This item was submitted to Loughborough's Institutional Repository (https://dspace.lboro.ac.uk/) by the author and is made available under the following Creative Commons Licence conditions.

creative commons

COMMONS DEED

Attribution-NonCommercial-NoDerivs 2.5

You are free:

- to copy, distribute, display, and perform the work

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor.

Noncommercial. You may not use this work for commercial purposes.

No Derivative Works. You may not alter, transform, or build upon this work.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

This is a human-readable summary of the Legal Code (the full license).

Disclaimer

For the full text of this licence, please go to: http://creativecommons.org/licenses/by-nc-nd/2.5/

Advanced Control for Miniature Helicopters: Modelling, Design and Flight Test

Cunjia Liu

Department of Aeronautical and Automotive Engineering

Loughborough University

A thesis submitted for the degree of

Doctor of Philosophy

September 2011

I would like to dedicate this thesis to my loving parents ...

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Wen-Hua Chen, for his constant guidance, helps and precious advices in the past three and half years. I have greatly respected his erudite knowledge, hard working spirit and rigorous attitude towards science as a researcher and appreciated his kind advice and solicitude as a senior.

I would also like to thank Prof. John Andrews for his support and encouragement during the process of my research. And even after he moved to the Nottingham University, he still cared my progress. Moreover, I owe a thank to Prof. Zhang Ren in my home university (Beihang University), who has brought me into the area of aerospace engineering.

Thirdly, I am grateful to all of the members of Autonomous System Lab who make it a wonderful place for research and work with a lot of interactions and activities that keep everyone inspired and motivated. Special thanks to Jonathan Clarke who helped me on many practical things with his rich engineering experience.

My gratitude also goes to the Department of Aeronautical and Automotive Engineering at Loughborough University and the Chinese Scholarship Council who has financially supported my study in UK.

Last but not least, I must thank my parents for their endless love and constant support while I am away from them for 10 years. Extra thanks to Miss Xiang Li for her long time accompaniment and support.

Abstract

Unmanned aerial vehicles (UAV) have been receiving unprecedented development during the past two decades. Among different types of UAVs, unmanned helicopters exhibit promising features gained from vertical-takeoff-and-landing, which make them as a versatile platform for both military and civil applications.

The work reported in this thesis aims to apply advanced control techniques, in particular model predictive control (MPC), to an autonomous helicopter in order to enhance its performance and capability.

First, a rapid prototyping testbed is developed to enable indoor flight testing for miniature helicopters. This testbed is able to simultaneously observe the flight state, carry out complicated algorithms and realtime control of helicopters all in a Matlab/Simulink environment, which provides a streamline process from algorithm development, simulation to flight tests.

Next, the modelling and system identification for small-scale helicopters are studied. A parametric model is developed and the unknown parameters are estimated through the designed identification process. After a mathematical model of the selected helicopter is available, three MPC based control algorithms are developed focusing on different aspects in the operation of autonomous helicopters.

The first algorithm is a nonlinear MPC framework. A piecewise constant scheme is used in the MPC formulation to reduce the intensive computation load. A two-level framework is suggested where the nonlinear MPC is combined with a low-level linear controller to allow its application on the systems with fast dynamics. The second algorithm solves the local path planning and the successive tracking control by using nonlinear and linear MPC, respectively. The kinematics and obstacle information are incorporated in the path planning, and the linear dynamics are used to design a flight controller. A guidance compensator dynamically links the path planner and flight controller. The third algorithm focuses on the further reduction of computational load in a MPC scheme and the trajectory tracking control in the presence of uncertainties and disturbances. An explicit nonlinear MPC is developed for helicopters to avoid online optimisation, which is then integrated with a nonlinear disturbance observer to significantly improve its robustness and disturbance attenuation.

All these algorithms have been verified by flight tests for autonomous helicopters in the dedicated rapid prototyping testbed developed in this thesis.

Contents

List of Figures

Nomenclature

Helicopter Model

a, b	longitudinal and lateral flapping angles of main rotor
β	blade pitch angle
δ_{col}	collective pitch control input
δ_{lat}	cyclic lateral control input
δ_{lon}	cyclic longitudinal control input
δ_{ped}	pedal control input
g	acceleration of gravity
G_x, G_y, G_z	projections of gravitational force in the body coordinate
I_{xx}, I_{yy}, I_{zz}	principal moments of inertia of the helicopter
L, M, N	external moments in x, y and z directions of body axes
l_M, y_M, h_M	the position of the main rotor shafts relative to c.g.
l_T, y_T, h_T	the position of the tail rotor shafts relative to c.g.
Ω	rotor speed
p, q, r	roll, pitch and yaw rate in helicopter body coordinate
ϕ, θ, ψ	roll, pitch and yaw Euler angles
u, v, w	projections of inertial velocity on helicopter body axes
X, Y, Z	external forces in x, y and z directions of body axes

x, y, z	helicopter position in inertial coordinate

Acronyms

DOBC	disturbance observer based control
DOF	degrees of freedom
ENMPC	explicit nonlinear MPC
LAN	local area network
LMI	linear matrix inequality
LQR	linear quadratic regulator
LTI	linear time invariant
LTV	linear time varying
MILP	mixed integer linear programming
MIMO	multi-inputs-multi-outputs
MPC	model predictive control
NLP	nonlinear programming
NMPC	nonlinear model predictive control
OP	optimisation problem
PEM	prediction error method
PID	proportional integral derivative
QP	quadratic programming
RHC	receding horizon control
TPP	tip path plane
UAV	unmanned aerial vehicle
UDP	user datagram protocol
VTOL	vertical take-off and landing

Chapter 1

Introduction

1.1 Overview

Enabled by the advancement in technologies and driven by a broad range of application demands, the last two decades have witnessed the rapid development of Unmanned Aerial Vehicle (UAV) systems. UAVs are now in the process of replacing human piloted aircraft in many situations where they may provide a safer, cheaper and more efficient solution than their manned counterparts. Military tasks, such as Intelligence, Surveillance and Reconnaissance, have promoted the development of UAVs from the early days; great potentials have also been found in civil applications such as agriculture, line inspection, and search and rescue. The growing demands on applications stimulate the further development of UAV technologies, and impose higher requirements on autonomy. In turn, as the control and system integration techniques become more capable, UAVs will find an ever-expanding role undertaking tasks in more complicated scenarios.

Advanced control algorithms, combined with hardware and software enabling technologies, are playing a critical role in providing means to achieve desired capabilities that future UAVs should have. Among these various algorithms, model predictive control (MPC) is of particular interested in this thesis, because it offers a number of advantages such as a general framework for nonlinear systems, constraint handing, online planning with preview. These features make MPC practically attractive for UAV applications. The receding horizon property of MPC provides a natural framework for improving autonomous level of UAVs. Within this framework, the process of measurement-optimisation-execution mimics a human operator's behaviour "look-think-action". In addition, MPC algorithms utilise dynamic models of UAVs with real-time updates of the current status of the vehicle and its environment. With the progress of a mission, they can replan a new route in real-time for UAVs if necessary to accommodate changes of task priorities and dynamic environments.

To this end, this thesis tries to use MPC techniques to explore the dynamic properties of an UAV system, in particular a helicopter-like UAV, due to its versatile flight patterns and complicated dynamics. Flight control design and path planning for a miniature helicopter are studied, which span from modelling, control design and synthesis, and verification using flight tests. A particular attention is paid on the engineering implementation and flight testing of MPC based algorithms developed in this thesis.

1.2 Outlines

This thesis aims at providing a systematic framework of delivering advanced control, in particular MPC based techniques, to autonomous helicopters. The proposed control algorithms in this thesis try to address a number of tasks in the operation of an autonomous helicopter, namely trajectory tracking, local path planning, and disturbance attenuation. The overall control design process, including modelling, control development and validation, is covered. The outlines of the remaining thesis are organised as follows:

- Chapter 2 provides a literature review on the topics related to this thesis. First, the basic idea of MPC is explained, and issues associated with MPC are discussed. Next, existing MPC techniques used in UAV applications are surveyed and categorised in detail. At the end, popular flight control algorithms for autonomous helicopters are reviewed, where some classic issues and new trends are discussed.

- Chapter 3 describes the development of an indoor testbed for miniature helicopters, which can rapidly realise complicated control algorithms and implement them on physical vehicles. It effectively facilitates the flight control development and the following verification. This testbed is composed of a Vicon Motion system, aerial/ground vehicles and a ground station. All the components are linked and integrated into a Matlab/Simulink environment allowing a researcher to program in the same software environment from the modelling and analysis to control design and final experiments. This indoor flight test environment is extensively used in supporting the research work described in this thesis.

- Chapter 4 studies the modelling and system identification techniques for miniature helicopters. It is important to understand the dynamic characteristics of a helicopter before any attempts at the development of advanced control can be carried out. Moreover, system identification technique is essential in producing models of the target helicopter when validating proposed algorithms on a real helicopter, or applying well established algorithms on different helicopters. To this end, this chapter presents a simplified nonlinear dynamics model to capture the key dynamics of a helicopter. The model is constructed based on the first principle method, and the corresponding system identification process is conducted to obtain the unknown parameters build in the dynamic model.

- Chapter 5 describes a MPC based control framework for autonomous helicopters. It is a two-level control structure, where the high-level MPC generates baseline control profile by exploiting the nonlinear helicopter model, and the low-level linear controller, designed based on the linearisation around the state reference provided by the high-level controller, compensates the baseline control in the presence of disturbances and uncertainties. The computational load rising from using a nonlinear helicopter model is reduced by using a piecewise constant scheme in MPC. The stability analysis on such a control framework is carried out. This two-level control is implemented on the testbed to address the trajectory tracking of a miniature helicopter.

- Chapter 6 gives a hierarchical control framework for local path planning and tracking control of a miniature helicopter. Following the same two-level control structure from Chapter 5, a nonlinear MPC planner is employed as a high-level controller for local path planning subject to helicopter kinematics and obstacles; a guidance compensator is then introduced to compensate the low bandwidth of the MPC planner. The generated guidance command is tracked by the helicopter under the control of a linear constrained MPC, which on the other hand can guarantee the effectiveness of using the kinematic model for path planning. The overall hierarchical framework is also tested on the testbed in different scenarios where obstacles may appear on helicopter's routes.

- Chapter 7 aims to tackle the MPC based trajectory tracking control of autonomous helicopter from an alternative way. Efforts have been taken in Chapter 5 in reducing computational demand caused by nonlinear optimisation in MPC by using the piecewise constant scheme and a two-level control structure, and the flight test have confirmed the real-time property of the proposed control framework. However, there are still considerable concerns about the computational time, the demand on computing power and numerical properties of nonlinear optimisers. To this end, in Chapter 7 an explicit MPC is employed to undertake the tracking control to avoid the online optimisation. This explicit MPC is designed based on a modified helicopter model where the disturbances and uncertainties are explicitly taken into account as lumped unknown terms. A nonlinear disturbance observer is then designed to estimate them and feed them into the MPC framework to attenuate the influences from disturbances and uncertainties.

- Chapter 8 concludes this thesis with some discussions and future perspectives.

Chapter 2

Literature review

This chapter presents a literature review focusing on three aspects: MPC technique, its applications on UAV and flight control of autonomous helicopters.

2.1 Brief overview of MPC

MPC technology, also known as receding horizon control (RHC), was originally developed to meet control needs of power plants and petroleum refineries, and now is an attractive control methodology used in a wide variety of application areas including chemicals, food processing, automotive, and aerospace applications [93]. The popularity and success of MPC are mostly due to its ability to explicitly deal with constraints and explicitly exploit the dynamic model, leading to a safe operation of the plant under all circumstances. It can cope with various performance specifications for nonlinear systems and is able to embed specific criteria into the MPC formulation. Due to these advantages, although MPC is computationally intensive and initially developed for systems with slow dynamics, it has drawn more and more attentions in the UAV community where aircraft exhibit fast dynamics [88; 99].

Fig 2.1 depicts the basic principle of MPC. A predictive controller has a mathematical model that is used to predict the behaviour of the plant, starting from the current time t, based on the measurements $x(t)$ and over a future *prediction horizon* T_p. This predicted behaviour depends on the assumed input $\hat{u}(\tau; x(t))$, $\tau \in [t, t + T_p]$, applied over the prediction horizon. The idea is to select that input over the *control horizon* T_c by an optimisation process such that it gives the best predicted behaviour, in terms of a predetermined performance index.

Often the constraints on plant states and inputs are also taken into account in the optimisation. If there were no disturbances and model mismatch and if the optimisation problem could be solved over an infinite horizon, then the optimal input signal found at t can be applied to the system in an open-loop fashion. However, due to disturbances and model mismatch, the actual system behaves different from the predicted one. To incorporate feedback, the optimal open-loop input is implemented only until the next sampling instant $t+\delta$, when new measurements become available and the optimisation is then repeated. This means the MPC is implemented in a *receding horizon* fashion, and this constitutes a feedback control strategy.

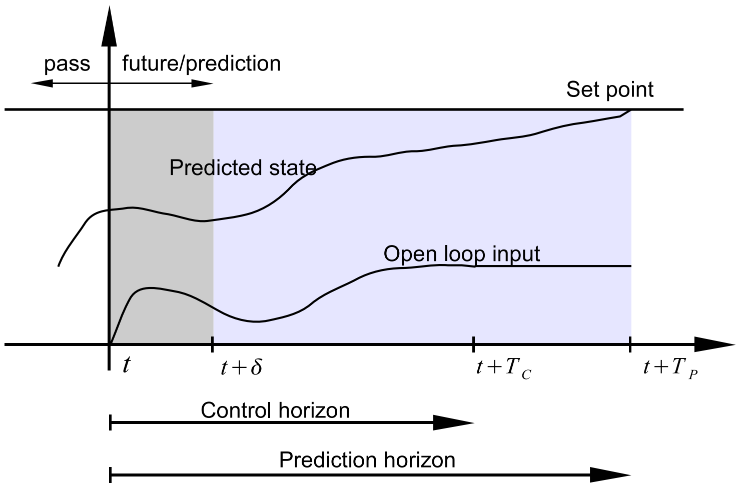


Figure 2.1: Principle of MPC

2.1.1 MPC formulation

Different types of prediction models can be employed for MPC design, resulting into different optimisation problems (OP) which need to be solved repetitively. Generally, a dynamic system can be described in the discrete-time domain by a state-space equation:

$$x(t+1) = f(x(t), u(t)) \tag{2.1}$$

where $x(t) \in \mathbb{R}^n$ is the state vector at time instant t, $x(t+1)$ denotes the successor state, $u(t) \in \mathbb{R}^m$ is the vector of control inputs, and $f(\cdot)$ is the state-update function. It is assumed that states and inputs are subject to constraints.

$$\begin{aligned} x(t) &\in \mathbb{X} \in \mathbb{R}^n \\ u(t) &\in \mathbb{U} \in \mathbb{R}^m \end{aligned} \tag{2.2}$$

where these constraints should hold $\forall t \geq 0$. Denote by $x_0 = x(t)$ the value of the measured state at time t, and by $\hat{x}_k$ $(\hat{u}_k)$ the predicted value of the state (input) at time $x(t+k)$ $(u(t+k))$.

The prediction is carried out over $k = 0, \ldots, N$, where N is called the prediction horizon. Then the corresponding finite time optimisation is stated as:

$$J_N^*(x_0) = \min_{U_N} F(x_N) + \sum_{k=0}^{k-1} l(x_k, u_k) \tag{2.3a}$$

$$\text{s.t.} \quad x_{k+1} = f(x_k, u_k), \tag{2.3b}$$

$$x_k \in \mathbb{X}, \forall k = 0, \ldots, N, \tag{2.3c}$$

$$u_k \in \mathbb{U}, \forall k = 0, \ldots, N-1 \tag{2.3d}$$

$$x_N \in \mathbf{X}_f \tag{2.3e}$$

where $J_N^*(x_0) \in \mathbb{R}$ denotes the optimal value of the performance index (2.3a) as a function of the initial state x_0, $U_N = (u_0^T, u_1^T, \ldots, u_{N-1}^T)^T \in \mathbb{R}^{Nm}$ is the whole sequence of optimal control inputs to be determined, x_N is the final predicted state, $F(x_N)$ is the *terminal penalty* function, $l(x_k, u_k)$ is the *stage cost* at step k, and $\mathbf{X}_f$ represents a *terminal set* constraint which is often added to obtain certain properties for formulated MPC (e.g. stability and all-time constraint satisfaction). This formulation is called nonlinear MPC (NMPC) and is general enough to describe a wide range of systems, including UAV applications.

In particular, if the system (2.3b) is in the discrete linear form as:

$$x(k+1) = Ax(k) + Bu(k) \tag{2.4}$$

where $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$, the cost function (2.3a) is in quadratic form, i.e. $F(x_N) = x_N^T P x_N$ and $l(x_k, u_k) = x_k^T Q x_N + u_k^T R u_k$ with P, Q and R being positive definite weighting matrices, and the constraints are also in the linear form, the formulation can be referred as linear MPC.

2.1.2 Issues of MPC

One of the open issues in MPC application is the computational burden associated with the solution of the MPC optimisation problem. The concept of receding horizon implementation of MPC assumes that the optimisation will be terminated within one sampling interval. Thus, the time required for solving optimisation problems sets a limit on the maximum admissible sampling rate of the control system.

For linear MPC, the resulting optimisation problem (2.3) can be converted into a standard Quadratic Programming (QP) form [70]:

$$\begin{aligned} & J_N^*(x_0) = x_0^T Y x_0 + \min_{U_N} \{U_N^T H U_N + x_0^T c U_N\} \\ \text{s.t.} \quad & Az < b, \\ & GU_N \leq W + Ex_0 \end{aligned} \tag{2.5}$$

where the input control sequence U_N is the optimisation variable, H and c are matrices formulated according to the system equation (2.4) and control horizon N, and G, W, E are also matrices reformulated based on the original optimal control problem. Such a QP problem can be solved online by efficient algorithms, such as active set method and interior point method. Both free and commercial packages are available for this problems. Moreover, the recently developed multi-parametric programming technique can solve this QP off-line to deliver an explicit MPC [63].

For nonlinear problems, the MPC optimisation problem in (2.3) can be cast as a nonlinear program (NLP) in the form of

$$\begin{aligned} & \min_z V(z) \\ \text{s.t.} \quad & c(z) \geq 0 \end{aligned} \tag{2.6}$$

where the minimisation variable z is typically the input control sequence U_N to be found. The cost function from (2.3a) is reformulated into $V(z)$. The nonlinear system dynamics (2.3b) are integrated into the constraint $c(z) > 0$. The state, input and terminal set constraints in (2.3c)-(2.3e) are evaluated over the full time horizon and also integrated into the constraint $c(z) > 0$.

It is very difficult to solve this constrained nonlinear optimisation problem to reach the global minimum because of the possible local minima and saddle points. However, in real-time implementation, it can be solved locally by employing an approximation. Common choices are sequential quadratic programming (SQP) and interior point methods (IPM). SQP method are based upon the assumption that the problem can locally be modelled as a quadratic program. The result is then used to find the next quadratic program, and in this way the problem is solved by sequential iterations. The commercial software NPSOL [43] implements this technique. Newer methods use an IPM, which is based upon eliminating the inequality constraints and replacing them by equality constraints through the use of slack variables and logarithmic barrier functions. KNITRO [43] is a software package using IPM for solving nonlinear programs. Similarly, IPOPT is an open source code that can tackle nonlinear optimisation using this algorithm [119]. In addition, a multiple shooting algorithm used in the OptCon package [110] provides an extremely efficient solution for nonlinear MPC by exploiting the special structure of the optimisation problems. Nevertheless, the computation time increases rapidly with the dimension of the nonlinear system, which will be an obstacle for applying it on systems with fast dynamics.

Another issue associated with the MPC approach is the stability of the closed-loop system. The general MPC form does not guarantee the stability due to a finite prediction horizon used in optimisation. With the development of the MPC theory many methods have been proposed to find sufficient conditions for stability. For nonlinear MPC, the widely used methods are to modify the optimisation problem by adding a terminal equality constraint [74], a terminal region constraint with dual mode control [84], or a terminal region constraint with a terminal penalty [14; 19]. A comprehensive literature review on this topic can be found in [75].

MPC has made its success in the process control area. The improving of technology and control theory enables the application of MPC in many new problems [71]. There is now a great interest in introducing MPC in other process and non-process applications such as paper-making, supply chain management, control of many kinds of vehicles, including marine, air, space, road and off-road. Some interesting biomedical applications are also very promising. Finally, the interest in the control of complex systems and networks is also significantly increasing.

These new applications frequently involve tight performance specifications, model updates and formulation adaptations because of changing operating points, environment, and safety-criticality. To this end, NMPC formulations which offer guarantees of stability, robustness and output feedback are expected to be more promising and to attract more research attention [3]. Moreover, the significant effort in developing efficient solutions of optimisation problems both using an explicit and a numerical approach is also critical for a wider diffusion of NMPC.

2.2 MPC on UAV applications

Many aspects need to be considered in order to achieve a higher degree of autonomy for a UAV system, such as task assignment, route planning, path optimisation, trajectory tracking and vehicle stabilisation. Simultaneously meeting requirements in all these aspects is very complicated and almost intractable, especially when an UAV is operated in a dynamically changing environment. As a result, a hierarchical decomposition (see Fig 2.2) is widely used in UAV community to divide the autonomy into three decision making and control layers, namely mission planning, trajectory planning and flight control [116]. The three operational layers cover necessary functions that support UAVs to execute tasks autonomously with necessary information provided by a situation awareness system.

The highest layer is the mission planning, where a planner selects and priori a sequential list of missions/tasks to be carried by the UAVs. In the case of multiple UAV, the mission planner also deals with task assignment and allocation. A mission planner usually sets up a global goal for each UAV in the operation area by taking into account the task requirements, vehicle capability and environment knowledge.

The path planning layer stays in between the mission planning and low-level flight control. Its main function is to generate a trajectory that fits vehicle's performance limits and environment constraints like obstacles while still being able to carry out tasks given by the mission planner. Moreover, additional constraints or requirements, such as generating shortest paths, minimum risk paths, and minimum fuel and energy consumption paths, can also be included for better performance and efficiency of the mission. As one of the key areas in UAV autonomy, numerous algorithms have been developed within this layer, where a thorough discussion can be found in the book [115].

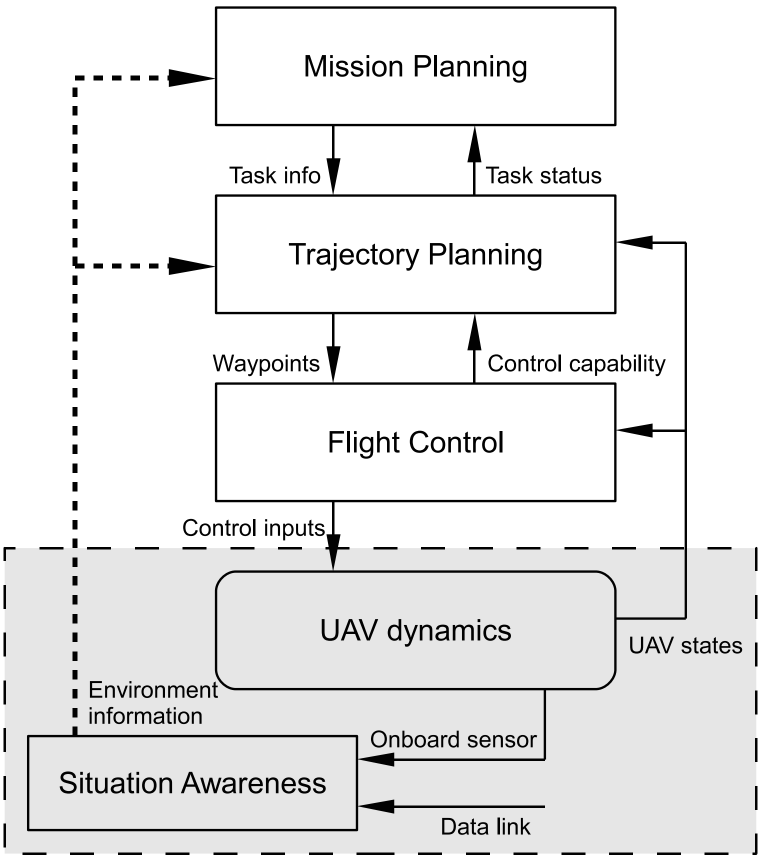


Figure 2.2: Hierarchical Control Architecture of UAV

The flight control layer is the lowest level. It stabilises the vehicle dynamics in the presence of uncertainties and disturbances and endows them with the capability to track a trajectory generated by a path planner. It is the fundamental function for an autonomous vehicle, especially when vehicle dynamics are complex and unstable.

In UAV applications, the boundaries among the different hierarchies are not strictly defined. Both mission planning and path planning involve a process of determining a series of waypoints leading UAVs to goal positions. The difference may lie in the time-scale on which planners operate and the horizon that planning expands. In this sense, path planing actions can be split into two categories: global path planning and local path planning [36]. The former one needs to generate a path from the current position of vehicle to the final target or at least being able to attain the final target in the future. The global knowledge of the entire operational area is needed beforehand, and the global path needs to update in response to the new mission goal or environment information. The time-scale of the global path planning is relatively large such that the lower layer may consider it as static. On the other hand, the local path planner only responds to the surrounding environment of the vehicle. It re-plans local trajectories that avoid immediate dangers such as collision with other vehicles, pop-up obstacles and in some case threats, and rejoin to the global path after manoeuvres. Local path planning usually works in a short time-scale, ideally in real-time, to react to newly detected obstacles. Moreover, it needs to respect the vehicle's dynamic characteristics, such as velocity and acceleration constraints and potentially the higher-order differential equation constraints associated with vehicle dynamics. It can be noted that in both the local path planning level and the flight control level, vehicle dynamics are taken into account, although in different degrees of completeness. If the path planner uses full vehicle dynamics and updates frequently enough to external disturbance, or from an opposite point of view, if a flight controller with the prediction ability uses longer horizon, it is possible to integrate low-level and mid-level into a single planning/control layer.

In each layer the planning and/or control problem can be abstracted as an optimisation problem with the vehicle states and goals or reference trajectory integrated in the performance index, which is in turn subjected to differential constraints (vehicle dynamics), logical constraints (decision making) and inequality constraints (state limitation and input saturation). However, the optimisation along the entire mission period is computationally prohibitive, and is also not applicable as new information is updated during the operation. To this end, the feature of MPC or RHC makes it a suitable strategy, because the feedback nature allows it to incorporate the latest environmental information in updating the original plan, and a finite planning horizon used in optimisation requires only the local information and reduces the computational effort. In the following, we will give a brief overview of MPC techniques used in UAV in terms of both trajectory planning and flight control.

2.2.1 Trajectory planning

Various algorithms for UAV trajectory planning have been developed to tackle application problems in different scenarios. Comprehensive reviews can be found in [36; 87]. In terms of MPC or RHC based algorithms, they may have different formulations (linear, nonlinear, etc.), functions (path-planning, cooperation, formation, etc.), and properties (computation load, prediction horizon, stability, etc). This subsection tries to categorise them according to problem formulations that in turn very much depend on the adopted vehicle models.

A very popular trajectory planning approach is to formulate problems into Mixed Integer Linear Programming (MILP). MILP is a powerful optimisation framework allowing the inclusion of integer variables and discrete logic in a continuous linear optimisation [31]. Obstacle avoidance, as well as collision avoidance, can be enforced with logical constraints by specifying that the vehicle must be either above, below, left or right of a non-fly zone, while the dynamics or kinematic properties of the vehicle are retained as continuous linear constraints. Moreover, other decision feature like task assignment can also be included into this optimisation framework.

Early work of using MILP for path planning can be found in [101] and [97] applied to spacecraft manoeuvring. Incorporating MILP into RHC framework provides efficient solutions to path planning in terms of obstacles avoidance [4; 95], multi-vehicle coordination [5; 76], and task assignment and scheduling [1; 2; 56]. More advanced algorithms based on the combination of RHC and MILP have also been developed to enhance UAV planning capabilities. For instant, [102] considers safety issue to guarantee the feasibility of future planning. In [62], the authors use the robust-safe-but-knowledgeable trajectory planning to achieve robustness to external disturbances and ensure safety with respect to changing environment. A upgraded version of this algorithm is given in [60] for distributed cooperative control of a UAV fleet. Similarly, in [61; 96], decentralised MPC algorithms for cooperating multi-vehicles are developed to reduce the computational and communicational demand. Except for these achievements on algorithm and theoretic aspects, various experiments and flight tests of this kind of technique are also reported in [23; 57; 98; 103].

Among the existing algorithms of using MILP, cost-to-go functions are usually incorporated in MPC formulations as terminal penalties added on the cost functions. In controlling general nonlinear systems, such a function is designed based on control Lyapunov functions to guarantee the stability [48]. For trajectory planning, a cost-to-go function is used to capture the trajectory beyond the prediction horizon, so that the resulting MPC is able to overcome the finite planning horizon and guide UAVs to global goals. Cost-to-go functions can be heuristically calculated based on a visibility graph representation of the environment with Dijkstra's searching algorithm [4]. More recently, a finite-state *spatial value function* is proposed to serve as a cost-to-go function, which captures the critical interaction between the vehicle dynamics and environment [79].

It shall mention that although MILP is versatile and powerful, it is restricted by using a linear model to represent UAV dynamics. A typical approximation is to consider a UAV as a point mass moving with limited speed and limited acceleration, such as

$$\begin{bmatrix} \boldsymbol{p}(k+1) \\ \boldsymbol{v}(k+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & \Delta t\boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{p}(k) \\ \boldsymbol{v}(k) \end{bmatrix} + \begin{bmatrix} \frac{(\Delta t)^2}{2}\boldsymbol{I} \\ \Delta t\boldsymbol{I} \end{bmatrix} \boldsymbol{a} \tag{2.7}$$

where $\boldsymbol{p}$ is the position vector, $\boldsymbol{v}$ is the velocity vector and $\boldsymbol{a}$ is the acceleration vector. Δt is the discretisation time step, and matrices $\boldsymbol{I}$ and $\boldsymbol{0}$ express an identity matrix and a zero matrix, respectively. This representation may be acceptable in global planning, but is not suitable for local planning of agile movements as the resulting trajectory may be dynamically infeasible.

For more realistic predictions, nonlinear models of vehicle dynamics are involved in MPC framework and result in nonlinear MPC. Initial trails have been applied in a "collision-free" environment for the trajectory generation for Caltech ducted fan [47] and for multi-vehicle formations [27]. In [111], the authors aim at guiding a vehicle with nonlinear dynamics through a urban area with obstacles. A two-step approach is adopted, where a feasible nominal trajectory is generated first, then linearisation is performed around the nominal trajectory to convert the nonlinear optimisation problem into a linear time varying one, which is finally solved within a RHC framework. Ref [68], proposes a nonlinear MPC path planner for an information gathering task, which shows that MPC framework can effectively deal with UAV dynamic constraints, multiple vehicle situations and a range of objective functions. Another interesting planning framework using NMPC is proposed for sensing missions [114]. In this work, onboard sensor model is incorporated in the online planning for the evaluation of information gathering, and a variable prediction horizon is adopted to handle range-limited sensors. Cooperation among UAV teams are achieved by sharing sensing information and future actions. Formation flight of UAVs can also be handled by nonlinear MPC as in [108], where the authors compared performance and computation loads of three different MPC formulations, i.e. centralised, sequential decentralised, and fully decentralised methods.

The aforementioned nonlinear MPC methods for path planning use general dynamic models in their predictions. The complexity of such models directly affects the computational load of the online optimisation. To this end, a kinematic model is usually adopted in nonlinear MPC formulation to represent aircraft's planar movement, such as

$$\begin{aligned} \dot{x} &= v \cdot \cos\psi \\ \dot{y} &= v \cdot \sin\psi \\ \dot{\psi} &= \omega \end{aligned} \tag{2.8}$$

where the state vector is the 2-D position and heading $\boldsymbol{x} = [x, y, \psi]^T$, and the inputs are the velocity and turn rate $\boldsymbol{u} = [v, \omega]^T$. Constraints can be imposed on the input vector to limit the aircraft's ability.

Another important factor associated with the computation is the length of the prediction horizon. Hence, even if it also depends on the hardware capability, there is a three way trade-off between the planning horizon, vehicle model accuracy and re-planning frequency, which consequently derives different MPC formulations for different planning purposes. For example, in ref [114] a unicycle model like (2.8) is incorporated to predict for 8-15 future steps, and the resulting re-planning cycle is 1-3 seconds. In [100], a detailed UAV dynamic model with 12 states is employed and the resulting MPC can reach 10Hz (0.1 second) updating rate but with a short prediction horizon of 5 steps. The former one is suitable for mid-range path planning, whereas the latter one is adequate for planning aggressive manoeuvring, of course in a local range.

To facilitate local path planning for agile movements of UAVs, more dynamics information of the vehicle must be taken into account, so that the optimised trajectory is dynamically feasible. Furthermore, the formulated optimisation problem has to be solved quickly enough to provide a high updating rate in order to respond to external environments. To this end, there is a class of methods that design a trajectory directly in the *output space* rather than by forward simulating a vehicle model [66; 112]. The constraints arising from vehicle dynamics can impose on the trajectory through the differential flatness property. This property allows vehicle's states and controls to be expressed in terms of the output vector, i.e. the trajectory, and its derivatives. Such a strategy can be found in [22] for generating an obstacle-free and time-deterministic trajectory for a quadrotor. In [7], a similar technique is applied to local path planning of a micro air vehicle in obstacle-rich environments. The vehicle performance limit, in the form of speed, acceleration and jerk constraints, is generated off-line and available online via look-up tables. A multiple vehicle situation of this algorithm is reported in [6].

In these applications, trajectories in receding horizon optimisation are represented by polynomial curves such as Chebyshev polynomials, Laguerre polynomials, Bezier polynomials, etc. Generally, they can be expressed as a finite series involving a product of coefficient k_i and a basis function B_i

$$p(t) = \sum_{i=0}^{n} k_i B_i(t) \tag{2.9}$$

where $p(t)$ is a curve as a function of parameter t, and n is the order of basis function. Trajectory design and optimisation involves tuning the scaling factors k_i to determine the shape of trajectories. Planning in output space reduces computational load while being able to provide a feasible trajectory for UAVs. However, it needs a dedicated control system to track such a trajectory.

2.2.2 Flight control

Using MPC techniques to stabilise an UAV with complicated dynamics in the presence of disturbances and uncertainties, and ultimately to enable the trajectory tracking function is a challenging work. It is mainly because the conflicts between the fast dynamics of aerial vehicles and the computational burden of MPC techniques. However, the benefits, like constraint handling, of using MPC encourage researchers investigate the possibility of implementing such techniques.

For most aircraft, their dynamic systems can be further divided into two subcomponents: the outer-loop kinematics and inner-loop attitude dynamics [77]. Consequently, the flight control system can be considered at two levels: guidance control and stability augmentation, governing the outer-loop and inner-loop of an aircraft, respectively [88]. The guidance control aims to track the trajectory or waypoints given by a path planner to achieve a given mission. The stability augmentation is also referred as inner-loop control, and in some case as attitude control. Its objectives are to provide stability for unstable dynamics and improve the response when the higher level guidance provides commands. No matter for guidance problems or stab1ilising problems, from a control point of view it is essentially to deal with a nonlinear plant, which cannot be abstracted by a linear model as in the path planning level.

On the other hand, the control bandwidth associated with flight control, especially for the inner-loop, is considerably higher than the path planing layer. Although it may also depend on particular UAV models, a common control updating rate is 30Hz for a fixed-wind aircraft and 50Hz for a vehicle like a helicopter or quadrotor [13].

As a trade-off between the plant complexity and computational demands, some MPC based algorithms tend to use linear settings so that the formulated optimisation problem can be solved by efficient QP solvers. To convert a nonlinear flight control problem into a linear form, various linearisation techniques are adopted. Standard linear MPC has been used on helicopter inner-loop control in [11], where helicopter dynamics model is linearised around the hovering condition and included in the MPC framework. Whereas the nonlinear kinematics is controlled by a multiple PID (proportionalintegralderivative) controller to achieve the trajectory tracking function. Another linear but explicit MPC is designed for controlling a toy helicopter [26]. Similarly, a model linearised around the hover condition is adopted. In [51], a UAV guidance control problem is solved by using MPC, in which again a linearised model is used. However, this model is updated every sampling time based on the measured states, and the nonlinear constraints are converted to linear forms by using a polyhedral approximation. Such a strategy is also applied on a F-16 aircraft for longitudinal control [52]. This study shows that in a MPC framework a linearised model dependent on flight condition is a necessary requirement for providing good performance as opposed to a single linear time invariant (LTI) model based method. MPC formulated based on a linear time varying (LTV) model is also reported for controlling autonomous vehicles [28]. In this application, successive online linearisation is performed at the current operating point at each time step and a linear MPC is designed for the resulting model. The stability issue arising from using LTV model instead of the original nonlinear model is discussed in [29]. In addition, linear MPC combining with feedback linearisation techniques can be used to a delivery flight control solution. In [117] the full attitude dynamics of a re-entry vehicle is feedback linearised, and the resulting linear system with new control inputs is incorporated into the MPC framework. Input and state constraints are applied on the MPC formulation with a constraint mapping algorithm developed to map the input and state constraints on the new inputs after feedback linearisation.

