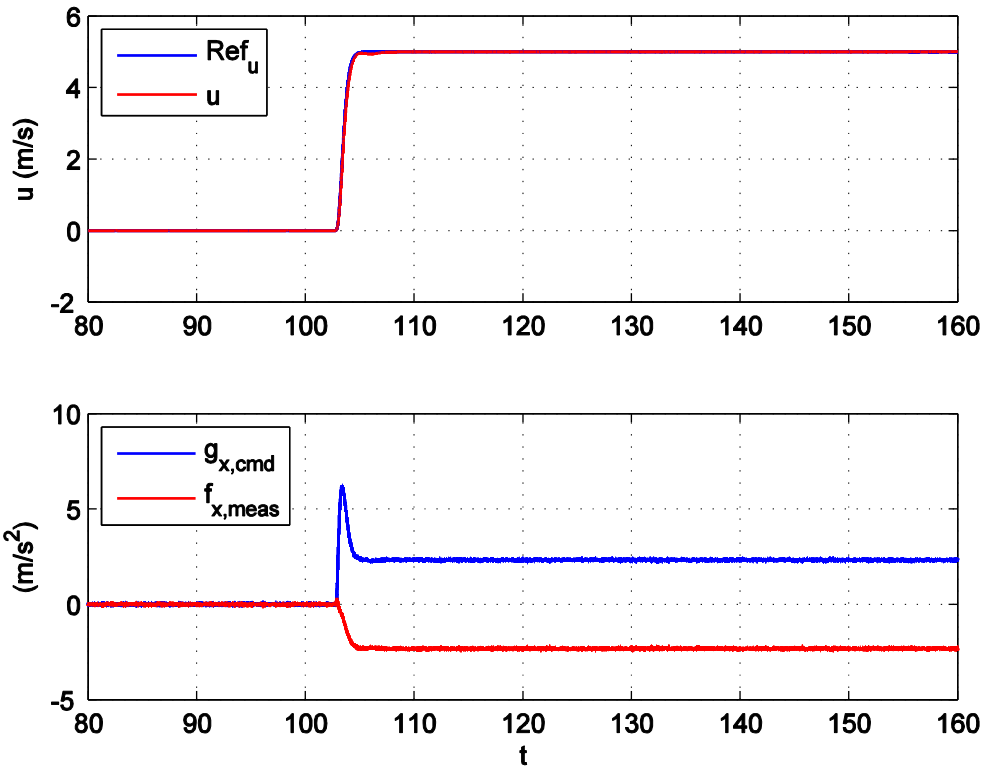


Figure 6.15 Direct usage of accelerometer in the case of constant speed flight

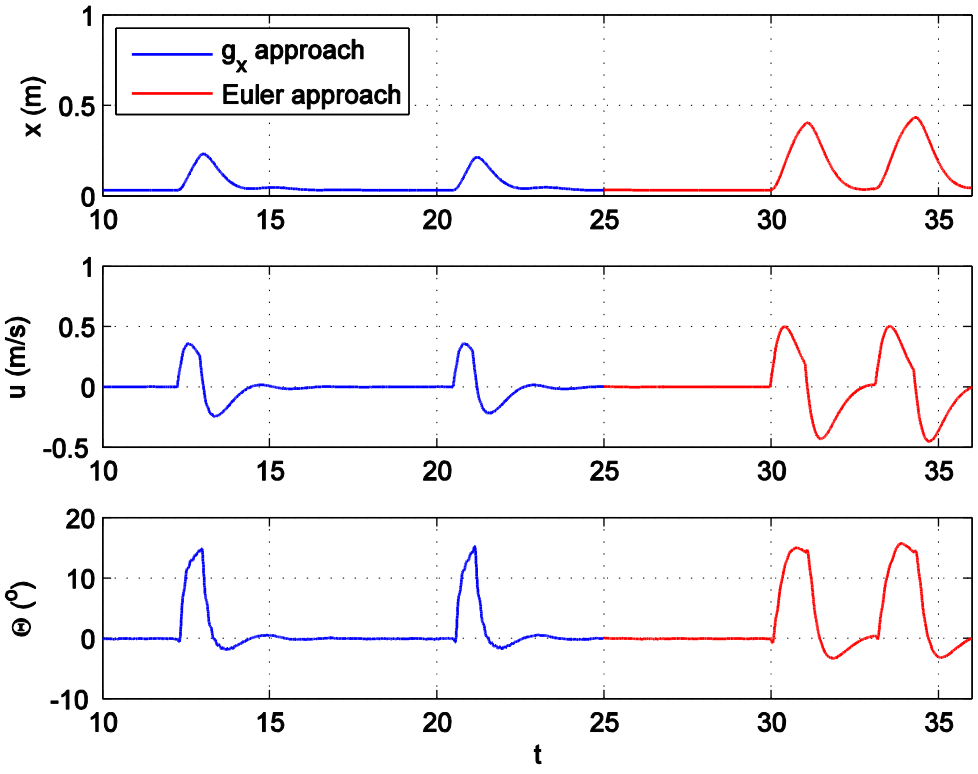


Figure 6.16 Disturbance rejections of the two approaches

6.4.2. NDI Design Comparisons: RD2 VS RD3 VS RD4

The structural difference between design b1), and b2) is the separation of inner and outer loops. Design b1) has RD2 outer loop and RD2 inner loop. Design b2) extends the outer loop to the next inner state, from the acceleration to the acceleration derivative. In terms of control states, the attitude state (either gravitational vector or Euler angles) is replaced by the acceleration $(\vec{\mathbf{a}})_W^{WW}$ but the rest stays the same. Design a) extends it even further and the inner loop is simply the actuator dynamics.

This modification makes an impact mainly on the feedforward control. In design b1), the feedforward signals are the reference position, velocity and acceleration because those are in the outer loop. Due to the stability assumption of TSS, the feedforward signals cannot go into the inner loop. However, in design b2), the outer loop extends to the acceleration derivative, so the reference acceleration derivative can be also used in the feedforward control, which could potentially increase the control bandwidth. The feedback control is merely affected by the different structures.

The design a) is a non-cascaded design: one loop control of RD4. This enables the feedforward of the 2nd derivative of the reference acceleration (or fourth derivative of the position). Theoretically, this structure could result the highest control bandwidth.

To see clearly the differences between the RD2, RD3 and RD4 structures, the reference trajectory in Figure 6.12 is commanded to all three controllers in an ideal simulation without uncertainties (e.g. sensor errors, parameter uncertainties and external disturbances). Therefore, the actuator system is the main limiting factor. The resulting position tracking errors are shown in Figure 6.17. As expected, the design a) gives the smallest tracking error as well as error oscillations. The tracking performance of design b2) is rather close to design a). Design b1) has much bigger tracking error compared to the other two designs. This error decreases if the bandwidth of reference signal is tuned to be smaller, which simply means the design b1) has smaller control bandwidth than the other two designs.

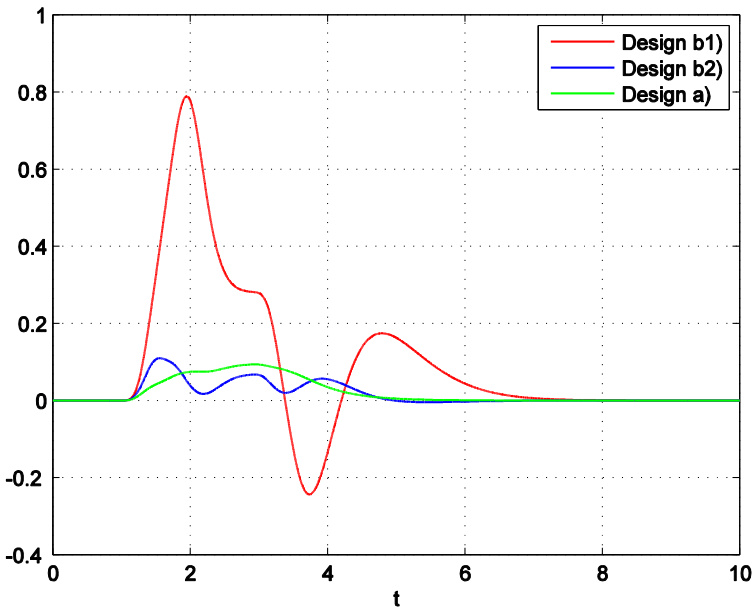


Figure 6.17 Position tracking errors of three NDI designs in ideal simulation

In high fidelity simulation and real flight test, the comparison results are not the same as in the ideal simulation environment. On the contrast, the design a) shows poor performance and robustness. The feedforward of the acceleration second derivative cause two problems:

- Oscillation and even instability under external disturbances. The model uncertainties increase dramatically from RD3 model to RD4 model. The rate gyro noises, structural vibrations and propeller turbulences are not taken into account in the control system design. When performing high bandwidth maneuvers in the presence of large uncertainties, the feedforward signal of the reference acceleration 2nd derivative, which will be converted to the angular accelerations, might be far away from the actual desired values and thus deteriorate the performance and stability. The result is larger tracking errors, oscillations and even instability.
- The rate gyro equipped on the 'Hummingbird' has a measurement range of 400°/s. A pure rate-feedback control design could limit the angular rate command to avoid sensor saturation. However, in the design a), unmatched feedforward angular accelerations (due to e.g. disturbances) could potentially saturate the sensor, which could jeopardize the whole data fusion. This failure happened during the flight test: The rate saturated for a few second, then the attitude estimation is directly affected and the quadrotor lost its stability.

To sum up, the quadrotor rate loop is not suitable to apply any feedforward control. For agile platform like the quadrotor, the rate control loop should mainly target on disturbance rejection rather than command taking. Much slower reference model has to be used for design a) in order to have a stable performance. It is not a robust structure for real flight.

In the high fidelity simulation test, the design b1) and b2) are compared. In Figure 6.18, the position tracking performance subject to a 3-2-1-1 signal is presented. The design b2) (in green) have obviously larger overshoot and tracking errors compared to the design b1) (in red). The velocity comparison result in Figure 6.19 corresponds to the same conclusion. It can be noted that the maneuvers are not slow motion, which any controllers can track very well. The velocity peak reaches 8m/s.

In the VICON flight test, similar comparison results are obtained as shown in Figure 6.20 and Figure 6.21. The difference may not be as obvious as in the high fidelity simulation. The gain settings for the reference model and error controllers are smaller so more conservative for robustness considerations. This could be one of the reasons for the smaller performance differences. For low bandwidth trajectories, even minor difference could be observed as both are tracking the trajectories very well. Nevertheless, the design b2) still has better tracking performance in terms of overshoot and error margins.

Overall, the design b2) is the best NDI design for its high control bandwidth, performance and robustness. The design a) is not robustness enough for quadrotor application as there are too much model assumptions. The design b1) has less control bandwidth but robust and intuitive structure so it may be suitable for low bandwidth control problem, e.g. video/photo taking task.

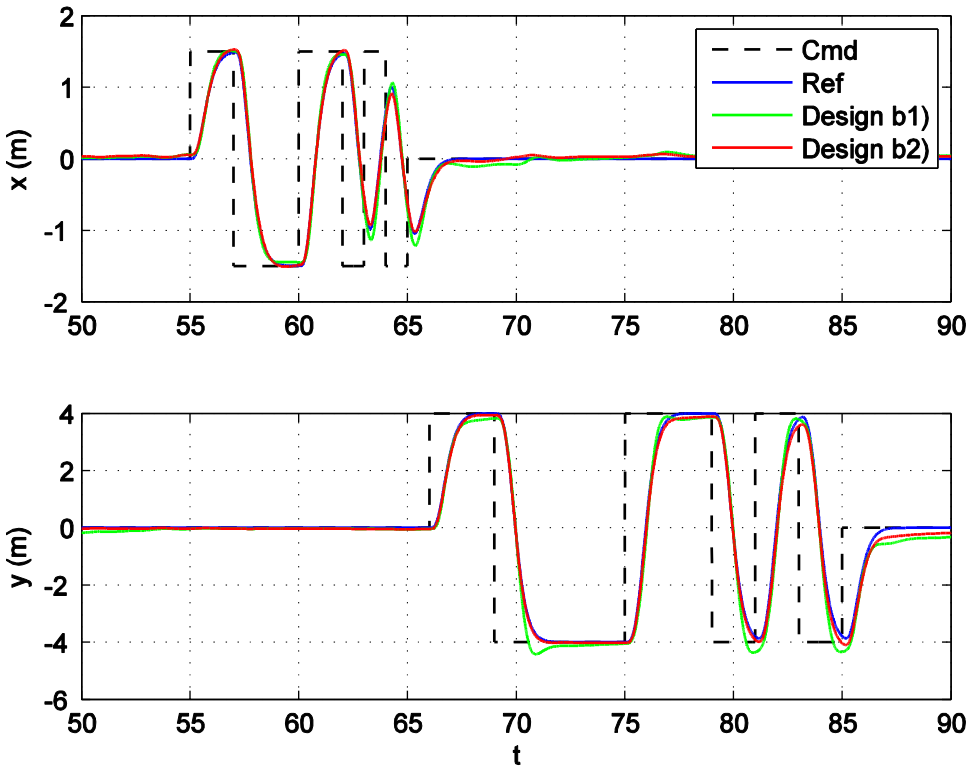


Figure 6.18 Position tracking comparison between design b1) and b2) in high fidelity simulation