As listed above, although flight control using NMPC can be found in various places, most applications use auxiliary techniques to avoid using full dynamics models of vehicles in their prediction. Therefore, they can only focus on either outer-loop guidance or inner-loop stabilisation. If both outer and inner loops of an aircraft are incorporated into a MPC framework, the resulting control system has the so-called integrated guidance and control (IGC) property. Few applications have followed this philosophy. Ref [85] describes the receding horizon control of Caltech ducted fan. The constrained RHC technique used in this application includes the full dynamics of the vehicle and considers the computational time. The resulting optimisation problem is parametrised by the B-spline technique and solved by using Nonlinear Trajectory Generation (NTG) software package. However, this method only works efficiently with a system that is differential flatness. In [100] an integrated estimation, planning and control framework is developed for an autonomous aircraft. A nonlinear MPC based on full aircraft dynamics is designed for trajectory tracking. The formulated nonlinear optimisation problem is solved by a FP-SQP algorithm, which allows the online calculation to be terminated to a sub-optimal solution to accommodate the time critical control. The proposed estimation and control algorithms have been verified on a SeaScan UAV through simulations and flight tests [10].

A series of remarkable work has been carried by Kim, Shim and their co-workers [53; 54; 106] to develop a NMPC based tracking control for helicopters. At beginning, the feasibility of using NMPC on autonomous helicopters is investigated in [53]. As a detailed helicopter dynamic model is included in the MPC framework, the resulting control signals can be directly applied to simultaneously stabilise the helicopter and achieve the desired tracking performance. Simulations show that the NMPC outperforms multi-loop PID controllers and has good robustness to parameter uncertainty. Later, a decentralised NMPC is proposed for multiple vehicles [106]. To produce the collision-free trajectories among multi-vehicles, a potential field method is incorporated in the NMPC framework. The formulated optimisation problems, including potential field terms and the full vehicle dynamics model, is very complicated. To overcome the difficulty, a gradient-search-based optimisation is used to solve it in real-time. Extended applications from this framework can be found in [105] for autonomous exploration in unknown area, in [107] for see-and-avoid manoeuvring, and in [113] for vision-based landing and terrain mapping.

Although successful achievements of implementing NMPC to fast dynamics have been demonstrated in these applications, the fundamental problem still exists. The inclusion of the complicated dynamics model dramatically increases computational demands. It can be noted that in order to enable numerical tractability and improve reliability, a hierarchical flight control system actually takes in charges of helicopter dynamics in the flight test. This setting means that the NMPC is employed as a trajectory planner and/or guidance controller, which sends a dynamically feasible and obstacle-free trajectory to a low-level controller used for tracking and control purposes. However, this configuration can be attributed to the low bandwidth of the NMPC algorithm.

2.3 Autonomous helicopter and flight control

Autonomous helicopters are versatile flying machines that have drawn considerable interests from both industry and academia. Comparing to the fixed-wind counterpart, they are capable of vertical take-off and landing (VTOL), fixed-point hovering, low-speed and low-altitude cruise, and performing aggressive manoeuvres. These features make them suitable for a broad range of applications. In the military side, missions like reconnaissance and surveillance, have already been carried out by unmanned helicopters like Northrop Grumman MQ-8 Fire Scout. Civil applications also have a wide perspective such as power line inspection, aerial video, search and rescue, etc. Although great success has been made, the development and application of autonomous helicopters are still at their initial stage. More sophisticated applications propose demands on better dynamics performance and higher autonomous ability. To this end, many research projects and industry trials on autonomous helicopters have been carried out. A few examples are listed below to show the activities in this area:

- Aalborg University – ASETA project;
- Carnegie Mellon University – Yamaha R50 Based UAV helicopters;
- ETH Zurich – AkroHeli;
- Georgia Institute of Technology Program – GTMax project;
- Massachusetts Institute of Technology – Draper Autonomous Helicopter;

- National University of Singapore – HeLion and SheLion project;
- Stanford University – X-Cell Tempest;
- University of California at Berkeley – BEAR project;
- University of Southern California – AVATAR project.

Nevertheless, the complicated dynamics of helicopters pose challenges for the flight control design, which include nonlinearity, multiple-inputs-multiple-outputs (MIMO), natural instability, and internal couplings. Other issue like model accuracy also affects the control design and performance. In the following, a brief overview regarding the modelling and control techniques for unmanned helicopters are provided.

2.3.1 Helicopter Modelling

There are two well-established approaches for modelling system behaviour in the control community: first-principles modelling, where models describing dynamics are built from scratch based on underlying physical laws; and system identification, where algorithms are used to find the relationship between the inputs and outputs of a given system using data collected from experiments.

A typical first-principle mathematical model for a small helicopter is nonlinear and contains many states, since it has to account for interactions between each component, such as motor, rotors and fuselage. The advantage of this kind of model is that it can cover a wide range of flight conditions. However, the quality may be degraded by inappropriate simplification and assumptions within modelling. Moreover, there are dozens of inherent physical parameters that need to be determined [42]. Some physical parameters can be measured directly, but the others may have to be measured by experiments or calculated based on assumptions or experience. All of these factors will cause inaccuracy and uncertainty in such models. Although, this issue could be addressed to some extent by tedious and trivial experimental validations and refinements.

A good example of using the first-principle method to modelling helicopter is the Minimum-Complex Helicopter Simulation Math Model [42]. A more recent and elaborate model for small-scale helicopter is given by Gavrilets [34], where the dynamics equations were developed using basic helicopter theory, and nonlinear expressions for external forces and moments were provided. The model was successfully used for aerobatic flight control design [33]. In addition, researchers in Aalborg University carried out a series of projects involving autonomous helicopters including the development of first-principle nonlinear models [40].

System identification, on the other hand, naturally focuses on the observation of input-output data, trying to provide a model that can interpret experiment phenomena. This method is more direct and more effective since it integrates validation into the modelling process. System identification is not a standalone process as it requires a good understanding of helicopter dynamics, hence the identification and first-principle approaches are essentially complementary and shall be integrated together to develop a promising model [78].

An early attempt to identify a small helicopter can be found in the Caltech experiment [86], where a linear model for attitude motions were derived and used. In order to guarantee the linear behaviour of a Kyosho EP Concept 30 helicopter, it was mounted on a stand allowing only angular movements. The authors used the linearised rigid-body equation of motion as a parametrised model, and identified these parameters by using a prediction error method (PEM). The resulting model was successfully used for control design for attitude movements. However, helicopter dynamics on the stand are not representative of a free helicopter, so there was still a big gap between the model and real flight response. Another notable work was carried out by Kim and Tilbury [55]. The authors developed a parametrised model using first-principle method which explicitly accounted for the flybar (stabilizer bar) dynamics. However their identification was based on single-input-single-output transfer functions, and the coefficients in the transfer functions had no physical meanings.

Actually there was no significant progress in the system identification approach of modelling a small-scale helicopter until Mettler and his co-workers completed their remarkable work [81; 82]. Their modelling and identification were based on Carnegie Mellon's Yamaha R-50 helicopter. First, a linear parametrised model was developed, which was mainly based on the rotor-fuselage equations with the extension of including stabilizer bar dynamics. With a proper simplification, the main rotor was modelled through first-order tip-path-plane dynamics, and the stabilizer bar was regarded as a secondary rotor. All the coupled components were connected by cross-coupling derivatives. The final model resulted in a linear MIMO state space model with 13 orders. Authors then applied frequency domain identification tool CIFER (Comprehensive Identification from Frequency Response) to identify these unknown parameters in the state-space model. The results showed that the model had a high fidelity even with a relative simple model structure. His parametrised model appears to be general enough to describe other small-scale rotorcrafts. Later on the same methodology has been successfully applied to MIT X-Cell 60 helicopter [80].

Following the model structure proposed by Mettler, a number of modelling projects have been carried out by researchers. Some applications used this model structure but relied on linear time-domain identification tool PEM to estimate the parameters, including Ref [54; 109]. Although there were difficulties reported in converging to the global minima during the identification, the control designs based on these models were successful. A more comprehensive study has been carried out by LaCivita et al [64]. The authors combined nonlinear modelling with linear system identification to produce a high fidelity model that can cover a larger operating condition. More recently, Grauer and Conroy adopted a two-step equation-error/output-error process in both time and frequency domain to identify a miniature helicopter [37].

2.3.2 Control design

For an ordinary helicopter, its position and orientation are usually controlled by means of five control inputs: the main rotor throttle (power to the rotor), the main rotor collective pitch inputs, which combining the throttle input can directly affect the helicopter height (altitude control), the tail rotor input which affects the heading of the helicopter (yaw motion) and produces the anti-torque to the main rotor, the longitudinal cyclic which alters the helicopter pitch motion and the longitudinal translation, and the lateral cyclic, which affects the helicopter roll motion and the lateral translation. Hence, a helicopter is a multivariable nonlinear uderactuated system with strong coupling in some control loops.

Initial trails on controlling an unmanned helicopter are based on heuristic methods [12; 54]. By decomposing helicopter dynamics into outer-loop and inner-loop and ignoring the coupling effects under the assumption of non-aggressive flight, a cascaded control architecture can be designed based on the observation that the translational motion depends on the attitude motion which is further related to control inputs. In this kind of control structure, multiple PID controllers are usually employed to augment individual channels respectively. The benefit of using PID is that its parameters may be easily adjusted according to the behaviour of the plant, which allows for online tuning when the helicopter model is unknown. However, the drawbacks are obvious, such as overlooking coupling among different control variables, limited control bandwidth and agility.

With the mature of helicopter modelling techniques, model based control design has played more and more important roles in enabling autonomous flight of helicopters. Due to the complicated dynamics of a helicopter, various control techniques, including both linear and nonlinear methods, have tried to tackle the problem from different direction and focusing on different aspects.

Linear control designs based on the system decomposition and linearisation are still popular. In [109] two linear-quadratic-integral (LQI) controllers are designed separately for inner-loop attitude and out-loop position control of a SF-40 small helicopter. The control system is carefully designed to maintain the helicopter out of the nonlinear region, and its performance is validated through simulations and flight experiments. Ref [65] proposes the design and implementation of an H-infinity loop shaping controller on the Carnegie Mellon University (CMU) Yamaha R-50 robotic helicopter. The proposed controller consists of one multivariable (MIMO) inner-loop for stabilisation and four separate (SISO) guidance loops for velocity and position control. Due to its ability to handle uncertainties, although the controller is designed for hovering, it performs manoeuvres (square, forward turn, backward turn and nose-out circle) efficiently in flight tests. Another controller based on H-infinite is reported in [32], where the output-feedback design procedure is simplified to solve only two coupled-matrix design equations and an efficient algorithm is provided for solving these. This procedure allows output-feedback control design with pre-specified controller structures and guaranteed performance.

Although linear control design can achieve good performance and robustness, its performance may be restricted from the beginning due to the linear model is not adequate to describe all the motions of a helicopter. Therefore, nonlinear control methods have been widely applied. Feedback linearisation as a common nonlinear control technique is implemented on helicopter tracking control [58]. This study shows that the helicopter model cannot be converted into a controllable linear system via exact state space linearisation; however, by neglecting weak couplings, input-output linearisation based on the approximated model can achieve bounded tracking error.

A flight control approach based on a state-dependent Riccati equation (SDRE) can be found in [9]. The control design uses a six-degree-of-freedom nonlinear helicopter model that is manipulated into a pseudolinear form. Therefore, system matrices are given explicitly as a function of the current state and the control effort can be calculated by minimising a quadratic-like performance index. The formulated standard Riccati equation is then solved numerically at each step of a 50 Hz control loop to deliver the nonlinear state feedback control online. In addition, the SDRE control is augmented with a nonlinear compensator that addresses issues with the mismatch between the original nonlinear dynamics and its pseudolinear transformation. The feasibility of real-time implementation of the proposed algorithm has been demonstrated by flight testing on a XCell-90 and a Yamaha R-MAX helicopters. However, the constraint handling feature of SDRE is not included in this application.

Interests of designing nonlinear robust controllers are also pursued. In [46] the problem of controlling the vertical motion of a nonlinear model of a helicopter, while stabilizing the lateral and horizontal position and maintaining a constant attitude, is investigated. A nonlinear controller, which combines nonlinear adaptive output regulations and robust stabilisation of systems in feedforward form by means of saturated controls, is designed and tested in simulations. The simulation results show robustness against uncertainties on the model and on the exogenous reference signal. Following the inspiration from this study, [73] solves the problem of controlling the vertical, lateral, longitudinal and yaw attitude motion of a helicopter along desired arbitrary trajectories. The proposed control structure is a mixture of feedforward actions (computed according to reference signals and a nominal model inversion), and feedback terms obtained by combining high gain and nested saturation control laws. Experimental results obtained on a small scale helicopter are also presented to show its performance and robustness.

Examples of using adaptive control methods can be found in [49] and [67]. In the first work, neural network based on adaptation to uncertainty in the attitude, as well as the translational dynamics, is introduced, thus minimising the model error and leading to more accurate position tracking. The pseudocontrol hedging method is used to enable adaptation to occur in the outer loop without interacting with the attitude dynamics. A pole-placement approach is used that alleviates time-scale separation requirements, allowing the outer loop bandwidth to be closer to that of the inner loop to increase position tracking performance. The second work presents a nonlinear adaptive controller to endow a small helicopter with the ability of aggressive flight. Adaptive backstepping technique is employed to systematically integrate the proposed controller with the online parameter adaptation rule to the vehicle mass variations and with the recurrent neural network (RNN) approximation to the coupling effect between the force and moment controls.

An interest part of those two studies is that both works realise (and overcome) the drawback in the conventional flight control design of two time-scale separation, which assumes that the bandwidth of the outer-loop dynamics is much lower than that of the inner-loop dynamics. This assumption is actually unfavourable to the aggressive and precise control of trajectory tracking. This also reflects the trend in developing flight control for autonomous helicopter: synthesis entire dynamics as an integrated system and increase the control bandwidth.

Except for trajectory tracking of helicopters, a few publications have focused on specific control problems imposed by autonomous helicopters. Autorotation as an emergency manoeuvre for helicopters safe landing in case of losing power has been studied in [25], where an NMPC with the optimisation problem solved by RNN is used to handle the problem. Ref [8] investigates the controlling of a helicopter flying a slung load. An estimation and control system is designed for this purpose where the estimator provides position and velocity estimates of the slung load and is designed to augment existing navigation in autonomous helicopters, and the controller is a combined feedforward and feedback scheme for simultaneous avoidance of swing excitation and active swing damping.

As the research on autonomous control of helicopters booms rapidly in recent years, it is very difficult to cover all aspects in this area. However, a number of observations can be made based on above literature review in terms of control design for autonomous helicopters.

First, classic control methods, like SISO PID control, are able to stabilise a helicopter. Most recent control designs pay attentions on specific requirements such as good robustness, aggressive flight and disturbance rejection.

Second, a common way to breakdown the design challenge is to use the separation of the slow translational movement and fast attitude motion. The coupling effect between them (which is rather significant for helicopters) is either ignored (linear control), tolerated (robust control) or estimated and compensated (adaptive control) in the control framework.

Third, there still a gap between theory and practice. Implementing and validating a control method on a real helicopter is not straightforward. The most of advanced algorithms are exercised through simulations. This is partially because the difficulty in development of a helicopter hardware including avionics and is partially due to the difficulty of acquiring an accurate model that most model based techniques rely on.

2.4 Summary

This chapter provides a literature review on three aspects that will be covered in this thesis, namely the MPC technique, its applications on UAV and autonomous helicopters. First, the basic principle of MPC is explained, and issues associated with MPC are discussed. Next, existing MPC techniques used in UAV applications are surveyed and categorised in detail. In the end, popular flight control algorithms for autonomous helicopters are reviewed, where some classic issues and new trends are discussed.

Chapter 3

A rapid prototyping platform for UAV research

3.1 Introduction

Research efforts have been increasingly devoted to the field of UAV since it is widely believed that numerous civil and military applications can be found for UAVs. To enhance intelligence and autonomy of UAVs, advanced methodologies ranging from individual flight control, multi-vehicle coordination control to mission planning and decision making have been developed. These algorithms need to be evaluated and verified in order to assess their practical performance, and to pave the way for inserting them into real world applications.

It is well known that flight tests are very expensive, impose high risks for personnel and assets, and require a large airfield and heavy logistic support. Due to these reasons, most of the research and development work in aerospace vehicles such as aircraft, missiles, and rotorcrafts are still evaluated by numerical simulations. This has been identified as one of the main obstacles for transferring advanced control concepts and methods into real engineering applications [91]. On the other hand, the research on miniature helicopters carried out in this thesis needs a facility to study the behaviour of the helicopter, design the control algorithms and evaluate their performance in a realistic environment. Therefore, it is imperative to have a proper test facility to facilitate these research activities and de-risk the new research ideas generated from these activities. For this purpose a unique indoor rapid prototyping platform has been developed in the Autonomous System Lab at Loughborough University. This chapter details the development of the platform and discusses its basic features and potential functions. Some initial test results are also included to show its capabilities.

A number of attempts have been made to develop various hardware-in-the-loop simulation and flight test facilities. For example, at Georgia Institute of Technology, an open system UAV testbed referred to as RTMax was developed to investigate flight control algorithms [50]. Researchers in the University of California at Berkeley use a platform comprised of a fleet of commercially available rotary-wing and fixed-wing UAVs to study UAV applications such as situation awareness and collision avoidance in unknown urban areas [104]. In the Aerospace Controls Laboratory at the Massachusetts Institute of Technology (MIT), an outdoor flight test platform integrated with a fleet of eight fixed-wing autonomous UAVs provides a means for evaluating coordination and control algorithms [57]. In general, these outdoor platforms provide the most realistic flight tests. However, they suffer a number of drawbacks. Firstly, they are expensive and have limitations on how quickly they can perform flight tests due to the constraints on accessing a large airfield. Secondly, most outdoor UAVs can only be flown in good weather conditions to avoid risks. Thirdly, UAVs typically require a large logistic support team, which makes testing logistically difficult and expensive.

In contrast, indoor test environment may provide a much more flexible, accessible and cheaper facility for testing UAVs and for general flight control research. The main constraints for indoor testbeds are confined space and strict requirements on avionic systems. To this end, MIT's RAVEN tesbed is the most impressive indoor testbed, where an environment with a number of quad-rotor aircraft has been developed to investigate long duration missions and health management research [44]. Although, most of the hardware components are commercial-off-shelf parts, the software environment, known as open control platform, was initially developed and provided by The Boeing Company. Another promising testbed named GRASP has been developed in the University of Pennsylvania more recently [83], which is a multiple micro aerial vehicle (MAV) testbed used to support coordinated, dynamic flight of MAV and associated applications.

The indoor testbed described in this chapter is characterised as a rapid prototyping platform. It can implement and verify algorithms at both control level and mission level in the real world in a seamless way; speeding up the development process from theory to practice. The main features of this platform are:

- Flexibility and versatility. Almost all small-sized commercial model vehicles and rotorcrafts can be operated in this environment to construct different scenarios.

- Low cost and low maintenance. Commercial-off-shelf vehicles are affordable, easy to repair from crashes and no running costs except for charging batteries.
- User friendly. This platform allows a user to carry out flight tests without being an expert in coding or electronic.
- Rapid prototyping. This allows researchers to start from algorithm development, complete numerical simulations and final physical implementation all in the same software environment.

3.2 Platform architecture and components

3.2.1 Design challenges and philosophy

One objective of the rapid prototyping platform is to enable researchers to test a variety of algorithms applicable to UAV systems, not limited to helicopter flight control, in nearly real-world scenarios. Thereby, vehicles with good handling quality and manoeuvrability are needed in this platform. Another objective is to simplify the operation procedure of flight tests and allow an individual to carry out the entire process from algorithm design to evaluation. This also means that components must require a minimal modification, have good reliability and are easily maintained. To meet these demands, the platform adopts proper commercial-off-the-shelf equipment and combines them effectively. The key constraint of an indoor test facility for flight testing is the limited operation space. Consequently, only small unmanned vehicles can be used to perform various realistic flight tests, other than just taking off and landing. This implies very little payload or no payload can be put on these small aerial vehicles, therefore onboard sensors are not appropriate. The core technique in the platform is an object tracking system known as Vicon [118]. Vicon allows a ground station computer to perceive the position and attitude of vehicles in the test area, instead of mounting an onboard computer and a sensor suite on vehicles in the conventional way. In this way, low-cost off-the-shelf radio controlled (R/C) vehicles can be used in the platform without modifications since there is no significant payload requirement on the aerial vehicles except markers for the Vicon system. In addition, all these separate components are finally integrated into the Matlab/Simulink environment, which is widely used in academia and industry for research and development.

The architecture of the platform follows a hierarchical design. Both high-level autonomous algorithms and low-level control algorithms are built into the ground station, but control algorithms can be modularised and customised for different vehicles. Modularisation allows easy addition or removal of different types of vehicles, as needed for different scenarios. Each low level control model has the capability to access hardware directly to enable their functions. The structure of the platform is shown in Fig.3.1.

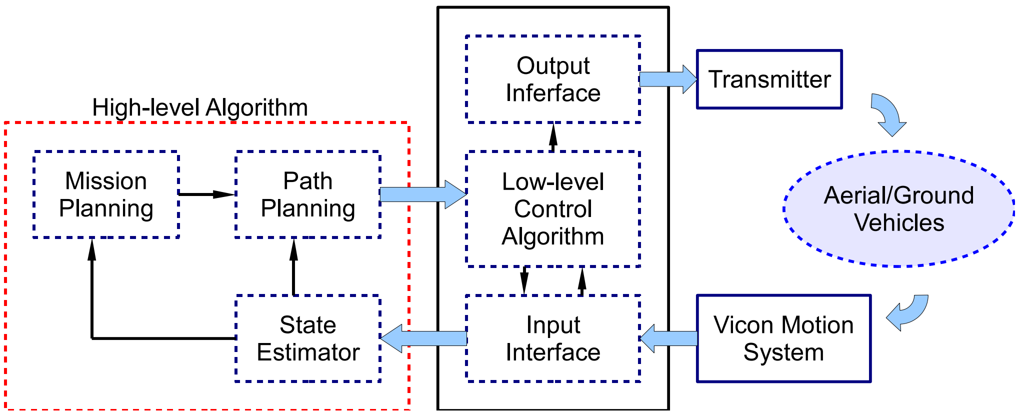


Figure 3.1: Platform structure

This hierarchical architecture also reflects the configuration in realistic UAV applications. In practice, the low-level control algorithm has to be executed by an onboard autopilot to react to the dynamics of an UAV. The path planning and mission planning algorithms can be carried out at remote ground control station if the planning focuses on a large time-scale operation, or alternatively by a secondary onboard computer to fast respond the surroundings in a local range.

3.2.2 Aerial/Ground vehicles

Small-scale vehicles play an important role in indoor tests for emulating the behaviour of UAVs and for setting various scenarios. These vehicles in the platform, especially aerial vehicles, need to be low-cost, low risk but be highly flexible and have good maintainability. As a result, standard radio-controlled helicopters are chosen as test vehicles, due to their suitability for flight in confined area, and more importantly, to their interesting flight dynamics. One model helicopter adopted is called Century Hummingbird (see Fig.3.2a). It is a fixed pitch electric helicopter with a relatively low rotor tip speed, which means a low power consumption and less energy in the main rotor system making the helicopter considerably safe to operate in an indoor environment. Its plastic components can also be easily replaced after a crash. This kind of helicopter is mainly used for UAV operational research and high-level tasks, such as the mission planning and task allocation, where a detailed consideration of the aircraft dynamics are not needed. Another helicopter used in the testbed is the Align Trex-250 helicopter (Fig.3.2b), which is a collective pitch helicopter with well-designed Bell-Hiller stabiliser. It is capable of aerobatics with good control handling, and therefore is used as the target helicopter in the following research on the helicopter dynamics and flight control development.

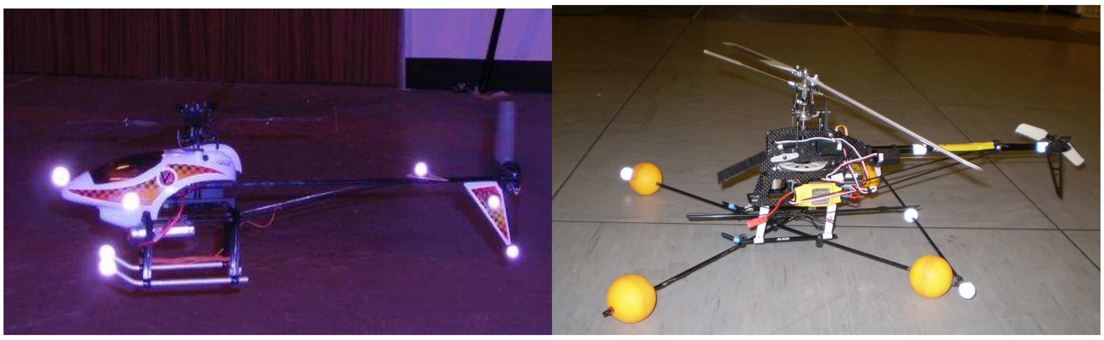

(a) Hummingbird (b) Trex-250

Figure 3.2: Aerial vehicles used in the platform

The ground vehicles adopted in the platform are Tamiya TT01 cars, which are R/C electric model cars. These aerial and ground vehicles enable the user to construct various scenarios such as formation, surveillance, tracking, and so on. It shall be highlighted that the dynamics and mechanisms of these helicopters and ground vehicles are very much the same as the normal ones except the scale or the change of certain coefficients.

3.2.3 Vicon motion system

The most important component in the indoor flight test environment is the navigation system used to obtain the flight (or driving) information for the vehicles. The Vicon motion capture system provides a powerful tracking facility suitable for the indoor environment, which uses dedicated cameras to sense the lightweight reflective balls or stripes in the operating area [118]. Therefore, by attaching some reflective markers to a vehicle, the vehicle can be detected by Vicon cameras. The marker position information is then transmitted via Ethernet using TCP/IP protocol to a computer where the Vicon Nexus software calculates the position and orientation of the vehicle.

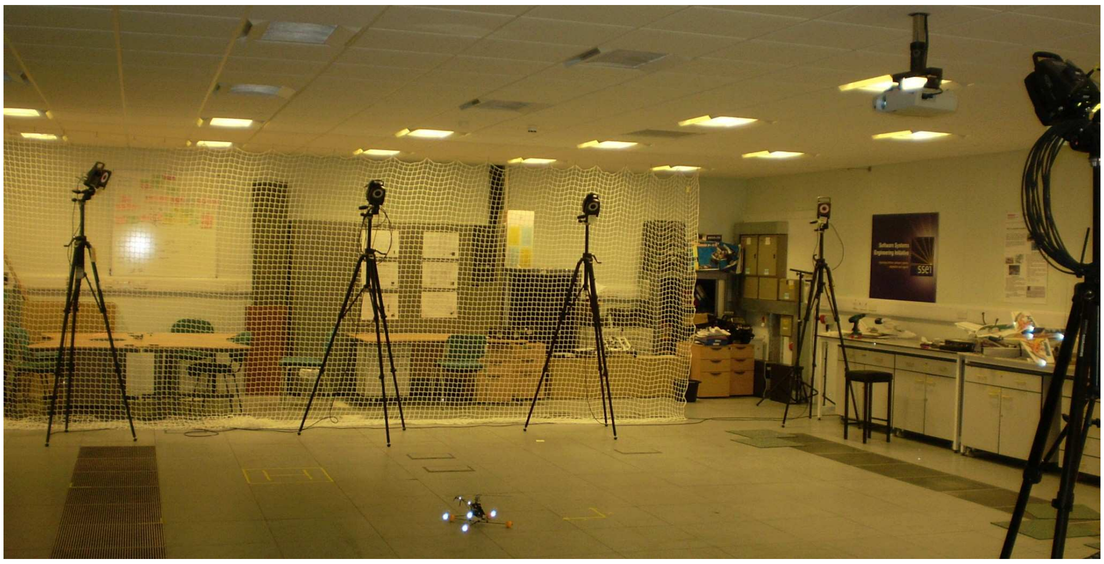


Figure 3.3: Flight test environment

Currently, the Vicon system consists of 5 MX cameras and 3 T10 cameras. MX cameras equipped with fish-eye lenses are used to cover a wider but relatively shallow area, whereas T10 cameras have a higher resolution and a longer detection range. The combination of those two kinds of camera gives a 5 m by 4.5 m by 2 m testing volume (see Fig.3.3). The static accuracy was assessed by measuring the position of a helicopter sitting on the floor. The result in Fig.3.4 shows that the drift in two hours is less than 0.25 mm. Due to the principle of the motion capture system; this gives a fair indication of the position and attitude accuracy of the motion-capture system during flight operations. The Vicon system can capture an object motion with a refresh rate up to 200 Hz and can still track the vehicle even if one or two markers on a vehicle are missing. Therefore, the Vicon system can be regarded as a high bandwidth and robust navigation system in this platform. It can be considered as an indoor replacement for the global positioning systems (GPS) but providing not only the position but also attitude information.

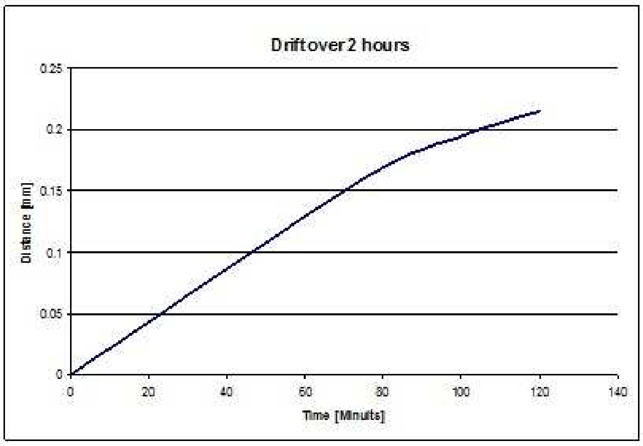


Figure 3.4: Two hours drift of Vicon system

3.2.4 Ground station

The ground station consists of several personal computers (PCs) with Intel Core2 6600 CPU at 2.4GHz and some accessories. The ground station acts as the brain of the platform, as vehicle information is processed here and control commands are transmitted from here after the calculations are performed.

To control vehicles in the test area, the computers running Vicon Nexus and Matlab applications provide the position and attitude information of vehicles and calculate corresponding control commands based on autonomous algorithms, respectively. In terms of high-level tasks, the ground station also manages tasks such as mission planning and trajectory design. In this case, several computers may be involved as will be discussed in later chapters.

In order to send control signals to the vehicles, the ground station is equipped with JR9X2 computer transmitters. The bridge between computers and transmitters is a PCTx adapter, which connects the computer to the R/C transmitter through the USB port and the trainer port interface as shown in Fig.3.5. In the actual tests, a PCTx adapter converts digital commands from the computer into PPM (Pulse Position Modulation) signals for the transmitter, and the latter transmits the corresponding commands to test vehicles.

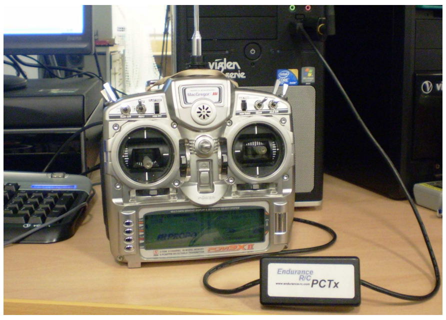


Figure 3.5: Transmitter and PCTx adapter

3.3 System integration in Matlab/Simulink

This section describes the details of integrating both hardware and software to construct a rapid prototyping environment based on Matlab/Simulink. Although the commercial-off-the-shelf components provide the necessary functionalities for the platform, a considerable effort on system integration is required to link all these components together within a single environment, because they have software drivers/packages developed by different venders. In addition, the implementation environment should be governed by a real-time operation system or similar software environment which enables all the components to communicate with the others in real-time and synchronously during tests.

3.3.1 Initial feasibility analysis

Since multiple subsystems are involved in the platform, it is essential to synchronise the execution among these subsystems in order to guarantee data compatibility, particularly when a physical helicopter or ground vehicle is involved. However, there exists a challenge that each hardware product has its own software or driver and is operated independently as originally designed. Vicon motion system processes data captured by Vicon cameras using the Nexus software, which contains a real-time engine providing processed position and attitude information of the objects in the test area. This engine can only be accessed through Vicon real-time application programming interface (API) written in C language. Moreover, the adapter connected to the computer through USB port is driven by a C++ API under Windows operating system (OS). Fortunately, the process of transmitting command signals from a PCTx adapter to a helicopter is straightforward and needs no modification.

To achieve our purpose, Matlab/Simulink is used to build a software environment to manage all the hardware. Matlab/Simulink is a very powerful and convenient tool for control system design and simulation, which also provides a number of communication means and mechanism for integrating with C/C++ languages. On the other hand, UAV autonomous algorithms, at the core of the platform are usually developed and implemented by utilising Matlab/Simulink. This makes it a very promising candidate for the seamless transition from design to numerical simulation and real-time validation in a single software environment.

At the initial test stage, a simple program was developed to implement a basic closed-loop control. First, a dynamic link library (DLL) is developed in the C language, which contained API functions for the Vicon system. Similarly, another DLL is created to talk to the PCTx adapter. Finally, a Matlab script program (M-code) is coded to assemble these two DLLs with a simple control law to control a ground vehicle running around a circle autonomously. This program is actually a simple closed loop continually calculating and sending control signals, and is executed in non-deterministic sampling time. However, it proved that the structure of platform is feasible. To overcome the problem that the common Matlab/Simulink programs are executed in computer time rather than real-time, two different real-time implementation environments based on Matlab/Simulink were tested by utilising different techniques.

3.3.2 Real-time environment using real-time blockset

An integrated implementation environment requires the capabilities of communication and execution in real-time. Simulink provides a powerful mechanism for extending its capability, namely S-function, which is a computer language description of Simulink block. S-function can be written in C/C++, which can have access to the Vicon API receiving vehicle state data and can also drive the adapter sending command signals. Thus, two dedicated S-function blocks were developed to communicate with the Vicon system as well as the adapter within the Matlab/Simulink environment. Despite the C/C++ programming required in the development progress, the completed blocks can be treated as normal Simulink blocks. Thus, the control of hardware is ultimately realised in the software environment .

After solving communication problems, a real-time block set is added into the Simulink environment to ensure that the implementation is attached to real-time [24]. This block set has the function of holding the execution of Simulink simulations to real time flow. This means that if the execution time of one simulation step is lower than the corresponding real time, this block set stops the simulink simulation and releases the remaining CPU time to all the other Windows processes or just idles. This concept is very simple but effective. A typical task execution time (TET) with a normal set of control algorithms at a sampling interval of 50ms is given in Fig.3.6. The lower line represents the execution time spend on the Simulink program during the simulation, while the upper line represents the remaining time for which Simulink waits for the next step (and leaves the CPU to remaining Windows applications).

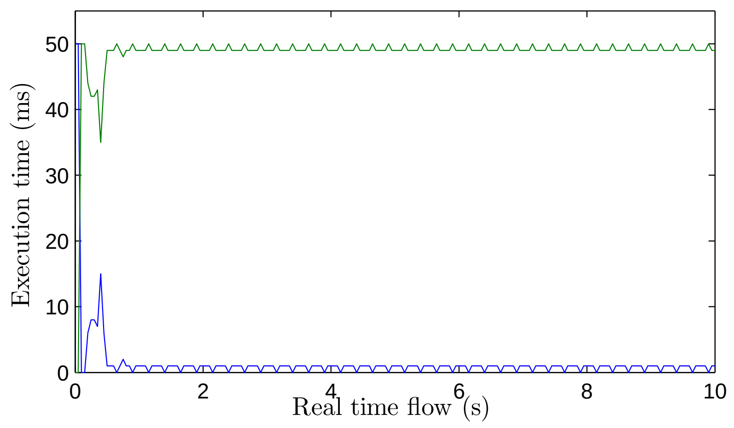


Figure 3.6: Task execution time under real-time block set (lower line: execution time of Simulink program; upper line: remaining time for other applications)

During the real-time tests, Vicon Nexus and Simulink programs are executed on the same PC that connects to Vicon MX cameras through Ethernet and the adapter thought USB port, respectively. In this manner, the time delay of data transferring can be minimised. The latency of the calculated Vicon data due to the network is less than 1 ms, while the latency of sending out control signals is about 5 ms on average due to the property of the USB port. Currently, control algorithms in this environment are running at the sampling interval of 20 ms, which is adequate for most flight control applications.

The software environment based on the Simulink can manage the data exchange and hardware communication autonomously in the background. Therefore, it is possible to use this platform as if it is a normal Simulink environment. A detailed structure of this environment is shown in Fig.3.7. The advantages of this environment are that it provides many powerful toolboxes and other useful built-in resources in Simulink, and it is also a very convenient way to observe and record signals during the flight tests. The latter property is very important for prototyping advanced and complicated control algorithms, since intermediate states of the controller can be easily monitored compared to similar realisation using embedded systems. To this end, it accelerates the development significantly, because there is no obstacle between algorithm development and rapid prototyping. One can implement developed algorithms into this platform directly for experiments as long as numerical simulations are completed in the Simulink environment.

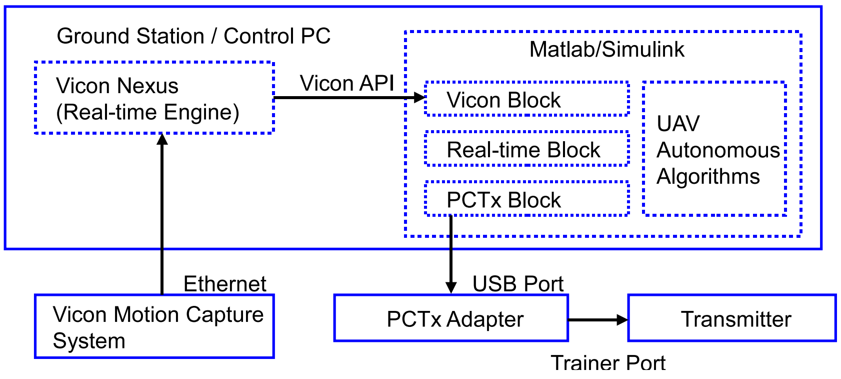


Figure 3.7: Structure of real-time block set environment

Nevertheless, there are some drawbacks which might influence flight tests for some scenarios in the future. For a more complicated mission, due to the heavy computation burden, it might be difficult to complete the calculation within the sampling interval when the RT Block set is used. Furthermore, from the OS point of view, Matlab/Simulink is running on Windows OS, which is not a true real-time operating system (RTOS). That means that other applications with a higher priority in Windows system may interrupt real-time experiments and possibly cause loss of control of vehicles. To avoid these negative aspects, another real-time environment based on xPC Target was then developed.

3.3.3 xPC Target environment

xPC Target is a special product in Matlab for prototyping, testing, and deploying real-time systems using standard PC hardware. It is an environment that uses a target PC, separate from a host PC, for running real-time applications. Theses applications are created from Simulink programs on the host PC, compiled and downloaded onto the target PC through Ethernet or serial connection. xPC Target can significantly enhance the reliability and have the capability of dealing with more complicated algorithms.

The structure of xPC Target environment is different from the previous one. Since the real-time execution is essentially guaranteed, the synchronising of communication is the remaining issue that is of concerned in this structure. xPC Target executes its applications on a real-time kernel, where Vicon Nexus is not compatible and USB port is not supported. Therefore, the target PC has to communicate with another server PC that can provide the vehicle's states calculated by Vicon Nexus and send command signals calculated by the target PC to transmitters.

There are two basic communication methods built in xPC Target. One is RS232 serial port transport, while another is user datagram protocol (UDP) technology. The latter one is chosen in our case, because of its high bandwidth and the ability of talking to multiple clients in the same network. Although UDP protocol eliminates error check and recovery, it ensures that real-time applications have a maximum chance of succeeding in real-time execution by only using the most recent data. On the other hand, the target PC, host PC and server PC shall be connected through a local area network (LAN) so that the network latency can be minimised.

Next, a C/C++ server program is developed running on the server PC. This program takes charge of the data transmission between target PC and server PC, where Vicon data are converted into UDP packets to send out, meanwhile received UDP packets from target PC are decoded into control signals to drive the transmitter. The synchronisation of data transfer is implicitly dealt with in a manner that the main application on the target PC calls each communication port at a fixed interval, whereas the server program receive and send packets passively. The entire structure of this environment is shown in Fig.3.8.

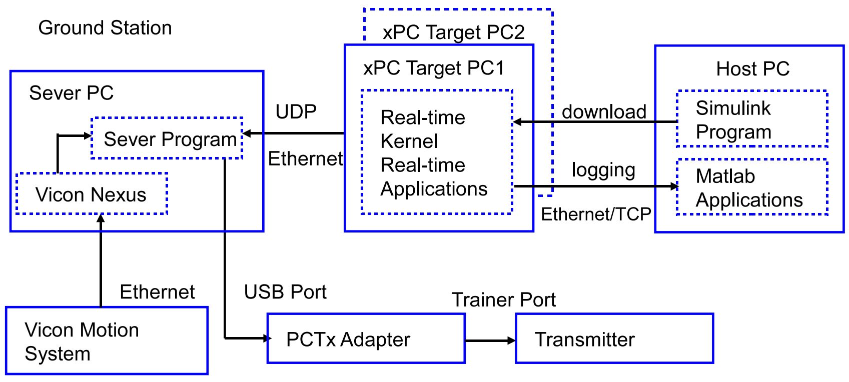


Figure 3.8: Structure of xPC Target environment

When doing the real-time implementation using this environment, the user needs to configure the network among these PCs first, and then compile Simulink programs into executable real-time applications and download to the target PC. This work is implemented in the Matlab environment. During flight tests, vehicle states can be visualised by the Vicon Nexus on the server PC or can be displayed in numerical or curve forms on the monitor of the target PC. After flight test all the data can be logged back to the host PC for recording or further analysing.

The merits of xPC target environment are that it provides a considerably more reliable environment for implementing various algorithms and has the potential of expansion to meet the real-time requirement for sophisticated algorithms. Although, the operation of this environment is not as convenient as the real-time block set environment, it is very suitable to demonstrate relatively mature algorithms into a complicated scenario such as multiple vehicle coordination and long duration mission management.

3.4 Potential usage of the test platform

The rapid prototyping platform is very versatile and flexible and can meet flight test requirements for various purposes. It can provide supports for many research activities, not just those reported in this thesis, as outlined in this section.

- System identification and modelling
  Modelling is always the first step in delivering control solutions for compli-

cated dynamic systems. Various tests of helicopters and ground vehicles can be performed under human remote control. All the control commands and the response of the vehicles captured by Vicon system can be recorded synchronously. All these data can be used to study and analyse the behaviour of the target vehicle. Hence, this platform provides an ideal environment for system identification and modelling, which is very much similar to wind tunnels for fixed-wing aircraft.

- Flight control
  Helicopters have very complicated dynamics, with strong non-linearities and coupling between different channels. To some extent, control of small scale helicopters is even more challenging than that of conventional helicopters since they are more susceptible to ground effects and the change of structure and propulsion. Various control algorithms are developed in this thesis using advanced control methodologies such as non-linear control and robust control and then evaluated in this flight test environment. The control calculations can be performed in Matlab/Simulink on standard PCs, which not only eases the implementation but also provides enough computing power for complicated algorithms such as model predictive control where one-line optimisation is required.

- Avionic systems
  Navigation systems are a very important part of the onboard avionic systems, and provide essential information for aircraft control and positioning. The Vicon optical tracking system can be used as a reference system to assess the performance of various new navigation systems. For example, one research topic is to investigate the integration of low cost inertial measurement sensors with computer vision. Together with inertial sensors, a small camera can be installed on the helicopter to perform various flight tests to investigate the performance of these new concepts and algorithms. It can provide support for similar work on vehicle navigation systems.

- UAV path and mission planning
  One of the main motivations for developing this flight test environment is to support research in UAV autonomous algorithms such as path planning and mission planning. In these high-level algorithms, the UAVs are treated as a mass point, so the algorithms are largely independent of the platforms.

To this end, this test facility provides an environment to verify and de-risk research work on UAV autonomy for various aerial vehicles including fixed wing aircraft.

- Teaching
  This flight test environment also provides support for teaching activities. Two modules, flight control systems and avionic systems directly benefit from it by setting the coursework and having experimental tests in this environment. It has also been used to support various final year projects and other group design projects in Loughborough University.

3.5 Real-time control example

In this section a real-time control example is presented to show the basic function of the proposed rapid prototyping platform. The task is to control two ground vehicles to follow circles with different radii at a constant speed. In this case the control algorithm is fairly simple. It measures the distances of the vehicles with respect to the centre of circles and compares them to the set-point radius. The errors are fed into proportional-derivative-integral (PID) controllers to generate steering signals. The controllers are realised in Simulink using standard blocks in the simulink library. One controller is shown in Fig.3.9.

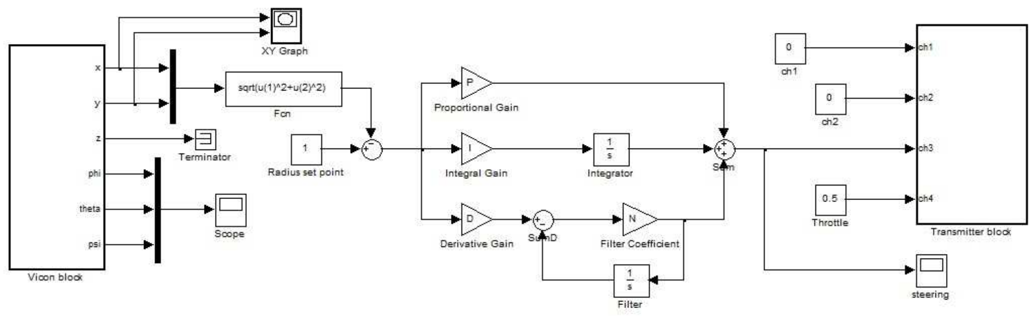


Figure 3.9: Simulink programme for real-time control

The real-time control can be achieved by using this diagram, where the Vicon block provides vehicle information and the Transmitter block sends out commands. More advanced algorithms can be added into the diagram to enhance the control performance or to execute more complicated tasks. The test result of this control example is shown in Fig.3.10. It can be seen that although the start points were not on the expected circles, the controller can make the cars following the trajectories.

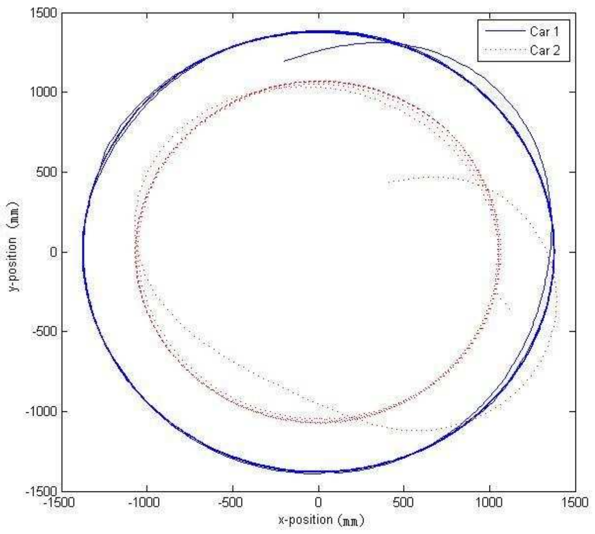
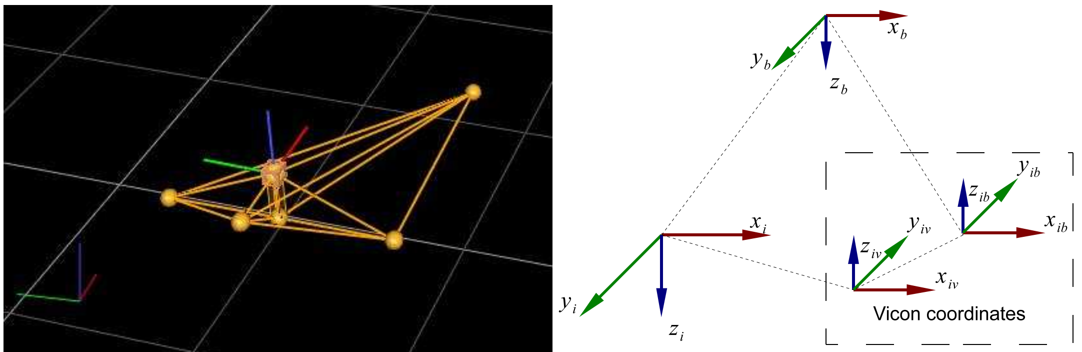


Figure 3.10: Test result of two vehicles control

It shall mention that the global coordinate system in Vicon Nexus is a Cartesian coordinate system (x_{iv}, y_{iv}, z_{iv}) with z_{iv} axis upwards (see Fig.3.11a). The rotation of an object is represented by a set of Euler angles in the sequence of X-Y-Z. It may be suitable for describing a ground vehicle, but is inconsistent with conventional coordinates system in aerospace engineering. The latter one adopts global coordinates (x_i, y_i, z_i) with z_i axis downwards as shown in Fig.3.11b and represents rotations using Euler angles yaw-pitch-roll corresponding to the sequence Z-Y-X.

To avoid misleading in implementing an algorithm developed from standard flight control techniques and prevent crashes due to mis-match of coordinates, it is important to convert default Vicon coordinates into conventional aerospace coordinates. The position information can be corrected by rotating Vicon coordinates about its x_{iv} axis for 180°, which is equivalent to change directions of z and y axes. The rotation information can be extracted from a corrected rotation matrix that is derived by first rotating (x_i, y_i, z_i) about x_i axis into Vicon global coordinates (x_{iv}, y_{iv}, z_{iv}), then following Vicon's rotation matrix to Vicon



(a) Vicon coordinates (b) Conventional aerospace coordinates

Figure 3.11: Coordinates system

body coordinates (x_{bv}, y_{bv}, z_{bv}), and finally rotating about x_{bv} axis to get body coordinates (x_b, y_b, z_b). The corrected rotation matrix is actually equivalent to the Vicon rotation matrix derived using negative second and third Euler angles as their axes are inverse. Therefore, this process can be implemented in Simulink as shown in Fig.3.12 and is embedded in the Vicon block.

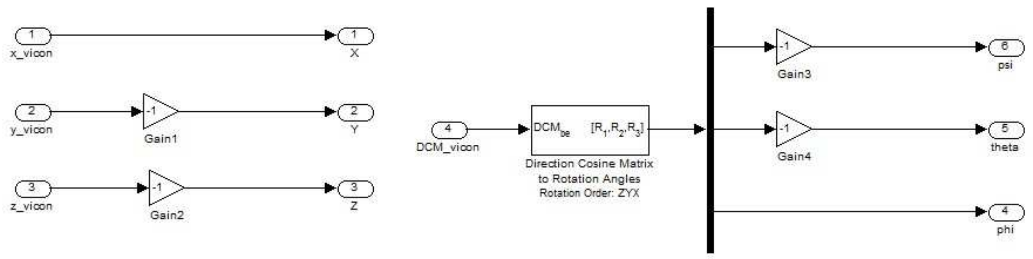


Figure 3.12: Coordinates conversion

3.6 Conclusion

In this chapter, the hardware and software architecture of a rapid prototyping flight test environment for autonomous UAVs is discussed in detail. This platform provides a convenient and effective facility for evaluating UAV related algorithms and supporting general flight control research using real vehicles. This is a versatile testbed and supports various scenarios, including single aerial vehicle, multi-vehicles, mixture of aerial and ground vehicles. It can also provide support for a variety of research, teaching and demonstration activities. The adoption of the widely used Matlab/Simulink environment enables researchers to test new research outputs seamlessly on real vehicles. This multifunctional, low cost and flexible indoor flight testbed also enables one researcher to manage and coordinate UAV missions with multi-vehicles, which significantly reduces manpower and logistic supports required for this kind of study. Another important feature of this platform is that model helicopters are adopted as test vehicles. On one hand, the nature of helicopter dynamics, such as hovering, vertical take-off and landing and low speed cruise, allows realistic and complicated missions to be simulated in a confined space. On the another hand, it provide an opportunity to carry out comprehensive study on the helicopter dynamics and flight control design as shown in the rest of this thesis.

Chapter 4

Modelling and system identification of a miniature helicopter

4.1 Introduction

This chapter describes a systematic methodology of obtaining a mathematical model of a miniature helicopter towards the control engineering practice. Helicopters are versatile flying machines that are ideally suited to tasks such as surveillance and reconnaissance in confined and dynamic surroundings due to their integrated abilities of hovering, low altitudes cruise, and agile manoeuvre.

To this end, the complicated dynamics of helicopters need to be well understood; otherwise the operational capabilities of autonomous systems based on small-scale helicopters will remain limited. On the other hand, in order to apply modern control strategies and explore a broad range of flight conditions, a precise mathematical model is a necessary. The model should not only have an accurate reflection of the main dynamic characteristic of the helicopter, but also open to a fair degree of simplification for control design and real-time prediction. Due to the complexity of the dynamics of helicopters, developing such a model is a challenge.

The previous works listed in the literature review (Chapter 2) have set a good starting point for the continuous development of a model for miniature helicopters, particularly for those used by the Autonomous System Lab. There are still open problems in modelling and system identification, especially in some practical aspects. Firstly from a control point of view, a helicopter model that is simple but can accurately predict the dynamic response is desired. Thus, a complicated model built from first-principles is excluded, and the model should be validated using real flight data to guarantee its fidelity. Secondly, in system identification experiments, input signals to the helicopter should have sufficiently large magnitude so as to guarantee that the helicopter is fully actuated and the corresponding flight modes are excited. Therefore, the model needs to account for the nonlinearity caused by manoeuvres. Thirdly, since the model structure needs nonlinear terms, the selected identification algorithm should be able to process nonlinear models, which implies the frequency domain methods are not appropriate.

To solve these difficulties, a helicopter model is derived with nonlinear terms to account for helicopter dynamics in a broad range of flight conditions, but retains the first-order approximation of external forces and moments, so as to reduce complexity to facilitate control design. The unknown parameters in the model structure are identified from flight data by using the PEM algorithm in the Matlab System Identification Toolbox [69]. This algorithm supports nonlinear grey-box identification (with unknown structure and unknown parameters), which makes it well suited for our purpose. Furthermore, to improve the accuracy of the parameter estimation, the entire nonlinear model is broken down to several small identification processes. The corresponding flight tests are also carefully designed and performed to support the identification process. All these efforts guarantee the fidelity of the model and the accuracy of the identification. The resulting parametrised nonlinear model shows a consistent match with flight data, even when both lateral and longitudinal channels are excited. Of particular interest is that the prediction of translational speeds is further improved compared with previous works. As there is no dedicated software involved in the identification process, the proposed method can easily be picked up by other research groups to accelerate the development of rotorcraft based autonomous systems.

4.2 Model development

4.2.1 Overview

The miniature helicopter adopted in the research is a Trex-250 electronic helicopter, as shown in Fig 4.1. It is a 200-sized helicopter with the main rotor diameter of 460mm and the trail rotor diameter of 108mm. The miniaturised size and 3D aerobatic ability make it well-suited for flight tests in the environment described in Chapter 3. Trex-250 has a collective pitch rotor and well designed Bell-Hiller stabilizer mechanism that makes it representative of most widely used small-scale helicopters found in literature. In addition, the Trex family features a series of helicopters from the Trex-250 micro-size to the Trex-700 small-size all with a similar structure and handing qualities meaning that the research outcomes can be easily transferred into much wider applications.

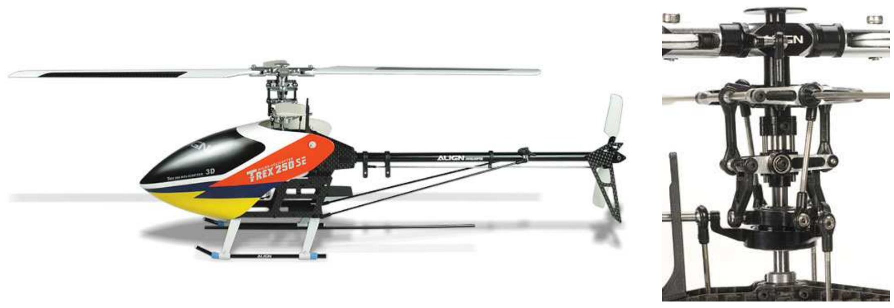


Figure 4.1: Trex-250 Helicopter

A helicopter's versatile flight ability is supported by its effective yet complicated rotor system. The main rotor blades can change their pitch angle simultaneously and cyclically while rotating around the main shaft. This movement is achieved via a swashplate mechanism. When the swashplate moves vertically along the rotor shaft under the control of collective input δ_{col}, the pitch angle of all blades are altered at the same time, so that the thrust is changed. When the swashplate is tilted, the blade pitching follows the swashplate's perimeter cyclically to produce different lifts at different angles. Then the whole rotor disc demonstrates a flapping motion to generate torques on the helicopter fuselage. The cyclic control inputs can be divided into lateral (roll) and longitudinal (pitch) components δ_{lat} and δ_{lon}, respectively. On the other hand, the tail rotor only has a collective input to alter its thrust used to counteract the main rotor moment and to change the heading. As in a real size helicopter where this control is accessible via pedals, it is denoted as δ_{ped}. In addition, there is a Bell-Hiller stabiliser bar in the rotor system used to improve the ability to fly a model helicopter, which actually provides a mechanical rate feedback (damping).

To study the behaviours of a helicopter, a body-fixed reference coordinate is established and attached to the helicopter as shown in Fig.4.2. Its origin is at the helicopter's centre of gravity (c.g.), the x axis is along the forward direction of the helicopter, the z axis is downward and the y axis is determined based on the "right-hand" rule. The other principal variables are also shown on the x, y and z body axes. They include: the projections of inertial velocity in the 3 body axes u, v, w; the Euler angles ϕ, θ, ψ representing the helicopter orientation; and the body angular rates p, q, r. The main rotor is represented by a disc which can tilt about the rotor hub in both longitudinal and lateral directions. This motion is described through the angles β_{1c} and β_{1s} measured with respect to a plane perpendicular to the rotor shaft.

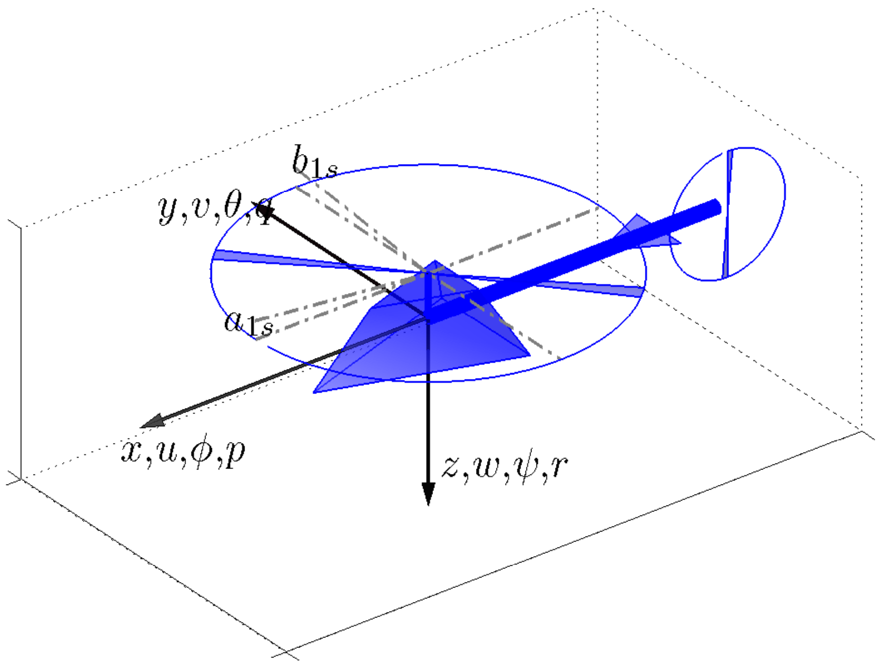


Figure 4.2: Body coordinates of the helicopter

The transformation between the inertia coordinates and the body-fixed coordinates can be parametrised in terms of Euler angles. The Euler angles refer to a specific sequence of rotations about the vehicle body axes. A widely used definition in flight dynamics is a sequence of yaw-pitch-roll, which are angle ϕ about the z axis, angle θ about the current y axis, and ϕ about the current x axis [21], respectively. In this way, the kinematic relationship is

$$[\ \dot{x} \quad \dot{y} \quad \dot{z}\ ]^T = \mathbf{R}_b^i[\ u \quad v \quad w\ ]^T \tag{4.1}$$

where the transformation matrix is (with s denote for $sin(\cdot)$ and c for $\cos(\cdot)$)

$$\mathbf{R}_b^i = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{4.2}$$

In addition, the relationship between the angular rate (p, q, r) and the Euler angle rate $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ can be expressed in

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.3}$$

4.2.2 Rigid-body model

Rigid-body equations of motion are extensively used in aerospace engineering to describe the dynamics of a vehicle in air or in outer space which is free to translate and rotate in all six-degrees-of-freedom (6DOF) [21]. The rigid-body equations of motion can be developed from Newton-Euler equations of motion expressed in the inertial reference coordinates. For a constant vehicle mass m and moment of inertia (inertial tensor) $\mathbf{I}$, they are:

$$m\frac{d\mathbf{v}_I}{dt} = \mathbf{F} \tag{4.4}$$

$$\mathbf{I}\frac{d\omega_I}{dt} = \mathbf{M} \tag{4.5}$$

where $\mathbf{F} = [\ X \quad Y \quad Z\ ]^T$ is the vector of external forces acting on the vehicle c.g., and $\mathbf{M} = [\ L \quad M \quad N\ ]^T$ is the vector of external moments. For a helicopter, the external forces and moments are produced by the main rotor and the tail rotor, the gravitational force, and the aerodynamic forces produced by the fuselage and the tail surfaces.

Using the kinematic principle of moving reference coordinates, the equations of motion can be expressed with respect to the helicopter body-fixed reference coordinate:

$$m\dot{\mathbf{v}} + m(\omega \times \mathbf{v}) = \mathbf{F} \tag{4.6}$$

$$\mathbf{I}\dot{\omega} + (\omega \times \mathbf{I}\omega) = \mathbf{M} \tag{4.7}$$

where $\mathbf{v} = [\ u \quad v \quad w\ ]^T$ and $\omega = [\ p \quad q \quad r\ ]^T$ are the fuselage velocities and angular rates in the body coordinates, respectively, and $\times$ denotes the cross-product. For a 6 DOF rigid-body system, Eq.(4.6) produces the three differential equations describing the helicopter's translational dynamics about the its three body axes:

$$\begin{aligned} \dot{u} &= (-wq + vr) + X/m \\ \dot{v} &= (-ur + wp) + Y/m \\ \dot{w} &= (-vp + uq) + Z/m \end{aligned} \tag{4.8}$$

Similarly, Eq.(4.7) produces the three ordinary differential equations describing the vehicle's rotational dynamics with the assumption that the cross-products of the inertia are small:

$$\begin{aligned} \dot{p} &= -qr(I_{yy} - I_{zz})/I_{xx} + L/I_{xx} \\ \dot{q} &= -pr(I_{zz} - I_{xx})/I_{yy} + M/I_{yy} \\ \dot{r} &= -pq(I_{xx} - I_{yy})/I_{zz} + N/I_{zz} \end{aligned} \tag{4.9}$$

After the rigid-body dynamics of the helicopter are expressed in Eq.(4.8) and (4.9) as a function of the external forces $\mathbf{F}$ and moments $\mathbf{M}$, the next step is to find the components that construct the external forces and moments, and express them as functions of control inputs and/or vehicle states. This is the most difficult step in the modelling of a helicopter due to the complicated nature of rotor aerodynamics and interactions. However, the principle components of $\mathbf{F}$

and $\mathbf{M}$ are generated by the main and tail rotors, the gravitational forces, and the aerodynamic forces caused by a fuselage (including tail boom, vertical and horizontal fins), such that:

$$\begin{aligned}
X &= X_M + X_F + G_x \\
Y &= Y_M + Y_F + Y_T + G_y \\
Z &= Z_M + Z_Z + G_z \\
L &= L_M + Y_M h_M + Z_M y_M + Y_T h_T \\
M &= M_M + M_T - X_M h_M + Z_M l_M \\
N &= N_M + Y_M l_M - Y_T l_T
\end{aligned} \tag{4.10}$$

where X_*, Y_*, Z_* and L_*, M_*, N_* denote, respectively, the forces and moments from different sources, with the subscripts $_M$, $_T$, $_F$ indicating the main rotor, tail rotor, and fuselage, respectively; (l_M, y_M, h_M) and (l_T, y_T, h_T) denote the coordinates of the main rotor and tail rotor shafts relative to the c.g. in the body coordinates; and (G_x, G_y, G_z) denote the projections of gravitational force in the body coordinate in the form of:

$$\begin{bmatrix} G_x & G_y & G_z \end{bmatrix}^T = \mathbf{R}_i^b \cdot \begin{bmatrix} 0 & 0 & g \end{bmatrix}^T \tag{4.11}$$

where g is acceleration of gravity.

Note that in the construction of Eq.(4.10), it is assumed that the tail rotor only generates the force and torque along the y direction and that torques arising from the aerodynamic forces of the fuselage are negligible. This is reasonable because among all the components in Eq.(4.10), the forces and moments produced by the main rotor and the thrust of the tail rotor have the most significant influences on a small-size helicopter. Moreover, they can be altered by the control inputs so that a helicopter can be controlled. To account for their mechanism and complete the helicopter model accordingly, the rigid-body model needs to be extended to include the main rotor behaviour.

4.2.3 Extension of rigid-body model

The objective of extending the rigid-body model discussed in the previous subsection is to take into account the higher-order effects that exist in helicopter dynamics, and to investigate their interactions with the rigid-body dynamics.

Except for the rigid-body motion, the most obvious and important movements in helicopter dynamics are the tilting or flapping movements of the main rotor. A combination of these two parts gives a hybrid model that is sufficiently simple, but captures the main helicopter characteristics.

4.2.3.1 Simplified rotor dynamics

The rotor dynamics are naturally complicated due to the high degrees of freedom of the motion of the rotor blades plus elusive aerodynamic characteristics. Consequently, detailed rotor equations of motion can be extremely complex [89], but they are not suitable for system identification and control design. However, earlier work on modelling of small-scale rotorcraft showed that the main rotor system with a stabiliser bar (flybar) can be lumped and represented by tip-path-plane (TPP) flapping dynamics with only two states. In the following, this chapter will describe the development of a highly simplified TPP rotor model for system identification and later for the control design purpose.

The flapping motion quantified by a flapping angle β is a 2π periodic function as it associated with blades rotating about the rotor shaft (speed Ω, position Ψ). Thus the general solution to the flapping equation can be approximated by first-order Fourier series

$$\beta(\Psi) \approx \beta_0(t) - \beta_{1c}(t)\cos\Psi - \beta_{1s}(t)\sin\Psi \tag{4.12}$$

This type of motion results in a cone-shaped rotor shown in Fig.4.3. The top of the cone is the so-called tip-path-plane. The constant term β_0 describes the cone angle and the coefficients of the first harmonic β_{1c} and β_{1s} describe the tilting of the rotor TPP in the longitudinal and lateral directions, respectively. By using the TPP presentation, the rotor flapping dynamics can be further simplified, because a tilting movement of plane replaces the detailed rotation movements of blades. The complete derivation and simplification can be found in [78]. The solution is directly adopted in this thesis to extend the rigid-body model.

To investigate the dynamic modes of the rotor TPP, simplified notations, a instead of β_{1c}, b instead of β_{1s} and a_0 instead of β_0, are used to formulate a tip-path-plane state vector $\mathbf{a} = [\ a_0 \quad a \quad b\ ]^T$. Note that for the Trex-250 helicopter, the coining angle β_0 is exactly equal zero due to its hingeless rotor head. For system identification, the rotor dynamics can be further simplified as not all effects

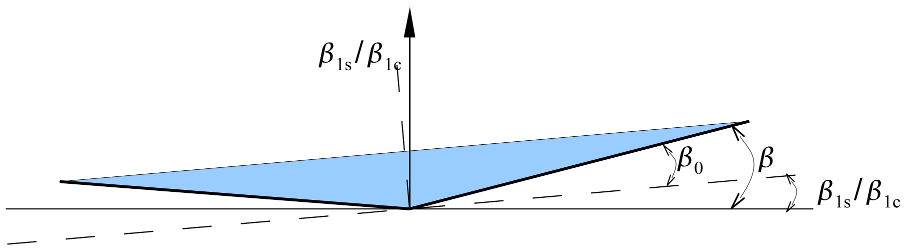


Figure 4.3: Tip-path-plane rotor representation

can be identified from flight data. By focusing on the low frequency flapping dynamics, the longitudinal and lateral flapping dynamics are approximated by first-order equations [78]:

$$\frac{16}{\gamma\Omega}\dot{a} = -a - \frac{16}{\gamma\Omega}q + \frac{p}{\Omega} + \frac{8}{\gamma\Omega^2}\frac{k_\beta}{I_\beta}b - B_1 \tag{4.13}$$

$$\frac{16}{\gamma\Omega}\dot{b} = -b - \frac{16}{\gamma\Omega}p + \frac{q}{\Omega} + \frac{8}{\gamma\Omega^2}\frac{k_\beta}{I_\beta}b + A_1 \tag{4.14}$$

where Ω is the rotor speed, k_β is the spring constant of the flapping restraint; I_β is the blade moment of inertia about the flapping hinge; A_1 and B_1 are, respectively, the lateral and longitudinal cyclic blade pitch angles, γ is the blade Lock number which represents the ratio between the aerodynamic and inertial forces acting on the blade given by

$$\gamma = \frac{\rho c C_{la} R^4}{I_\beta} \tag{4.15}$$

where R is the rotor radius, ρ is the air density, c is the blade chord length, and C_{la} is the lift curve slope.

These equations capture the key TTP responses due to control inputs and helicopter motion. Moreover, these equations reflect rotor dynamics in a low frequency range within which they will be coupled into fuselage dynamics. In Eq.(4.13) and (4.14), one important coefficient is $\frac{16}{\gamma\Omega}$, which is defined as the rotor time constant τ. Note that it depends on the rotor Lock number γ and rotor angular speed Ω. By using the notation $\tau = \frac{16}{\gamma\Omega}$, we have two terms τq and $-\tau p$, which are important pitch and roll damping terms in the lateral and longitudinal directions, respectively. They can also be seen as a rate feedback mechanism which is actually provided by the flybar. In contrast, $-\frac{q}{\Omega}$ or $+\frac{p}{\Omega}$ is the lateral (or longitudinal) flapping produced by a body roll rate p (or pitch rate q). This is one of the main rotor cross-coupling effects. $\frac{8}{\gamma\Omega^2}\frac{k_\beta}{I_\beta}$ is another cross-coupling effect arising in the presence of a flapping restraint. It can be denoted by the derivatives A_b and B_a for the longitudinal and lateral flapping, respectively. Finally, the blade pitch angles A_1 and B_1 at lateral and longitudinal directions are decided by the cyclic movement of the swashplate, which can be modelled to be proportional to control inputs δ_{lat} and δ_{lon}:

$$A_1 = A_{lat}\delta_{lat} + A_{lon}\delta_{lon} \tag{4.16}$$

$$B_1 = -B_{lat}\delta_{lat} - B_{lon}\delta_{lon} \tag{4.17}$$

where A_{lat}, A_{lon}, B_{lat} and B_{lon} are control derivatives, in which A_{lat} and B_{lon} are used to account for unmodelled cross-coupling effects.

The simplified rotor dynamic equations that are suitable for system identification can be summarised as:

$$\tau\dot{a} = -a - \tau q + A_b b + B_{lat}\delta_{lat} + B_{lon}\delta_{lon} \tag{4.18}$$

$$\tau\dot{b} = -b - \tau p + B_a b + A_{lat}\delta_{lat} + A_{lon}\delta_{lon} \tag{4.19}$$

where $-\frac{q}{\Omega}$ or $+\frac{p}{\Omega}$ are dropped, as their coefficients have one order of magnitude less than $\frac{16}{\gamma\Omega}$ based on the estimation using its physical definition and final identification results.