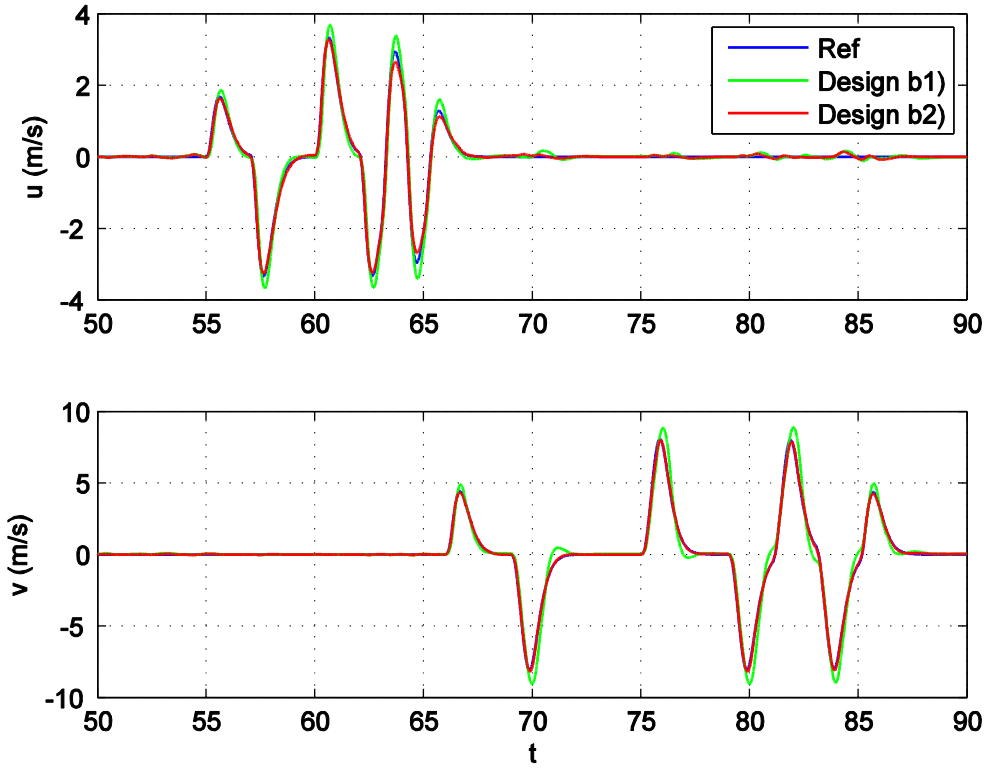


Figure 6.19 velocity comparison between design b1) and b2) in high fidelity simulation

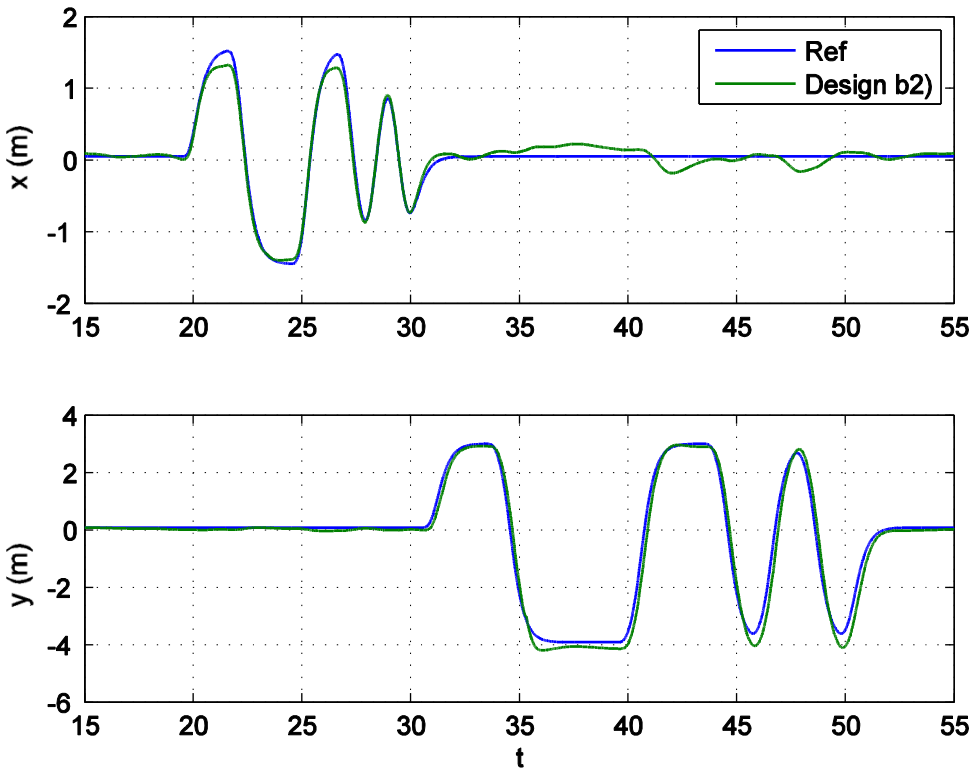


Figure 6.20 Position tracking performance of design b1) in VICON test

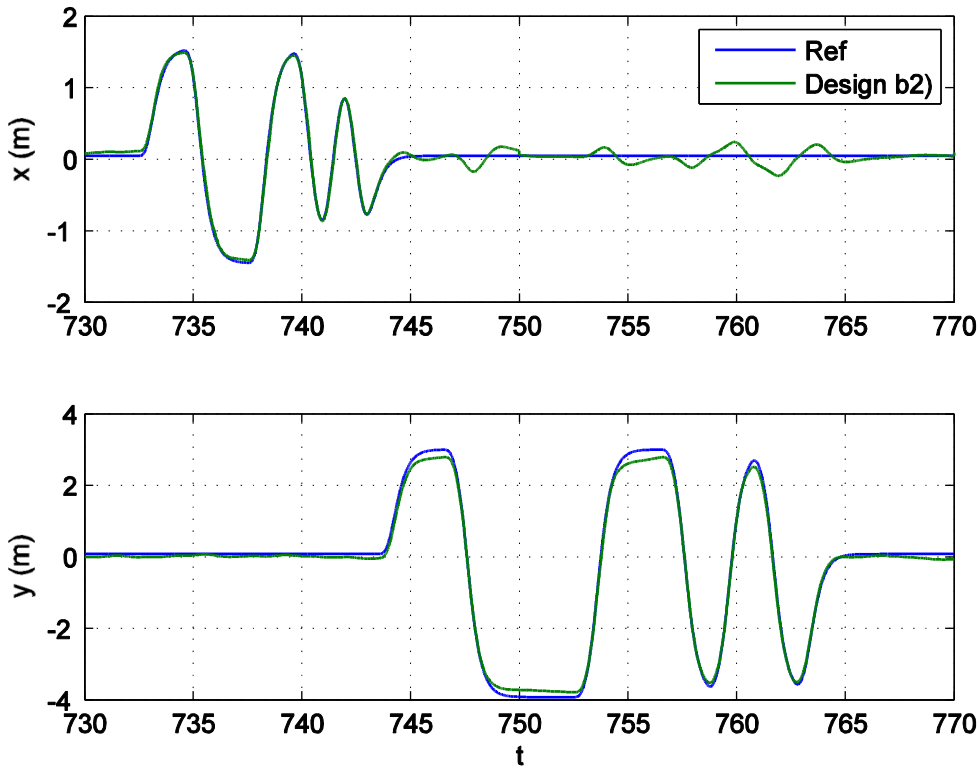


Figure 6.21 Position tracking performance of design b2) in VICON test

6.4.3. Backstepping Designs VS NDI RD3 Design

In this section, the three Backstepping designs are compared first, and then the best of Backstepping is compared with the best of the NDI designs.

Due to the non-triangular form, the Backstepping position controllers have to use a command filter on the angular rate commands. There are two 'Backstepping design with command filter', g1) and g2). The design g1) uses the new gravitation parameter as attitude control but the design g2) uses the acceleration instead plus a dynamic extension as the NDI design b2). The design h) is the CFB design that is derived based on design g2).

First, the design structures of g1) and g2) are similar to the NDI RD3 control structure. The feedforward signals contain the same information, from reference position to the reference acceleration. The difference is on the thrust control. Design g1) has direct thrust command to the actuators. Design g2) commands thrust derivative due to the dynamic extension. However, their performance is still almost the same due to the following two reasons.

- The gain design on the thrust axis can be higher so that the thrust derivative is still high bandwidth signals.
- The small phase lag between the direct thrust command and the integrated thrust command is negligible due to the thrust dynamics.

The CFB design h) modified the design g2) by adding compensated filter error feedback, the $\boldsymbol{\xi}_i$ terms. The main structure remains the same as design g2). As analyzed in the previous sections with the generic example and attitude control example, these modifications have little effect on the overall performance. The control laws for the design b2), g2) and h) are collected below for comparison,

$$\begin{cases} b2): \mathbf{u}_2 = \boldsymbol{G}_4^{-1}\left(-\mathbf{f}_4 + \dot{\vec{\boldsymbol{\omega}}}_d\right) = \mathbf{G}_4^{-1}\left(-\mathbf{f}_4 + \dot{\hat{\boldsymbol{\alpha}}}_{3f} + \mathbf{A}_4\hat{\mathbf{z}}_4\right) \\ g2): \mathbf{u}_2 = \mathbf{G}_4^{-1}\left(-\mathbf{f}_4 + \dot{\hat{\boldsymbol{\alpha}}}_{3f} + \begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3 + \mathbf{A}_4\hat{\mathbf{z}}_4\right) \\ h): \mathbf{u}_2 = \mathbf{G}_4^{-1}\left(-\mathbf{f}_4 + \dot{\hat{\boldsymbol{\alpha}}}_{3f,CFB} + \begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}(\mathbf{z}_3 + \boldsymbol{\xi}_3) + \mathbf{A}_4\hat{\mathbf{z}}_4\right) \end{cases}$$

In the high-fidelity simulation, the four designs have almost the same performance, as shown in Figure 6.22. The position tracking errors are at the same magnitude level as in Figure 6.23. The control signals are compared in Figure 6.24: the control signal $\dot{\vec{\boldsymbol{\omega}}}_d$ in the design b2), the additional Backstepping term $\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\mathbf{z}_3$ in the design g2), and the additional error compensation term $\begin{bmatrix}\mathbf{G}_3^T(1:2,:)\\ \mathbf{0}_{1\times 3}\end{bmatrix}\boldsymbol{\xi}_3$ in the design h).

The results show that the differences between the four controllers are again negligible and the additional modification terms in the Backstepping designs have no effect on the position control as well. The best performing controller is the NDI RD3 design b2), for the high control bandwidth and high robustness.

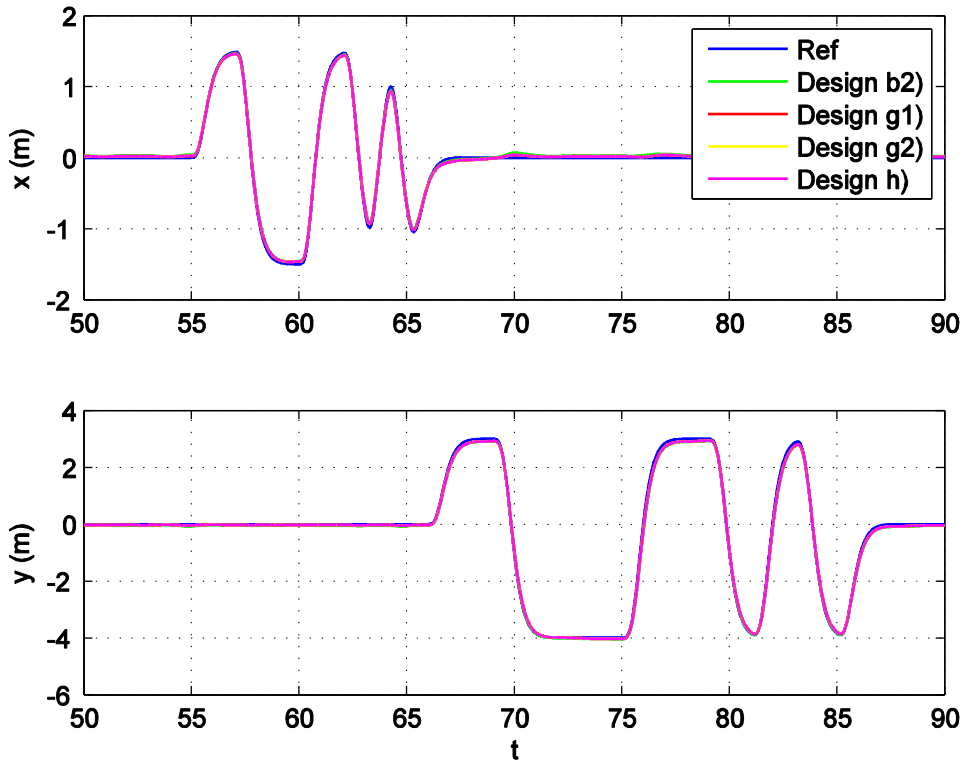


Figure 6.22 Position tracking performance of design b2), g1), g2) and h) in high fidelity simulation

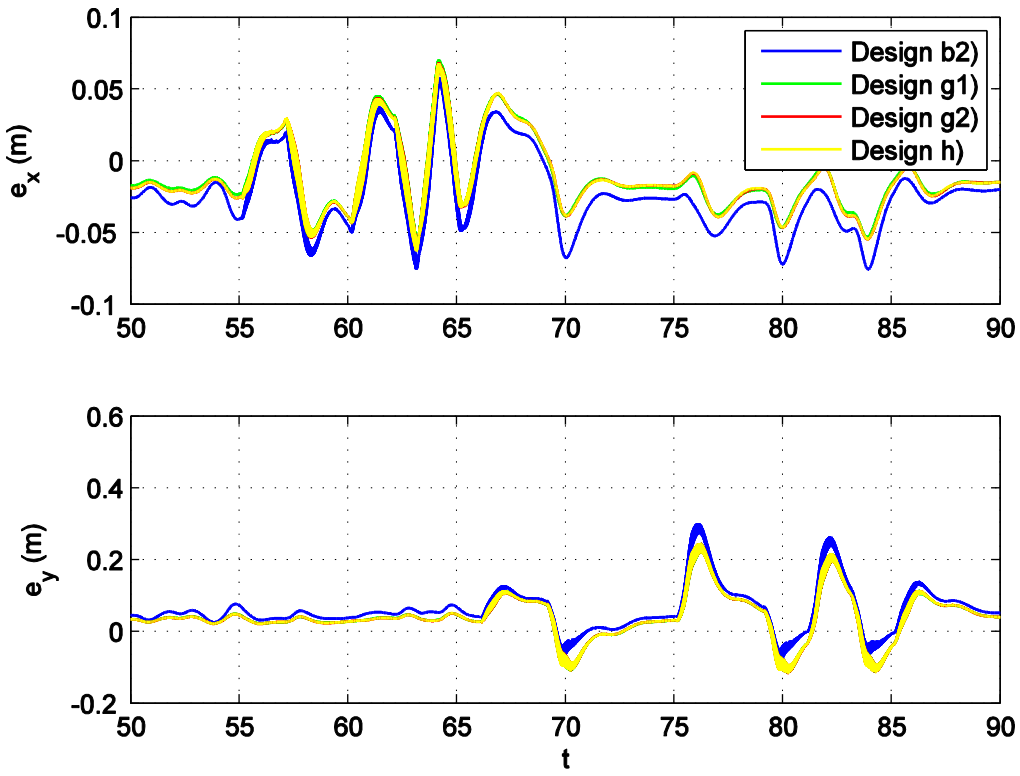


Figure 6.23 Position tracking error of design b2) g1), g2) and h) in high fidelity simulation