After the flapping dynamics of the rotor are studied, it is now possible to investigate the basic function of the main rotor and its by-product, which are main rotor thrust T and torque Q_M, respectively. The first one can be modelled as

$$T = K_T P_M \Omega^2 \tag{4.20}$$

where K_T denotes the aerodynamic constant of the rotor's blade, and P_M is the collective pitch angle of the blade, which is controlled by input δ_{col}. The main rotor's torque has a more complicated form, which according to [34] can be modelled as:

$$\begin{aligned} I_{rot}\dot{\Omega} &= Q_M - Q_M^R \\ Q_M^R &= c\Omega^2 + dP_M^2\Omega^2 \\ Q_M &= P_e/\Omega \\ P_e &= \bar{P}_e\delta_{thr} \end{aligned} \tag{4.21}$$

where I_{rot} is the moment of inertia of the rotor blades about the main shaft, Q_M^R is a reaction torque due to the aerodynamic resistance of blade (with physical parameters c and d depending on blade characteristic), and P_e denotes the engine power assumed to be proportional to throttle input δ_{thr} with coefficient $\bar{P}_e$. Although the main rotor torque varies in such a complicated way, modern unmanned helicopters are equipped with onbard heading gyros, which help the tail rotor to generate the correct thrust to cancel the main rotor torque.

It can be noticed from the above discussion that many of the coefficients in rotor dynamics depend on the rotor speed. Fortunately, in most modern unmanned rotorcraft, the rotor speed Ω is retained constant by an onboard electronic governor. As a result the corresponding coefficients can be considered as constant at the current modelling level.

4.2.3.2 Rotor and fuselage interaction

The interaction between the main rotor and the fuselage is reflected by the relation between the rotor tip-path-plane angle, the thrust vector, and the forces and moments produced by the rotor, which is shown in Fig.4.4.

The forces produced by the main rotor are considered to be the result of the thrust vector tilting. With the assumption that the direction of the thrust vector is perpendicular to the rotor TPP (common assumption for hover and low-speed flight), X_M, Y_M and Z_M in Eq.(4.10) are the projections of the thrust vector on the x, y and z body axes, such that

$$\begin{aligned} X_M &= -T\sin a\cos b \approx -Ta \\ Y_M &= T\sin b\cos a \approx Tb \\ Z_M &= -T\cos a\cos b \approx -T \end{aligned} \tag{4.22}$$

Note that here we use the fact that the magnitude of flapping angle is small.

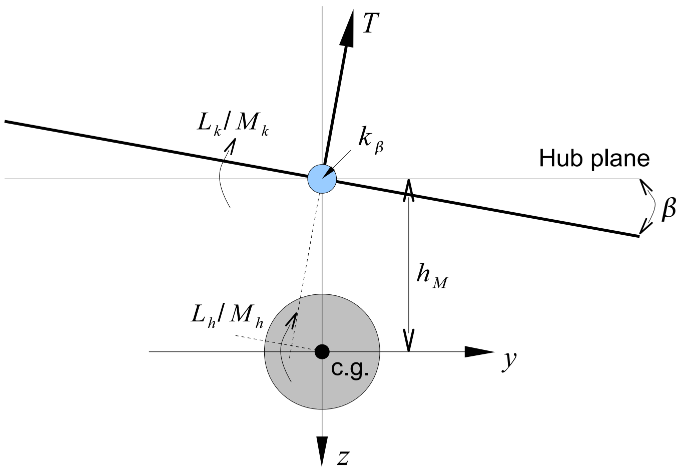


Figure 4.4: Rotor force and moments acting on the helicopter fuselage

The moment produced by rotor flapping acting of the fuselage consists of two parts. The first contribution results from the restraint in the blade attachment to the rotor hub. This restraint can be approximated by a linear torsional spring model. Using the TPP lateral and longitudinal flapping b and a, the corresponding torsional moments are

$$\begin{aligned} L_k &= k_\beta b \\ M_k &= k_\beta a \end{aligned} \tag{4.23}$$

The second moment contribution is from the tilting of the thrust vector. As the thrust vector will tilt proportionally to the rotor flapping angles, its projection in the hub plane produces a moment with a moment arm of length equal to the distance h_M. Taking into account Eq.(4.22), the resulting roll and pitch moments are

$$\begin{aligned} L_T &= h_M Y_M \approx h_M T b \\ M_T &= h_M(-X_M) \approx h_M T a \end{aligned} \tag{4.24}$$

The total moments due to rotor flapping action on the fuselage are obtained by summing the hub restraint and thrust tilting contributions.

4.2.3.3 Complete helicopter model

After the rotor dynamics and their coupling with rigid-body dynamics have been investigated, the complete helicopter model used for system identification can be formulated by combining every force and moment component in Eq.(4.10). In the first-principle modelling of helicopters, every noticeable contribution to external forces and moments needs to be taken into account. However, in the development of a model for system identification, some terms in Eq.(4.10) need to be selectively dropped or further simplified. Firstly, this is due to their limited contributions that cannot be identified from experiment data. Effects from trivial contributions can be easily corrupted by disturbances in flight test or submerged in sensor noise. Secondly, the extra terms need more free parameters to describe, which will increase the degrees of freedom of parameters, thereby may cause difficulties in identification.

It is a recursive process to refine the model structure, i.e. which term in the equations which should remain or be dropped, by assessing the final identification results. According to the literature and initial trails in our system identification exercises, it was found that it is adequate to approximate some forces (or moments) using their first-order Taylor series. To this end, we express forces and moments using stability derivatives and control derivatives, which are first partial derivatives in Taylor series with respect to vehicle states and control inputs, respectively. Note that in the translational and rotational equations, the forces are normalised by the helicopter mass, and the moments are normalised by the related moment of inertia.

In hover and slow flight conditions, the main rotor thrust is approximately equal to the vehicle gravity, so Eq.(4.20) is approximated as

$$\frac{T}{m} = -Z_{col}\delta_{col} + g \approx g \tag{4.25}$$

where Z_{col} is the collective pitch derivatives. By following this approximation and combining Eq.(4.11) and (4.22), the forces in Eq.(4.10) can be reformulated as:

$$X/m = X_u u - g\sin\theta - ga \tag{4.26}$$

$$Y/m = Y_v v + g\cos\theta\sin\phi + gb \tag{4.27}$$

$$Z/m = Z_w w + g\cos\theta\cos\phi + Z_{col}\delta_{col} - g \tag{4.28}$$

where, X_u, Y_v and Z_w are velocity damping derivatives, which account for the relative fuselage drag forces. The approximation $\frac{T}{m} = g$ is only used in the horizontal forces of the main rotor as their effects are reduced by small flapping angles. Note that the lateral force Y_T produced by the tail rotor is dropped, as it is compensated by a partition of the main thrust generated by a “sitting angle” in flight and this effect can be eliminated by dropping the mean values in the identification process.

Next, it follows from Eq.(4.23) and (4.24), that the pitch and roll moment derivatives with respect to flapping angles can be expressed by

$$M_a = (k_\beta + h_M T)/I_{yy} \tag{4.29}$$

$$L_b = (k_\beta + h_M T)/I_{xx} \tag{4.30}$$

Thereby, the normalised moments are

$$L/I_{xx} = L_a a + L_b b \tag{4.31}$$

$$M/I_{yy} = M_a a + M_b b \tag{4.32}$$

$$N/I_{zz} = N_r r + N_{ped}\delta_{ped} + N_{col}\delta_{col} \tag{4.33}$$

where L_a and M_b are cross-coupling derivatives used to capture unmodelled coupling effects, N_r is the yawing damping derivative accounting for the rate feedback from the onboard heading gyro, N_{col} is collective control derivative, and N_{ped} is the yawing control derivative. Note that the torque N_M produced by the main rotor are ignored, because it can be compensated by the anti-spinning torque $Y_T l_T$ from the tail rotor.

So far, the model structure used for describing helicopter dynamics is completed by substituting Eq.(4.26)-(4.28) and (4.31)-(4.33) back to Eq.(4.10). The schematic of the model structure is illustrated in Fig.4.5. It shows that the control signals first alter the states of the main and tail rotors, which produce corresponding thrusts, torques and flapping angles, then these elements act on the main rotor hub and tail boom to produce different forces and torques on the rigid-body of the helicopter. These forces and torques combining fuselage forces and the gravity through a mixer block completes the helicopter model. The equations contained in each block are summarised in Table.4.1. The question remained in modelling is how to decide the specific values of unknown parameters in the model structure.

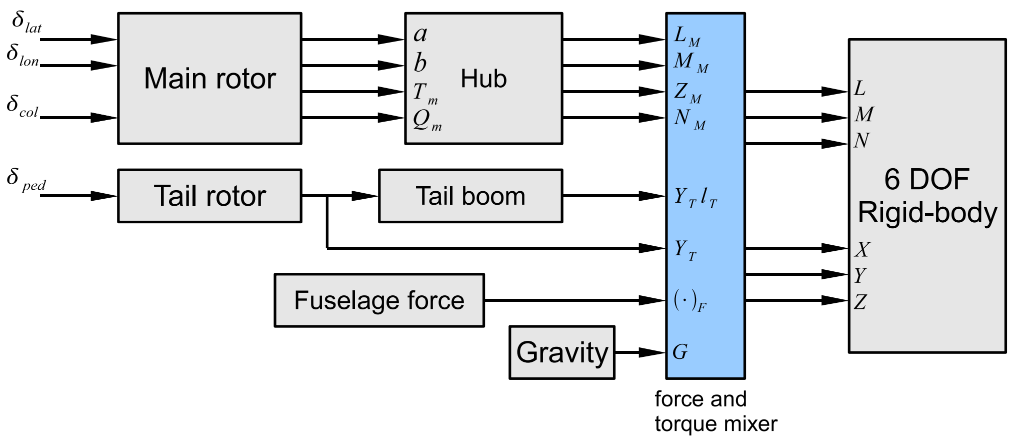


Figure 4.5: Schematic of the model structure

Table 4.1: Equations in the model schematic

Blocks	Corresponding equations
Main rotor	Eq.(4.18), (4.19), (4.20), (4.21)
Tail rotor	Y_T, $Y_T l_T$
Hub	Eq.(4.22), (4.23), (4.24)
Fuselage force	$X_u u$, $Y_v v$, $Z_w w$
Gravity	Eq.(4.11)
Mixer	Eq.(4.10), (4.26)-(4.28), (4.31)-(4.33)
Rigid-body	Eq.(4.8), (4.9), (4.1), (4.3)

4.3 System identification

This section introduces the identification procedure including the design of flight experiments, the identification algorithm, model breakdown and parameter estimation.

4.3.1 Preparation

Flight experiments play an important role in the system identification. Ideally, the flight experiments are executed in an open-loop style. In order to excite the desired mode and obtain the dynamic response in a large frequency range, the pilot should input sweep frequency signals and try not to add too much control efforts to remedy the stability of the helicopter (as long as the helicopter does not crash). Moreover, the input signals in the interested channels should have sufficient magnitudes to suppress the interferences of disturbances. This is very critical since the Trex-250 is small and light, and the response to disturbances may submerge the real control responses. In the identification, the model needs to be broken down into small blocks, and corresponding flight tests need to be performed. During the flight tests, at least two sets of data need to be collected for each pattern of flight. One is for identification and another is for validation.

The algorithm for parameter estimation used in this thesis is PEM, which is a conventional identification technique and has been included in the Matlab System Identification Toolbox [69]. PEM needs a parametrised model and determines the unknown parameter in a way that the prediction based on the model matches measured data as accurately as possible. It can be considered as an optimization process, where the optimisation variables are the unknown parameters of the model, the constraints are parameter ranges and model equations, and the cost function is used to penalise the deviation between model prediction and measured helicopter responses. The cost function that PEM tends to minimise can be defined as follows:

$$V_N(\theta, Z^N) = \frac{1}{N}\sum_{k=1}^{N} l(\varepsilon(k,\theta)) \tag{4.34}$$

where θ is the vector of parameters to be estimated, $Z^N = [\mathbf{u_k}, \mathbf{y_k}]$, $k = 1, 2, \cdots, N$, is the given experiment data set, $l(\cdot)$ is a positive defined scalar function, and ε is the error between measured data and predicted responses from model which depended on θ.

Since essentially there is a nonlinear optimisation involved in the estimation, there is a chance that the algorithm may trap in a local minimum that is not the actual value for a specific system, and the results can be sensitive to the initial guesses of the parameters. To overcome these problems, the full helicopter is partitioned into a few subsystems to describe the specific helicopter movement. The flight tests for each kind of movement are carried out in order to serve the corresponding subsystem identification.

4.3.2 Model Breakdown and Identification

As stated before, identifying all the parameters at once is very difficult or even impossible since there are 16 unknown parameters in the model structure. However, by breaking down the full helicopter model into several parts, it is possible to find suitable initial values for the unknown parameters. The helicopter dynamics can be divided into four channels, namely roll, pitch, yaw and heave. Thereby, the identification and the corresponding flight tests are implemented according to this sequence. When one channel is excited, the other channel should be kept as calm as possible. To investigate the coupling effect between roll and pitch, those two channels also need to be excited simultaneously. The identification flowchart is illustrated in Fig.4.6. Note that in the identification process, the inner-loop attitude dynamics are first identified, and then the outer-loop translational movements are investigated.

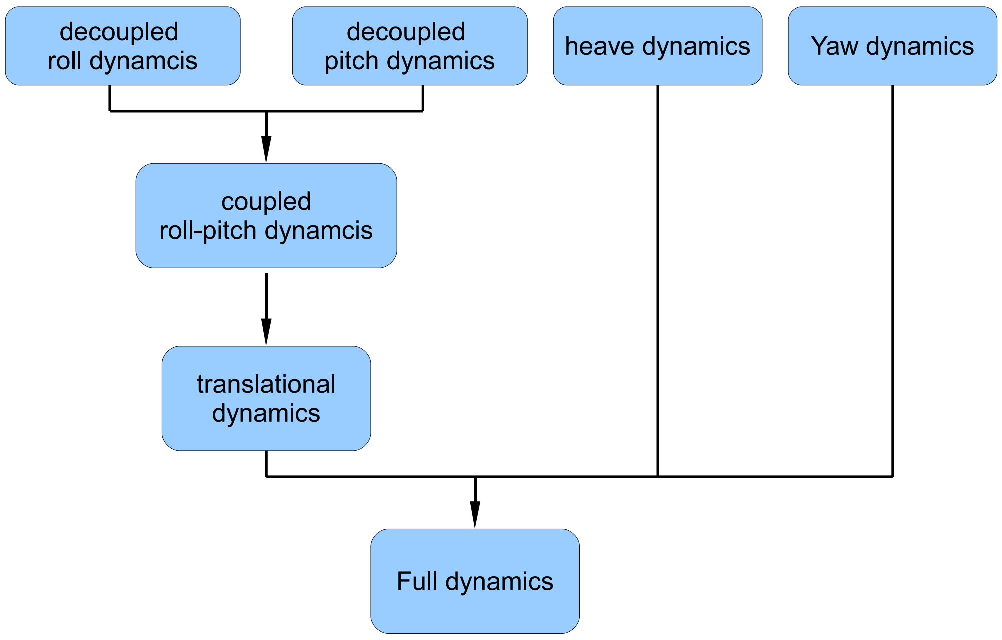


Figure 4.6: Flowchart of identification process

After the system identification process is broken down, the related model in each step is also simplified as there are less unknown parameters to estimate. Furthermore, the results obtained in the current step is set as the initial values for the following identification procedure to avoid the sensitivity in parameter's initial guess. The detailed steps and results in the flowchart Fig.4.6 are given as follows:

1. Decoupled roll and pitch dynamics
   The roll and pitch dynamics are initially identified in this step, respectively. Cross-coupling effects are avoided in the flight experiments, so these terms in the equations can be dropped. Specifically, in identifying the roll channel, the coefficients for flapping angle a and pitch rate q in Eq.(4.19) and (4.31) are ignored to yield the following equations

$$\begin{aligned} \dot{p} &= L_b b \\ \dot{b} &= -p - b/\tau + B_{lat}\delta_{lat} + B_{lon}\delta_{lon} \end{aligned} \tag{4.35}$$

   where L_b, B_{lat} and B_{lon} are unknown parameters to be estimated. Similarly, the equations for identifying the pitch channel can also be derived from Eq.(4.18) and (4.32) as:

$$\begin{aligned} \dot{q} &= M_a a \\ \dot{a} &= -q - a/\tau + A_{lat}\delta_{lat} + A_{lon}\delta_{lon} \end{aligned} \tag{4.36}$$

   where M_a, A_{lat} and A_{lon} need to be identified. The results in this step are given in Fig.4.7, where the estimated responses and experimental responses are compared.

2. Coupled roll-pitch dynamics
   In this step, roll and pitch dynamics are integrated together through cross-coupling terms L_a and M_b, such that

$$\begin{aligned} \dot{p} &= L_a a + L_b b \\ \dot{q} &= M_a a + M_b b \end{aligned} \tag{4.37}$$

   The corresponding flight test should excite both channels in an interactive way to identify L_a and M_b and refine the other parameters from the previous step. The control signals and identification results are shown in Fig 4.8.

3. Translational equations of motion

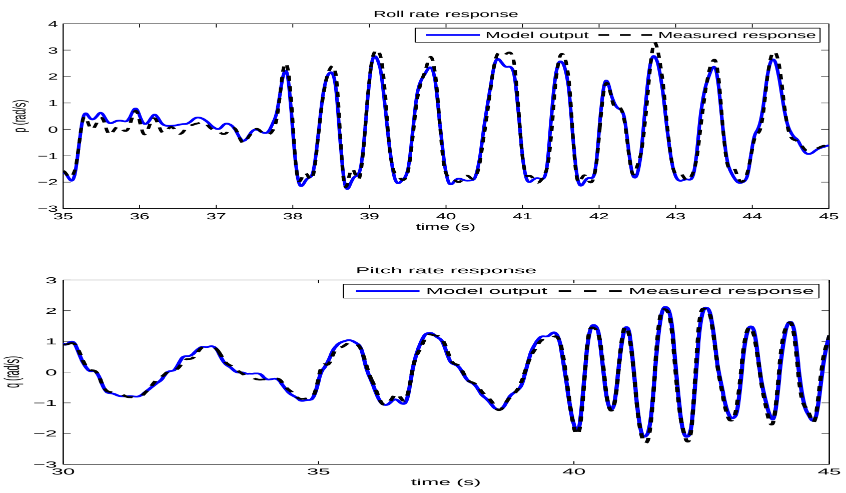


Figure 4.7: Decoupled roll and pitch identification

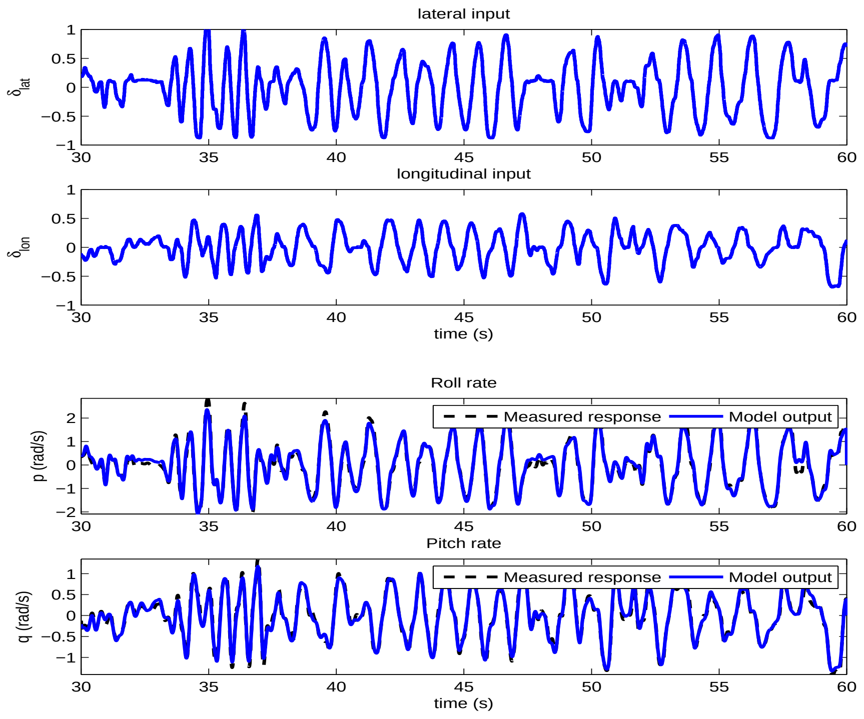


Figure 4.8: Coupled roll and pitch identification

In this step, roll-pitch dynamics Eq.(4.37) and translational dynamics composed of Eq.(4.26)-(4.27) and (4.8) are considered. To reduce the complication, two variables are defined in Eq.(4.38) to substitute the corresponding nonlinear terms in Eq.(4.26) and (4.27). Since the newly defined variables can be measured in the flight test, they can be treated as the control inputs. The key derivatives to be identified in this step are the translational damping terms X_u and Y_v.

$$\begin{aligned} termX &= -wq + vr - g\sin\theta \\ termY &= -ur + wp + g\sin\phi\cos\theta \end{aligned} \tag{4.38}$$

The same set of data used in step 2 is used for identifying the translational movements. Comparing with the existing state-space helicopter model [82], the model proposed in this paper introduces the two nonlinear terms to increase the fidelity. The *termX*, *termY* and identification results are shown in Fig 4.9.

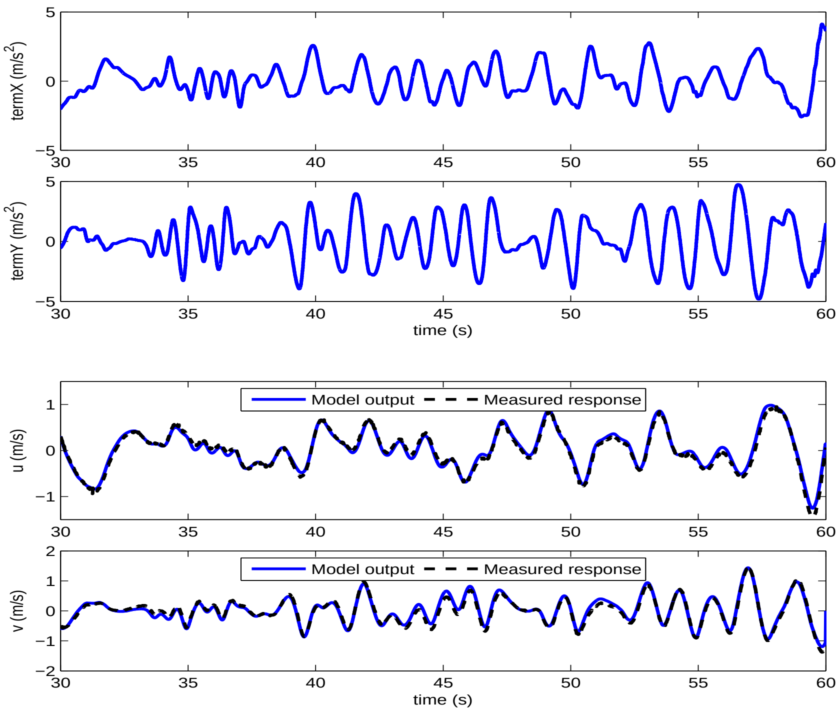


Figure 4.9: Translational dynamic identification

4. Yaw dynamics

Trex-250 helicopter is equipped with a head holding gyro, which makes the yaw channel quite stable and decoupled from the other channels. Hence, a first-order system is adequate to model this dynamics with derivatives of N_r, N_{ped} and N_{col} from Eq.(4.33). The control signal and identification result are given in Fig 4.10.

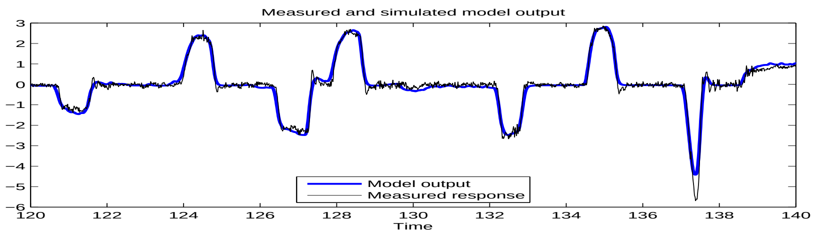


Figure 4.10: Yaw dynamic identification

5. Heave dynamics
   In this step, the unknown parameters Z_w and Z_{col} in Eq.(4.28) are identified. Similarly to the strategy in step 3, the nonlinear term is defined as an input variable in Eq.(4.39).

$$termZ = -vp + uq + g\cos\phi\cos\theta - g \tag{4.39}$$

6. Full helicopter model
   After previous steps, all of the unknown parameters have been initially estimated. The full helicopter model is evaluated in this step, where all the parameters are further refined using the previous results as initial values. The parameters identified are given in table 4.2.

4.3.3 Result validation

The model validation is conducted by driving the identified model using another set of flight data that was not used in the identification process. The validation flight data include excitations on all the channels to verify the model accuracy with respect to coupled helicopter dynamics. The validation results are presented in Fig 4.11. The results show a good agreement between the flight data and predicted data especially for longitudinal and lateral movements.

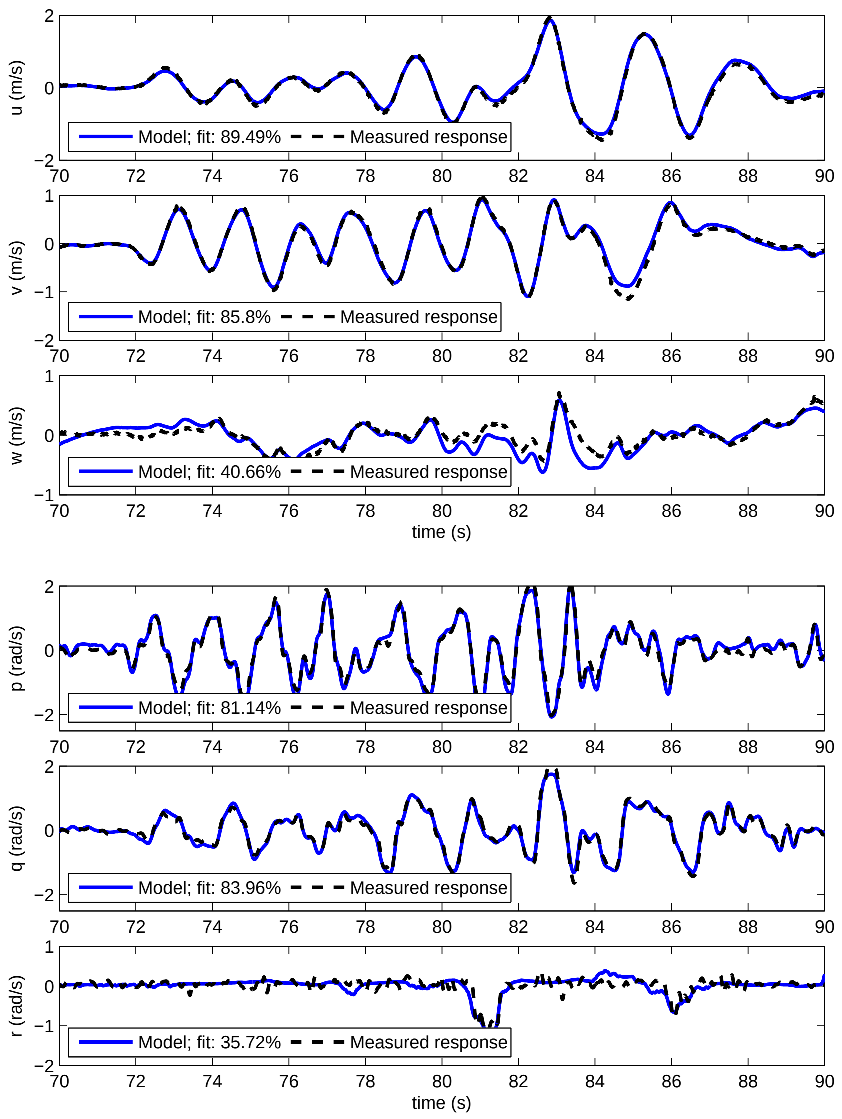


Figure 4.11: Model validation

Table 4.2: Identified parameter for Trex-250

Parameter (Unit)	Identified value	Standard deviation	Parameter (Unit)	Identified value	Standard deviation
X_u (s^{-1})	-0.233	0.006	Y_v (s^{-1})	-0.329	0.006
Z_w (s^{-1})	-0.878	0.022			
L_a (s^{-2})	83.98	11.21	L_b (s^{-2})	745.67	13.85
M_a (s^{-2})	555.52	9.89	M_b (s^{-2})	11.03	3.44
τ (s)	0.045	0.001	N_r (s^{-1})	-23.98	0.63
A_{lat} (rad)	0.196	0.003	A_{lon} (rad)	1.945	0.006
B_{lat} (rad)	2.120	0.007	B_{lon} (rad)	-0.38	0.010
Z_{col} (m/s^2)	-5.71	0.06			
N_{col} (rad/s^2)	8.89	2.9	N_{ped} (rad/s^2)	113.65	2.9

To examine the result quantitatively, a measurement of the Best Fit [69] is introduced, which is defined as the percentage of the output that the model reproduces:

$$\text{BestFit} = \left(1 - \frac{|\mathbf{y} - \hat{\mathbf{y}}|}{|\mathbf{y} - \bar{\mathbf{y}}|}\right) \times 100 \tag{4.40}$$

where $\mathbf{y}$ is the measured helicopter response, $\hat{\mathbf{y}}$ is the output predicted by the identified model and $\bar{\mathbf{y}}$ is the mean of $\mathbf{y}$. 100% corresponds to a perfect fit, and 0% indicates that the fit is no better than approximating the output by a constant ($\hat{\mathbf{y}} = \bar{\mathbf{y}}$). The Best Fit values are also displayed in the Fig.4.11.

In the flight test for model validation, all channels in the helicopter dynamics are excited simultaneously, and the model predictions still match the experimental data with a high accuracy. In contrast, in most of the literature, e.g. [82; 109], their time domain validations carried out with only one channel excited at one time. This means that the proposed identification process captures the majority of cross-coupling effects. It can also be seen that there are some mis-matches in both the heave and yaw channels. For the former, it is mainly due to the high nonlinearity inherited in the rotor system and excited by the aggressive flight. The effects such as inflow air instability and rotor speed variation cannot be captured by the present thrust model. The mis-match in the yaw channel mainly exists in the high frequency range. This is due to the presence of the head holding gyro, which always tries to compensate for the external disturbances. The identified model is able to capture the trend of the yaw movement and it is adequate for control design.

It is worth pointing out that although efforts have been taken to obtain an accuracy model from the flight test data, there is still a modelling uncertainty with respect to the real helicopter dynamics. The uncertainty, on one hand, is caused by the limitation of the current model structure and identification method as discussed above. On the other hand, it can be introduced by the physical changes of the helicopter. Because of the light structure of a miniature helicopter, its dynamics are vulnerable to mechanical changes, such different payloads, component upgrading and repair from crash. Therefore, the control algorithm designed for the miniature helicopter requires certain robustness to perform well in practice.

4.4 Basic control design

After a mathematical model of the helicopter is obtained, a control system can be designed by using model-based techniques. This section describes the controller design to support the non-aggressive flight of a small helicopter. Around the hover mode, helicopter dynamics can be considered as a linear system with multi-inputs-multi-outputs. Next, by decoupling the helicopter dynamics into several subsystems, a cascaded PID controller can be developed.

4.4.1 Model analysis

A helicopter has four primary input commands δ_{lat}, δ_{lon}, δ_{col} and δ_{ped} for controlling the roll and pitch rates, vertical velocity, and yaw rate, respectively. In piloting a helicopter, the cyclic inputs produce rotor moments and then change the fuselage attitude, but they cannot control the vehicle's position and velocity directly. After the fuselage's roll and pitch angles are changed, the rotor thrust is tilted accordingly to produce horizontal projections as propulsive forces. In this way, the helicopter's translational states can be controlled. On the other hand, the dynamics of a helicopter can be divided into slow translational movements (outer loop) and fast attitude movements (inner loop). Although this decomposition is only valid in non-aggressive flight, it provides a guideline for control analysis. So when a desired vehicle position is given, the controller first transfers a position requirement into an attitude command in a low bandwidth, and then controls the helicopter to track such a command in a high bandwidth.

4.4.2 PID controller

By following the above principle, a cascaded control structure can be derived for a helicopter's low-speed flight. First, an inner-loop controller is used to stabilise the attitude dynamics, which follows attitude commands and generates cyclic controls. Then, an outer-loop controller is designed for tracking the position and heading reference. It generates the collective pitch and pedal control directly based on height and heading errors and provides desired attitude commands for an inner-loop controller based on lateral and longitudinal errors. The control diagram is shown in Fig.4.12.

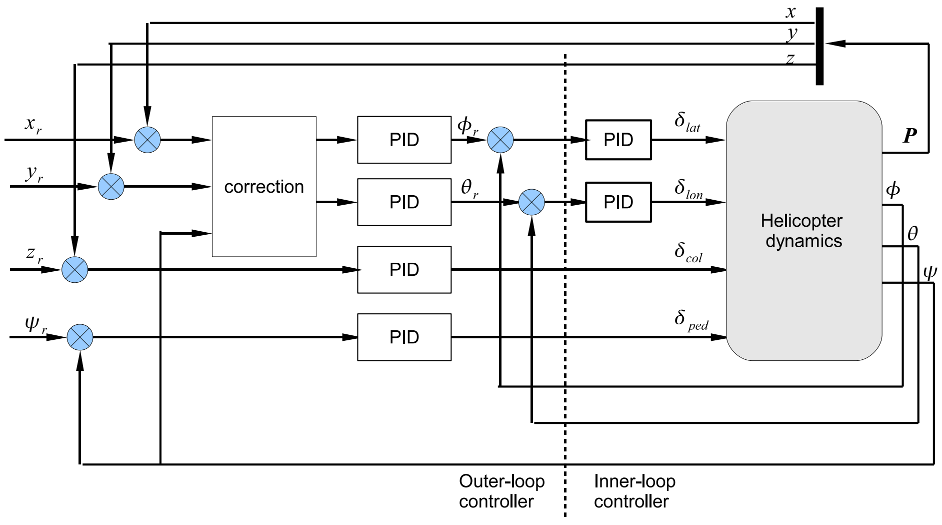


Figure 4.12: Cascaded control sturcure

In this control structure, the outer-loop controller employs four PID controllers. The first two measure the lateral and longitudinal error e_x^b and e_y^b in body-fixed coordinates and generate desired attitude requirements ϕ_r and θ_r. Note that the position error is usually compared in the inertial coordinate, so a correction block is introduced to transfer error $e_x^i = x_r - x$ and $e_y^i = y_r - y$ into body coordinates according to the heading angle ψ:

$$\begin{aligned} e_x^b &= e_x^i \cos\psi + e_y^i \sin\psi \\ e_y^b &= -e_x^i \sin\psi + e_y^i \cos\psi \end{aligned} \tag{4.41}$$

The other two PID controllers in the outer-loop eliminate the height and heading

Table 4.3: PID gains

	K_p	K_i	K_d
lateral	0.415 2	0.002	0.592
longitudinal	-0.313	-0.002	-0.450
height	-0.455	-0.010	-0.470
heading	0.750	0.010	0.500
roll	0.507	0.010	0.318
pitch	0.751	0.010	0.696

errors directly. In the inner-loop, there are another two PID controllers used to track the attitude command.

An important feature of PID based control is that it can be implemented without a specific model of the vehicle dynamics. All of the feedback gains in the control structure can be tuned empirically based on the observation of flight performance. However, this is a very tedious process and will be very risk if "trial and error" is directly applied on a real helicopter. Since the helicopter model is available, the tuning of feedback gains can be carried out using model-based design. To this end, a Simulink model is created including both the helicopter model and the control structure. The overall control system is then decomposed into four single-input-single-output (SISO) subsystems corresponding to four control channels: δ_{lat} to position y, δ_{lon} to position x, δ_{col} to height z and δ_{ped} to heading ψ. For each subsystem, we use Matlab's control design toolbox to optimise PID gains based on a step response criterion. The optimised gains are given in Table 4.3.

Simulations are carried out to verify the performance of the basic PID control. Step response results are shown in Fig.4.13. It can be seen that the cascaded PID can provide a satisfactory performance for non-aggressive flight.

It shall be noticed that the cascaded PID has its performance limit as it is designed based on a linearised and decoupled model. The decoupling means not only the translational and attitude separation, i.e. slow outer-loop and fast inner-loop, but also the four separated control channels, i.e. lateral, longitudinal, heading and heave. In addition, PID control exhibits poor robustness in the experiments. The control parameters tuned for the same helicopter may not work at the next experiment due to the trivial mechanical changes on the helicopter. Therefore, for more demanding flight requirements and various flight conditions, a comprehensive control analysis and design is required.