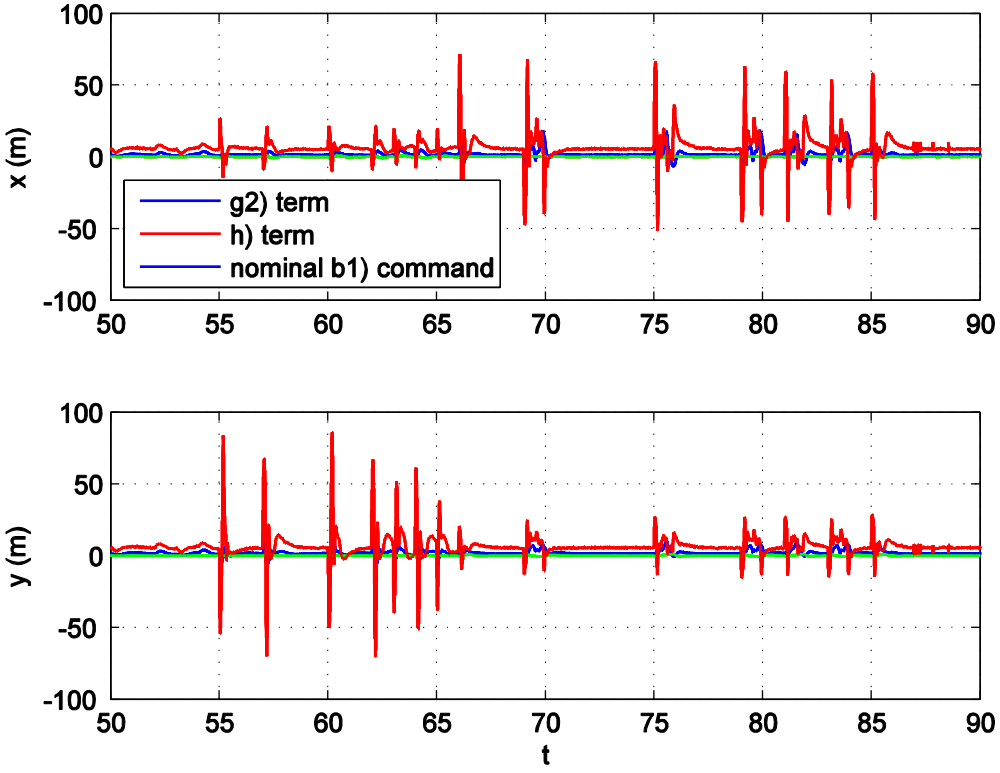


Figure 6.24 Angular rate commands of design b2) g1), g2) and h) in high fidelity simulation

6.4.4. Overall performance comparison

The overall performance of the different nonlinear position control designs in this thesis is now compared using several important criteria, such as tracking performance and design complexity. A table with the results of this comparison can be found in Table 6.1. It can be seen that the NDI RD3 position control design outperforms the other designs.

	NDI b1)	NDI b2)	NDI a)	BC g1)	BC g2)	BC h)
Tracking Performance	★★	★★★	★★	★★★	★★★	★★★
Control Bandwidth	★	★★	★★★	★★	★★	★★
Disturbance Rejection	★★★	★★	★	★★	★★	★★
Design Complexity	★★★	★★★	★★	★	★★	★
Tuning Complexity	★★★	★★★	★★★	★	★★	★★

Table 6.1 Comparison of nonlinear position control designs

(three-star indicates the best, two-star is good / acceptable and one-star is bad / negative. These three categories are also indicated by green, yellow and red, respectively)

Tracking Performance

The NDI RD3 control design has the best tracking performance. It has more model assumptions than the RD2 design, but not as many as the RD4 design to ruin the structural robustness. The Backstepping designs are based on the RD3 concepts, so the tracking performance is rather similar.

Control Bandwidth

The control bandwidth is predominately determined by the actuator dynamics but can be improved by feedforward control. The NDI RD2 design has feedforward control until the attitude loop, the NDI RD3 design until the rate loop, the RD4 design until the actuator commands. Thus, theoretically the NDI RD4 design has the highest control bandwidth. The Backstepping design has similar RD3 structure thus similar feedforward control and control bandwidth.

Disturbance Rejection

The NDI RD4 design has too much model assumptions thus poor performance against external disturbances. The RD3 designs have acceptable disturbance rejection performance. In the implementation, the aggressiveness of the disturbance response can be tunable by the modifications of accelerometer feedback. The NDI RD2 design is most stable against disturbances, as more model uncertainties can be compensated.

Design Complexity

The NDI in general are much simpler than the Backstepping designs, even the RD4 design is acceptable. The Backstepping design g1) is quite complicated for the nonlinear states are involved in the analytical derivation. The design g2) is slightly better because the analytical part only involves linear states. The CFB design h) also gets complicated due to the derivation of the compensated tracking error.

Tuning Complexity

The tuning effort for the NDI designs is rather small, linear gain design plus minor experimental tuning. For the Backstepping designs, the gain design and tuning is difficult, especially with original model in the design g1).

7. Augmented Adaptive Control

It is an imperfect world, and there are always disturbances and uncertainties in the real applications. Adaptive control is well known to account for disturbances and uncertainties. For an agile platform like the quadrotor, fast adaptation with good transient performance is essential. The $\mathcal{L}_1$ adaptive scheme is chosen because it has guaranteed fast adaptation with transient stability without persistent excitation [42]. Besides good performance, its linear-like structure is very simple to implement.

It should be noted that the adaptive control is only augmented to the baseline controllers, either NDI or Backstepping. The state vector in the adaptive controller is an error state but not the full state. The overall augmented control structure is shown in Figure 7.1. The baseline controller (in blue) controls the nominal plant, while the augmented adaptive control accounts for uncertainties and disturbances.

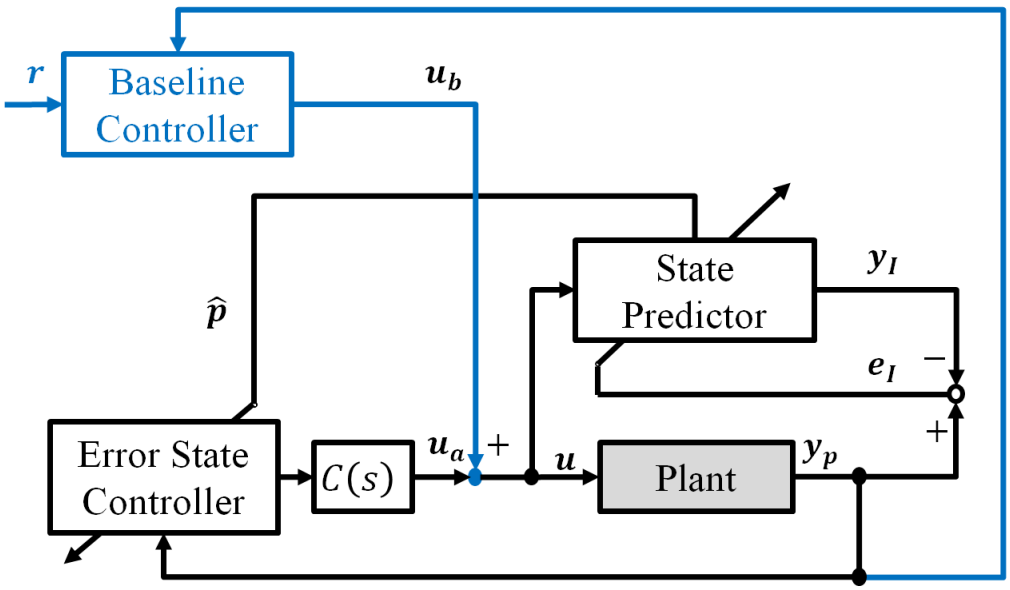


Figure 7.1 Overall control structure with augmented $\mathcal{L}_1$ control

The advantage over full adaptive control scheme is that the knowledge of nominal plant dynamics can be actively used in the baseline controller while the adaptive control only needs to do what it does best: compensate uncertainties and disturbances. The total control input is splitted into two parts, one parts is a confident part which can be operated with maximum control bandwidth limited by the plant physics and the other parts is a estimation part whose control bandwidth is not only limited by the plant physics but also the transient dynamics of the adaptation. Hence, the overall control bandwidth of the combined approach can be higher than the full adaptive control approach.

Recall the full adaptive $\mathcal{L}_1$ controller from Figure 2.11, we can see that the controller output needs go through the $\mathcal{L}_1$ filter $C(s)$ before entering into the plant. However in the combined approach in Figure 7.1, the baseline controller output $\mathbf{u}_b$ goes to the plant directly, i.e., it is only limited by the actuator dynamics, and the adaptive control output $\mathbf{u}_a$ has to filtered by

the $\mathcal{L}_1$ filter. For quadrotor application, where most of plant dynamics can be easily identified, the combined approach is certainly favorable. Nevertheless, for systems with little knowledge or large uncertainties, full adaptive maybe more appropriate.

7.1. Disturbances and Uncertainties

Reviewing the mathematical model in section 4.3, the position and attitude propagations are simple kinematics therefore have few uncertainties. Most of the system uncertainties lie in the rotational dynamics and the control allocation. These uncertainties will affect the cancellation of nonlinearities in NDI or Backstepping controls. In control designs, the nominal or estimated values are used. The errors are often called inversion errors.

Let's first recall the dynamic equations from the angular accelerations to the motor RPMs, Eq. (4.39) and Eq. (4.41).

$$\dot{\vec{\boldsymbol{\omega}}} = \mathbf{I}^{-1} \cdot \vec{\mathbf{M}} - \mathbf{I}^{-1} \cdot \vec{\boldsymbol{\omega}} \times \mathbf{I}\vec{\boldsymbol{\omega}}$$

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} 0 & -R & 0 & R \\ R & 0 & -R & 0 \\ -k_m & k_m & -k_m & k_m \end{bmatrix} \begin{bmatrix} k_n \cdot n_1^p \\ k_n \cdot n_2^p \\ k_n \cdot n_3^p \\ k_n \cdot n_4^p \end{bmatrix}$$

The uncertain parameters include the moment of inertia $\mathbf{I}$, the force and torque constant k_n and k_m, and the power index p. Moreover, the misalignment of the motor position and axis, the performance differences between the motors and propellers, and the propeller deformations at high speed are all sources of uncertainties. The adaptive control can play an important role here and take some load off the error controller in the baseline controller.

For quadrotors, disturbances are mainly the aerodynamic forces. These mainly lie in the translational dynamics. There are few uncertainties. Recall the dynamic equations represented by the new parameter,

$$\left(\dot{\vec{\mathbf{v}}}_k^G\right)_W^{WW} = \mathbf{M}_{WB} \cdot \left(\begin{bmatrix} f_x \\ f_y \\ g_z \end{bmatrix}_B + \begin{bmatrix} g_x \\ g_y \\ \Delta f_z - T/m \end{bmatrix}_B \right)$$

$$T = k_n\left(n_1^p + n_2^p + n_3^p + n_4^p\right)$$

From practical point of view, the adaptive control does not perform better than linear error control for the aerodynamic disturbances on the quadrotor. Moreover, the disturbances in the Body-fixed x-y-axis can be measured by the accelerometer, while the z-component can be assumed negligible, because in the same axis the uncertainty in thrust is much larger. The mass m can be easily measured with good accuracy.

After analyzing the disturbances and uncertainties, the necessary place to augment the adaptive controller is at the rate control loop, because most of the parameter uncertainties are there. In addition, the best sensor onboard is often the rate gyros, which has high update rate and relatively low sensor bias and noises, so the adaptive controller can provide accurate estimate and its performance will not be affected a lot by the sensor errors. On the contrast, for the force dynamics, possible adaptive element can also be implemented, e.g. for the thrust/height dynamics. However, considering the sensor quality of Image sensor or GPS, the performance of the adaptive controller can be largely affected. That is the main reason no adaptive element is implemented elsewhere than the rate loop.

7.2. Adaptation Law

The adaptive control on the rate loop can be augmented to any of nonlinear control designs above. The dynamic equation for the angular acceleration can be written as the sum of desired acceleration to be commanded and a nonlinear function, which groups all the parameter uncertainties including external disturbances into a single term,

$$\dot{\vec{\boldsymbol{\omega}}} = \dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{d}} + \Delta\vec{\mathbf{f}}_{\boldsymbol{\omega}} \tag{7.1}$$

The desired angular acceleration $\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{d}}$ is commanded to the plant, which contains two parts:

$$\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{d}} = \dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{b}} + \dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{a}} \tag{7.2}$$

The baseline command $\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{b}}$ is from the reference signals and the error controls, which assign the desired dynamics to the system. The second term $\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{a}}$ is augmented adaptive control, which tries to compensate the parametric uncertainties grouped as $\Delta\vec{\mathbf{f}}_{\boldsymbol{\omega}}$.

$$\dot{\vec{\boldsymbol{\omega}}} = \dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{b}} + \underbrace{\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{a}} + \Delta\vec{\mathbf{f}}_{\boldsymbol{\omega}}}_{\to \mathbf{0}} \tag{7.3}$$

To estimate the parametric uncertainties, the following state predictor is proposed,

$$\dot{\hat{\vec{\boldsymbol{\omega}}}} = \dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{d}} + \Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} + \mathbf{K}_{\boldsymbol{sp}}\left(\hat{\vec{\boldsymbol{\omega}}} - \vec{\boldsymbol{\omega}}\right) \tag{7.4}$$

The desired acceleration $\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{d}}$ is the same as commanded to the plant. The nonlinear estimate $\Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}}$ tries to estimate the nonlinearities in the real plant. The last term is simply a feedback term to make sure the stability of the state predictor.

The predictor error and parameter error are defined as,

$$\begin{cases} \tilde{\vec{\boldsymbol{\omega}}} = \hat{\vec{\boldsymbol{\omega}}} - \vec{\boldsymbol{\omega}} \\ \Delta\tilde{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} = \Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} - \Delta\vec{\mathbf{f}}_{\boldsymbol{\omega}} \end{cases} \tag{7.5}$$

The prediction error dynamic can be derived by subtracting the state predictor dynamics and the real dynamics.

$$\dot{\tilde{\vec{\boldsymbol{\omega}}}} = \Delta\tilde{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} + \mathbf{K}_{\boldsymbol{sp}}\tilde{\vec{\boldsymbol{\omega}}} \tag{7.6}$$

Following a similar argument as in Section 3.3 of [42], one can derive the adaptation law of piecewise constant type for the unknown parameter,

$$\Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} = \left(1 - e^{\mathbf{K}_{\boldsymbol{sp}}T_S}\right)^{-1}\mathbf{K}_{\boldsymbol{sp}}e^{\mathbf{K}_{\boldsymbol{sp}}T_S}\tilde{\vec{\boldsymbol{\omega}}} \tag{7.7}$$

where T_S is the time step of the control loop. In our quadrotor processer, the time step is 1ms. The adaptation gain is fixed for a fixed time step T_S and state predictor gain $\mathbf{K}_{\boldsymbol{sp}}$.

The adaptive control law is,

$$\dot{\vec{\boldsymbol{\omega}}}_a = -\boldsymbol{C}(s)\Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} \tag{7.8}$$

where $\boldsymbol{C}(s)$ is the low pass filter to filter high bandwidth content of the estimation due to the high gain adaptation.

The adaptive control law can be substitute into the dynamic equation (7.3),

$$\dot{\vec{\boldsymbol{\omega}}} = \dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{b}} - \boldsymbol{C}(s)\Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}} + \Delta\vec{\mathbf{f}}_{\boldsymbol{\omega}} \tag{7.9}$$

The baseline control $\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{b}}$ remains as it is, while the adaptive controller aims to compensate the nonlinearities $\Delta\vec{\mathbf{f}}_{\boldsymbol{\omega}}$ using the estimate $\Delta\hat{\vec{\mathbf{f}}}_{\boldsymbol{\omega}}$ obtained from the state predictor. The low pass filter $\boldsymbol{C}(s)$ is a tunable element so that the adaptive control only tries to compensate the nonlinearities within its actuator bandwidth yet not to excite the system by the sensor errors.

7.3. Results

The $\mathcal{L}_1$ adaptive control is augmented to the rate loop of the NDI RD3 position baseline control (design b2). To test the augmented adaptive controller, a weight of 180 gram is used to create some artificial disturbances, as shown in Figure 7.2. Considering the maximum payload of the Hummingbird is 200 gram, this is very large disturbance applied to the vehicle. Moreover, it is not attached in the CG point, but at the arm tip, directly under the motor, which can produce a large moment disturbance. The weight is attached and cut off during the flight so that the adaptation can be clearly visible.

Figure 7.2 Hummingbird with artificial moment disturbance

The adaptive parameter $\dot{\bar{\omega}}_a$ during the flight is shown in Figure 7.3 and the position tracking of the flight is given in Figure 7.4. The adaptive augmentation is switched on around t=230s and the adaptive parameter starts to adapt to disturbances and uncertainties. During the circular motion, there could be periodic disturbances or inversion errors as indicated by the adaptive parameter. At t=320s the weight is attached to the vehicle. The adaptive element $\dot{p}_a$ on the roll acceleration adapts to about 2 rad/s^2 to account for the moment disturbance of the weight. The fast adaptation of $\mathcal{L}_1$ control can be observed when the weight is cut off at t=390s, the adaptive element jumps back within half a second.

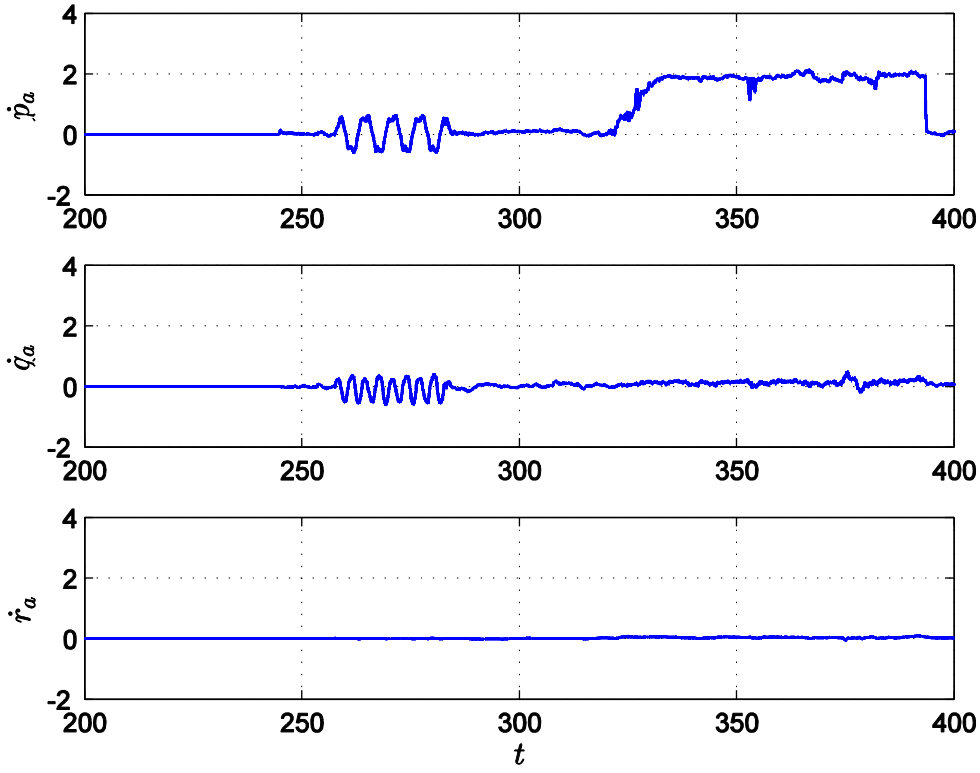


Figure 7.3 The adaptive parameter $\dot{\bar{\omega}}_a$ during the flight

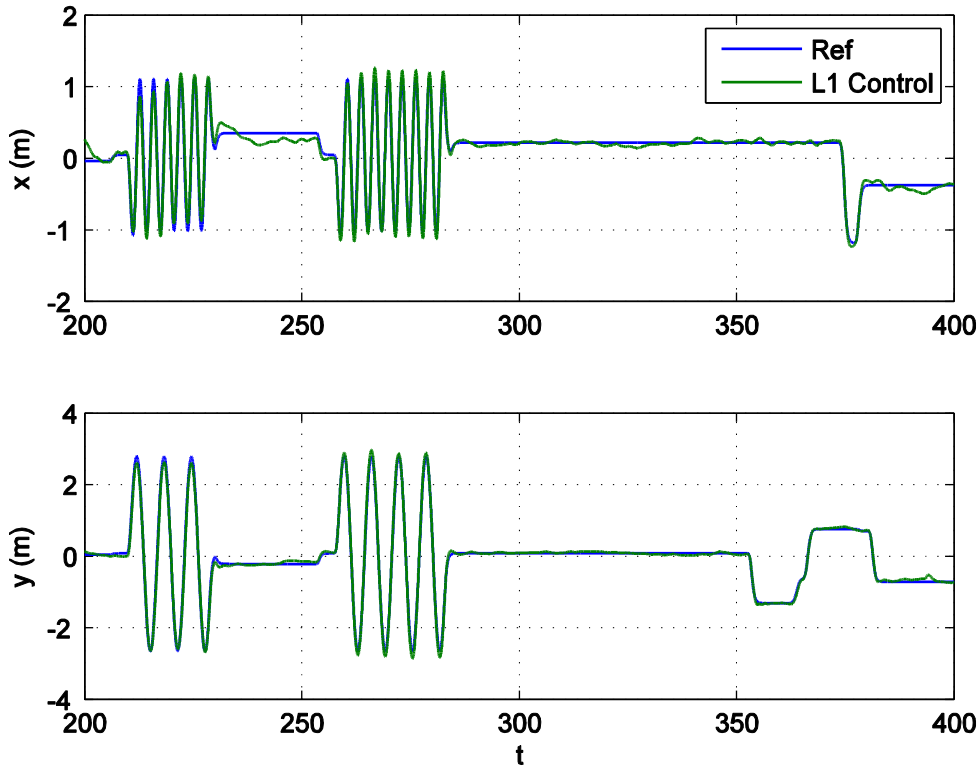


Figure 7.4 The position tracking of the augmented $\mathcal{L}_1$ control

8. Implementations

In this chapter, the hardware and software implementations are described in details. As mentioned earlier, there are three systems used for flight tests, webcam system, VICON system and GPS system. The flight test results are also presented in the respective sections.

8.1. Vision Position Systems

The two vision systems share the same experimental set-up (Figure 8.1), though certain parameters maybe necessary to be adjusted. The difference between the VICON and webcam systems is the quality of the measurements, and of course the huge cost difference. The position measurement performance of both systems is given in Table 8.1. The latency of the VICON system is mainly due to the communication delay, while the webcam system has additional about 60 ms delay due to the image processing. Additionally, as it utilizes the quadrotor edge contrast information for the detection, the speed is quite limited with the given frame rate. The edge in the image will be blurred when the speed is more than 2m/s. Nevertheless, the performance of the webcam is very satisfactory compared with its low cost.

Figure 8.1 VICON system and Webcam system

	VICON	Webcam
Accuracy	Millimeter	5~10 cm
Update rate	100 Hz	25 Hz
Latency	40 ms	100 ms
Range (x-y-z)	3mX7mX2m	1mX2mX1.5m

Table 8.1 VICON and Webcam system parameters

The overall system set-up is presented in Figure 8.2. The different processing speeds of the different subsystems are shown in red. The ground computer is used mainly as data communication platform. The commands to the Hummingbird are given by a joystick connected to the ground computer. The vision outputs from the vision system are transmitted

to the Hummingbird via. the ground computer. And the hummingbird send outputs back to the ground computer for debug and display.

All the data fusion and control algorithms are running on the Hummingbird. They are first programmed in the Matlab/Simulink and then converted to C code by Mathworks Real-time Workshop Embedded Coder. The implementation of the data fusion and control algorithms, and their Simulink development environment will be introduced in the next sections.

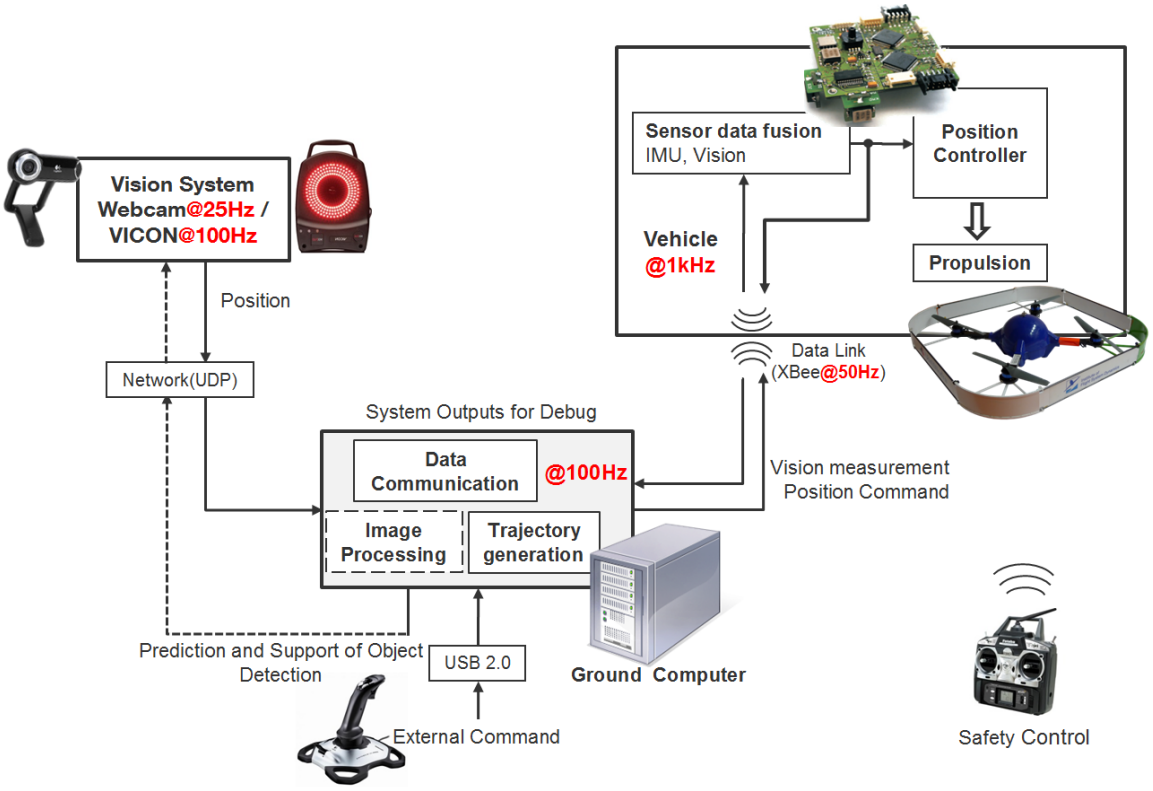


Figure 8.2 Vision system set-up

8.1.1. Position Observer with time delayed measurement

The cascaded data fusion structure is implemented for the vision system. The AHRS is responsible for the attitude estimation and the position observer responsible for the position and velocity estimation. Due to the cascaded structure, the vision system errors do not affect the pitch and bank angle estimation. The AHRS uses only the vision heading measurement for the azimuth angle correction. Standard AHRS implementation [52] can be applied. However, minor modification in the position observer is necessary to account for the time delay.

As stated earlier, the webcam system often has unmatched covariance, which is a common problem in vision detection. In our case, a constant covariance is assumed for the position measurement. The position observer is a simple but suitable filtering algorithm.

The fundamental problem in the design of an observer is the determination of the observer gain matrix such that the error dynamics is stable. The observer gain can be solved by either pole placement or optimization, i.e. solving an algebraic Riccati equation.