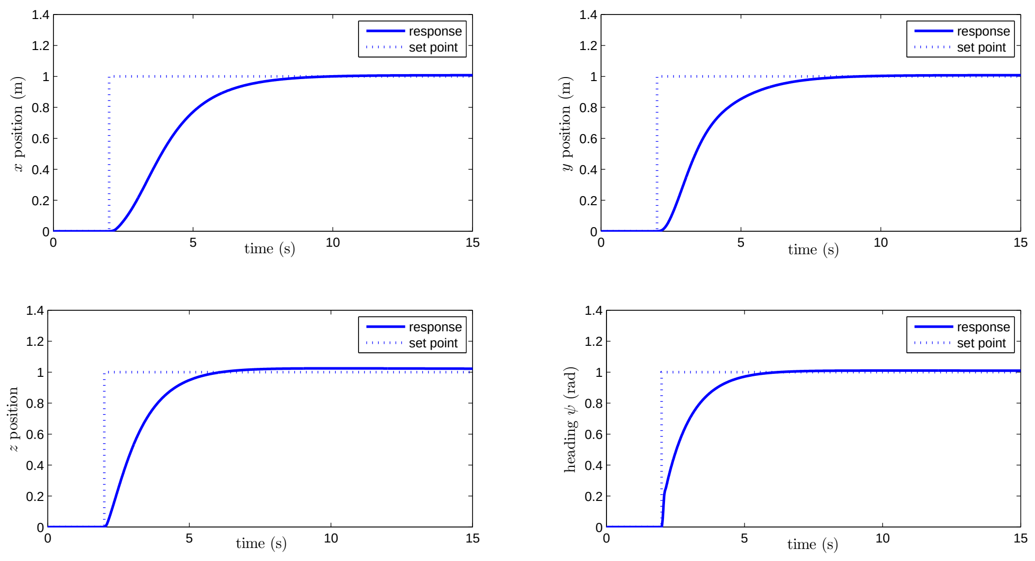


Figure 4.13: Step responses under PID control

4.5 Conclusion

This chapter describes a modelling and system identification procedure for a Trex-250 miniature helicopter. The identified model aims for control design and online prediction in the following work. Thereby, it has a simplified structure but contains the main nonlinear effects to cover a large flight region. The identification process, including model breakdown and flight test design, is discussed in detail. A specific flight test pattern has been used in each step to provide more deterministic estimations. The system identification results show a good match between the model predictions and the real flight data. Finally, a cascaded PID based controller is designed for the Trex-250 helicopter, which will be used as a baseline for future studies.

Chapter 5

Piecewise constant MPC for autonomous helicopters

5.1 Introduction

Chapter 4 describes the modelling process of unmanned helicopters. This chapter is to develop advanced control strategies based on the developed model. More specifically, the nonlinear model predictive control of autonomous helicopters will be investigated in this chapter.

Comparing to other control techniques, the NMPC provides a number of unique advantages for autonomous helicopter flight:

- It can deal with nonlinear, multiple-inputs-multiple-outputs dynamics of helicopters by directly using their mathematical models in the control loop;
- It explicitly takes into account states and control constraints to guarantee flight safety and prevent control saturation;
- The outer-loop and inner-loop of helicopters are considered as an entire system, which results in an integrated guidance and control fashion that enhances flight agility;
- It provides a local path planning function by combining future reference and environment information such as obstacles and collisions.

However, MPC techniques, especially the nonlinear MPC, imposes challenges in real-time implementation because a computationally intensive nonlinear optimisation problem is required to solve at each sampling instant. This issue has been discussed in the literature review in Chapter 2.

Although with the development of avionics and microprocessor technology online optimisation becomes feasible, there are still difficulties in directly applying MPC algorithms into plants with fast dynamics like helicopters. Initial trials on implementation of MPC algorithms on helicopters with short prediction horizons have been reported in [53; 106]. Nevertheless, the further reduction of computational burden in control algorithms can always benefit applications. The extra computation power can be put on extending the prediction horizon, including a more detailed model, and/or taking into account more information such as obstacles in the flight environment.

This chapter proposes a control framework for autonomous helicopters, which explores the advantages of nonlinear MPC while being able to apply to systems with fast dynamics. Instead of attempting to implement a single NMPC, the proposed framework employs a two-level control structure where the high-level MPC generates baseline control by exploiting the nonlinear helicopter model and environment information, and the low-level linear controller, designed based on linearisation around the state reference provided by the high-level controller, compensates the baseline control in the presence of disturbances and uncertainties. These two-level controllers are parallelly operated at different time scales. The high-level MPC strategy runs in a lower sampling rate allowing enough time to perform online optimisation, while the low-level controller is executed in a much higher sampling rate to accommodate helicopter dynamics.

One feature of the proposed control framework is that the high-level MPC adopts the piecewise constant control scheme [72]. The modified algorithm allows the online optimisation occurring at scattered sampling instants without losing the prediction accuracy. More importantly, the piecewise constant MPC significantly reduces the number of control variables to be decided (also known as optimisation variables) in the optimisation problem, which helps to ease the workload of the online optimisation.

The stability of the control framework is investigated particularly focusing on the modified piecewise constant MPC. The design procedure of the terminal region and terminal penalty that guarantee the close-loop stability is discussed in detail and illustrated through an example. In addition, the stability analysis also provides a way to construct a feasible initial control sequence for each optimisation, on which the stability is assured even if only the suboptimal solution can be found during the online optimisation.

To verify the proposed control framework flight tests on a small-scale helicopter are carried out. The flight test utilises the indoor test facility developed in Chapter 3, where ground station is constructed with the capabilities of observing the helicopter states and integrating the OP solver with the real-time controller.

5.2 Helicopter model

The dynamic model of a small-scale helicopter has been developed in Chapter 4. Here the basic structure of the model is recalled in (5.1) for the completeness.

$$
\begin{aligned}
&[\ \dot{x} \quad \dot{y} \quad \dot{z}\ ]^T = \mathbf{R}_b^i(\phi,\theta,\psi)[\ u \quad v \quad w\ ]^T && \text{(5.1a)}\\
&\dot{u} = vr - wq - g\sin\theta + X_u u + X_a a && \text{(5.1b)}\\
&\dot{v} = wp - ur + g\cos\theta\sin\phi + Y_v v + Y_b b && \text{(5.1c)}\\
&\dot{w} = uq - vp + g\cos\theta\cos\phi + Z_w w + Z_{col}\delta_{col} - g && \text{(5.1d)}\\
&\dot{p} = -qr(I_{yy} - I_{zz})/I_{xx} + L_a a + L_b b && \text{(5.1e)}\\
&\dot{q} = -pr(I_{zz} - I_{xx})/I_{yy} + M_a a + M_b b && \text{(5.1f)}\\
&\dot{r} = -pq(I_{xx} - I_{yy})/I_{zz} + N_r r + N_{col}\delta_{col} + N_{ped}\delta_{ped} && \text{(5.1g)}\\
&\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta && \text{(5.1h)}\\
&\dot{\theta} = q\cos\phi - r\sin\phi && \text{(5.1i)}\\
&\dot{\psi} = q\sin\phi\sec\theta + r\ cos\phi\sec\theta && \text{(5.1j)}\\
&\dot{a} = -q - \frac{a}{\tau} + \frac{A_{lat}}{\tau}\delta_{lat} + \frac{A_{lon}}{\tau}\delta_{lon} && \text{(5.1k)}\\
&\dot{b} = -p - \frac{b}{\tau} + \frac{B_{lat}}{\tau}\delta_{lat} + \frac{B_{lon}}{\tau}\delta_{lon} && \text{(5.1l)}
\end{aligned}
$$

where $\mathbf{x} = [\ x \quad y \quad z \quad u \quad v \quad w \quad p \quad q \quad r \quad \phi \quad \theta \quad \psi\ ]^T$ is the state of the rigid-body of the helicopter consisting of inertial position, local velocity, angular rate and attitude, respectively, $\mathbf{R}_b^i$ is a transformation matrix from body to inertial coordinates defined in Eq.(4.2), $\mathbf{u} = [\ \delta_{lat} \quad \delta_{lon} \quad \delta_{ped} \quad \delta_{col}\ ]^T$ is the control inputs including lateral and longitudinal cyclic, pedal and collective pitch, respectively, and the other parameters in the model are the stability and control derivatives, whose values are obtained using system identification. Note that the moment of inertia along body axes I_{xx}, I_{yy} and I_{zz} are remained in the model structure for the sake of completeness, whose values can be measured through a pendulum experiment. The dynamics of the main rotor is described by the flapping angles $[\ a \quad b\ ]^T$ with the effective time constant τ. However, the rotor flapping states a and b cannot be directly measured, which usually relies on a state observer. In order to reduce the complexity and focus on the control design, a steady state approximation is adopted as a measurement of the flapping angles [9]:

$$\begin{aligned} a &= -\tau q + A_{lat}\delta_{lat} + A_{lon}\delta_{lon} \\ b &= -\tau p + B_{lat}\delta_{lat} + B_{lon}\delta_{lon} \end{aligned} \tag{5.2}$$

By substituting (5.2) into (5.1), it is able to represent the helicopter model into a compact form to facilitate the following analysis:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{5.3}$$

Note that a more detailed model can be used to describe the helicopter dynamics without affecting the controller design. However, to facilitate the flight test, the presented model developed in Chapter 4 is adopted, because its parameters for the selected helicopter has been obtained through system identification.

5.3 MPC based control framework

MPC techniques need to solve a formulated optimisation problem at each sampling instant. The computational load will be significantly increased for a highly nonlinear system with multi-inputs-multi-outputs. On the other hand, the fast dynamics of a helicopter require quick responses to the external environment, which means the controller has to work at a high bandwidth. In order to overcome this problem, a control strategy which comprises of a high-level piecewise constant MPC and a low-level linear controller is developed.

5.3.1 Piecewise constant MPC

The traditional MPC is either developed based on a continuous system model or a discrete counterpart. A continuous-time model is much more natural and accurate in terms of describing the behaviour of a system, but the corresponding MPC algorithm involves continuously solving an optimisation problem, which is difficult to implement as a computational time is required for performing online optimisation. In contrast, the discrete MPC uses the discrete representation of the system and makes online implementation feasible by solving the optimisation problem only at each sampling instant. The computational demand reduces when the discretisation sampling time increases, but as a result the accuracy of the approximated discrete representation degrades. Moreover, system states and constraints can only be evaluated at the sampling instants leaving those within sampling intervals being ignored.

In this section, a modified MPC strategy that uses piecewise constant controls to drive a continuous system or an accurate discrete approximation is introduced. The piecewise constant control suggests that the control signals keep constant values (i.e. zero-order holding) for several discretisation intervals, which makes the proposed algorithm different from normal sampled data MPC or discrete time MPC that has been investigated by many researchers [75]. This strategy allows optimisation to be performed at the individual instants as in discrete MPC while using a more accurate model to predict the evolution of the system.

By trading-off between the prediction accuracy and computational burden, the discrete model approximated from a continuous model (5.3) with a high sampling frequency is chosen as the prediction model. The discretisation sampling time is defined as T_d, which is also the integration step used in prediction. The error between the discrete model and continuous model increases monotonically with T_d. The MPC designed on a discrete model can stabilise the original continuous model if T_d is small enough [39]. However, the small T_d increases the computational burden, as there are more variables to be decided with respect to the same prediction length.

To avoid this problem, it is necessary to introduce another important parameter, namely the MPC sampling time T_s, defined as the interval of the MPC updating the current states and generating a new control sequence. In the conventional discrete MPC setting, $T_d = T_s$. However, with respect to T_d and T_s there is a *control holding horizon* N such that $T_s = N \cdot T_d$, which also implies the control inputs keeping the constant values for N integration steps.

To clearly explain the time setting, an example is illustrated in Fig 5.1. The control holding horizon N is set to 4 steps, the same with the prediction horizon H. Within the period of T_s the control variables are set to constant, while the integration of system equation follows the discrete sampling time T_d. This setting maintains the accuracy of the prediction but significantly reduces the number of optimisation variables covering the same length of prediction. For example, when $N = 4$ and $H = 4$, in the conventional MPC there are $N \times H = 16$ variables to be optimised, but in the control holding scheme, only 4 variables need to be optimised.

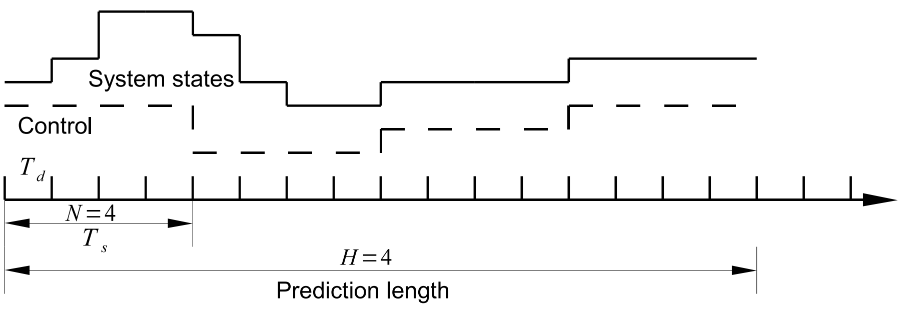


Figure 5.1: Time setting example

Under this setting, a discrete MPC form is employed for the flight control. By defining the reference trajectory as $\mathbf{x}_r$ and the tracking error $\mathbf{x}_e = \mathbf{x}_r - \mathbf{x}$, the objective function to be minimised can be formulated as:

$$\begin{aligned} J(k) = {} & F(\mathbf{x}(k+HN)) + \\ & \sum_{i=0}^{H-1}\sum_{j=0}^{N-1} L(\mathbf{x}(k+iN+j),\, \mathbf{u}(k+iN+j)) \end{aligned} \tag{5.4}$$

$$\begin{aligned} L(\mathbf{x}(k),\, \mathbf{x}_r(k),\, \mathbf{u}(k)) &= \mathbf{x}_e(k)' Q \mathbf{x}_e(k) + \mathbf{u}(k)' R \mathbf{u}(k) \\ F(\mathbf{x}(k+HN),\, \mathbf{x}_r(k)) &= \mathbf{x}_e(k+HN)' P \mathbf{x}_e(k+HN) \end{aligned}$$

where $L(\mathbf{x}(k),\, \mathbf{x}_r(k),\, \mathbf{u}(k))$ is the penalty term for each integration step, $F(\mathbf{x}(k+HN),\, \mathbf{x}_r(k))$ is the terminal penalty, H is the prediction horizon, N is the control holding horizon, and P, Q and R are the semi-positive definite weighting matrices.

Remark 5.1. The control holding horizon N plays an important role in the modified MPC formulation. If $N = 1$ the modified MPC reverts to the conventional MPC setting. For the given prediction duration required by the closed-loop stability, increase of N can reduce the number of variables to be optimised, hence significantly reduce the computational burden. On the other hand, for a given number of optimisation variables that the online solver can handle, increase of N can expand the prediction length.

Remark 5.2. With the modified time setting the time allowed for OP solving is increased to T_s, while maintaining the resolution of the integration to T_d in the prediction. Note that other MPC formulations may also have different sampling time for discretisation and realtime implementation [41]. This is due to the heavy computational load rather than actively using the control holding mechanism to reduce the computational load.

The nonlinear optimisation problem that minimises the objective function subjected to various constraints can be stated as:

$$\mathbf{x}_m,\, \mathbf{u}_m = \arg\min_{\hat{\mathbf{x}},\, \hat{\mathbf{u}}} J(k) \tag{5.5}$$

subject to:

$$\begin{aligned}
&\hat{\mathbf{x}}(k+j+1) = f(\hat{\mathbf{x}}(k+j),\, \hat{\mathbf{u}}(k+j)) \\
&\hat{\mathbf{x}}(k+j) \in \mathbb{X} \\
&\hat{\mathbf{u}}(k+j) \in \mathbb{U} \\
&\hat{\mathbf{x}}(k+HN) \in \boldsymbol{\Omega} \\
&j = 0,\, 1,\, \cdots, N-1 \\
&\hat{\mathbf{x}}(k) = \mathbf{x}(k)
\end{aligned}$$

where $\mathbf{x}(k+1) = f(\mathbf{x}(k),\, \mathbf{u}(k))$ is the discrete form of the helicopter dynamics with the discretisation time T_d, and $\mathbb{X}$, $\mathbb{U}$ and $\boldsymbol{\Omega}$ are control constraints, state constraints and terminal region, respectively. The hat symbol is used to indicate the variables in the prediction distinguishing from the real variables. This optimisation problem is solved at each sampling instant, producing the state reference $\mathbf{x}_m$ and the baseline control sequence $\mathbf{u}_m$, in which the first element is applied into the helicopter.

5.3.2 Two-level control framework

The proposed MPC scheme eases the computational burden by (i) increasing the computational interval to give more time for optimisation and (ii) reducing the number of variables to be optimised for a given predictive horizon. However, the MPC strategy becomes an open-loop optimal control within the interval T_s. Unfortunately, due to the mis-match between the mathematical model and the real helicopter, the noises and disturbances in the process, this kind of optimal control may result in a significantly degraded performance. Within the interval T_s, the MPC cannot suppress any tracking error caused by these sources. Experiments have shown that the bandwidth associated with the MPC may not be adequate for stabilising and controlling the helicopters that have fast dynamics.

In order to overcome these difficulties in implementing NMPC with online optimisation, a two-level structure is adopted in the control framework. The high-level controller is the MPC strategy described before, which can provide optimised state reference $\mathbf{x}_m$ and the corresponding baseline control $\mathbf{u}_m$, whereas the low-level controller is a linear feedback controller that can provide stability around the optimised state reference in the presence of disturbances and uncertainties. The high-level controller runs at a lower sampling rate T_s to adapt the calculation time caused by solving the nonlinear OPs. In contrast, the low-level controller works at a much higher sampling rate to reject disturbances. The control structure is shown in Fig 5.2.

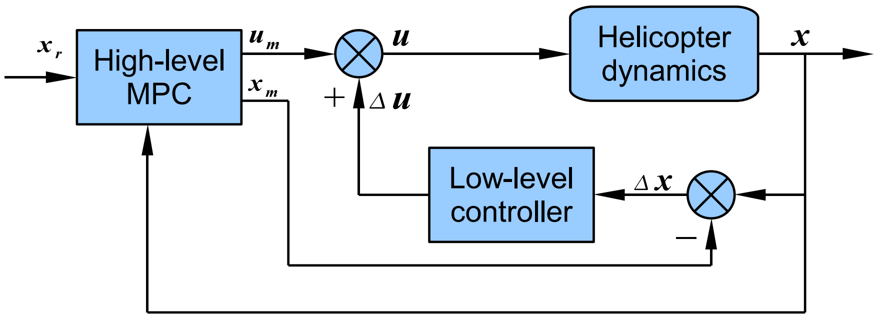


Figure 5.2: Two-level control framework

In the implementation, the low-level controller measures real helicopter states $\mathbf{x}$ and compares them against the state reference $\mathbf{x}_m$ from the high-level MPC. The error signals $\Delta\mathbf{x}$ are used to generate local compensation control $\Delta\mathbf{u}$. The overall control inputs $\mathbf{u}$ applied to the vehicle consist of two parts: the nominal control inputs and the compensation control generated by the local controller, i.e. $\mathbf{u} = \mathbf{u}_m + \Delta\mathbf{u}$.

The low-level controller is designed based on perturbation models around the reference state $\mathbf{x}_m$ and control $\mathbf{u}_m$. Since the low-level controller works in a much higher sampling rate, the controller design can be performed in the continuous time domain. The helicopter model can be linearised around the nominal reference and input as:

$$\begin{aligned}\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \approx f(\mathbf{x}_m, \mathbf{u}_m) &+ \left.\frac{\partial f}{\partial x}\right|_{\mathbf{x}_m, \mathbf{u}_m} (\mathbf{x} - \mathbf{x}_m) \\ &+ \left.\frac{\partial f}{\partial u}\right|_{\mathbf{x}_m, \mathbf{u}_m} (\mathbf{u} - \mathbf{u}_m)\end{aligned} \tag{5.6}$$

By defining the error state $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_m$ and control compensation $\Delta\mathbf{u} = \mathbf{u} - \mathbf{u}_m$. The system (5.6) can be stated as a parameter dependent system (5.7).

$$\Delta\dot{\mathbf{x}} = \left.\frac{\partial f}{\partial x}\right|_{\mathbf{x}_m, \mathbf{u}_m} \Delta\mathbf{x} + \left.\frac{\partial f}{\partial u}\right|_{\mathbf{x}_m, \mathbf{u}_m} \Delta\mathbf{u} = A_m(\mathbf{x}_m, \mathbf{u}_m)\Delta\mathbf{x} + B_m(\mathbf{x}_m, \mathbf{u}_m)\Delta\mathbf{u} \tag{5.7}$$

Considering a static state feedback K, i.e. $\Delta\mathbf{u} = -K\Delta\mathbf{x}$, the close-loop system can further be expressed as:

$$\Delta\dot{\mathbf{x}} = (A_m(\cdot) - B_m(\cdot)K)\Delta\mathbf{x} = A_{cl}(\cdot)\Delta\mathbf{x} \tag{5.8}$$

The parameters in $A_{cl}(\cdot)$ are dependent on $\mathbf{x}_m$ and $\mathbf{u}_m$, which are bounded in the high-level optimisation. Hence, the system (5.8) can be converted into a polytopic system with its vertices computed by the uncertainty parameters with defined boundary values, and the robust stability of such a system can be guaranteed by using the parameter dependent Lyapunov function technique [35; 38].

5.4 Stability analysis

This section investigates the stability of the two-level control framework proposed in the last section. To this end, the stability of the piecewise constant MPC as the high-level controller will be analysed first.

5.4.1 Stability of the piecewise constant MPC

The stability of proposed piecewise constant MPC is investigated by using Lyapunov technique inspired by [72]. The analysis considers the helicopter performing hovering flight which is a typical flight mode for helicopters. Since all the states in the hovering are zeros, the error state $\mathbf{x}_e$ in (5.4) can be replaced by $\mathbf{x}$.

Next, a terminal region $\mathbf{\Omega}$ and an associated terminal controller $k_f(\cdot)$ are defined to satisfy a number of assumptions:

Assumption 5.1. The terminal region is a neighbourhood of the origin where state constraints are satisfied in this region. $0 \in \mathbf{\Omega}$, and $\mathbf{\Omega} \in \mathbb{X}$ is closed.

Assumption 5.2. There exists a terminal controller $k_f(\cdot)$ such that control constraints are satisfied for all the states in the terminal region. $k_f(x) \in \mathbb{U}$, $\forall x \in \mathbf{\Omega}$.

Assumption 5.3. The N step evolution of the system under the terminal control $k_f(\cdot)$ stays in the terminal region. $\varphi_N(x, k_f(x)) \in \mathbf{\Omega}$, $\forall x \in \mathbf{\Omega}$, where $\varphi_i(x,u)$ denotes the states of the system at i step from an initial state x under the constant control signal $u = k_f(\cdot)$.

Considering the evolution of the system along the time, at sampling instant k the optimised cost function is denoted as:

$$\bar{V}_m(x_k) = \|\bar{x}_{k+HN}\|_P^2 + \sum_{j=0}^{H-1}\sum_{i=0}^{N-1} \|\bar{x}_{k+j\cdot N+i}\|_Q^2 + \sum_{j=0}^{H-1} N \cdot \|\bar{u}_{k+j\cdot N}\|_R^2 \tag{5.9}$$

where $\bar{x}_{k+j\cdot N+i}$ denotes the optimised states evolving from the time instant k, and $\bar{u}_{k+j\cdot N}$ denotes the control sequence that is generated in a zero-holding fashion. Note that the bar symbol is used to indicate variables with the optimised values.

Following Assumption 5.1-5.3, at the next MPC sampling instant $k+N$ there is a feasible control sequence defined as:

$$\hat{\mathbf{u}}_{k+N:k+HN} = \left\{\bar{u}_{k+N}, \bar{u}_{k+2N}, \ldots, \bar{u}_{k+(H-1)N}, k_f(\bar{x}_{k+HN})\right\} \tag{5.10}$$

where $\bar{x}_{k+HN}$ is the terminal state in the previous prediction, implicitly staying in the terminal region $\mathbf{\Omega}$. Thus, the corresponding cost function at time instant $k+N$ is:

$$\begin{aligned} V_m(x_{k+N}) &= \|x_{k+HN+N}\|_P^2 + \sum_{j=0}^{H-1}\sum_{i=0}^{N-1} \left\|x_{k+(j+1)\cdot N+i}\right\|_Q^2 + \sum_{j=0}^{H-1} N \cdot \left\|u_{k+(j+1)\cdot N}\right\|_R^2 \\ &= \left\|x_{k+(H+1)N}\right\|_P^2 + \sum_{j=1}^{H}\sum_{i=0}^{N-1} \|x_{k+j\cdot N+i}\|_Q^2 + \sum_{j=1}^{H} N \cdot \|u_{k+j\cdot N}\|_R^2 \end{aligned} \tag{5.11}$$

When there is no error between the model and the real plant and in the absence of disturbances, the measured state x_{k+N} at instant $k+N$ should be equal to the state $\bar{x}_{k+N}$ predicted at instant k. Therefore, it is possible to inspect the following relationship.

$$\begin{aligned}
&V_m(x_{k+N}) - \bar{V}_m(x_k) \\
&= \left\|x_{k+(H+1)N}\right\|_P^2 - \|x_{k+HN}\|_P^2 + \sum_{j=1}^{H}\sum_{i=0}^{N-1}\|x_{k+j\cdot N+i}\|_Q^2 - \sum_{j=0}^{H-1}\sum_{i=0}^{N-1}\|x_{k+j\cdot N+i}\|_Q^2 \\
&\quad + \sum_{j=1}^{H} N\cdot\|u_{k+j\cdot N}\|_R^2 - \sum_{j=0}^{H-1} N\cdot\|u_{k+j\cdot N}\|_R^2 \\
&= \left\|x_{k+(H+1)N}\right\|_P^2 - \|x_{k+HN}\|_P^2 + \sum_{i=0}^{N-1}\|x_{k+H\cdot N+i}\|_Q^2 + N\cdot\|u_{k+H\cdot N}\|_R^2 \\
&\quad - \sum_{i=0}^{N-1}\|x_{k+i}\|_Q^2 - N\cdot\|u_k\|_R^2
\end{aligned} \tag{5.12}$$

Note that at time instant $k+N$, the online optimisation is performed to minimise the cost function initialised by (5.11). Thereby the optimised cost function implies:

$$\bar{V}_m(x_{k+N}) \le V_m(x_{k+N}) \tag{5.13}$$

The cost function $V_m(x)$ is adopted as the Lyapunov function candidate for the proposed piecewise constant MPC. The stability condition based on Lyapunov theory requires:

$$\bar{V}_m(x_{k+N}) \le \bar{V}_m(x_k) \tag{5.14}$$

which means the Lyapunov function is non-increasing. This is met if

$$V_m(x_{k+N}) - \bar{V}_m(x_k) < 0 \tag{5.15}$$

By recalling Eq.(5.12) the above stability condition is satisfied if:

$$\left\|x_{k+(H+1)N}\right\|_P^2 - \|x_{k+HN}\|_P^2 + \sum_{i=0}^{N-1}\|x_{k+H\cdot N+i}\|_Q^2 + N\cdot\|u_{k+H\cdot N}\|_R^2 < 0 \tag{5.16}$$

To complete the stability analysis, the next step is to find a terminal penalty P, a suitable terminal region $\boldsymbol{\Omega}$ and an associated terminal control $k_f(\cdot)$ such that condition (5.16) and Assumption 5.1-5.3 are fulfilled for any $x_{k+HN} \in \boldsymbol{\Omega}$. This is a coupled problem, and some practical procedures will be provided in the next subsection to determine these parameters.

Remark 5.3. The terminal controller will never be applied to the real system, but it can be used to construct an initial solution of the OP through Eq.(5.10). This initial solution is feasible and available at each sampling instant as long as an solution can be found for the formulated OP at beginning, and it speeds up the convergence of online optimisation during the implementation.

5.4.2 Terminal region and controller

Assuming the linearised discrete model in the hovering mode is as follows (note that the notations with the index in the round bracket are used to represent the states of the linearised model):

$$x(k+1) = Ax(k) + Bu(k) \tag{5.17}$$

Because the control inputs remain constant during the MPC sampling time $T_s = N \cdot T_d$ as discussed before, it can be obtained that:

$$\begin{aligned} x(k+2) &= A^2x(k) + ABu(k) + Bu(k) \\ &\vdots \\ x(k+N) &= A^N x(k) + (A^{N-1}B + \cdots + AB + B)u(k) \end{aligned} \tag{5.18}$$

By defining two matrices $A_i = A^i$ and $B_i = A^{i-1}B + \cdots + AB + B$, a compact form of Eq.(5.18) is found:

$$x(k+i) = A_i x(i) + B_i u(k), \quad i = 1, \ldots, N \tag{5.19}$$

Moreover, if one can find a linear terminal control

$$u(k) = -k_f x(k), \quad \forall x(k) \in \mathbf{\Omega} \tag{5.20}$$

the equation (5.19) can be further written as:

$$x(k+i) = A_i x(k) + B_i k_f x(k) = A_i^{cl} x(k), \quad i = 1, \ldots, N \tag{5.21}$$

where A_i^{cl} is the close-loop system matrix. When designing the terminal controller k_f, the linear discrete control theory can be used. The resulting control has to guarantee that the eigenvalues of A_N^{cl} stay in the unit disc.

Then, from a terminal state $x_k = x(k)$ to i step of the evolution under the terminal control, the difference between the nonlinear model states x_{k+i} and linearised model states $x(k+i)$ can be described by:

$$\Phi_i(x_k) = \varphi_i(x_k, k_f x_k) - A_i^{cl} x_k \tag{5.22}$$

By invoking (5.22), one can derive the following relationship from the stability condition (5.16):

$$\begin{aligned}
&\left\|x_{k+(H+1)N}\right\|_P^2 - \|x_{k+HN}\|_P^2 + \sum_{i=0}^{N-1} \|x_{k+H\cdot N+i}\|_Q^2 + N\cdot\|u_{k+H\cdot N}\|_R^2 \\
&= \left\|\Phi_N(x_{k+HN}) + A_N^{cl} x_{k+HN}\right\|_P^2 - \|x_{k+HN}\|_P^2 \\
&\quad + \sum_{i=0}^{N-1} \left\|\Phi_i(x_{k+HN}) + A_i^{cl} x_{k+HN}\right\|_Q^2 + N\cdot\|u_{k+H\cdot N}\|_R^2 \\
&= \|\Phi_N(x_{k+HN})\|_P^2 + 2\cdot\Phi_N(x_{k+HN})' P A_N^{cl} x_{k+HN} + \left\|A_N^{cl} x_{k+HN}\right\|_P^2 - \|x_{k+HN}\|_P^2 \\
&\quad + \sum_{i=0}^{N-1} \|\Phi_i(x_{k+HN})\|_Q^2 + \sum_{i=0}^{N-1} 2\cdot\Phi_i(x_{k+HN})' Q A_i^{cl} x_{k+HN} \\
&\quad + \sum_{i=0}^{N-1} \left\|A_i^{cl} x_{k+HN}\right\|_Q^2 + N\cdot\|u_{k+H\cdot N}\|_R^2
\end{aligned} \tag{5.23}$$

Notice that in the optimisation problem (5.5), the terminal state x_{k+HN} is forced in the terminal region $\mathbf{\Omega}$, which can be specified as the neighbourhood of the origin with the radius α:

$$\mathbf{\Omega}(k_f, N) = \{x \in \mathbb{R}^n : \|x\|_P^2 < \alpha\} \tag{5.24}$$

It follows from the definition of $\Phi_i(x_k)$ that it depends on a high order of x, so that $\|\Phi_i(x_k)\|$ is approaching zero faster than $\|x_k\|$ when the radius of the terminal region α approaching zero. Therefore, for a small enough α, a positive scalar γ and a positive definite matrix $\tilde{Q}$, where eigenvalues $\lambda(\tilde{Q}) > \lambda(Q)$ can be found, such that:

$$\|\Phi_N(x_{k+HN})\|_P^2 + 2\cdot\Phi_N(x_{k+HN})' P A_N^{cl} x_{k+HN} \leq \gamma\,\|x_{k+HN}\|^2 \tag{5.25}$$

and

$$\|\Phi_i(x_{k+HN})\|_Q^2 + 2 \cdot \Phi_i(x_{k+HN})' Q A_i^{cl} x_{k+HN} + \left\|A_i^{cl} x_{k+HN}\right\|_Q^2 \leq \left\|A_i^{cl} x_{k+HN}\right\|_{\tilde{Q}}^2 \tag{5.26}$$

By substituting Eq.(5.25), (5.26) and terminal control (5.20) into the stability condition (5.23), one can derive:

$$\begin{aligned}
&\left\|x_{k+(H+1)N}\right\|_P^2 - \|x_{k+HN}\|_P^2 + \sum_{i=0}^{N-1} \|x_{k+H\cdot N+i}\|_Q^2 + N \cdot \|u_{k+H\cdot N}\|_R^2 \\
\leq &\left\|A_N^{cl} x_{k+HN}\right\|_P^2 - \|x_{k+HN}\|_P^2 + x_{k+HN}{}' (\sum_{i=0}^{N-1} \left\|A_i^{cl}\right\|_{\tilde{Q}}^2) x_{k+HN} \\
&+ N \cdot \|k_f x_{k+HN}\|_R^2 + \gamma \|x_{k+HN}\|^2 \\
= &x_{k+HN}{}' \left(A_N^{cl}{}' P A_N^{cl} - P + \sum_{i=0}^{N-1} \left\|A_i^{cl}\right\|_{\tilde{Q}}^2 + N \cdot k_f{}' R k_f + \gamma I_n \right) x_{k+HN}
\end{aligned} \tag{5.27}$$

Solving the discrete Lyapunov equation

$$A_N^{cl}{}' P A_N^{cl} - P + \sum_{i=0}^{N-1} \left\|A_i^{cl}\right\|_{\tilde{Q}}^2 + N \cdot k_f{}' R k_f + \gamma I_n = 0 \tag{5.28}$$

yields the terminal penalty weighting matrix P, which is also used to defined the terminal region using (5.24).

At this stage, Assumption 5.1-5.3 and condition (5.16) are fulfilled by properly choosing control parameters in the MPC design procedure, which is summarised as follows:

1. Determine the process weighting matrix Q and control weighting matrix R based on performance specifications.

2. Determine control holding horizon N according to the computation power and burden, and calculate the corresponding linearised system matrices A_i and B_i.

3. Design the terminal control gain k_f with respect to A_i and B_i, with eigenvalues of A_N^{cl} staying in the unit disc .

4. Determine γ and $\tilde{Q}$, and solve the Lyapunov equation (5.28) to obtain the terminal weighting matrix P.

5. Determine the terminal region radius α. For any states within the terminal region, check if the resulting terminal control and evolved states $\varphi_i(x, k_f x)$ stay in the corresponding constraints.

6. Check the maximum value of expression (5.16) with respect to all the states in the terminal region. If the value is larger than zero, reduce the radius α and repeat this process until its value is smaller than zero.

If Step 5 or Step 6 are failed or the resulting radius of the terminal region is too small, one needs to go back to tune the design parameters (usually by increasing γ or reducing N) to obtain a proper terminal control gain and terminal region that guarantee the stability of the piecewise constant MPC. The process outlined in [14] can be adopted to implement Step 5 and 6. This procedure can be further improved by adopting the method proposed in [16], where the terminal region can be maximised using an optimisation algorithm to search the best terminal penalty P and terminal control.

5.4.3 Stability of the two-level control framework

The control framework proposed for autonomous helicopters consists of two parts: high-level MPC and low-level linear controller. Although the stability of the high-level MPC and the low-level linear controller has been discussed separately, the stability of the overall system needs to be investigated. There are different sampling rates for the high-level and low-level controllers. The low-level linear controller acts at a high sampling rate even could be in a continuous-time fashion, so does the overall control signal applied to helicopters. It is convenient to analyse the stability of the overall system in the continuous-time domain.

In the absence of uncertainty and disturbance, the helicopter state $\mathbf{x}(t)$ always follows MPC trajectory $\mathbf{x}_m(t)$ driven by the control $\mathbf{u}(t) = \mathbf{u}_m(t)$. Therefore, the stability of the two-level control framework naturally follows the argument made for piecewise constant MPC in the last subsection.