As stated earlier, the observer has the same structure of a Kalman filter, just without the covariance. Recall the math model of the position observer from section 0, the system equation and measurement equation are (The reference frame W is ignored for better visualization),

$$\begin{bmatrix}\dot{\vec{\mathbf{r}}}\\ \dot{\vec{\mathbf{v}}}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\vec{\mathbf{r}}\\ \vec{\mathbf{v}}\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathbf{I}\end{bmatrix}\vec{\mathbf{a}}$$
$$\vec{\mathbf{y}} = \vec{\mathbf{r}}_{vision} = [\mathbf{I} \quad \mathbf{0}]\begin{bmatrix}\vec{\mathbf{r}}\\ \vec{\mathbf{v}}\end{bmatrix} \tag{8.1}$$

The observer equation is then,

$$\begin{bmatrix}\dot{\hat{\vec{\mathbf{r}}}}\\ \dot{\hat{\vec{\mathbf{v}}}}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\hat{\vec{\mathbf{r}}}\\ \hat{\vec{\mathbf{v}}}\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathbf{I}\end{bmatrix}\vec{\mathbf{a}} - \begin{bmatrix}k_1\mathbf{I}\\ k_2\mathbf{I}\end{bmatrix}\left(\hat{\vec{\mathbf{r}}} - \vec{\mathbf{r}}\right) \tag{8.2}$$

The estimation error dynamics is,

$$\begin{bmatrix}\dot{\hat{\vec{\mathbf{r}}}} - \dot{\vec{\mathbf{r}}}\\ \dot{\hat{\vec{\mathbf{v}}}} - \dot{\vec{\mathbf{v}}}\end{bmatrix} = \begin{bmatrix}-k_1\mathbf{I} & \mathbf{I}\\ -k_2\mathbf{I} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\hat{\vec{\mathbf{r}}} - \vec{\mathbf{r}}\\ \hat{\vec{\mathbf{v}}} - \vec{\mathbf{v}}\end{bmatrix} \tag{8.3}$$

In the Matlab implementation, the observer gains can be design offline using the 'Kalman' commands, i.e. solving an algebraic Riccati equation. The measurement covariance of the accelerometer and position sensor is design parameters, and minor tuning of the covariance is necessary for the best performance.

Based on the classical Luenberger observer, two modifications are introduced for better performance in the implementation.

The first modification considers the accelerator bias: the x-y-axis bias cannot be observed due to the cascaded structure, i.e. the coupling with the attitude estimation; the z-axis bias, however, can be observed. Additional bias state b_z can be added to the observer equation (8.2). The assumption is to ignore the nonlinearities due to the frame rotation from W frame to the B frame.

The accelerometer bias on the z-axis is added to the velocity differential equation,

$$\dot{\hat{\vec{\mathbf{v}}}} = \mathbf{M}_{WB}(:,3)b_z + \mathbf{M}_{WB}\vec{\mathbf{f}}_B^{II} + \vec{\mathbf{g}}_W \tag{8.4}$$

Secondly, the bias is estimated by the position correction. However, the position error is better rotated into the body-fixed frame.

$$\dot{b}_z = -k_3\mathbf{M}_{BW}(3,:)\left(\hat{\vec{\mathbf{r}}} - \vec{\mathbf{r}}\right) \tag{8.5}$$

The expanded observer equation is,

$$\begin{bmatrix}\dot{\hat{\vec{\mathbf{r}}}}\\ \dot{\hat{\vec{\mathbf{v}}}}\\ \dot{b}_z\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{M}_{WB}(:,3)\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\hat{\vec{\mathbf{r}}}\\ \hat{\vec{\mathbf{v}}}\\ b_z\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ \mathbf{I}\\ \mathbf{0}\end{bmatrix}\left(\mathbf{M}_{WB}\vec{\mathbf{f}}_B^{II} + \vec{\mathbf{g}}_W\right) - \begin{bmatrix}k_1\mathbf{I}\\ k_2\mathbf{I}\\ k_3\mathbf{M}_{BW}(3,:)\end{bmatrix}\left(\hat{\vec{\mathbf{r}}} - \vec{\mathbf{r}}\right) \tag{8.6}$$

The three gains k_1, k_2 and k_3 are determined by the Matlab '*kalman*' function as shown in Eq. (8.7), where the covariance $\mathbf{Q}$ and $\mathbf{R}$ is determined by the accelerometer noise $\boldsymbol{w}_a$, bias random walk $\boldsymbol{w}_b$ and position measurement noise $\boldsymbol{v}$. The final values are determined in experiment given in Table 8.2. It shall be noted that although the steady state accelerator noise level can be easily measured, the in-flight accelerator variance is not the same level. In-flight tuning is absolute necessary.

$$[\blacksquare,\quad \mathbf{K},\quad \blacksquare] = kalman(sys,\quad \mathbf{Q},\quad \mathbf{R}) \tag{8.7}$$

$$\mathbf{Q} = \begin{bmatrix} E(\boldsymbol{w}_a \boldsymbol{w}_a^T) & \mathbf{0} \\ \mathbf{0} & E\left(\boldsymbol{w}_b \boldsymbol{w}_b^T\right) \end{bmatrix}$$

$$\mathbf{R} = E(\boldsymbol{v}\boldsymbol{v}^T)$$

	VICON	Webcam
$\boldsymbol{w}_a\left(\frac{m}{s^2}\right)$	0.3	0.3
$\boldsymbol{w}_b\left(\frac{m}{s^2}\right)$	0.01	0.01
$\boldsymbol{v}(m)$	0.01	0.02
$\mathbf{K}$	[7.8 30.3 1]	[5.5 15.2 0.5]

Table 8.2 Position observer parameters

The second modification is on the time delay problem of the position measurement. An easy remedy is to synchronize the signal by delaying the prediction signals and assume the error dynamics remains in the last 40 or 100 ms. If the current time is denoted by t and time delay is τ, the generic form is,

$$\dot{\hat{\mathbf{x}}}_t = \mathbf{A}_t\hat{\mathbf{x}}_t + \mathbf{B}_t\mathbf{u}_t + \mathbf{K}(\mathbf{y}_{t-\tau} - \mathbf{H}\hat{\mathbf{x}}_{t-\tau}) \tag{8.8}$$

The final implemented equation and Simulink structure are as follows,

$$\begin{bmatrix} \dot{\hat{\vec{\mathbf{r}}}} \\ \dot{\hat{\vec{\mathbf{v}}}} \\ \dot{b}_z \end{bmatrix}_t = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M}_{WB}(:,3) \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\vec{\mathbf{r}}} \\ \hat{\vec{\mathbf{v}}} \\ b_z \end{bmatrix}_t + \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \\ \mathbf{0} \end{bmatrix} \vec{\mathbf{a}} - \begin{bmatrix} k_1\mathbf{I} \\ k_2\mathbf{I} \\ k_3\mathbf{M}_{BW}(3,:) \end{bmatrix} \left(\hat{\vec{\mathbf{r}}}_{\mathbf{t-\tau}} - \vec{\mathbf{r}}_{vision}\right) \tag{8.9}$$

The implementation structure in the Simulink is shown in Figure 8.3.

A close look of the fused signals and the measured signals is presented in Figure 8.4. The measured signals (green) are logged on the onboard processer, so it has the image processing delay and the transmission delays. After filtering, the fused signals (blue) get smooth and continuous; moreover, there is an obvious shift in time indicating that the time-compensation in the filter has an effect on the performance.

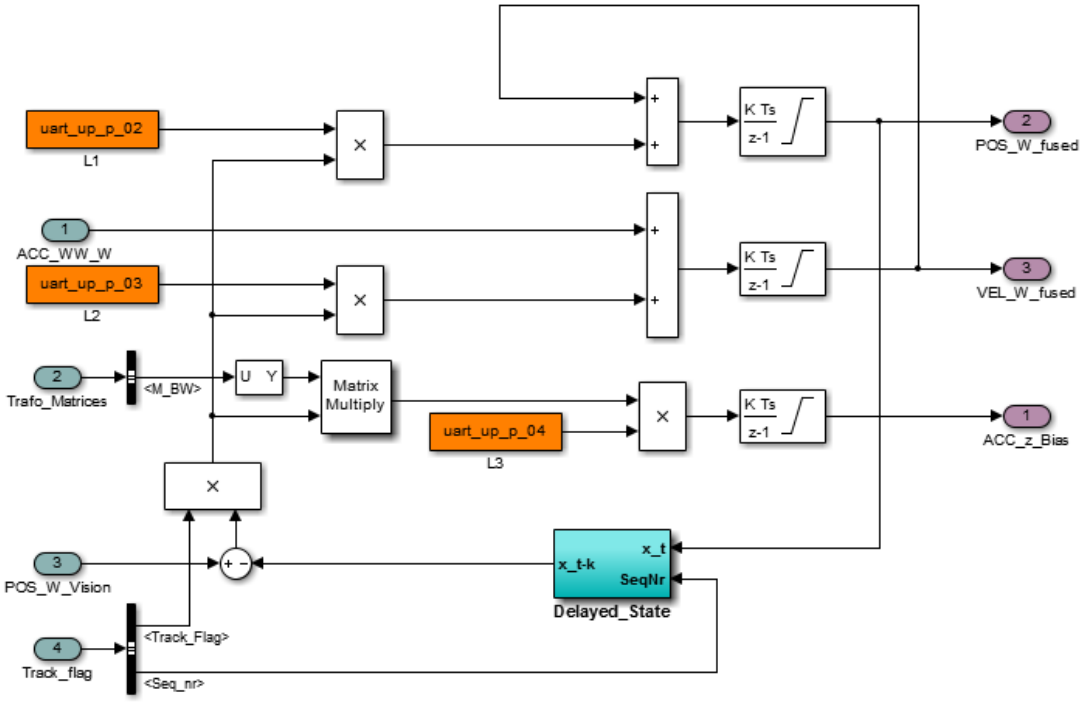


Figure 8.3 Vision system set-up

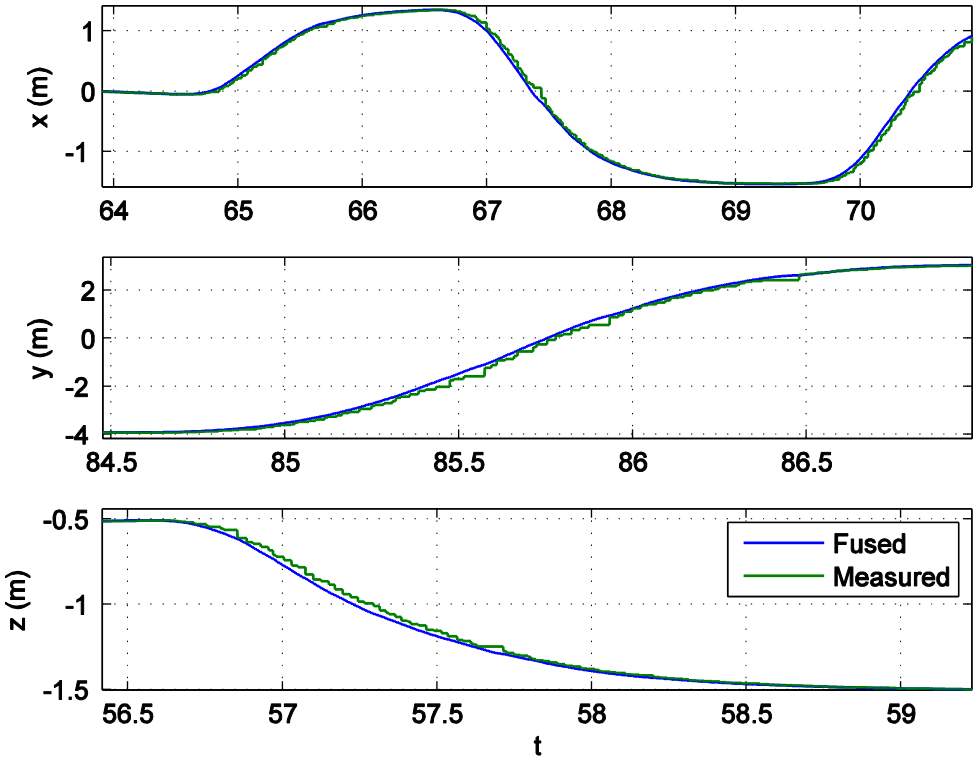


Figure 8.4 Vision measurement and fused position signal in VICON system

8.1.2. Position controller implementation

The implementations of the different position controllers are rather similar. The NDI RD3 design (b2) is presented here as an example.

8.1.2.1. Simulink implementation

The implemented Simulink structure of the position controller is given in Figure 8.5

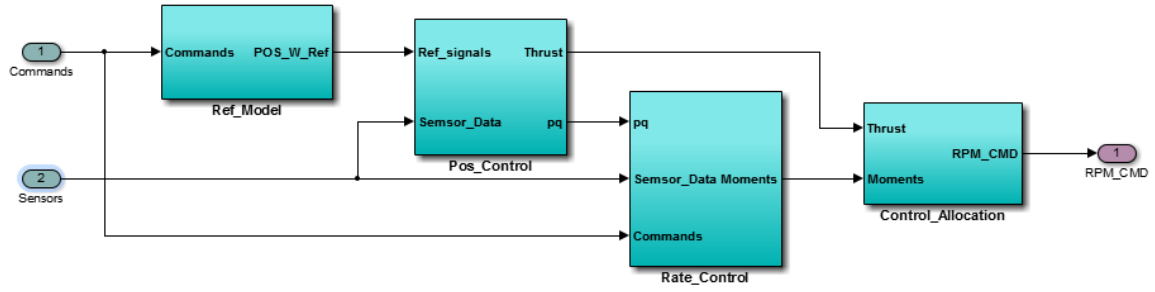


Figure 8.5 Simulink structure of NDI RD3 position controller

The reference mode is a linear fourth order reference system as described in section 6.3.2. The implementation is straightforward as given in Figure 8.6. The parameter A1 defines the natural frequency of the fourth order system.

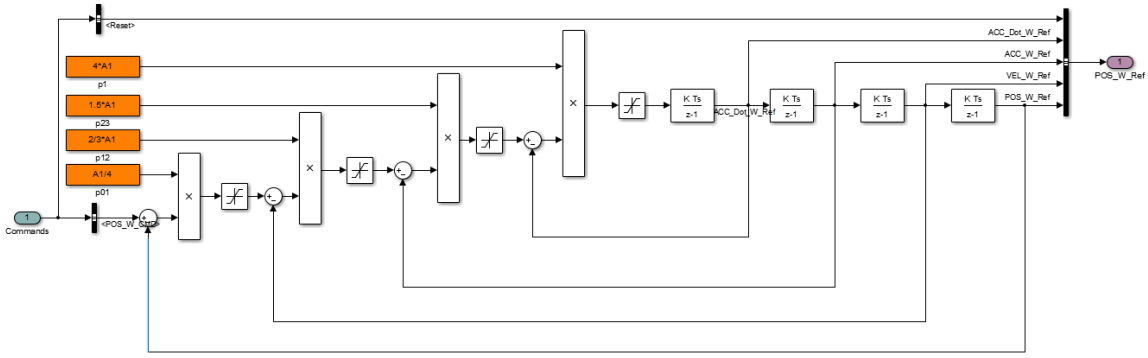


Figure 8.6 Position reference model in Simulink

The error controller of the outer loop follows similar structure of the position reference model, just with additional feedforward signals. The yellow blocks are signal debug channels.