In reality, within a MPC sampling interval $t \in [t_k, t_{k+1})$, the actual helicopter state $\mathbf{x}(t)$ may diverge from $\mathbf{x}_m(t)$ under the constant control $\mathbf{u}_m(t_k)$ due to uncertainties and/or disturbances. However, within a surrounding area of $\mathbf{x}_m(t)$, the error dynamics can be described by a linearised system (5.7), where the surrounding is defined as $\Omega(\mathbf{x}_m) = \{\mathbf{x} | \|\mathbf{x} - \mathbf{x}_m\| \leq \bar{e}\}$. When considering a bounded constant disturbance $\mathbf{d}$ and a low-level controller, the behaviour of the divergence $\mathbf{x}_\Delta$ can be modified from Eq.(5.8) as:

$$\Delta\dot{\mathbf{x}} = A_{cl}\Delta\mathbf{x} + E\mathbf{d} \tag{5.29}$$

where E is a disturbance input matrix. It is important to point out that for any perturbed state $\mathbf{x}(t) \in \Omega(\mathbf{x}_m)$, $t \in (t_k, t_{k+1})$, its divergence $\Delta\mathbf{x}(t)$ is bounded for all $t > 0$. As at the time $t = t_k$, the high-level MPC reset the error state to zero, i.e. $\Delta\mathbf{x} = 0$, the state response in the period $t \in (t_k, t_{k+1})$ can be written as:

$$\Delta\mathbf{x}(t) = A_{cl}^{-1}(e^{A_{cl}(t-t_k)} - I)E\mathbf{d} \tag{5.30}$$

Therefore, $\mathbf{x}_\Delta(t)$ is bounded by a limited value, which may depend on the magnitude of the disturbance. Furthermore, the high-level MPC resets the error state to zero at $t = t_{k+1}$, i.e. $\Delta\mathbf{x}(t_{k+1}^+) = 0$. After MPC synchronises the real helicopter state $\mathbf{x}$ and MPC state $\mathbf{x}_m$, the overall stability is guaranteed as MPC runs in an closed-loop fashion (with a long sampling time), rather than in an open-loop fashion. Therefore, the low-level controller improves the performance of the high-level MPC in the presence of uncertainties and disturbances, without scarifying the original MPC stability.

This phenomenon can be further explained in Fig 5.3, where the dash lines represent the MPC solutions, the dotted line is the real state of the system and the reference trajectory is plotted as the solid line on the top. At sampling instant $t_k = kT_s$ there is a trajectory $\mathbf{x}_m(t)$, $t_k \leq t \leq t_{k+1}$, yielded by the high-level MPC, towards the reference $\mathbf{x}_r$. If the disturbance occurs or there is model mismatching, there is discrepancy between the real helicopter and MPC trajectories, i.e. $\Delta\mathbf{x} \neq 0$. The low-level controller generates compensation controls to eliminate $\Delta\mathbf{x}$. Even if it cannot be reduced to zero within a short time period, at next MPC sampling point $t_{k+1} = (k+1)T_s$ the MPC measures the current state of the system and produces a new trajectory $\mathbf{x}_m(t)$, $t_{k+1} \leq t \leq t_{k+1+H}$ towards $\mathbf{x}_r$ based on the measured new state, and $\Delta\mathbf{x}(t_{k+1}^+)$ is reset to zeros automatically.

It shall note that the stability analysis presented in this section is to investigate the theoretical properties of the proposed MPC based control algorithm. It also shows that the two-level MPC framework has certain robustness against the bounded uncertainty. However, this theoretical investigation do not always provide guarantee for the stability of the actual control system implemented due to the significant model/plant mismatch. Careful design and tuning of control

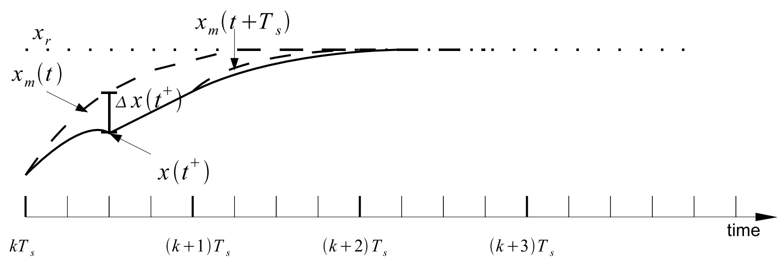


Figure 5.3: State trajectory under the disturbance

parameters will be involved in flight experiments.

5.5 Controller design

5.5.1 High-level MPC

The aims of MPC design are to choose appropriate weight matrices for the desired control performance, to determine the time settings for prediction accuracy and online solvability, and to find appropriate terminal weighting and constraints for MPC stability. After a number of trials, the MPC parameters used for simulation and flight tests presented in this chapter are given in Table 5.1.

Table 5.1: MPC design parameters

prediction horizon H	10
control holding horizon N	10
discretisation time T_d	0.02s
MPC sampling time T_s	0.2s
weighting matrix Q	$diag(\ 0.1\quad 0.1\quad 1\quad 1\quad 2\quad 2\quad 2\quad 2\ )$
weighting matrix R	$diag(\ 0.02\quad 0.02\ )$

The design procedure of the terminal weighting P, control k_f and region $\boldsymbol{\Omega}$ developed in Section 5.4 is illustrated by an example where only the lateral and longitudinal channels of helicopter dynamics are considered as they contain the majority of the helicopter's nonlinearity and avoid to illustrate the complicated full helicopter states. In this simplified model the state is defined as $\mathbf{x}_s = [x \quad y \quad u \quad v \quad \phi \quad \theta \quad p \quad q]^T$, the control input is $\mathbf{u}_s = [\delta_{lat} \quad \delta_{lon}]^T$, and the system is described by the corresponding equations in Eq.(5.1).

The auxiliary parameters used to determine the terminal region and control are as follows: $\tilde{Q} = 1.1 \cdot Q$, $\gamma = 0.2$ and $\alpha = 125$. The terminal penalty matrix P in Eq.(5.31) is calculated by solving the Lyapunov equation (5.28), and the terminal controller k_f in Eq.(5.32) is designed by using the Linear Quadratic Regulator (LQR) algorithm based on the linearised model.

$$P = \begin{bmatrix} 30.89 & 0.02 & 15.49 & 0.04 & 0.02 & -1.05 & 0.18 & -39.92 \\ 0.02 & 30.80 & 0.05 & 14.91 & 0.81 & -0.10 & 38.58 & -0.34 \\ 15.49 & 0.05 & 20.68 & 0.07 & 0.03 & -1.29 & 0.25 & -53.72 \\ 0.04 & 14.91 & 0.07 & 19.36 & 0.98 & -0.09 & 49.97 & -0.38 \\ 0.02 & 0.81 & 0.03 & 0.98 & 1.08 & -0.04 & 4.59 & -0.16 \\ -1.05 & -0.10 & -1.29 & -0.09 & -0.04 & 1.28 & -0.36 & 5.90 \\ 0.18 & 38.58 & 0.25 & 49.97 & 4.59 & -0.36 & 219.09 & -1.41 \\ -39.92 & -0.34 & -53.72 & -0.38 & -0.16 & 5.90 & -1.41 & 230.96 \end{bmatrix} \tag{5.31}$$

$$k_f = \begin{bmatrix} -0.01 & 0.07 & -0.04 & 0.21 & 0.04 & 0.00 & 1.51 & 0.2937 \\ -0.07 & -0.01 & -0.25 & -0.02 & -0.01 & 0.06 & -0.15 & 1.7024 \end{bmatrix} \tag{5.32}$$

The terminal region yielded by $\|x\|_P^2 < \alpha$ is in a high dimension space which cannot be illustrated directly. To show the properties of the terminal region and terminal control, it is assumed that the helicopter is in the hovering status and only has the initial position errors. In this case, since all states other than the position are zeros, the terminal region reduce to a constraint on the position:

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} 30.89 & 0.02 \\ 0.02 & 30.80 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \leq 125 \tag{5.33}$$

Starting from the boundary defined by (5.33), the position phase portrait plotted in Fig.5.4 is calculated by using the helicopter model under the terminal control. It can be observed that all position trajectories of the helicopter are driven towards zero by the terminal control, which suggests that the terminal region is an invariant set and Assumption 5.3 is fulfilled. During this process, the other states are also bounded and within the range of their state constraints as shown in Fig.5.5-5.7. Moreover, by focusing on one trajectory labelled by thick lines in phase portraits, the helicopter movement under the terminal control can be examined. In this trajectory, the helicopter starts from left rear of the origin, and needs to fly towards it. Hence, both the lateral and longitudinal velocities increase first and then decrease as shown in Fig.5.5. Consequently, the helicopter needs to roll right and pitch down to gain positive speed and then roll left and pitch up to reduce speed (see Fig.5.6 and 5.7).

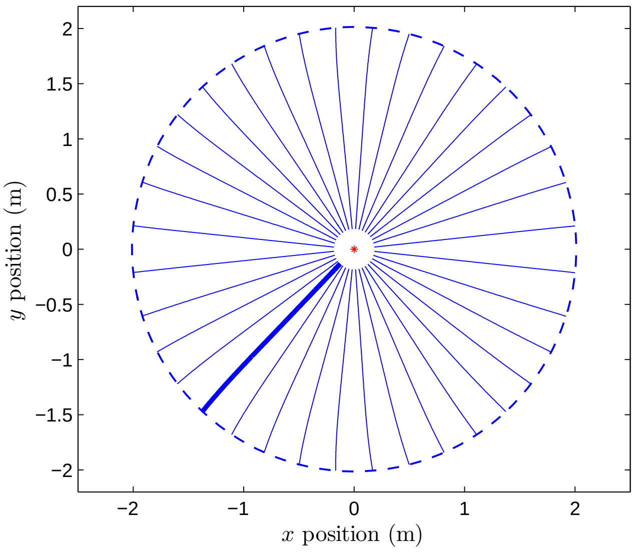


Figure 5.4: Position phase portrait

5.5.2 Low-level control

The task of low-level control is to regulate the error dynamic system (5.7). The error state is caused by uncertainties and disturbances, which cannot be fully suppressed by the high-level MPC due to its low bandwidth. Under the current flight test configuration and hardware including sensor, computing power and communication, the bandwidth associated with MPC can reach 10Hz, which is adequate for translational movements of a helicopter, but not fast enough for helicopter's attitude movements. Therefore, the low-level controller design focused on regulating the attitude tracking error.

As described in previous sections, the low-level controller is designed based on an error system obtained by linearisation around the nominal state and control generated by the high-level controller. Particularly for the attitude dynamics, the

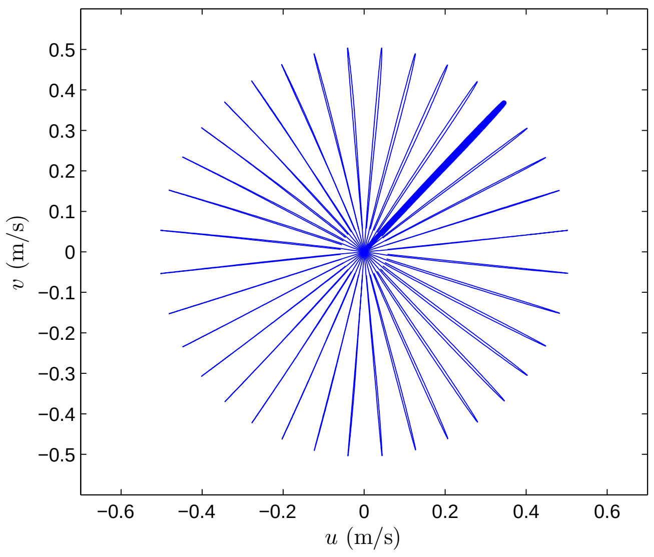


Figure 5.5: Velocity phase portrait

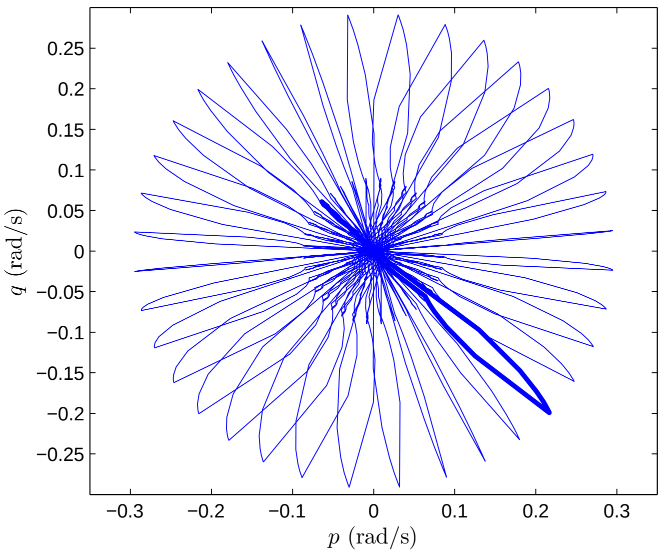


Figure 5.6: Angular rate phase portrait

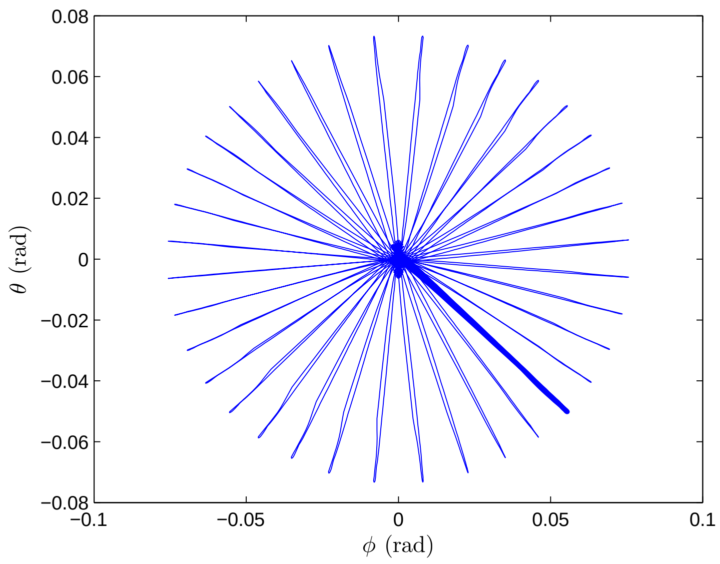


Figure 5.7: Attitude phase portrait

state is defined as $\mathbf{x}_a = [\phi \quad \theta \quad \psi \quad p \quad q \quad r]^T$, the control is $\mathbf{u}_a = [\delta_{lat} \quad \delta_{lon} \quad \delta_{ped}]^T$ and matrices in error system (5.7) can be specified as:

$$A_m(\mathbf{x}_m) = \begin{bmatrix} a_{11} & a_{12} & 0 & 1 & a_{15} & a_{16} \\ a_{21} & a_{22} & 0 & 0 & 1+a_{25} & a_{26} \\ a_{31} & a_{32} & 0 & 0 & a_{35} & 1+a_{36} \\ 0 & 0 & 0 & -33.42 & -3.76 & 0 \\ 0 & 0 & 0 & -0.49 & -24.90 & 0 \\ 0 & 0 & 0 & 0 & 0 & -23.98 \end{bmatrix}, \tag{5.34}$$

and

$$B_m = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 71.59 & -5.39 & 0 \\ 5.93 & 48.24 & 0 \\ 0 & 0 & 113.65 \end{bmatrix} \tag{5.35}$$

where some elements are depended on the reference state $\mathbf{x}_m$ from high-level

MPC, such as

$$\begin{aligned}
a_{11} &= (q\cos\phi - r\sin\phi)\tan\theta, & a_{12} &= (q\sin\phi + r\cos\phi)\sec^2\theta \\
a_{15} &= \sin\phi\tan\theta, & a_{16} &= \cos\phi\tan\theta \\
a_{21} &= -q\sin\phi, & a_{22} &= -r\cos\theta \\
a_{25} &= \cos\phi - 1, & a_{26} &= -\sin\theta \\
a_{31} &= (q\cos\phi - r\sin\phi)\sec\theta, & a_{32} &= (q\sin\phi + r\cos\phi)\tfrac{\sin\theta}{\cos^2\theta} \\
a_{35} &= \sin\phi\sec\theta, & a_{36} &= \cos\phi\sec\theta - 1
\end{aligned} \tag{5.36}$$

To avoid investigating the complicated relationship in the state matrix $A(\mathbf{x}_m)$ during the control design, this matrix is converted into a form that depends affinely on parameters α_i, $i = 1, \cdots, 12$, such that

$$A(\mathbf{x}_m) = A(a) = A_0 + \alpha_1 A_1 + \cdots + \alpha_{12} A_{12} \tag{5.37}$$

where A_0 is the nominal state matrix obtained from the hovering statue, A_i, $i = 1, \cdots, k$, $k = 12$ are known constructive matrices and $\alpha = (\alpha_1, \cdots, \alpha_k)$, $k = 12$, is a vector of uncertain and varying parameters, whose values are determined according to Eq(5.36). As state $\mathbf{x}_m$ from high-level MPC is bounded, the bounds on each uncertainty parameter also can be found, i.e. $\alpha_i \in [\underline{\alpha}_i, \overline{\alpha}_i]$. This means that the parameter vector α takes values in a hyper-rectangle called the parameter box. In the sequel,

$$\mathcal{V} := \{(v_1, v_2, \ldots, v_k) : v_i \in \{\underline{\alpha}_i, \overline{\alpha}_i\}\} \tag{5.38}$$

denotes the set of 2^k vertices or corners of this parameter box. After the above processes, the error system is then categorised into a linear parameter varying (LPV) system $\Delta\dot{\mathbf{x}}_a = A(\alpha)\Delta\mathbf{x}_a + B\Delta\mathbf{u}_a$ as Eq.(5.8).

In this way, a feedback controller can be designed based on the nominal matrix A_0, as long as it is robust enough for any parameters within the parameter box. Suppose there is a static feedback gain K, the closed-loop system for attitude error dynamics can be written as

$$\Delta\dot{\mathbf{x}}_a = A_{cl}(\alpha)\Delta\mathbf{x}_a \tag{5.39}$$

where $A_{cl} = A_0 - BK + \alpha_1 A_1 + \cdots + \alpha_{12} A_{12}$. After the control gain is designed, a systematic way is required to examine the robustness of the closed-loop system against any values in the parameter box. Accordingly, the following lemma is adopted in this study:

Lemma 5.1 ([38]). *Consider the linear uncertain system (5.39) and an affine quadratic Lyapunov function*

$$P(\alpha) = P_0 + \alpha_1 P_1 + \cdots + \alpha_k P_k \tag{5.40}$$

The continuous-time system (5.39) is affine quadratically stable if $A_0 - BK$ is stable and there exist $k+1$ symmetric matrices $P_0, P_1, \ldots, P_k$, such that $P(\alpha) > 0$ satisfies

$$L(v) = A^T(v)P(v) + P(v)A(v) + \sum_{j=1}^{k} v_j^2 M_j < 0 \tag{5.41}$$

for all $v \in \mathcal{V}$ and

$$A_j^T P_j + P_j A_j + M_j \geq 0, j = 1, 2, \ldots, k \tag{5.42}$$

where, $M_j = M_j^T \geq 0$ are some positive semidefinite matrices.

The feedback gain K is calculated by using LQR technique, and then the robustness of the closed-loop system is evaluated by solving LMIs (Linear Matrix Inequality) in Lemma 5.1. The bounds on the uncertain parameters can be calculated based on their definition in (5.36) subjected to state constraints given by the high-level MPC. Since the state constraints are ± 1, ± 1, $\pm\pi/6$ and $\pm\pi/6$, on q, r, ϕ and θ, respectively, the parameter box is defined by

$$\begin{aligned}
&a_{11} \in [-0.8165, 0.8165], && a_{12} \in [-1.8856, 1.8856] \\
&a_{15} \in [-0.2887, 0.2887], && a_{16} \in [-0.5774, 0.5774] \\
&a_{21} \in [-0.5, 0.5], && a_{22} \in [-1, 1] \\
&a_{25} \in [-0.0670, 0.0670], && a_{26} \in [-0.5, 0.5] \\
&a_{31} \in [-1.6330, 1.6330], && a_{32} \in [-0.9428, 0.9428] \\
&a_{35} \in [-0.5774, 0.5774], && a_{36} \in [-0.1444, 0.1444]
\end{aligned} \tag{5.43}$$

After several trials, the weighting matrices for LQR design are determined as $Q = \text{diag}\{10, 10, 40, 0.1, 0.1, 0.1\}$ and $R = \text{diag}\{0.1, 0.1, 0.1\}$, and the robust low-level gain K that satisfies Lemma 5.1 is:

$$K = \begin{bmatrix} 7.02 & 0.81 & -0.00 & 0.49 & 0.03 & 0.00 \\ -0.81 & 7.02 & -0.00 & -0.06 & 0.51 & 0.00 \\ 0.00 & 0.00 & 11.8 & 0.00 & 0.00 & 0.65 \end{bmatrix} \tag{5.44}$$

5.6 Simulation

Before the real flight test, simulations are first carried out. Since there is no disturbance in numerical simulations, the high-level MPC along is able to control the helicopter. The aims of numerical simulations are to investigate the computational attributes of the proposed MPC scheme, and to compare with the conventional MPC.

One simulation is to track a square trajectory containing sharp 90° turns, which poses extra burdens on the OP solver as it has to replan a smooth trajectory to adapt to helicopter's dynamics. The time related settings for piecewise constant MPC are $T_d = 0.02$s, $H = 10$, and $N = 5$. Therefore, the MPC sampling time is $T_s = 0.1$s and the prediction length is 1s. On the other hand, as $N = 1$ in a conventional MPC, one has to increase H to 50 steps to cover the same prediction length.[1] The settings for the OP solver remain the same for both scenarios and the full helicopter dynamics are used in prediction.

The simulation is performed on the computer with a 2.4 GHz CPU and 2GB memory, where the OP is solved by the KNITRO solver [43]. From Fig.5.8, it can be seen that the piecewise constant and conventional MPC gives almost the same tracking performance. This observation is also supported by the integrated squared error (ISE) performance indexes calculated form the simulation as shown in Table.5.2. Nevertheless, the computational burdens in two MPC schemes are quite different. Fig.5.9 compares the computation time spent at each sampling instant along the simulation time. It is shown that in the piecewise constant scheme the calculation time is around 0.05s and the maximal value is below the sampling interval suggesting that it is suitable for online execution. In contrast, the conventional MPC scheme needs more time to solve the OP due to more variables (200 instead of 40 in the piecewise constant scheme) need to be handled,

[1]The reason of using prediction length of 1 second instead of 2 seconds in this comparison is because in the latter case conventional MPC requires handling 400 variables, which exceeds the ability of the OP solver.

which means it has to scarify the control bandwidth or the prediction horizon in order to be applied on helicopter control.

Table 5.2: ISE index performance comparison

States	Piecewise Constant MPC	Conventional MPC
x	3.8226	3.7715
y	3.6170	3.5458
z	0.0016	0.0015

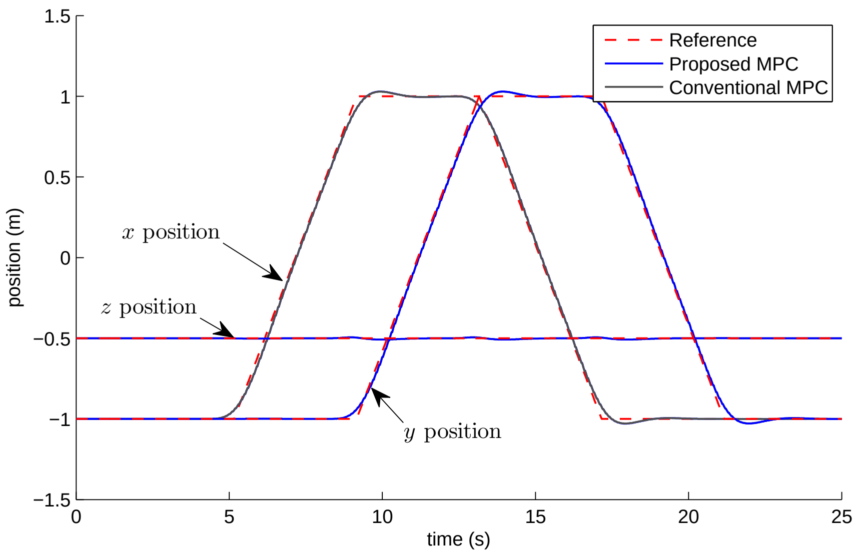


Figure 5.8: Square tracking

5.7 Flight experiment

The flight tests are executed on in the indoor flight testbed described in Chapter 3. To realise the proposed two-level control framework with online optimisation function, a multi-computer configuration is adopted in the ground station. One computer is used to achieve real-time control, whereas another one is dedicated for online optimisation. The communication between the two computers relies on LAN with UDP/IP protocol. The testbed configuration is given in Fig.5.10

Many flight tests have been carried out in our flight testbed to verify the proposed controller in different scenarios, one of which presented in this section

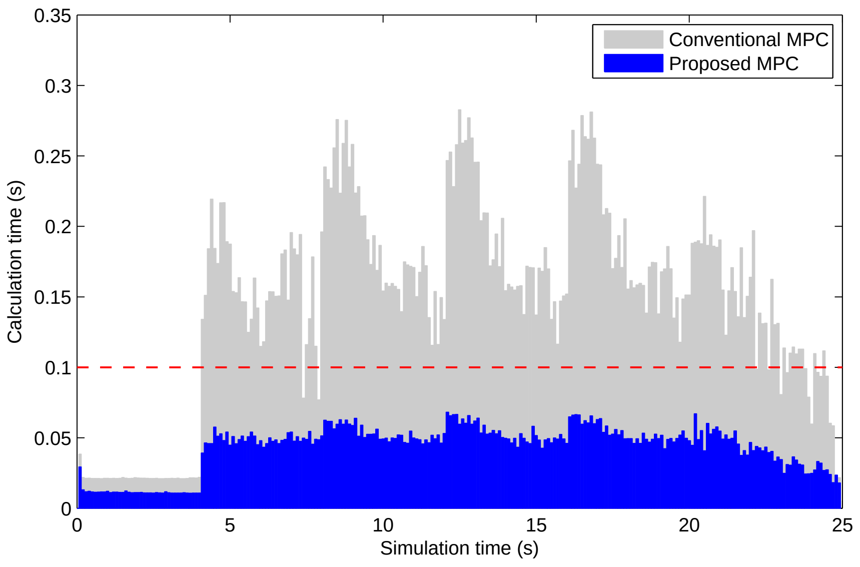


Figure 5.9: Computational time

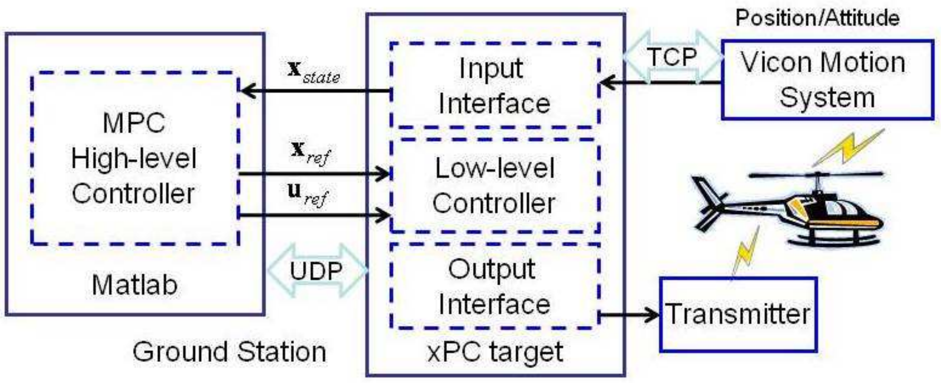


Figure 5.10: Testbed configuration with online optimisation

is to execute the same flight pattern used in the simulation that tracks a square trajectory with 2m length. The reference progresses at a constant speed of 1m/s, so the helicopter needs to complete the whole manoeuvre in 8 seconds. Moreover, this reference requires the helicopter starts from stationary at one corner and finishes in stationary at the next corner and then keeps going. Such a trajectory is dynamically infeasible for helicopters, but it is deliberately used to demonstrate the prediction feature of MPC, which uses online optimisation to generate a smooth trajectory allowing the helicopter to fly along the reference as close as possible and keep stable.

The tracking result is shown in Fig.5.11 in a 3 dimensional plot. In the flight test the helicopter was controlled to hover at the start point first and started to track after 40s. During this process, the roll angle and pitch angle are cooperated to increase the translational speed at one direction and decrease at another as shown in Fig.5.12. Note that a positive roll angle gives a positive lateral acceleration and a positive pitch angle generates a negative longitudinal acceleration, vice versa. The corresponding control signals are provided in Fig.5.13, where the baseline control from the high-level is plotted in solid line, whereas the overall control is given in dash line. It can be observed that the high-level MPC gives the basic trend of the control signal and the low-level controller adds compensations on it to achieve the required control performance.

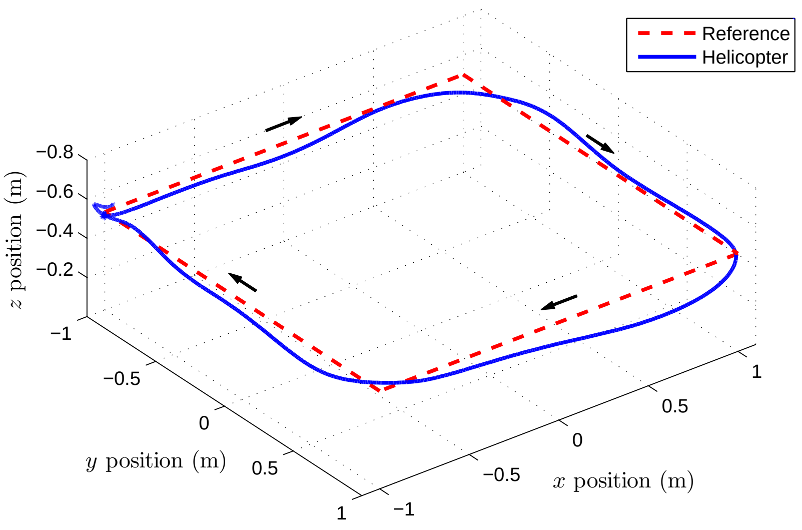


Figure 5.11: Flight tracking result

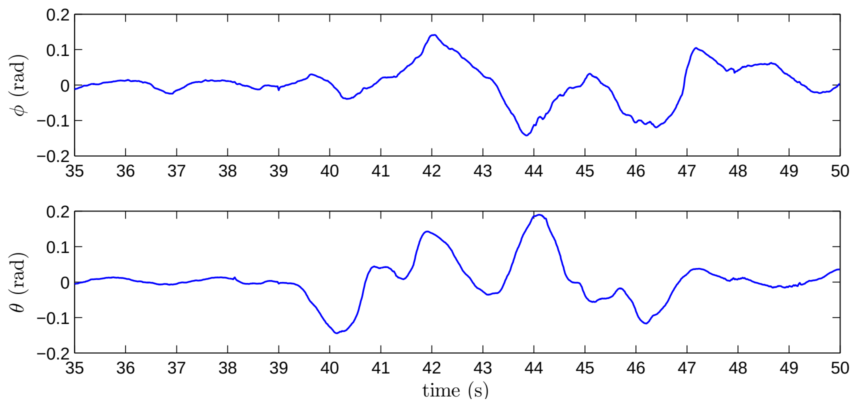


Figure 5.12: Attitude angles

The flight test demonstrates a good tracking performance under the proposed two-level control scheme except for a small steady state error. The steady state error is introduced by untrimmed helicopter dynamics, and it can be eliminated by carefully trimming or modifying the current control to incorporate an integral action. It shall be reminded that control of a small-scale helicopter is even more difficult than a large one as small ones are quite sensitive to any wind gust and turbulence, and any small change in helicopter structure and propulsion systems. To this end, a good robustness of the proposed scheme has been clearly demonstrated in the flight tests. The model used in MPC online calculation is simplified and the parameters are estimated through system identification. There are certainly mis-match between the model and the real helicopter dynamics.

5.8 Algorithm modification

In previous sections, a two-level online optimisation based control framework is established to address the flight control of small helicopters. The computational burden in MPC is reduced by a piecewise constant scheme and the overall control bandwidth is increased by introducing a low-level linear controller. A very satisfactory performance of proposed framework has been demonstrated through both simulations and experiments. However, by investigating the process of execution of such a control scheme, the existing control framework can be further improved.

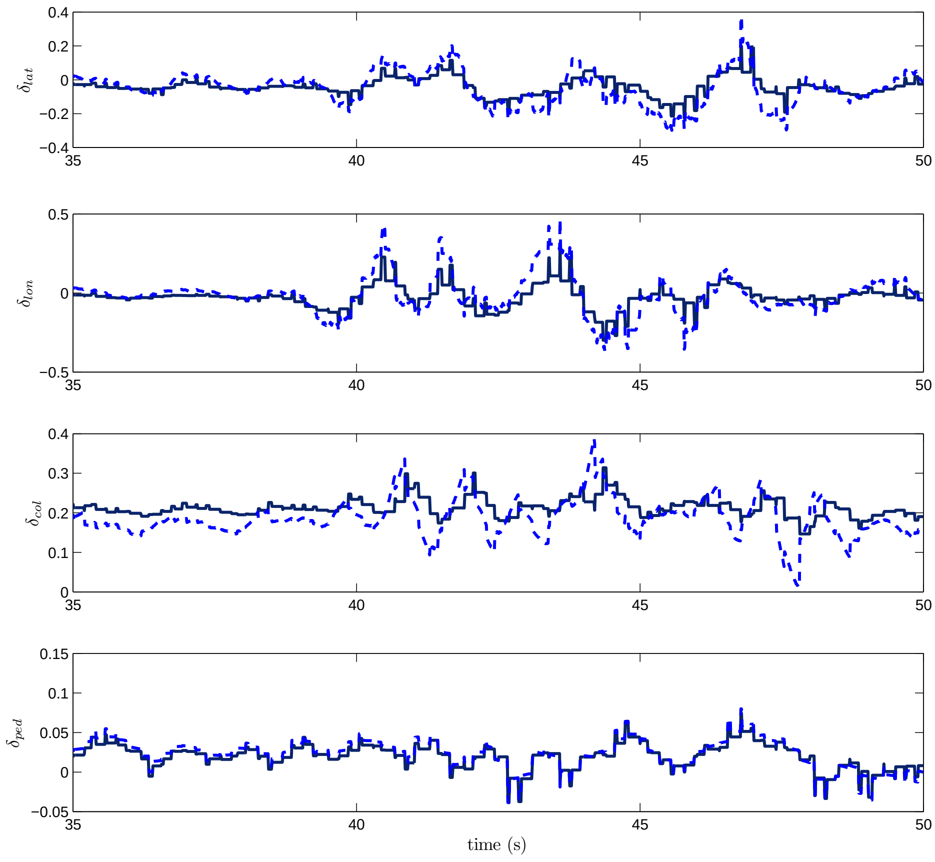


Figure 5.13: Control signals: High-level MPC control (solid line) and overall control with low-level compensation (dotted line)

5.8.1 High-level MPC with computational delay

Computational delay is another issue in the implementation of MPC algorithms. Unlike conventional control techniques, the control signal in MPC is not instantaneously available after the plant state is updated. An online optimisation takes time to generate the new control signals, which causes computational delay. In the proposed piecewise constant MPC, although the computation load is reduced, the time spend on online calculation is still not negligible.

Computational delay degrades the desired control performance, as depicted by an example in Fig 5.14. A general continuous MPC strategy is considered in this case. At sampling instant t, the system state $x(t)$ is measured. Online optimisation is then carried out to produce the control profile $u(\tau; x(t))$, $\tau \in [t, t+T]$, where T is the prediction horizon. In fact, this control is not available to the system until time $t + \delta$, where δ is the computational delay. During the calculation period $[t, t + \delta]$, the system is under the control of previous control profile $u(\tau; x(t-\delta))$, $\tau \in [t-\delta, t-\delta+T]$ (there is always an exception for the first step). In Fig.5.14, the desired control is plotted in dotted lines and then in solid line after available, whereas the actual control applied using thick solid lines. It can be seen that the desired control profile cannot be fully applied to systems, and it makes no sense to optimise the control profile $u(\tau; x(t))$,$\tau \in [t, t + \delta]$ at sampling instant t, since this part of control will never be applied. The above phenomena also results in state mismatching in both online predictions and state references for low-level control, as the predicted state and the real state at time $t + \delta$ are evolved from $x(t)$ under different controls.

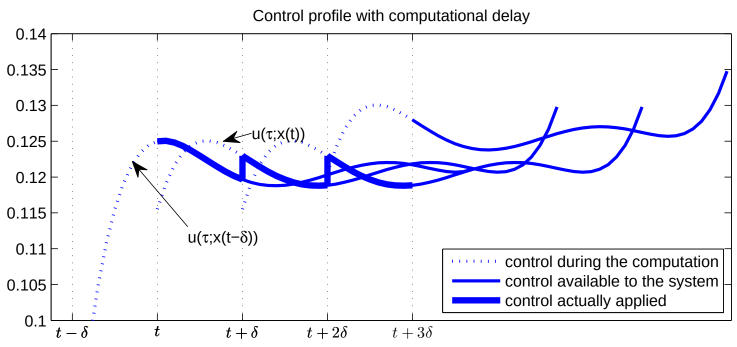


Figure 5.14: Control profile with computational delay

In order to take into account the computational delay and improve the control performance, a modified objective function for helicopter autonomous flight is adopted. By defining the reference trajectory as x_r and the tracking error $x_e = x_r - x$, we can reformulate the objective function to be minimised [19]:

$$\begin{aligned} J(t) =& \hat{x}_e(t+T)' P \hat{x}_e(t+T) + \int_{\delta}^{T} \hat{x}_e(t+\tau)' Q \hat{x}_e(t+\tau) \\ &+ \hat{u}(t+\tau)' R \hat{u}(t+\tau) d\tau \end{aligned} \tag{5.45}$$

where P, Q and R are the positive definite weighting matrices. Note that the integration starts from state $\hat{x}(t+\delta)$, which can be calculated by using current state $x(t)$ and previous control $u(\tau; x(t-\delta))$, $\tau \in [t, t+\delta]$. In this modified setting, computational delay becomes an important parameter that has to be determined beforehand. Although it is known that the computational time for a nonlinear optimisation problem is not constant depending on initial guesses, optimum and other factors, the maximum calculation time can be determined when the optimisation process can be terminated with a suboptimal solution. This suboptimal solution is accepted as it still guarantees the MPC stability as discussed in previous sections.

In the piecewise constant scheme, the MPC sampling time T_s can be set equivalent to the maximum computational delay δ, so that the modified objective function can be written as:

$$\begin{aligned} J(k) =& \hat{x}_e(k+HN)' P \hat{x}_e(k+HN) + \\ & \sum_{i=1}^{H-1} \sum_{j=0}^{N-1} \hat{x}_e(k+iN+j)' Q \hat{x}_e(k+iN+j) + \\ & \hat{u}(k+iN+j)' R \hat{u}(k+iN+j) \end{aligned} \tag{5.46}$$

Therefore, the nonlinear optimisation problem that needs to be solved at sampling instant k can be stated as:

$$(x_m,\, u_m) = \arg\min_{\hat{\mathbf{x}},\, \hat{\mathbf{u}}} J(k) \tag{5.47}$$

subject to:

$$\begin{aligned}
&\hat{x}(k+j+1) = f(\hat{x}(k+j),\, \hat{u}(k+j)) \\
&\hat{x}(k+j) \in \mathbb{X} \\
&\hat{u}(k+j) \in \mathbb{U} \\
&j = N,\, N+1,\, \cdots, HN-1 \\
&\hat{x}(k+N) = x(k+N)
\end{aligned}$$

where $x(k+N)$ is calculated from the currently measured state $x(k)$ under the control $u(k; x(k-N))$ calculated at previous sampling instant, and the rest of parameters follows the definition in optimisation problem (5.5).

After the high-level MPC is modified, the way it provides state references is also changed. Owing to the computational delay, from sampling instant $t = kT_s$ to $t + \delta = (k+1)T_s$, the reference is replaced by the system evolution from the current state $x(t)$ under the previous control, such that:

$$x_m(\tau; x(t)) = \begin{cases} \hat{x}(\tau; x_m(t); u_m(t-\delta)), & \tau \in [t, t+T_s] \\ x_m(\tau; x_m(t); u_m(t)), & \tau \in [t+T_s, t+HT_s] \end{cases} \tag{5.48}$$

The benefit of this strategy is that the newly defined state reference (5.48) is smoother for the low-level controller to track comparing to directly using the previous optimised state, while the latter one causing state mismatching and low-level control signal jumping.

The improvement from the modified MPC formulation that takes into account computational delay can be verified by real-time simulations. Different from purely numeric simulations, real-time simulations are carried out on the flight test platform except that the real helicopter is replaced by a mathematical model. In this way, simulations are attached to the real time frame and computational delay is naturally embedded. A step response test is performed to compare the MPC formulations with and without computational delay in consideration in terms of helicopter flight control. The controller settings follow those in previous flight tests, so the computational delay is set to a fixed values of 0.2 second equal to the MPC sampling interval.

The position responses of the helicopter are given in Fig.5.15. It can be seen that when computational delay is taken into account the responses are smoother and have less oscillations. On the other hand, although proposed piecewise constant MPC is affected by computational delay, it still delivers a satisfactory performance due to the presence of low-level controller, which treats the mismatching

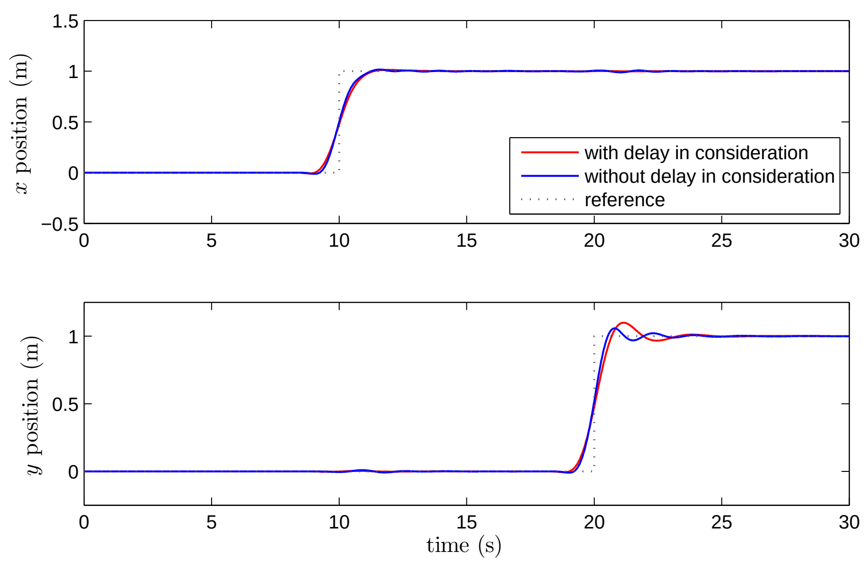


Figure 5.15: Position step responses with and without computational delay in MPC formulation (Numerical simulation)

in state references as the effect of disturbances and compensate for them. This is reflected from the differences between control signals provided by those two MPC schemes in Fig.5.16 and 5.17. In the former one, the control signals are smooth and low-level controller almost takes no actions since there are no external disturbances and artificial uncertainties. In contrast, when computational delay influences the real-time implementation, the low-level controller has to compensate the mis-matching occurred every time when new MPC control signals and corresponding state references are available.

5.9 Conclusion

Development of a control system to support autonomous flight of helicopters is very challenging as helicopters are unstable, highly nonlinear and exhibit fast dynamics. This chapter proposes a MPC based control framework for autonomous flight of small-scale helicopters. The framework has two levels of controls including a high-level MPC and low-level linear feedback control. The MPC works in a piecewise constant fashion to reduce the computation burden and to increase the time available for real-time optimisation. The linear feedback control responds to fast dynamics of the helicopter in the presence of disturbances and model mis-

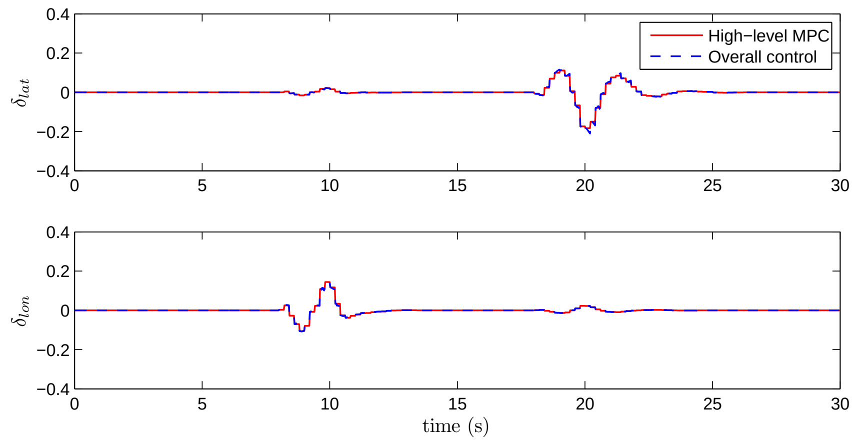


Figure 5.16: Control signals with computational delay in MPC formulation (Numerical simulation)

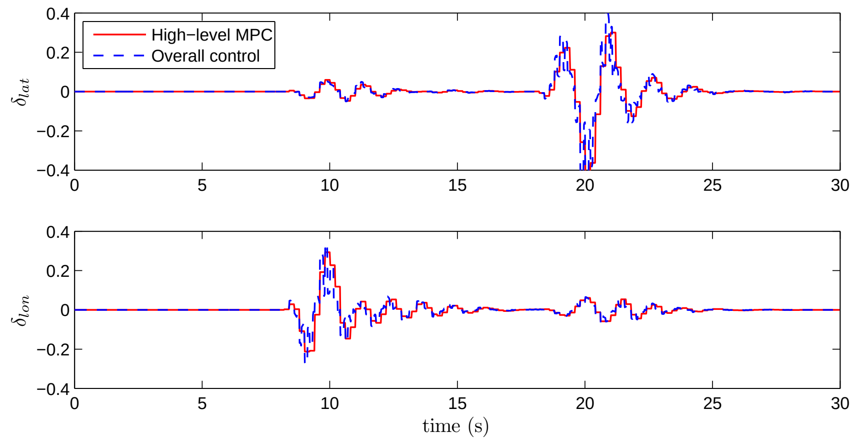


Figure 5.17: Control signals without computational delay in MPC formulation (Numerical simulation)

matching and compensate the low bandwidth of the high-level control due to the adoption of the piecewise constant control policy. With this configuration, it is possible to implement nonlinear MPC algorithms in system with fast dynamics such as helicopters. The stability issue of the high-level MPC and the overall control scheme are discussed and the design procedure is provided. The overall control framework was successfully tested on a Trex-250 helicopter through various flight experiments, and very satisfactory performance has been demonstrated. In the flight experiments, the proposed framework also shows the local path planning ability, which will be further explored in the next chapter.

Chapter 6

Local path planning using MPC techniques

6.1 Introduction

Local path planning aims at generating an obstacle-free trajectory that is dynamically feasible for an UAV to track and meanwhile leads an UAV to its global trajectory. Local planning does not concern the global goal as it can be achieved by following a reference provided by a higher level planner. It needs to re-plan a feasible local trajectory according to surroundings, aircraft dynamics and global references, so as to react to newly detected obstacles, and to guide the UAV back to the global reference after avoidance manoeuvres or perturbed by strong gusts. Moreover, local planning needs to cooperate with the flight control system so that the re-planned trajectory can be accurately followed. The "foresee" feature of MPC makes it as a very suitable strategy for both local path planning and control, because it takes into account the future values of references and information of surrounding area.

In the last chapter, a piecewise constant MPC framework is proposed mainly for tracking control of an autonomous helicopter. It utilises the full dynamic model for predictions so that the control signals can be directly applied to the helicopter to achieve high performance manoeuvres. However, when obstacles or other environment information is taken into account in the prediction, the computational load in online optimisation will increase dramatically. In this case, the benefits of using full dynamic model may be overweighted by its disadvantages, such as low update rate, short prediction horizon and potential loss of control due to a failure of optimisation.

This chapter, however, proposes a hierarchical planning and control framework for autonomous helicopters flying in a planar mode. First, local trajectory re-planning, required by obstacle avoidance and dynamical feasibility, is managed by a planning layer where the nonlinear MPC framework from the previous chapter is adopted. A kinematic model is incorporated into the MPC formulation to represent the motion characteristics of a helicopter and a potential field method is included to realise the obstacle avoidance. Furthermore, a novel linear time varying control law is developed as a low-level control in the MPC framework. As the planning layer only considers the kinematics, it provides the desired velocities and heading rate as guidance commands that the helicopter needs to track.

Next, after the desired velocity and heading rate are determined by the planning layer, a flight control layer is used to stabilise the helicopter and track the guidance commands. In this layer, the helicopter dynamics is approximated by a linear model, which shows a good fidelity in the normal flight mode. A linear MPC controller is designed based on this model to achieve tracking control. The main reason of using MPC in the control layer is because it can handle the constraints so that the normal flight mode can be guaranteed as well as the fidelity of the linear model.

The key component in the proposed hierarchical framework is the time varying control in the low-level of nonlinear MPC framework. It is designed based on the kinematics of the helicopter, so it acts as a guidance controller and serves as a link between the path planning and flight control.

6.2 Kinematic model and dynamic model

The kinematic model used to describe the planar motion of a helicopter can be extracted from the full kinematic model by assuming small pitch and roll angles, such that:

$$\begin{aligned} \dot{x} &= u\cos(\psi) - v\sin(\psi) \\ \dot{y} &= u\sin(\psi) + v\cos(\psi) \\ \dot{\psi} &= \omega \end{aligned} \tag{6.1}$$

where (x, y) is the position of the helicopter, ψ is the heading angle, (u, v) is the longitudinal and lateral velocities, respectively, and ω is the heading rate which is equivalent to yaw rate r in planar flight. In the path planning configuration, the state is defined as $\mathbf{x} = [x, y, \psi]^T$ as well as the output $\mathbf{y}$, whereas the input is $\mathbf{w} = [u, v, r]^T$, which is also the commands sent to the next layer to be followed. In the normal planar flight mode, u represents the forward speed of the helicopter, whereas v is the side slip velocity that should be eliminated.

Remark 6.1. A widely used unicycle model for fixed-wind aircraft can be derived from (6.1) by setting $v = 0$. This is because fixed-wind counterparts do no have the ability to move laterally like helicopters. The difference lies in that the model (6.1) enables the hovering flight mode when speed u or v is small.

For the purpose of flight control design, a linear model for hovering condition is well accepted for capturing non-aggressive flight [32; 92]. Such a model can be linearised from the full dynamic model developed in Chapter 4 as follows:

$$\begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} + \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} \tag{6.2a}$$

$$\mathbf{z}_1 = C_1 \mathbf{x}_1 \tag{6.2b}$$

$$\mathbf{z}_2 = C_2 \mathbf{x}_2 \tag{6.2c}$$

where the state variables $\mathbf{x}_1 = [u, v, \phi, \theta, p, q]^T$, $\mathbf{x}_2 = [z, w, r]^T$, control inputs $\mathbf{u}_1 = [\delta_{lat}, \delta_{lon}]^T$, $\mathbf{u}_2 = [\delta_{col}, \delta_{ped}]^T$, output variables $\mathbf{z}_1 = [u, v]^T$, $\mathbf{z}_2 = [r, z]^T$, and the elements in state and control matrices are given in Appendix A. It can been seen that the linear model is decoupled into two sub-systems: $\dot{\mathbf{x}}_1 = A_1\mathbf{x}_1 + B_1\mathbf{u}_1$ for describing longitudinal/lateral motions and $\dot{\mathbf{x}}_2 = A_2\mathbf{x}_2 + B_2\mathbf{u}_2$ for heaving/heading motions.

6.3 Path planning and flight control

The schematic diagram of the control structure for autonomous helicopters is shown in Fig 6.1, where it is hierarchically divided into three layers. The top layer is the global planning that provides the reference $\mathbf{y}_{ref} = [x_{ref}, y_{ref}, \psi_{ref}]^T$ but its function is beyond the scope of this chapter. The middle layer is the local path planning. Given the global reference $\mathbf{y}_{ref}$ and the obstacle positions, the local planner generates a desired local path $\mathbf{y}_o$ and its corresponding guidance command $\mathbf{w}_c$ by taking into account the kinematic model of the helicopter. It is of importance to include the kinematics and impose movement constraints to generate a smooth and feasible trajectory. The bottom layer is the flight control which is used to enable the stability of the helicopter motion with respect to the surrounding air and track guidance commands from local planner.

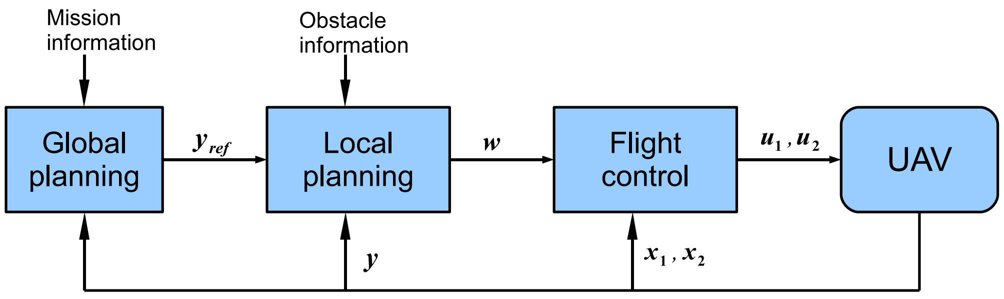


Figure 6.1: Structure of the overall autonomous flight control

6.3.1 MPC planner

The MPC framework discussed in Chapter 5 can be used to govern the kinematic system (6.1) for the path planning purpose. The time setting follows the piecewise constant scheme, so there are discretisation time T_d, control holding horizon N and prediction horizon H. The optimisation problem that MPC solves at each sampling instant can be stated as:

$$\mathbf{y}_o, \mathbf{u}_o = \arg\min_{\hat{\mathbf{x}}, \hat{\mathbf{u}}} J(k) \tag{6.3}$$

subject to:

$$\hat{\mathbf{x}}(k+j+1) = f(\hat{\mathbf{x}}(k+j), \hat{\mathbf{u}}(k+j)) \tag{6.4a}$$
$$\hat{\mathbf{u}}(k+j) \in \mathbb{U} \tag{6.4b}$$
$$j = 0, 1, \cdots, N-1 \tag{6.4c}$$
$$\hat{\mathbf{x}}(k) = \mathbf{x}(k) \tag{6.4d}$$

where $\mathbf{y}_o = \mathbf{x}_o$ is the desired local trajectory, $\mathbf{u}_o$ is the corresponding optimal input, $J(k)$ is the cost function to be minimised, Eq.(6.4a) is the discrete representation of kinematics (6.1) and $\mathbb{U}$ is the control input constraint. The hat variables is to distinguish the states in prediction from the true states. Note that although only movements in horizontal plane is considered in this formulation, the concept can be extended to 3 dimensions.

As in the local planning, multiple objectives are considered and the corresponding cost function $J(k)$ at time index k is constructed as

$$J(k) = J_f(k+HN) + \sum_{i=0}^{H-1}\sum_{j=0}^{N-1} J_{tk}(k+iN+j) + J_{obs}(k+iN+j) + J_u(k+iN+j) \tag{6.5}$$

where J_f is the terminal penalty, J_{tk}, J_{obs} and J_u are stage cost for tracking, obstacle avoidance and control effort, respectively. These cost functions, except for obstacle avoidance term J_{obs}, can be defined in quadratic forms:

$$J_f = \|\mathbf{y}(k+HN) - \mathbf{y}_{ref}(k+HN)\|_P^2 \tag{6.6}$$

$$J_{tk} = \|\mathbf{y}(k+iN+j) - \mathbf{y}_{ref}(k+iN+j)\|_Q^2 \tag{6.7}$$

$$J_u = \|\mathbf{u}(k+iN+j)\|_R^2 \tag{6.8}$$

where P, Q and R are positive definite matrices. Eq.(6.6) and (6.7) penalise the deviation from the reference along the prediction horizon, and the term of (6.8) penalises the control effort.

On the other hand, the cost penalty J_{obs} may consist of several contributors such as $J_{obs} = \sum_{i=1}^{n} J_{obs}^i$, where $n = 1, 2, \cdots$ is the number of obstacles being considered. For each obstacle, the penalty cost can be provided by a potential function like a Yukawa function:

$$J_{obs}^i = \beta \frac{e^{\alpha d_i}}{d_i + \epsilon}, \quad i = 1, \ldots, n, \tag{6.9}$$

where β is a scaling factor, α is the decay rate of the potential field, d_i is the distance between the helicopter and the nearest point on the i-th obstacle, and ϵ is a small positive scalar to prevent singularity.

An example of potential field around a point obstacle is shown in Fig.6.2, where it can be seen that the penalty cost approaches infinity as the distance to the obstacle gets close to zero. Acceptable safe clearance distance can be defined using the potential field design parameters α and β. For a detected obstacle with location (x_{obs}, y_{obs}) and a safety distance r_{obs}, the distance d_i can be calculated as $d_i = \sqrt{(x - x_{obs})^2 + (y - y_{obs})^2} - r_{obs}$. By incorporating the potential term Eq.(6.9), the overall cost function (6.5) can be seen as a trade-off performance index for tracking a predefined reference and diverging from obstacles while minimising the control effort.

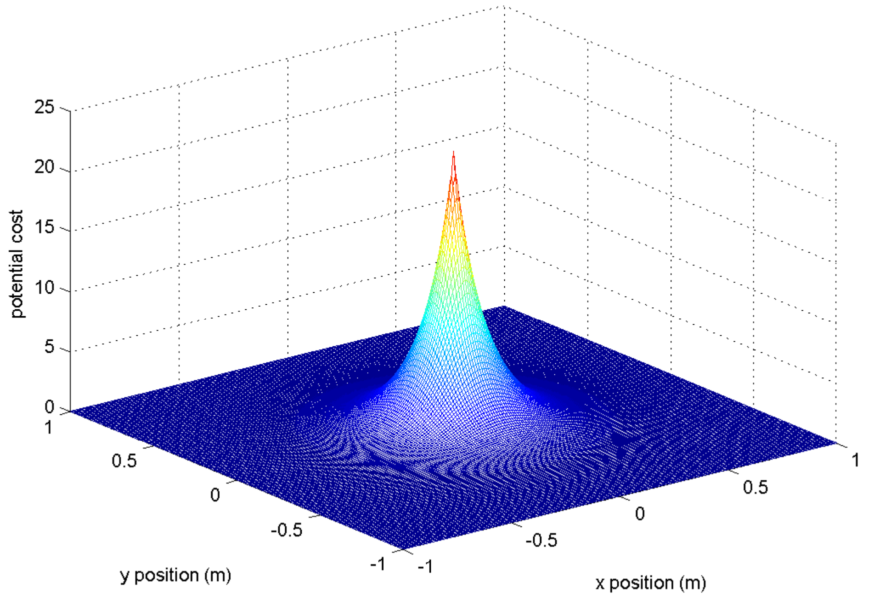


Figure 6.2: Potential field about a point at $(0, 0)$

6.3.2 Two-level guidance framework

Although only a kinematic model is used in optimisation, the inclusion of obstacle information significantly increases the computational load, especially in an obstacle rich environment. Moreover, as obstacles appear in an unexpected manner, it is of importance to allocate enough time for online optimisation. As a result, the update rate, i.e. the bandwidth, of the MPC planner is restricted. Therefore, it is necessary to introduce a guidance compensator as the low-level controller in the nonlinear MPC framework by following the same principle as in Chapter 5.

The linearised system of Eq.(6.1) around the operation point determined by

MPC planner's outcome $\mathbf{y}_o$ and $\mathbf{w}_o$ is given as:

$$\Delta\dot{\mathbf{x}} = A_o\Delta\mathbf{x} + B_o\Delta\mathbf{u} \tag{6.10}$$

where $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_o = [\ \Delta x \quad \Delta y \quad \Delta\psi\ ]^T$, $\Delta\mathbf{u} = \mathbf{w} - \mathbf{w}_o = [\ \Delta u \quad \Delta v \quad \Delta r\ ]^T$, state transition matrix

$$A_o = \begin{bmatrix} 0 & 0 & -u_o\sin(\psi_o) - v_o\sin(\psi_o) \\ 0 & 0 & u_o\cos(\psi_o) - v_o\sin(\psi_o) \\ 0 & 0 & 0 \end{bmatrix} \tag{6.11}$$

and control matrix:

$$B_o = \begin{bmatrix} \cos(\psi_o) & -\sin(\psi_o) & 0 \\ \sin(\psi_o) & \cos(\psi_o) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.12}$$

where the subscript $(\cdot)_o$ denotes the optimal state and control from the mid-level MPC planner.

For this linear time varying system, a novel feedback control is designed to regulate its state to the origin, i.e. reduce the tracking error $\Delta\mathbf{x}$ to zero. Let the error based feedback law be given by $\Delta\mathbf{x} = -K\Delta\mathbf{x}$, where the feedback control gain is proposed as

$$K = \begin{bmatrix} k_1\cos(\psi_o) & k_1\sin(\psi_o) & 0 \\ -k_2\sin(\psi_o) & k_2\cos(\psi_o) & 0 \\ -k_3\frac{\sin(\psi_o)}{u_o} & k_3\frac{\cos(\psi_o)}{u_o} & k_4 \end{bmatrix} \tag{6.13}$$

where k_1, k_2, k_3 and k_4 are positive gains, usually chosen to be constant.

The closed-loop error system is derived as:

$$\Delta\dot{\mathbf{x}} = (A_o - B_oK)\Delta\mathbf{x} = A_{cl}\Delta\mathbf{x} \tag{6.14}$$

where the state transition matrix A_{cl} has a complicated form:

$$A_{cl}(\mathbf{x}_o, \mathbf{w}_o) = \begin{bmatrix} -k_1\cos^2(\psi) - k_2\sin^2(\psi) & (k_2 - k_1)\cos(\psi)\sin(\psi) & -v\cos(\psi) - u\sin(\psi) \\ (k_2 - k_1)\cos(\psi)\sin(\psi) & -k_2\cos^2(\psi) - k_1\sin^2(\psi) & u\cos(\psi) - v\sin(\psi) \\ (k_3\sin(\psi))/u & -(k_3\cos(\psi))/u & -k_4 \end{bmatrix} \tag{6.15}$$

where the subscript $(\cdot)_o$ is eliminated for the sake of simplification.

This closed-loop error system (6.14) has a very nice property. That is, its eigenvalues directly and only depend on the control gains under the proposed control law (6.13). It can be shown from the characteristic equation:

$$\det(\lambda I - A_{cl}) = (\lambda + k_1)(\lambda^2 + (k_2 + k_4)\lambda + k_3) \tag{6.16}$$

where λ represents eigenvalues. It is easy to set all the gains k_i, $i = 1, \ldots, 4$ to positive constant, so that eigenvalues λ_i, $i = 1, 2, 3$, all have negative real parts. If the state transition matrix A_{cl} is constant, the above condition is adequate to guarantee the stability of error system (6.14) under the proposed feedback gain (6.13). However, A_{cl} is actually time varying and depends on the reference $\mathbf{x}_o$ and $\mathbf{w}_o$ from the MPC planner. A more rigorous stability analysis is needed, which is provided in Theorem (6.1). Theorem (6.1) is established by following the technique developed in [120], and it is included for the sake of completeness.

Theorem 6.1. *The closed-loop error system (6.14) is globally asymptotically stable if k_1, k_2, k_3 and k_4 are chosen as positive constants and such that different eigenvalues are resulted.*

Proof. Considers a Lyapunov function candidate

$$V(v, t) = v^T M(t) v e^{f(v,t)} \tag{6.17}$$

where v is used to denote the state $\Delta\mathbf{x}$, $f(v, t)$ is a scalar function and the functional matrix $M(t)$ is defined as:

$$M(t) = (L^{-1}(t))^T L^{-1}(t) \tag{6.18}$$

where $L(t)$ is the matrix consisting of all the eigenvectors of $A_{cl}(t)$. Thus, $M(t)$

is a positive definite matrix. Because $\lambda_i \neq \lambda_j$, $L(t)$ satisfies

$$L^{-1}(t)A_{cl}(t)L(t) = \Lambda = \text{diag}\{\lambda_1, \lambda_2, \lambda_3\} \tag{6.19}$$

Substituting Eq.(6.18) into Eq.(6.17) and using the property of Eq.(6.19) yields

$$\begin{aligned} V(v,t) &= v^T M(t) v e^{f(v,t)} \\ &= (L^{-1}(t)v)^T (L^{-1}(t)v) e^{f(v,t)} \\ &= \left\| L^{-1}(t)v \right\| e^{f(v,t)} \end{aligned} \tag{6.20}$$

With the properties of the norm and the exponential function, it can be seen that $V(v,t) \geq 0$ and the equality happens if and only if $L^{-1}(t)v = 0$. Consequently, this is implied by $v = 0$ as $L^{-1}(t)$ is invertible. Therefore, $V(v,t)$ satisfies the condition of being an Lyapunov function.

Taking the derivative of Lyapunov function $V(v,t)$ with respect to time and invoking the closed-loop error system (6.14) yields

$$\dot{V}(v,t) = [v^T A_{cl}(t)M(t)v + v^T M(t)A_{cl}(t)v + v^T \dot{M}(t)v + \dot{f}(v,t)v^T M(t)v]e^{f(v,t)} \tag{6.21}$$

where $\dot{M}(t)$ is the derivative of $M(t)$ with respect to time. Next, by choosing a function $f(v,t)$ in the following form

$$\dot{f}(v,t) = -\frac{v^t \dot{M}(t) v}{v^T M(t) v} \tag{6.22}$$

the derivative of Lyapunov function Eq.(6.21) can be written as

$$\dot{V}(v,t) = v^T \left[(A_{cl}(t))^T M(t) + M(t)A_{cl}(t) \right] v e^{f(v,t)} \tag{6.23}$$

Taking into account Eq.(6.18) and Eq.(6.19), Eq.(6.23) can be written as:

$$\begin{aligned} \dot{V} =& v^T \left[(A_{cl}(t))^T (L^{-1}(t))^T L^{-1}(t) + (L^{-1}(t))^T L^{-1}(t) A_{cl}(t) \right] v e^{f(v,t)} \\ =& v^T (L^{-1}(t))^T \left[(L(t))^T (A_{cl}(t))^T (L^{-1}(t))^T + L^{-1}(t) A_{cl}(t) L(t) \right] L^{-1}(t) v e^{f(v,t)} \\ =& ((L^{-1}(t))v)^T \left[(\Lambda(t))^T + \Lambda(t) \right] (L^{-1}(t)v) e^{f(v,t)} \end{aligned} \tag{6.24}$$

Note that $\Lambda^T + \Lambda = 2\text{diag}\{\Re(\lambda_1), \ldots, \Re(\lambda_2)\}$. Furthermore, it can be shown from Eq.(6.16) that all the eigenvalues have negative real parts if all the gain k_i, $i = 1, 2, 3, 4$, are chosen as positive constants. Let $-a \leq \Re(\lambda_i) \leq -b \leq 0$. Thus,

$$-2aV(v,t) \leq \dot{V}(v,t) \leq -2bV(v,t) < 0 \tag{6.25}$$

for all nonzero vectors $v = \Delta\mathbf{x}$. This completes the proof. □

Theorem 6.1 guarantees the stability of the error system (6.14) under the LTV guidance compensator, so that the composite signal $\mathbf{w}_c = \mathbf{w} + \Delta\mathbf{w}$ (provided by the MPC planner and the LTV compensator) can drive the kinematics to follow the reference $\mathbf{y}_{ref}$ and act as a desired guidance command for the flight control system to track.

The flight controller is designed based on the helicopter dynamic model which commands the helicopter to follow the guidance $\mathbf{w}_c$ generated from simplified kinematic model as described above. The over hierarchical control structure is given in Fig.6.3

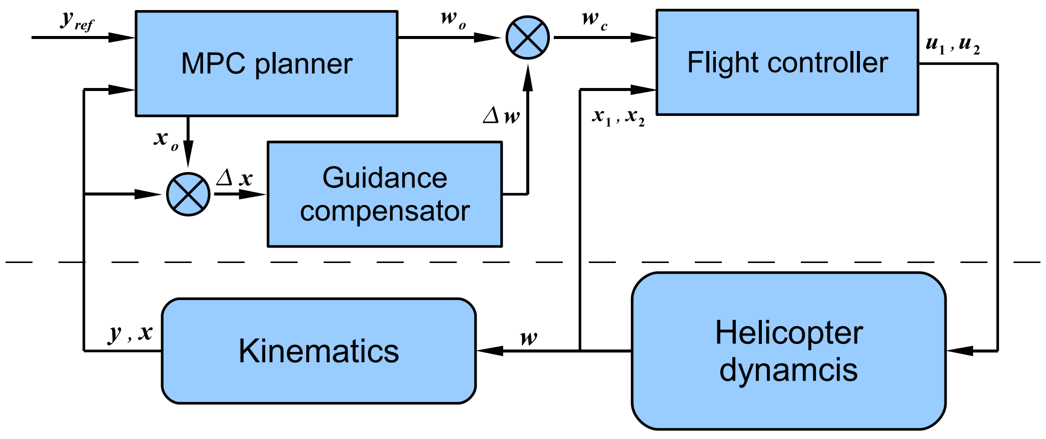


Figure 6.3: Hierarchical control structure

Due to the parallel two-level structure of the path planning layer, the guidance compensator can be used as an emergency guidance controller to track a pre-defined trajectory. For example, if the MPC planner failed to solve a nonlinear optimisation problem, the guidance compensator can stabilise the helicopter on the current position.

6.3.3 Flight controller

The flight controller described in this chapter adopts the linear MPC technique modified from [70] to achieve the tracking control of helicopter dynamics (6.2a). Two linear MPC controllers with the same structure are employed to govern the longitudinal/lateral and heave/heading subsystems, respectively. For each subsystem, an augmented formulation is adopted:

$$\underbrace{\begin{bmatrix} \mathbf{x}_i(k+1) \\ \mathbf{u}_i(k) \\ \mathbf{w}_i(k+1) \end{bmatrix}}_{\tilde{\mathbf{x}}_{k+1}} = \underbrace{\begin{bmatrix} A_i^d & B_i^d & 0 \\ 0 & B & 0 \\ 0 & 0 & I \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} \mathbf{x}_i(k) \\ \mathbf{u}_i(k-1) \\ \mathbf{w}_i(k) \end{bmatrix}}_{\tilde{\mathbf{x}}_k} + \underbrace{\begin{bmatrix} B_i \\ I \\ 0 \end{bmatrix}}_{\tilde{B}} \underbrace{\Delta \mathbf{u}_i}_{\tilde{\mathbf{u}}_k} \tag{6.26}$$

$$\tilde{\mathbf{y}}_k = \mathbf{z}_i(k) - \mathbf{w}_i(k) = \tilde{C}\tilde{\mathbf{x}}_k \tag{6.27}$$

where index $i = 1, 2$ indicates the longitudinal/lateral or heave/heading subsystem, respectively, A_i^d and B_i^d are system and control matrices in the discrete form of Eq.(6.2a) derived in Eq.(A.3), $\mathbf{w}_i$ denotes the corresponding guidance command in $\mathbf{w}_c$, $\tilde{C} = [\ C_i \quad 0 \quad -I\ ]$ is the output matrix, and $\tilde{\mathbf{y}}_k$ is the output vector of the augmented system, representing for the tracking error.

For such a system, a linear MPC is employed as the flight controller to stabilise the state and regulate the output, where the performance index is specified by a quadratic cost function to be minimised:

$$J(k) = \frac{1}{2} \sum_{i=1}^{H_p} \|\tilde{\mathbf{y}}_{k+i}\|_Q^2 + \frac{1}{2} \sum_{i=0}^{H_c-1} \|\tilde{\mathbf{u}}_{k+i}\|_R^2 \tag{6.28}$$

where k indicates the time step at which the state is updated, $\tilde{\mathbf{y}}_{k+i}$, $i = 1, \ldots, H_p$ is the i-step ahead prediction of the tracking error with H_p denoting the prediction horizon. The predictions of tracking errors are functions of the future control increments $\tilde{\mathbf{u}}_{k+i}$, $i = 1, \ldots, H_c - 1$, where H_c is the control horizon, beyond which the control keeps the same value. Note that the subscript used to indicate different subsystems is eliminated as the same MPC formulation can be applied to both subsystems.

The optimisation problem needs to be solved at each time step k is then formulated as:

$$\min_{\tilde{\mathbf{u}}} J(k) \tag{6.29a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_{k+i+1} = \tilde{A}\tilde{\mathbf{x}}_{k+i} + \tilde{B}\tilde{\mathbf{u}}_{k+i}, \tag{6.29b}$$

$$\tilde{\mathbf{u}}_{k+i} \in \mathbb{U}, \quad i = 0, 1, \ldots, H_c, \tag{6.29c}$$

$$\tilde{\mathbf{x}}_{k+i} \in \mathbb{X}, \quad i = 0, 1, \ldots, H_p, \tag{6.29d}$$

where Eq.(6.29a) is the cost function defined in Eq.(6.28), $\mathbb{U}$ is the control constraint which actually limits the increments of control signals, and $\mathbb{X}$ is the state constraints including limits on subsystem's state $\mathbf{x}_i$ and its control input $\mathbf{u}_i$, for $i = 1, 2$.