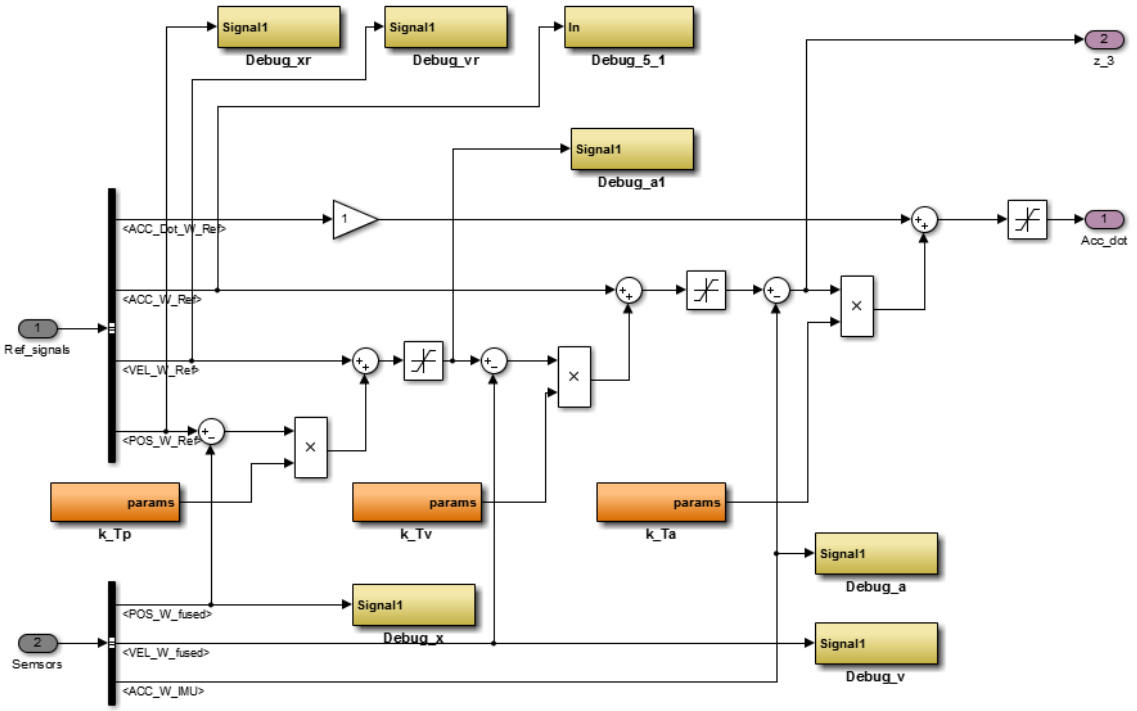


Figure 8.7 Outer Loop Controller in Simulink

The rest of the blocks are implemented likewise. One minor point in the Simulink implementation is the usage of trigonometric functions or square root functions. These functions can be computational expensive for small microprocessor like the ARM 7 cortex. If they can't be avoided, for instance, the trigonometric functions in the Euler angle approach of design b1) or the square root function for the control allocation (see Eq. **Error! Reference ource not found.**), lookup table is a less expensive alternative method in the implementation. The BLDC controller mapping in Figure 8.8 is given as an example.

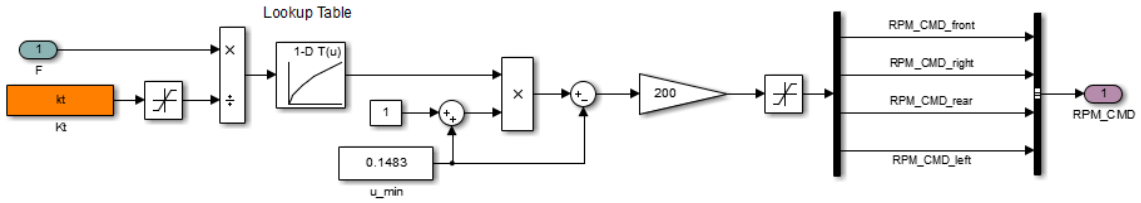


Figure 8.8 BLDC controller mapping in Simulink

8.1.2.2. Practical modifications from theoretical control law

There are two modifications made in the practical implementation compared with the theoretical control law.

The first one is about the nonlinear mapping from force to the RPM command of the BLDC controller. Theoretically, it is nearly a quadratic relationship between the force and rotation speed, i.e. the parameter p should be close to two in the mapping Eq. **Error! Reference source ot found.**.

$$n_i = \left(\frac{F_{prop,i}}{k_n}\right)^{p\approx 2}$$

However, with the actuator dynamics in the consideration, the quadratic mapping is not so optimal. The actuator or the motor dynamics would prefer a linear mapping so that the time scale separation is constant between the rate loop and the motor control loop. If the quadratic mapping is used, it will lead to higher gain at low RPM and lower gain at high RPM, which can easily cause oscillations at low thrust level. As a result, the actual implemented mapping is a trade-off between the force dynamics and actuator dynamics and an empirical value is chosen as 1.5.

The second modification is about the usage of the accelerometer measurements of the Body-fixed x and y-axis. As explained, the z-axis is most dominated as the thrust is aligned in this axis. The x-y- axis has mainly disturbance forces and aerodynamic drag forces in high-speed flight. In the conventional Euler angle approach, those measurements are simply ignored. However, with the new math model designs, these values can be actively used in the feedback control.

With the accelerometer feedback, the control state at this level is not the attitude but the acceleration. Theoretically, more accurate position tracking shall be obtained. However, in flight test it is too aggressive on the disturbances, for instance, if the quadrotor is poked on the side, it will try to account for the side force by pitching or rolling up until it flipped over. In addition, the measurement of the accelerometer equipped on the 'Hummingbird' can be very noisy, which may also excite the system to oscillations. The attitude behavior can be very jittering.

To account for these, the modification is to limit the maximum value on the feedback and to remove the noises by low pass filter. In the experiment, the absolute maximum of the x-and y- feedback is set to be $7 m/s^2$ and a second order filter with natural frequency of 20 is used to filter the noise. Another implementation technique is to reduce the x-y-axis accelerations into 50%. The disturbance rejection is still better than the Euler angle approach but the attitude behavior is still as smooth as the Euler angle approach.

8.1.3. Code generation and ground control

The software has been developed in two different platforms of the Hummingbird, the ARM 7 microprocessor and the Gumstix Overo board. They share the same top-level software structure and the operations are similar.

The data fusion, flight controller and other auxiliary functions are first implemented in Simulink with predefined input and output bus interface. For instance the bus interface for the Overo Simulink implementation is shown in Table 8.3. The FCS_IMU_Bus and FCS_GPS_Bus contains the IMU and GPS output. The FCS_Pilot_Bus takes the pilot commands from the transmitter. The FCS_Data_in_Bus contains the signals sent from the ground control station, which may contains ground control commands, vision sensor output, etc. There are just two output buses: the FCS_CMD_Bus is the RPM commands calculated by the controller and the FCS_Data_Out_Bus is the user specified debug signals. The detailed definitions of the signals can be found in the bus definition file in the appendix.

Input Bus	Output Bus
FCS_IMU_Bus	FCS_CMD_Bus
FCS_GPS_Bus	FCS_Data_Out_Bus
FCS_Pilot_Bus	
FCS_Data_in_Bus	

Table 8.3 Input and output bus for Overo

Next, they can be converted into C code using the Real time workshop embedded coder of Matlab/Simulink and uploaded to the ARM cortex.

The ground control is mainly responsible for the information exchange between the vision system and the quadrotor. The vision output are first sent to the Simulink ground model via UDP connection and then sent to the quadrotor together with ground commands (from a joystick) via X-bee [53]. It also receives debugging signals from the quadrotor via X-bee. The ground model is shown in Figure 8.9.

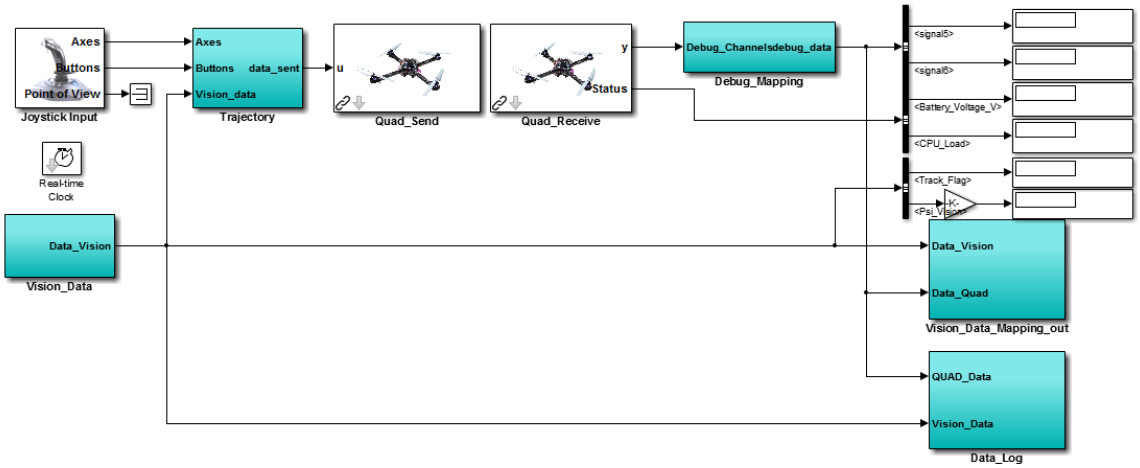


Figure 8.9 Ground Simulink model

8.2. GPS based Position System

The GPS based position system has three sensors to replace the vision system: GPS receiver, barometer and magnetometer. The GPS has relative good accuracy in the latitude and longitude measurement compared with altitude. The barometer can improve the altitude estimation and the magnetometer measures the magnetic field and provides a heading correction to the gyro measurement.

Switching to GPS based position system, the update rate of the position measurement drops to 5Hz and accuracy drops to 3-5 m. however, the range and speed limitation are gone and the quadrotor can thus cover larger flight envelope and fly more aggressive maneuver. The maximum speed in the automatic flight exceeds 10 m/s for instance.

The implementation of the GPS based position control system is not so much different with the vision system. The data fusion has to cope with different sensor inputs. The controller just needs minor gain tuning (smaller outer loop feedback gains due to reduced position quality). New path generator [39] is implemented for various choices of trajectories.

The vision sensor is replaced by GPS, barometer and magnetometer. Standard Extended Kalman filter is implemented. The time update of this filter can be found in section 4.2.3, the math model for the full state EKF. The measurement updates of different sensors has different frequencies and their synchronizations are critical for the fusion results. The state and covariance corrections are triggered by the data flag of the sensors. The implementation in the Matlab-embedded function is shown in Appendix.

Flight test results are given in the figures below. The tracking performance of position, velocity and attitude is good and robust. The position tracking accuracy is much worse than that with the VICON system. However, the flight envelope is much wider. More research and development in the outdoor flight can be done in the future.

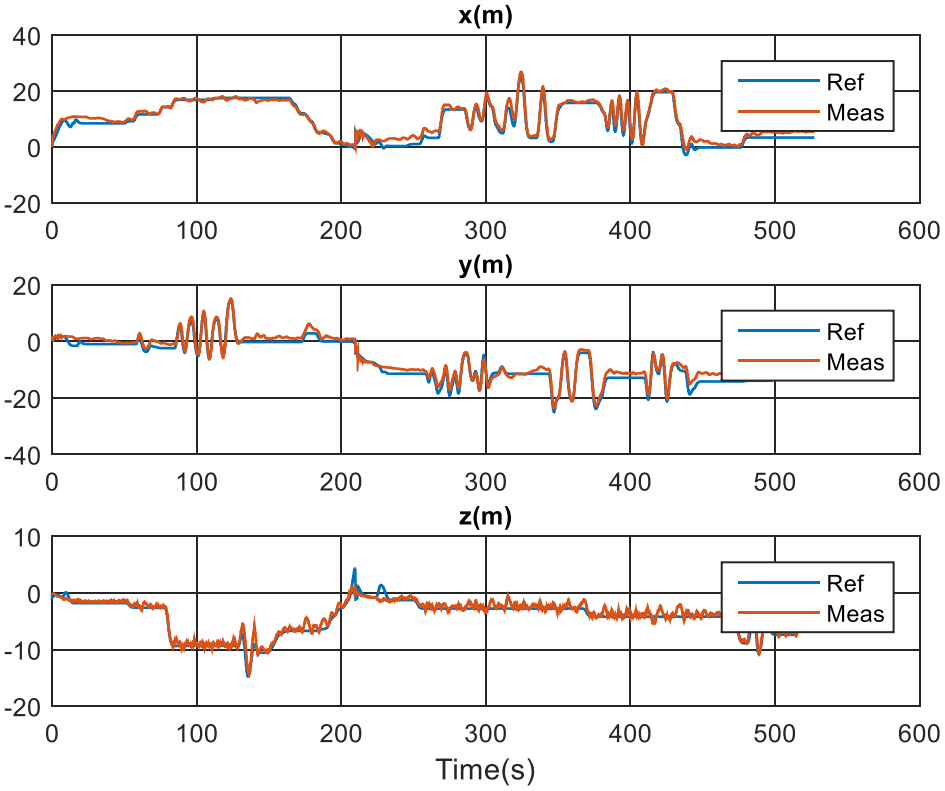


Figure 8.10 Position tracking in outdoor flight test with GPS

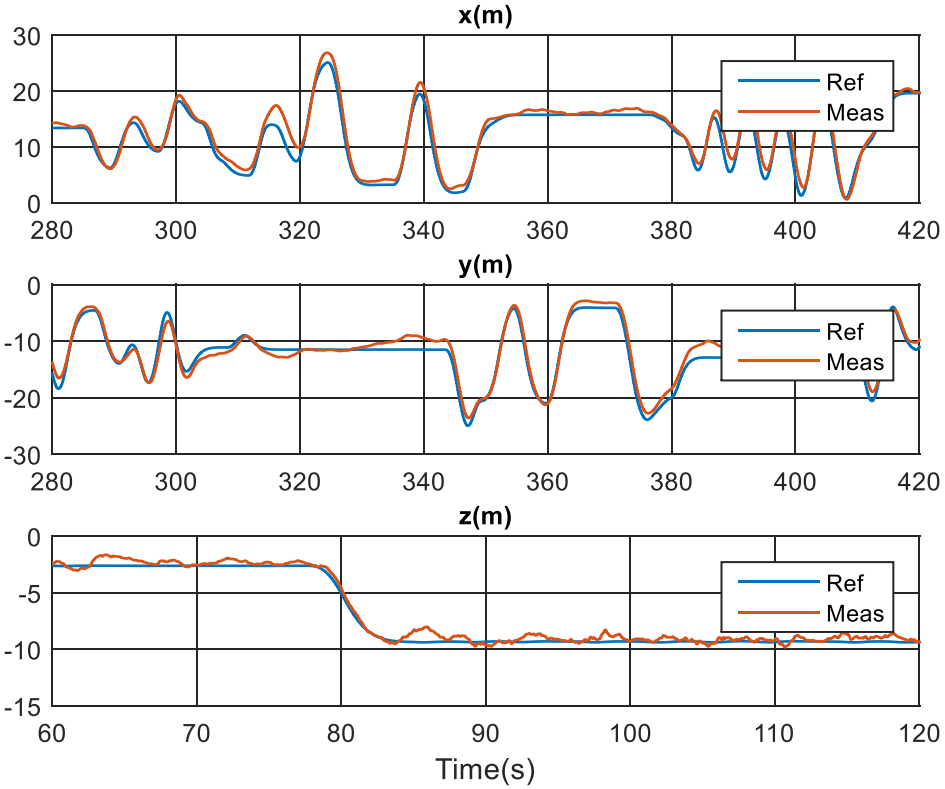


Figure 8.11 Close look of position tracking

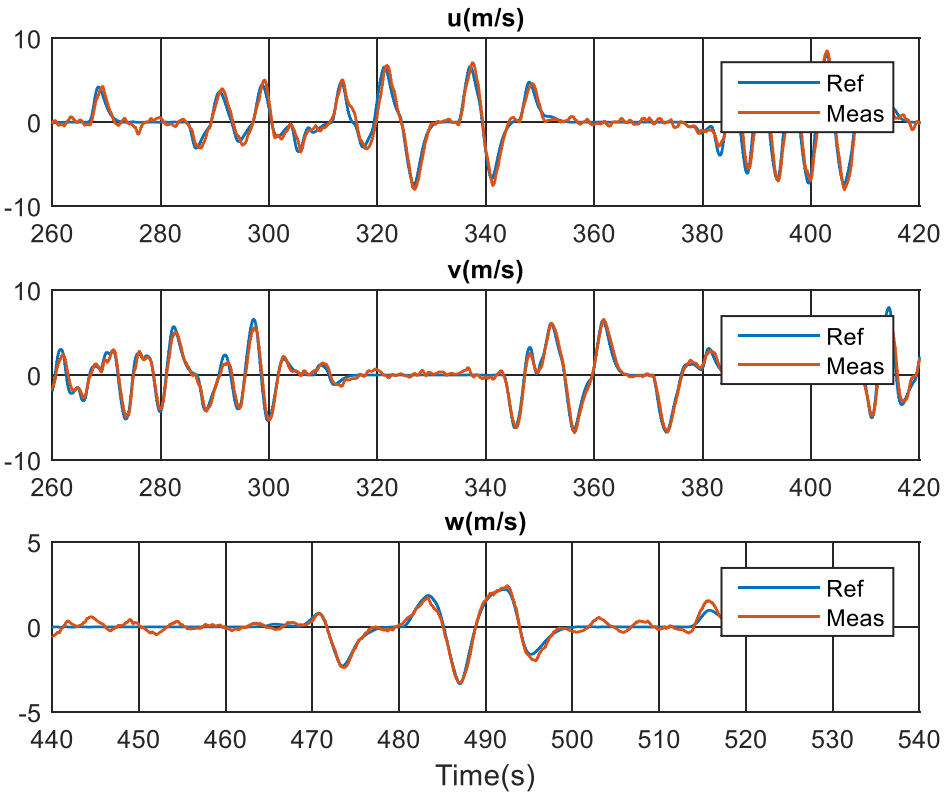


Figure 8.12 Velocity tracking in outdoor flight test with GPS

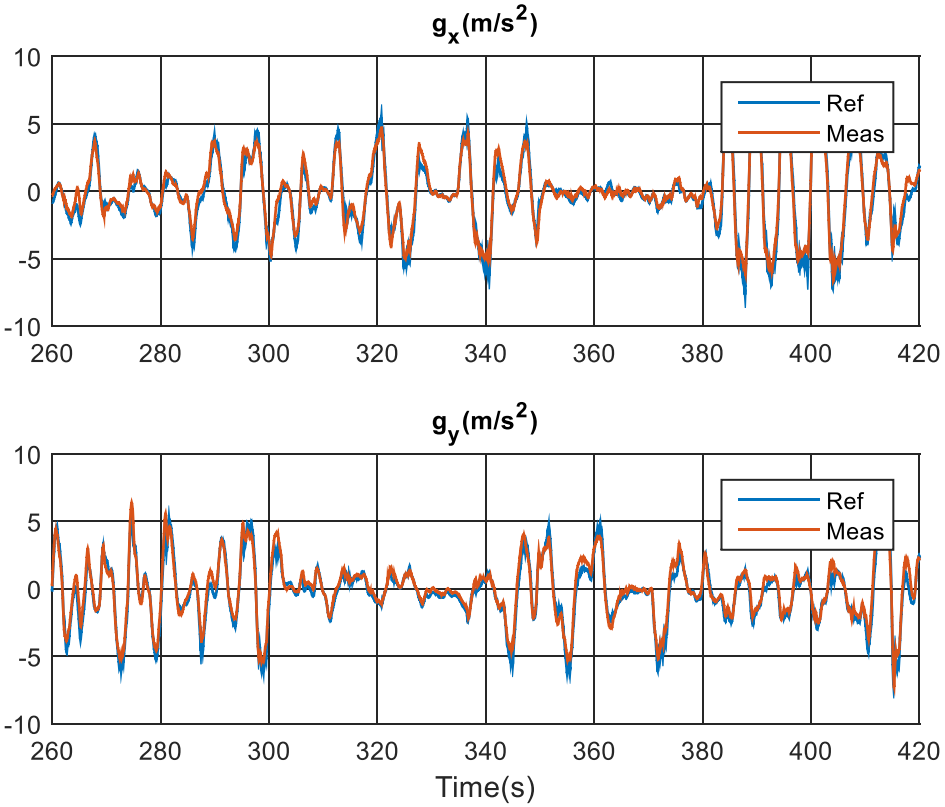


Figure 8.13 Attitude tracking (gxy) in outdoor flight test with GPS

9. Conclusions

This thesis has aimed to investigate the potential of the existing nonlinear adaptive control theories and to bridge the gap between the theories and reality by designing novel control architectures in experimental & operational systems. Different designs using NDI and Backstepping have been investigated and compared. Novel control structures and parameterizations have been proposed and their resulting controller has been evaluated throughout the thesis. The main conclusion and results of the thesis are summarized below.

9.1. Summary

In this thesis, the nonlinear control theories, NDI and Backstepping, have been analyzed and novel control structures have been designed to pushing the actuator limits and explore the flight envelop of the quadrotor without conservative margins. The baseline + augmented adaptive control structure serves as a good basis for future research.

Gravitational Vector Parameterization

The gravitational vector parameterization is recommended to replace the conventional Euler angle parameterization for its efficiency, simplicity and availability. As it is an acceleration state, it has less nonlinearity from the translational acceleration compared with the angle representation. The gravitational vector approach has efficient computation and intuitive representation of the quadrotor dynamics. In addition, the analytical Backstepping in Design g1) is made possible to the rate loop. However, it still has the singularity at Pitch or Bank angle of 90 degree.

NDI Control Design

The classical theoretical NDI aims to convert the system strictly into a chain of integrators. In the control design process, the NDI control can be extended in two directions. First, not every nonlinear dynamics needs to be inverted, for instance, the aerodynamic damping or the accelerometer x-y-axis feedback. Second, original model design as in design c) is possible. With the extension, the disadvantages of classical NDI compared with Backstepping are remedied. Overall, the NDI control designs have the same performance as the best Backstepping approach, either for system of triangular form or non-triangular form. What's more, the NDI design structures are often simple and intuitive because NDI is deeply rooted in the physics of the plants. As a result, the NDI RD3 position loop + RD1 rate loop design has the best overall performance with high control bandwidth., the RD2 position loop + RD2 attitude loop design is a better choice for quadrotor application required low control bandwidth but high robustness. To conclude, the NDI control is preferred compared with Backstepping for the quadrotor applications.

Backstepping Control Design

Backstepping control methods in general has more design freedom and more control variations. The best control design with Backstepping has the same performance as NDI,

however, the implementations like the gain designs and debugging is not as intuitive as NDI. More implementation effort is needed for the similar NDI designs. The additional Backstepping term compared with NDI is proved to have little impact on the controller performance, as well as the CFB modification.

Hybrid Control: Nonlinear Baseline Control + Augmented $\mathcal{L}_1$ Control

The hybrid control structure is most suitable for the quadrotor applications, where the knowledge of the plant dynamics can be used in the baseline control, and the uncertainties mainly from external disturbances and aerodynamics can be compensated with fast converging $\mathcal{L}_1$ controller.

9.2. Future Work

New question and research directions can be formulated based on the research presented in this thesis. They are summarized in this section

Path generation with consideration of actuator limits

For high demanding trajectories, the actuator can be easily saturated and impede the tracking performance. When switching from trajectory tracking to path following, additional degree of freedom of position control in the path direction can be used to improve the cross-path deviations. Online computation of the foot point on the path is necessary and the error dynamics may be formulated in the path frame.

What's more, online path generation with consideration of the actuator limits maybe very useful to avoid actuator saturations or only allow short time saturations. The PCH concept may be employed in the close-loop reference path generation.

Advanced Adaptive Control

The current $\mathcal{L}_1$ controller is one of the basic adaptive control implementation. Debates on the $\mathcal{L}_1$ adaptation are still going on. Other adaptive strategies include MRAC, Neural Network, Background Learning, etc. There are huge potential in the adaptive area and further research in this area.

Bibliography

[1] Ascending technologies GmbH, [Online]. Available: http://www.asctec.de. [Accessed 27 July 2012].

[2] "Overo Fire COM, GUM3503F," Gumstix Inc., [Online]. Available: https://www.gumstix.com/store/product_info.php?products_id=227. [Accessed 27 Nov 2012].

[3] "Rate Gyroscope, ADXRS610," Analog Device, [Online]. Available: http://www.analog.com. [Accessed 27 November 2012].

[4] "Accelerometer, MXR9500G/M," MEMSIC Inc., [Online]. Available: http://www.memsic.com. [Accessed 27 August 2012].

[5] S. Klose, J. Wang, M. Achtelik, G. Panin, F. Holzapfel and A. Knoll, "Markerless Vision Assisted Flight Control of a Quadrocopter," in *IEEE RSJ International Conference on Intelligent Robots and Systems*, Taipei, 2010.

[6] Vicon Motion System, [Online]. Available: http://www.vicon.com. [Accessed 27 July 2012].

[7] M. Achtelik, "Simulink Quadrocopter Framework," Institute of FSD, TU München, 2009.

[8] M. Valenti, B. Bethke, G. Fiore and J. How, "Indoor Multi-Vehicle Flight Testbed for Fault Detection, Isolation, and Recovery," in *AIAA Guidance, Navigation, and Control Conference*, Colorado, USA, 2006.

[9] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[10] G. Hoffmann, S. Waslander and C. Tomlin, "Quadrotor helicopter trajectory tracking control," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Honolulu, Hawaii, 2008.

[11] D. Mellinger, N. Michael and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Proceedings of the International Symposium on Experimental Robotics*, Delhi, India, 2010.

[12] A. Isidori, Nonlinear Control Systems, New York: Springer, 1995.

[13] J. J. E. Slotine and W. Li, Applied Nonlinear Control, Prentice Hall, 1991.

[14] H. Voos, "Nonlinear Control of a Quadrotor Micro-UAV using Feedback-Linearization," in *IEEE International Conference on Mechatronics*, 2009.

[15] V. Mistler, A. Benallegue and N. K. M'Sirdi, "Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback," in *10th IEEE International Workshop on Robot and Human Interactive Communication*, Roman, 2001.

[16] M. Buhl, O. Fritsch and B. Lohmann, "Exakte Ein-/Ausgangslinearisierung für die translatorische Dynamik eines Quadrocopters," in *Automatisierungstechnik*, 2011.

[17] M. Krstic, I. Kanellakopoulos and P. Kokotovic, Nonlinear and adaptive control design, New York: Wiley, 1995.

[18] S. Bouabdallah and R. Siegwart, "Backstepping and Sliding-mode Techniques Applied to an Indoor Micro Quadrotor," in *IEEE International Conference on Robotics and Automation*, 2005.

[19] L. Pollini and A. Metrangolo, "Simulation and robust backstepping control of a quadrotor aircraft," in *AIAA Modeling and Simulation technologies Conference and Exhibit*, 2008.

[20] T. Madani and A. Benallegue, "Backstepping control for a quadrotor helicopter," in *IEEE International Conference on Intelligent Robots and Systems*, 2006.

[21] J. A. Farrell and M. Polycarpou, "Command Filtered Backstepping," in *American Control Conference*, 2008.

[22] P. C. Parks, "Lyapunov redesign of model reference adaptive control systems," *IEEE Trans. Automat. Control,* pp. 362-367, 1966.

[23] B. Egardt, "Stability of Adaptive Controllers," in *Lecture Notes in Control and Inform. Sci.*, New York, Springer-Verlag, 1979.

[24] P. A. Loannou and P. V. Kokotovic, "Robust redesign of adaptive control," *IEEE Transactions on Automatic Control,* pp. 202-211, 1984.

[25] K. S. Narendra and A. M. Annaswamy, "A new adaptive law for robust adaptation without persistent excitation," *IEEE Transactions on Automatic Control,* pp. 134-145, 1987.

[26] C. Cao and N. Hovakimyan, "Design and analysis of a novel L1 adaptive control architecture with guaranteed transient performance," *IEEE Transactions on Automatic Control,* pp. 586-591, 2008.

[27] I. M. Gregory and C. Cao, "L1 adaptive control design for NASA AirSTAR flight test vehicle," in *AIAA Guidance, Navigation and Control Conference*, Chicago, 2009.

[28] K. A. Wise, E. Lavretsky and N. Hovakimyan, "Adaptive control in flight: Theory, application, and open problems," in *American Control Conference*, Minneapolis, 2006.

[29] T. Yucelen and E. Johnson, "Command governor based-adaptive control," in *AIAA Guidance, Navigation and Control Conference*, Mineapolis, 2012.

[30] T. Gibson, A. Annaswamy and E. Lavretsky, "Closed-Loop Reference Models in Adaptive Control, Part 1: Transient Performance,," in *American Control Conference*, 2013.

[31] E. Kharisov, N. Hovakimyan and J. Karl, "Comparison of Several Adaptive Controllers According to Their Robustness Metrics," in *AIAA Guidance, Navigation and Control Conference*, Toronto, 2012.

[32] M. Achtelik, T. Bierling, J. Wang and F. Holzapfel, "Adaptive Control of a Quadcopter in the Presence of large/complete Parameter Uncertainties," in *AIAA Infotech@Aerospace*, St. Louis, 2011.

[33] J. Wang, T. Bierling, M. Achtelik, L. Höcht, F. Holzapfel, W. Zhao and T. H. Go, "Attitude Fre Position Control of a Quadcopter using Dynamic Inversion," in *AIAA Infortech@Aerospace*, St. Louis, 2011.

[34] W. Zhao, T. H. Go, J. Wang and F. Holzapfel, "Leader Follower Quadrotor Formation Flight Control," in *CEAS EuroGNC*, Munich, 2011.

[35] J. Wang, T. Bierling and F. Holzapfel, "Novel Dynamic Inversion Architecture Design for Quadrocopter Cotnrol," in *Advances in Aerospace Guidance, Navigation and Control*, Munich, Springer Berlin Heidelberg, 2011, pp. 261-272.

[36] J. Wang, T. Raffler and F. Holzapfel, "Nonlinear Position Control Approaches for Quadcopters Using a Novel State Representation," in *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, 2012.

[37] J. Wang, F. Holzapfel and F. Peter, "Comparison of Dynamic Inversion and Backstepping Control with Application to a Quadrotor," in *CEAS EuroGNC*, Delft, 2013.

[38] J. Wang, F. Holzapfel, E. Xargay and N. Hovakimyan, "Non-cascaded Dynamic Inversion Design for Quadrotor Position Control with L1 Augmentation," in *CEAS EuroGNC*, Delft, 2013.

[39] T. Raffler, J. Wang and F. Holzapfel, "Path Generation and Control for Unmanned Multirotor Vehicles Using Nonlinear Dynamic Inversion and Pseudo Control Hedging," in *19th IFAC Symposium on Automatic Control in Aerospace*, Wuerzburg, 2013.

[40] H. J. Marquez, Nonlinear Contrl Systems Analysis and Design, New Jersey: Wiley , 2003.

[41] E. Johnson, "Limited Authority Adaptive Flight Control," Georgia Institute of Technology, 2000.

[42] N. Hovakimyan and C. Cao, L1 Adaptive Control Theory: guaranteed robustness with fast adaptation, Philadelphia: SIAM, 2010.

[43] K. S. Narendra and A. M. Annaswamy, Stable Adaptive Systems, New York: Prentice Hall, 2005.

[44] D. Simon, Optimal State Estimation, New Jersey: WILEY, 2006.

[45] W. Levine, The Control Handbook, Florida: CRC Press, 2000.

[46] D. Luenberger, "Obeserver for multivariable systems," *IEEE Trans. Autom. Control,* vol. 11, pp. 190-197, 1966.

[47] "Matlab Help Document on Racatti Equation," [Online]. Available: http://www.mathworks.de/help/search/doccenter/en/R2013a?qdoc=riccati+equation&submitsearch=Search. [Accessed 22 07 2013].

[48] D. Ito, J. Georgie, J. Valsek and D. Ward, "Reentry Vehicle Flight Controls Design Guidelines: Dynamic Inversion," NASA, Houston, 2002.

[49] F. Peter, M. Leitao and F. Holzapfel, "Adaptive Augmentation of a New Baseline Control Architecture for Tail-controlled Missiles Using a Nonlinear Reference Model," in *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, 2012.

[50] E. v. Oort, Adaptive Backstepping Control and Safety Analysis for Modern Fighter Aircraft, Zutphen, The Netherlands: Wohrmann Print Service, 2011.

[51] J. Kuipers, Quaternions and rotation Sequences: a Primer with Applications to Orbits, Aerospace, and Virtual Reality, Princeton University Press, 1999.

[52] J. A. Farrell, Aided Navigation, New York: McGraw-Hill, 2008.

[53] D. International, "XBee/XBee-Pro ZB RF Modules," 15 11 2010. [Online]. Available: http://ftp1.digi.com/support/documentation/90000976_G.pdf. [Accessed 18 03 2014].

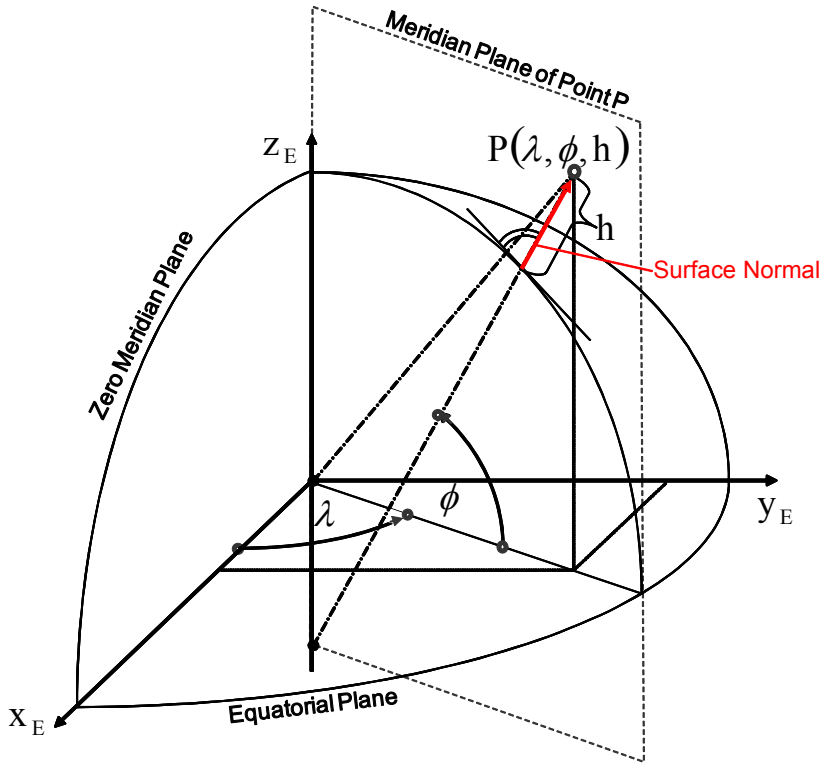
[54] C. J. Schumacher, P. P. Khargonekar and ,. N. H. McClamroch, "Stability Analysis of Dynamic Inversion Controllers Using Time-Scale Separation," in *Proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, Boston, MA, 1998.

Appendix

Coordinate Frames

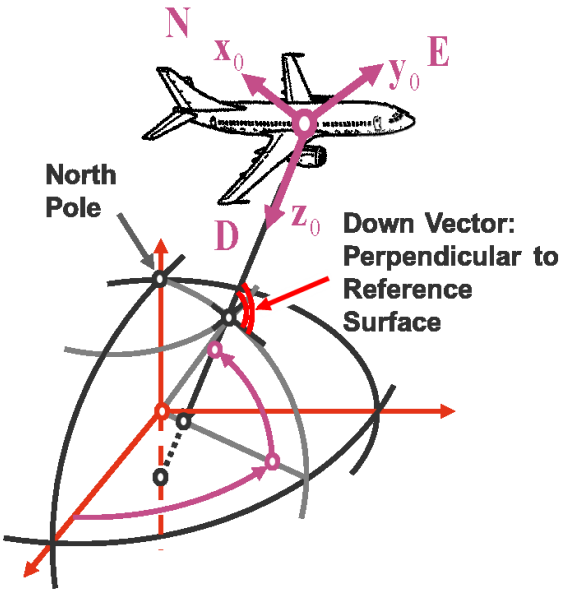
- **ECEF - Frame (Earth Centered Earth Fixed)**

Index:	E
Role:	Navigation Frame – used to specify positions
Origin:	Center of the Earth
Translation:	Moves with ECI-frame
Rotation:	Earth Rotation around z-axis at earth angular rate, ~2π/24 h
x-Axis:	In equatorial plane, points through Greenwich Meridian
y-Axis:	In equatorial plane to form a right hand system with the x-Axis and z-Axis
z-Axis:	Rotation axis of the earth



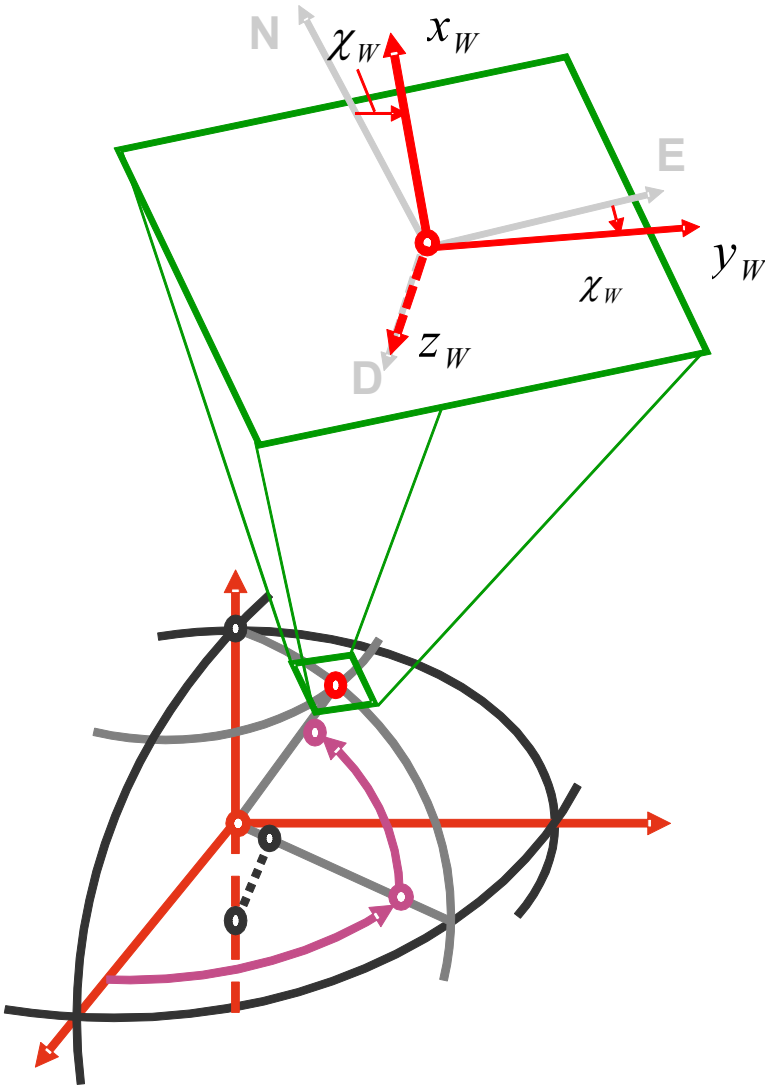
- **NED Frame (North-East-Down Frame)**

Index:	O
Role:	Orientation Frame – used to specify the attitude of vehicle
Origin:	Reference Point of the Aircraft
Translation:	Moves with Aircraft Reference Point
Rotation:	Rotates with transport rate to keep NED alignment
x-Axis:	Parallel to the local geoid surface, pointing to the geographic north pole
y-Axis:	Parallel to the local geoid surface, pointing east to form a right hand system with x-Axis and z-Axis
z-Axis:	Pointing downwards, perpendicular to the local geoid surface



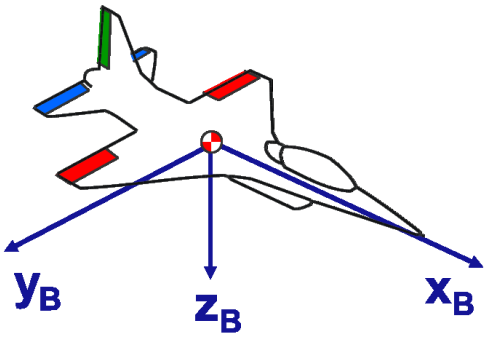
- **W Frame (World Frame)**

  Index: W
  Role: World Frame – used to specify the local position of vehicle, deduced from NED frame.
  Origin: Point on the surface of the earth
  Translation: None
  Rotation: None
  x-Axis: Parallel to the local geoid surface, pointing to a direction that deviates with alignment angle χ_W from north direction
  y-Axis: Parallel to the local geoid surface, to form a right hand system with x-Axis and z-Axis
  z-Axis: Pointing downwards, perpendicular to the local geoid surface



- **Body Fixed Frame**

Index:	B
Role:	Notation Frame – used for forces, moments, …
Origin:	Reference Point of the Aircraft
Translation:	Moves with Aircraft Reference Point
Rotation:	Rotates with the Rigid Body Aircraft
x-Axis:	Pointing towards the aircraft nose in the symmetry plane
y-Axis:	Pointing to the right wing to form an orthogonal right hand system
z-Axis:	Pointing downwards in the symmetry plane of the aircraft, perpendicular to the x- and y-axis



- **Kinematic Frame**

Index:	K
Role:	Flight-Path Axis Frame
Origin:	Reference point of the aircraft (R)
Translation:	Moves with Aircraft Reference Point
Rotation:	Rotates with the direction of the kinematic aircraft motion
x-Axis:	Aligned with the kinematic velocity, pointing into the direction of the kinematic velocity
y-Axis:	Pointing to the right perpendicular to the x- und z- Axes
z-Axis:	Pointing downwards parallel to the projection of the local normal to the WGS-84 ellipsoid into a plane perpendicular to the x-Axis (i.e. to the kinematic velocity)