This optimisation problem can be converted into a QP formation for which fast and numerically reliable algorithms are available. The reformulation can start with considering the system state in a matrix form $\bar{X} = [\tilde{\mathbf{x}}_1^T, \tilde{\mathbf{x}}_2^T, \cdots, \tilde{\mathbf{x}}_{H_p}^T]^T$. Note that the index k is dropped without losing generality. For each element $\tilde{\mathbf{x}}_i$, the evolution of the system (6.26), i.e. the equality constraint Eq.(6.29b), can be represented by

$$\tilde{\mathbf{x}}_i = \tilde{A}^i\tilde{\mathbf{x}}_0 + \sum_{j=0}^{i-1} \tilde{A}^j\tilde{B}\tilde{\mathbf{u}}_{i-1-j} \tag{6.30}$$

for $i = 1, 2, \ldots, H_p$. Thus, a matrix expression of the evolution of all the output of the system (6.26) can be derived as:

$$\bar{Y} = \bar{C}\bar{A}\tilde{\mathbf{x}}_0 + \bar{C}\bar{B}\bar{U} \tag{6.31}$$

where

$$\bar{Y} = \begin{bmatrix} \tilde{\mathbf{y}}_1^T & \tilde{\mathbf{y}}_2^T & \cdots & \tilde{\mathbf{y}}_{N_p}^T \end{bmatrix}^T \tag{6.32}$$

$$\bar{U} = \begin{bmatrix} \tilde{\mathbf{u}}_0^T, \tilde{\mathbf{u}}_1^T, \ldots, \tilde{\mathbf{u}}_{N_c-1}^T \end{bmatrix}^T \tag{6.33}$$

$$\bar{C} = \text{diag}\{\tilde{C}, \ldots, \tilde{C}\} \tag{6.34}$$

with the corresponding dimension, and

$$\bar{A} = \begin{bmatrix} \tilde{A} \\ \tilde{A}^2 \\ \vdots \\ \tilde{A}^{N_c-1} \\ \vdots \\ \tilde{A}^{N_p} \end{bmatrix} \quad \bar{B} = \begin{bmatrix} \tilde{B} & 0 & \cdots & 0 \\ \tilde{A}\tilde{B} & \tilde{B} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{A}^{H_c-1}\tilde{B} & \tilde{A}^{H_c-2}\tilde{B} & \cdots & \tilde{B} \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{A}^{H_p-1}\tilde{B} & \tilde{A}^{H_p-2}\tilde{B} & \cdots & \tilde{A}^{H_p-H_c}\tilde{B} \end{bmatrix} \tag{6.35}$$

Therefore, by inserting Eq.(6.31) into Eq.(6.28) the optimisation problem (6.29) can be written as the following QP formulation:

$$J^*(\tilde{\mathbf{x}}_0) = \frac{1}{2}\tilde{\mathbf{x}}_0^T Y \tilde{\mathbf{x}}_0 + \min_{\bar{U}} \{\frac{1}{2}\bar{U}^T H \bar{U} + \tilde{\mathbf{x}}_0^T F \bar{U}\} \tag{6.36a}$$

$$\text{s.t.} \quad G\bar{U} \leq W + E\tilde{\mathbf{x}}_0 \tag{6.36b}$$

where J^* is the optimal cost as a function of initial state $\tilde{\mathbf{x}}_0$, $Y = (\bar{C}\bar{A})^T\bar{Q}\bar{C}\bar{A}$, $H = (\bar{C}\bar{B})^T\bar{Q}\bar{C}\bar{B} + \bar{R}$, and $F = (\bar{C}\bar{A})^T\bar{Q}\bar{C}\bar{A}$, in which $\bar{Q} = \text{diag}\{Q, \ldots, Q\}$, $\bar{R} = \text{diag}\{R, \ldots, R\}$ in the corresponding dimensions. $G\bar{U} \leq W + E\tilde{\mathbf{x}}_0$ is the constraints translated from Eq.(6.29c)-(6.29d).

The QP problem (6.36) is solved at each sampling instant with a updated $\tilde{\mathbf{x}}_0$ to generate the control sequence $\bar{U}$, where the first one $\tilde{\mathbf{u}}_0$ is actually applied to the helicopter. The main advantage of using linear MPC as the flight controller is the ability to handle the state and control constraints, so that the attitude of helicopter can be retained in a reasonable range without violating the assumptions of the linear dynamics model and the planar flight mode. The computational load of solving a QP problem is now manageable by a microprocessor especially when using an efficient solver [30].

6.4 Implementation

The realisation and implementation of the hierarchical control framework, composed of a MPC planner, a guidance compensator and a flight controller, need to be investigated to achieve local path planning and tracking control simultaneously for an autonomous helicopter.

A modern avionic system for a UAV is usually configured to have two flight computers. The low-level one functions as an autopilot, whereas the high-level one executes some advanced functions like planning and management. Under this configuration of avionics, it is natural to locate the guidance compensator and flight controller on the low-level autopilot and the MPC planner on the high-level flight computer, respectively. The arrangement is summarised in the Table 6.1 in terms of three aspects: algorithms (or functions), the plants they deal with, and their implementation means. The linear MPC algorithm needs to be implemented on a low-level autopilot, because it directly controls the helicopter dynamics and reacts to external disturbances. The guidance compensator is also executed on the same computer so that it can cooperate with the flight controller and provide the guidance compensation. More importantly, if the high-level computer fails to provide guidance commands, the low-level autopilot itself can stabilise the helicopter. Both the algorithms located on the low-level autopilot are executed in a high sampling rate, as they are time critical functions. In the laboratory environment, this low-level autopilot is realised by the simulink real-time control environment. The nonlinear MPC planner is carried out by another high-level flight computer due to its high computational load and to reduce inferences with the time-critical low-level algorithms. In the test environment, this function is performed in Matlab using a NLP solver KNITRO [43]. The overall hardware configuration in the indoor testbed can refer to the diagram in Fig.5.10.

Table 6.1: Realisation of hierarchical control

Implementation	Algorithm	Plant/Model
High-level computer (Matlab NLP solver)	Nonlinear MPC (local path planning)	Obstacle info
		Kinematics (Outer-loop)
Low-level autopilot (Simulink environment) (time critical)	Guidance compensator	
	Linear MPC (tracking and stabilising)	Dynamics (Inner-loop)

6.4.1 Simulation

Simulations are carried out using the configuration introduced before. The full dynamics model of the Trex-250 developed in Chapter 4 is used as the plant, whereas the kinematic model and linearised dynamics model are used for designing the local path planner and flight controller, respectively.

The purpose of simulations is to choose the control parameters that can achieve a good planning and tracking performance. The parameters determined by simulations are given in Table.6.2. Simulation results are not presented here, as performance of the proposed algorithms can be further verified by flight experiments in a more realistic environment.

Table 6.2: Parameters for simulation

MPC planner	Guidance compensator	Flight controller
$T_d = 0.05$s	$k_1 = 1$	$\Delta T = 0.02$
$N = 2$	$k_2 = 1$	$H_p = 50$
$H = 20$	$k_3 = 0.5$	$H_c = 4$
$T_s = 0.5$s	$k_4 = 2$	$Q = \text{diag}\{1, 1\}$
$\alpha = 10$		$R = \text{diag}\{10, 10\}$
$A = 15$		

6.4.2 Experiment

After the control parameters are determined and the initial performance assessment is done by simulations, flight experiments can be carried out on the testbed to verify the proposed hierarchical control framework. Due to the confined test space, only simple scenarios can be set up. The experimental result shown here is to track a global square trajectory while avoiding two obstacles en route. The tracking speed is set at 0.25 m/s because the test focuses on path planning and is not aimed to excite aggressive manoeuvres. These obstacles are assumed to be detected by the helicopter when the distance is less than 0.5m. The combined local path planning and tracking result is shown in Fig.6.4, where the arrows are used to indicate helicopter's moving direction. The guidance command $\mathbf{w}_c$ composed of the MPC planner $\mathbf{w}_o$ plus the guidance compensator, and the corresponding response of the helicopter $\mathbf{w}$ are shown in Fig.6.5. It can be seen that the helicopter under the control of the linear MPC is able to track to guidance commands that eventually lead to a collision-free trajectory as shown in Fig.6.4. The calculation time of the online optimisation is given in Fig.6.6.

The linear MPC controller stabilises the helicopter and tracks the guidance command. Its tracking performance is given in Fig.6.5, and stabilising performance can be examined by observing the attitude history of the flight test in

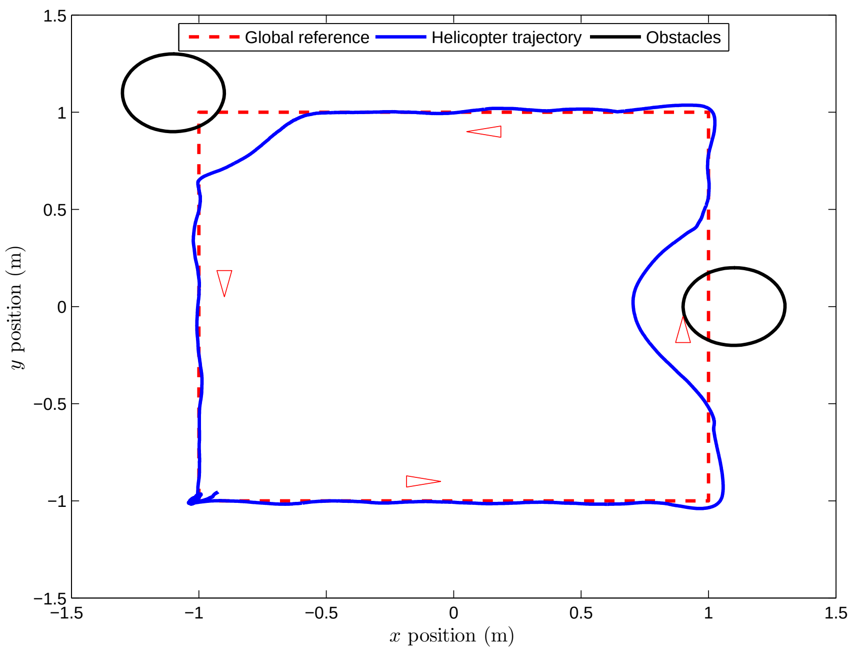


Figure 6.4: Flight test result

Fig.6.7. The corresponding control signals send to the helicopter are presented in Fig.6.8. It shall be noted that the roll angle ϕ and pitch angle θ are retained in small deviations during the normal flight phase, and can quickly reach high magnitudes during the manoeuvres. The different mean values of ϕ and θ are trim values for the helicopter used in experiments. The non-zero trim values do not compromise the control performance as the linear MPC has the integral action which automatically finds the corresponding control trims. As a result, the steady state error is eliminated from the tracking output (see Fig.6.4). However, it shall note that the flight test result outperforms those in Chapter 5 not only because of the integral action in the controller, but also the slow flight speed retained a linear dynamic region.

The emergency guidance provided by the guidance compensator is also tested in flight experiments. In the test, the high-level MPC planner is deliberately halted, but the helicopter can keep hovering on the last commanded position. The experimental result is not included in this thesis, but this function has actually saved the helicopter several times during the initial tuning tests.

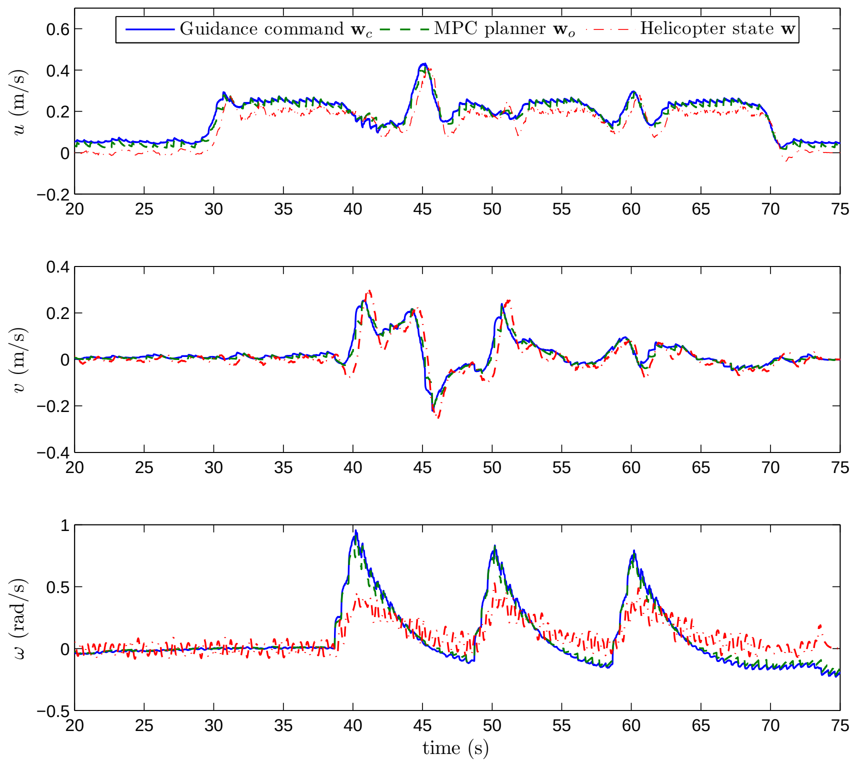


Figure 6.5: Guidance command and response

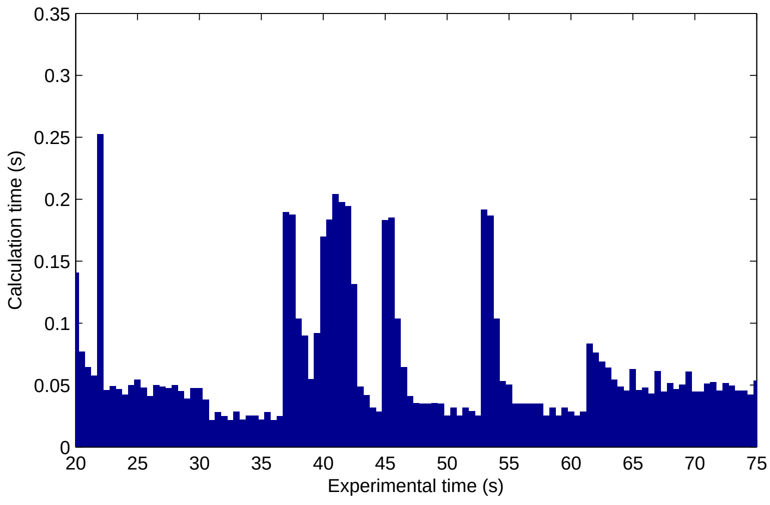


Figure 6.6: Calculation time

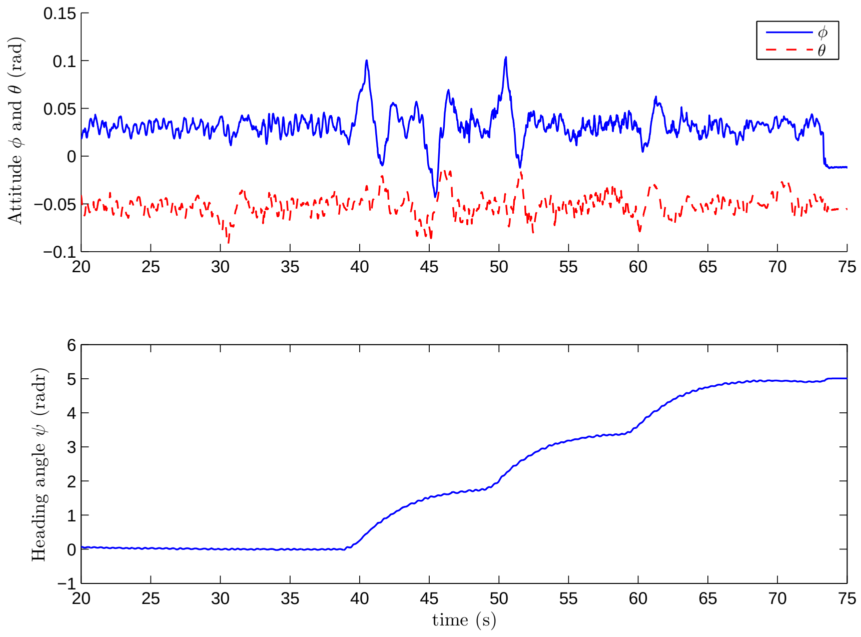


Figure 6.7: Attitude history

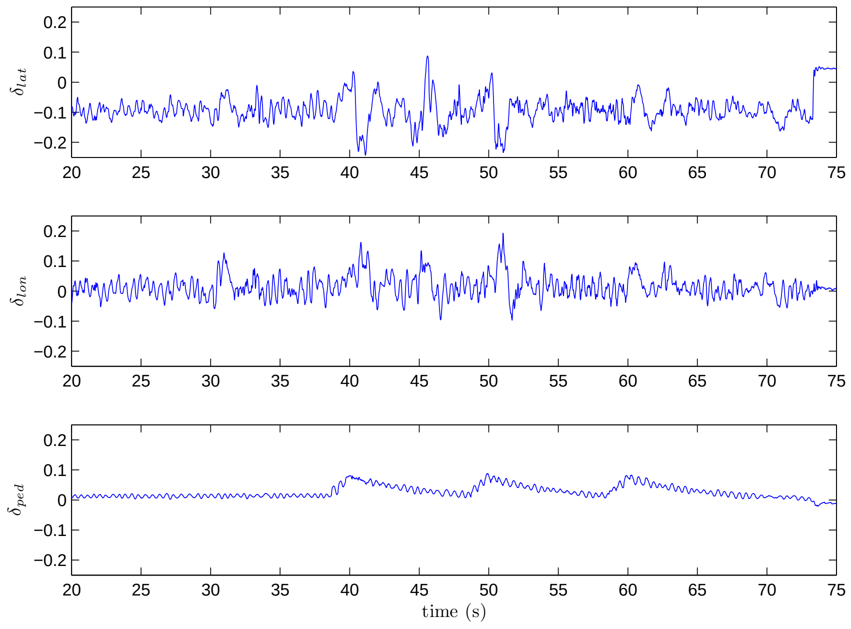


Figure 6.8: Control signals

6.5 Conclusion

This chapter describes a hierarchical control framework for local path planning and control of small helicopters. Two MPC techniques are used in this framework, including a nonlinear MPC planner integrated with the potential field function to achieve local path planning and obstacle avoidance, and a linear MPC used for stabilising and tracking control. A guidance compensator based on a linear time varying control law is then used to link the planning layer and the tracking control layer, which provides quick responses to translational disturbances and improves the robustness of the control framework. In addition, the integral action incorporated in the linear MPC helps to reduce the steady state error. Flight tests are carried out to verify this control framework, which show a good performance of this control framework.

Chapter 7

Explicit nonlinear MPC and disturbance observer based control for autonomous helicopters

7.1 Introduction

MPC has been widely recognised as a promising control strategy for UAV systems. Its capability and advantages on flight control and local path planning for autonomous helicopters have been demonstrated in Chapter 5 and 6. However, one main barrier in applications is that a solution of a nonlinear optimisation problem has to be found in each sampling instant. To overcome this problem, Chapter 5 combines the piecewise constant MPC with a two-level control framework to facilitate the real-time implementation. The formulated nonlinear optimisation problem still has to be solved online, usually by a secondary flight computer. The extra payload and power consumption are quite luxury for a small-scale helicopter.

This chapter further looks at the real-time implementation issue of MPC by avoiding online optimisation. An explicit nonlinear MPC (ENMPC) for trajectory tracking of autonomous helicopters is introduced in this chapter. By approximating the tracking error and control efforts in the receding horizon using their Taylor expansion to a specified order, an analytic solution to the nonlinear MPC can be found and consequently a closed form controller can be formulated without online optimisation [20]. The benefits of using this MPC algorithm are not only the elimination of the online optimisation and the associated resource, but also a higher control bandwidth, which is very important for helicopters in aggressive flight scenarios.

Apart from the control algorithm, there are practical issues in controlling autonomous helicopters from an engineering point of view. It is known that the control performance of MPC, or other model based control technologies, heavily relies on the quality of the model. However, the model of high accuracy for a helicopter is difficult to obtain due to the complicated aerodynamic nature of the rotor system. On the other hand, because of the light-weighted structure, small-scale helicopters are more likely to be affected by wind gusts and other disturbances than their full size counterpart, and the physical parameters such as mass and moments of inertia can be significantly altered due to the change of the payload and even its location. All these factors compromise the actual performance of the controller designed based on the nominal model.

Robust control techniques, especially H_∞ technique, have been used in handling the parametric uncertainty and ummodelled dynamics [32; 65; 73]. Although satisfactory performance has been demonstrated, robust control is known to result in conservative solutions and presents trade-offs between performance and robustness. On the other hand, adaptive control also shows promising results of controlling autonomous helicopters in the presence of uncertainties [49; 59]. However, the controllers usually have complicated structures and very high order. Other methods to compensate the wind disturbances are also available such as [9] where the authors provided a method of calculating the trim control by exploiting a detailed helicopter model. However, this method need either an estimation or direct measurement of wind conditions.

To enhance the performance of ENMPC in a complex operation environment, this chapter advocates a disturbance observer based control (DOBC) approach. Disturbance observers have been applied to estimate unknown disturbances in the control process [15; 18]. As the estimation of disturbances is provided, the control system can explicitly take them into account and compensate them. The advantage of the DOBC is that it preserves the tracking and other properties of the original baseline control while being able to compensate disturbances rather than resorting to a different control strategy.

In designing a disturbance observer augmented ENMPC for trajectory tracking of autonomous helicopters, two problems need to be addressed, namely, designing the nonlinear disturbance observer to estimate disturbances acting on the helicopter, and integrating the disturbance information into the ENMPC scheme to compensate their influences. To this end, another contribution of this chapter lies in the synthesis of the ENMPC and DOBC by exploiting the helicopter model structure. The disturbances are assumed to be a part of the helicopter dynamics where the coupling terms can also be lumped into disturbance terms. In this way an ENMPC is derived under the assumption that all the disturbances are measurable, and then these disturbances are replaced by their estimation provided by the proposed disturbance observers. In turn, the lumped disturbance terms simplify the model structure allowing the derivation of ENMPC for helicopters. The composite control framework provides a promising solution to autonomous helicopter trajectory tracking in the presence of uncertainties and disturbances. The performance of the proposed control system is tested through simulations and verified in the indoor flight testbed.

7.2 Helicopter model

As discussed in Chapter 4, a helicopter is a highly nonlinear system with multiple inputs multiple outputs and complex internal couplings. The complete model taking into account the flexibility of the rotors and fuselage usually results in high degrees-of-freedom and makes the following system identification much more difficult. Therefore, a practical way to deal with this issue is to capture the primary dynamics by a simplified model and treat the other trivial factors that affect dynamics as uncertainty or disturbances. This process has been demonstrated in Chapter 4, and the resulting model for control design purpose is presented in Eq.(5.1).

The helicopter model used in this chapter is modified from Eq.(5.1) by explicitly taking into account disturbances. With the translational kinematics in the standard form

$$[\ \dot{x} \quad \dot{y} \quad \dot{z}\ ]^T = \mathbf{R}_b^i(\phi, \theta, \psi)[\ u \quad v \quad w\ ]^T \tag{7.1}$$

the translational dynamics of the helicopter are modified by keeping the thrust of main rotor as a dominating force and considering other force contributions as disturbances, such that

$$\begin{aligned}\dot{u} &= vr - wq - g\sin\theta + d_x \\ \dot{v} &= wp - ur + g\cos\theta\sin\phi + d_y \\ \dot{w} &= uq - vp + g\cos\theta\cos\phi + T + d_z\end{aligned} \tag{7.2}$$

where, T is the normalised main rotor thrust controlled by collective pitch δ_{col}, in the way that $T = g + Z_w w + Z_{col}\delta_{col}$, and (d_x, d_y, d_z) are normalised force disturbances that include external wind gusts, internal couplings and unmodelled dynamics. These force disturbances directly affect the translational dynamics and result in tracking error. As force disturbances are not in the channels of control inputs, they are called "mis-matched" disturbances. This modification on one hand increases the valid range of the model compared to simplified helicopter models for control design that neglect all other forces other than the main thrust [58; 73; 94]. On the other hand it reduces the workload of deriving the ENMPC for helicopters as different forces are lumped into one term .

Apart from the force disturbances in Eq.(7.2), small-scale helicopters also subject to structural uncertainties and are vulnerable to physical alterations like payload change. These factors are commonly ignored in the control design, as they can be compensated by setting control trims in the implementation. To save the trim tuning process in the real life operation, trims errors in the control channel are considered as disturbances again. Thereby, combining helicopter's rotational dynamics in Eq.(5.1) and flapping angle approximation Eq.(5.2) yields

$$\begin{aligned}\dot{p} &= -L_{pq} + L_{lat}(\delta_{lat} + d_{lat}) + L_{lon}(\delta_{lon} + d_{lon}) \\ \dot{q} &= -M_{pq} + M_{lat}(\delta_{lat} + d_{lat}) + M_{lon}(\delta_{lon} + d_{lat}) \\ \dot{r} &= -N_{pr} + N_r r + N_{col}\delta_{col} + N_{ped}(\delta_{ped} + d_{ped})\end{aligned} \tag{7.3}$$

where

$$\begin{aligned}&L_{pq} = qr(I_{yy} - I_{zz})/I_{xx} + \tau(L_a q + L_b p), \\ &M_{pq} = pr(I_{zz} - I_{xx})/I_{yy} + \tau(M_a q + M_b p), \\ &N_{pq} = pq(I_{xx} - I_{yy})/I_{zz} \\ &L_{lat} = L_a A_{lat} + L_b B_{lat}, \qquad M_{lat} = M_a A_{lat} + M_b B_{lat}, \\ &L_{lon} = L_a A_{lon} + L_b B_{lon}, \qquad M_{lon} = M_a A_{lon} + M_b B_{lon},\end{aligned} \tag{7.4}$$

and d_{lat}, d_{lon} and d_{ped} account for different trim errors. In addition, since they are combined into the angular dynamics and affect the angular rate directly, they can be considered as torque disturbances.

The modified helicopter model by combining (7.1)-(7.3) can be expressed by a general affine form:

$$\begin{aligned} \dot{\mathbf{x}} &= f(\mathbf{x}) + g_1(\mathbf{x})\mathbf{u} + g_2(\mathbf{x})\mathbf{d} \\ \mathbf{y} &= h(\mathbf{x}) \end{aligned} \tag{7.5}$$

where $\mathbf{x} = [\ x \quad y \quad z \quad u \quad v \quad w \quad p \quad q \quad r \quad \phi \quad \theta \quad \psi\ ]^T$ is the helicopter state, $\mathbf{y}$ is the output of the helicopter, and $\mathbf{d} = [\ d_x \quad d_y \quad d_z \quad d_{lat} \quad d_{lon} \quad d_{ped}\ ]^T$ is the lumped disturbance acting on the helicopter. In the trajectory tracking control of an autonomous helicopter, the interested outputs are the position and heading angle. Thus, $\mathbf{y} = [\ x \quad y \quad z \quad \psi\ ]^T$.

7.3 Explicit nonlinear MPC with disturbances

Trajectory tracking is the basic function required when an autonomous helicopter performs a task. To this end, a controller is needed such that the output $\mathbf{y}(t)$ of the helicopter (7.5) tracks the prescribed reference $\mathbf{w}(t)$. In the MPC strategy, tracking control can be achieved by minimising a receding horizon performance index

$$J = \frac{1}{2}\int_0^T (\hat{\mathbf{y}}(t+\tau) - \mathbf{w}(t+\tau))^T Q (\hat{\mathbf{y}}(t+\tau) - \mathbf{w}(t+\tau)) d\tau \tag{7.6}$$

where weighting matrix $Q = \text{diag}\{q_1, q_2, q_3, q_4\}$, $q_i > 0$, $i = 1, 2, 3, 4$. Note that the hatted variables belong to the prediction time frame.

Conventional MPC algorithm requires solving of an optimisation problem at every sampling instant to obtain the control signals. To avoid the computationally intensive online optimisation, an explicit solution for the nonlinear MPC problem is obtained based on the approximation of the tracking error in the receding prediction horizon [20].

7.3.1 Output approximation

For a nonlinear MIMO system like the helicopter, it is well known that after differentiating the outputs for a specific number of times, the control inputs appear in the expressions. The number of times of differentiation is defined as *relative degree.* For the helicopter with output $\mathbf{y} = [\ x \quad y \quad z \quad \psi\ ]^T$ and the corresponding input $\mathbf{u} = [\ \delta_{lon} \quad \delta_{lat} \quad \delta_{col} \quad \delta_{ped}\ ]^T$, the relative degree is a vector, $\rho = [\ \rho_1 \quad \rho_2 \quad \rho_3 \quad \rho_4\ ]$. If continuously differentiating the output after the control input appears, the derivatives of control input appear, where the number of the input derivatives r is defined as the *control order.*

Since the helicopter model has different relative degrees, the control order r is first specified in the controller design. The ith output of the helicopter in the receding horizon can be approximated by its Taylor series expansion up to order $\rho_i + r$:

$$\begin{aligned}\hat{y}_i(t+\tau) &\approx y_i(t) + \tau\dot{y}_i(t) + \cdots + \frac{\tau^{r+\rho_i}}{(r+\rho_i)!}y_i^{[r+\rho_i]}(t) \\ &= \begin{bmatrix}1 & \tau & \cdots & \frac{\tau^{r+\rho_i}}{(r+\rho_i)!}\end{bmatrix}\begin{bmatrix} y_i(t) \\ \dot{y}_i(t) \\ \vdots \\ y_i^{[r+\rho_i]}(t)\end{bmatrix}, 0 \le \tau \le T \end{aligned} \tag{7.7}$$

where $i = 1, 2, 3, 4$. In this way, the approximation of the overall output of the helicopter can be cast in a matrix form:

$$\begin{aligned}\hat{\mathbf{y}}(t+\tau) = \begin{bmatrix}\hat{x}(t+\tau) \\ \hat{y}(t+\tau) \\ \hat{z}(t+\tau) \\ \hat{\psi}(t+\tau)\end{bmatrix} &= \begin{bmatrix}\hat{y}_1(t+\tau) \\ \hat{y}_2(t+\tau) \\ \hat{y}_3(t+\tau) \\ \hat{y}_4(t+\tau)\end{bmatrix} \\ = \begin{bmatrix} 1, \tau, \cdots, \frac{\tau^{r+\rho_1}}{(r+\rho_1)!} & \cdots & 0_{1\times(r+\rho_4+1)} \\ \cdots & \cdots & \cdots \\ 0_{1\times(r+\rho_1+1)} & \cdots & 1, \tau, \cdots, \frac{\tau^{r+\rho_4}}{(r+\rho_4)!}\end{bmatrix}& \\ \begin{bmatrix} y_1(t)^T, \dot{y}_1(t)^T, \cdots, y_1^{[r+\rho_1]}(t)^T & \cdots & y_4(t)^T, \dot{y}_4(t)^T, \cdots, y_4^{[r+\rho_4]}(t)^T\end{bmatrix}^T& \end{aligned} \tag{7.8}$$

For each channel in the output matrix, the control orders r are the same and can be decided during the control design, whereas the relative degrees ρ_i are different but determined by the helicopter model structure. Manipulating the output matrix (7.8) gives the following partition:

$$\hat{\mathbf{y}}(t+\tau) = \left[\begin{array}{ccc|ccc} \bar{\tau}_1 & \cdots & 0_{1\times\rho_4} & & & \\ \cdots & \cdots & \cdots & \tilde{\tau}_1 & \cdots & \tilde{\tau}_{r+1} \\ 0_{1\times\rho_1} & \cdots & \bar{\tau}_4 & & & \end{array}\right] \left[\bar{Y}_1(t)^T \quad \cdots \quad \bar{Y}_4(t)^T | \tilde{Y}_1(t)^T \quad \cdots \quad \tilde{Y}_r(t)^T\right]^T \tag{7.9}$$

where

$$\bar{Y}_i = \left[y_i(t) \quad \dot{y}_i(t) \quad \cdots \quad y_i^{[\rho_i-1]}\right]^T, i = 1,2,3,4 \tag{7.10}$$

$$\tilde{Y}_i = \left[\begin{array}{cccc} y_1^{[\rho_1+i-1]} & y_2^{[\rho_2+i-1]} & \cdots & y_4^{[\rho_4+i-1]} \end{array}\right]^T, i = 1,\ldots,r+1 \tag{7.11}$$

$$\bar{\tau}_i = \left[\begin{array}{cccc} 1 & \tau & \cdots & \frac{\tau^{\rho_i-1}}{(\rho_i-1)!} \end{array}\right], i = 1,2,3,4 \tag{7.12}$$

and

$$\tilde{\tau} = \operatorname{diag}\left\{\begin{array}{ccc} \frac{\tau^{\rho_1+i-1}}{(\rho_1+i-1)!} & \cdots & \frac{\tau^{\rho_4+i-1}}{(\rho_4+i-1)!} \end{array}\right\} \tag{7.13}$$

It can be observed from Eq(7.9) that the prediction of the helicopter output $\hat{\mathbf{y}}(t+\tau)$, $0 \le \tau \le T$, in the receding horizon needs the derivatives of each output of the helicopter up to $r+\rho_i$ order at time instant t. Except for the output $\mathbf{y}(t)$ itself that can be directly measured, the other derivatives have to be derived according to the helicopter model (7.5). During this process the control input will appear in the ρ_ith derivatives, where $i = 1,2,3,4$.

The first derivatives can be obtained from the helicopter's kinematics model:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \dot{y}_3 \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{R}_b^i \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{7.14}$$

$$\dot{y}_4 = \dot{\psi} = q \sin\phi \sec\theta + r \cos\phi \sec\theta \tag{7.15}$$

Differentiating (7.14) and (7.15) with substitution of helicopter kinematics (7.1) yields the second derivatives:

$$\begin{bmatrix}\ddot{y}_1\\ \ddot{y}_2\\ \ddot{y}_3\end{bmatrix} = \begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix} = \mathbf{R}_b^i \begin{bmatrix} d_x \\ d_y \\ T + d_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}, \tag{7.16}$$

where $T = Z_w w + Z_{col}\delta_{col} - g$ is the normalised main rotor thrust, and

$$\begin{aligned}\ddot{y}_4 =& \ddot{\psi} = q\tfrac{\cos\phi}{\cos\theta}\dot{\phi} + q\tfrac{\sin\phi\sin\theta}{\cos^2\theta}\dot{\theta} - r\tfrac{\sin\phi}{\cos\theta}\dot{\phi} + r\tfrac{\cos\phi\sin\theta}{\cos^2\theta}\dot{\theta} - L_{pq}\tfrac{\sin\phi}{\cos\theta} + N_r\tfrac{\cos\phi}{\cos\theta}r + \\ & L_{lat}\tfrac{\sin\phi}{\cos\theta}(\delta_{lat} + d_{lat}) + L_{lon}\tfrac{\sin\phi}{\cos\theta}(\delta_{lon} + d_{lon}) + N_{col}\tfrac{\cos\phi}{\cos\theta}\delta_{col} + N_{ped}\tfrac{\cos\phi}{\cos\theta}(\delta_{ped} + d_{ped})\end{aligned} \tag{7.17}$$

Note that although control input δ_{col} appears in (7.16), the other control inputs do not, so it needs to continue differentiating the first three outputs. To facilitate the derivation, the relationship $\dot{\mathbf{R}}_b^i = \mathbf{R}_b^i\hat{\omega}$ is adopted by using skew-symmetric matrix $\hat{\omega} \in \mathbb{R}^{3\times 3}$:

$$\hat{\omega} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}. \tag{7.18}$$

Thus, the third and fourth derivatives of the position output can be written in:

$$\begin{bmatrix}y_1^{[3]}\\ y_2^{[3]}\\ y_3^{[3]}\end{bmatrix} = \begin{bmatrix}x^{[3]}\\ y^{[3]}\\ z^{[3]}\end{bmatrix} = \mathbf{R}_b^i\hat{\omega} \begin{bmatrix} d_x \\ d_y \\ T + d_z \end{bmatrix} + \mathbf{R}_b^i \begin{bmatrix} 0 \\ 0 \\ Z_w\dot{w} + Z_{col}\dot{\delta}_{col} \end{bmatrix}, \tag{7.19}$$

and

$$\begin{aligned}\begin{bmatrix}y_1^{[4]}\\ y_2^{[4]}\\ y_3^{[4]}\end{bmatrix} = \begin{bmatrix}x^{[4]}\\ y^{[4]}\\ z^{[4]}\end{bmatrix} =& \mathbf{R}_b^i\hat{\omega}\hat{\omega} \begin{bmatrix} d_x \\ d_y \\ T + d_z \end{bmatrix} + 2\mathbf{R}_b^i\hat{\omega} \begin{bmatrix} 0 \\ 0 \\ Z_w\dot{w} + Z_{col}\dot{\delta}_{col} \end{bmatrix} + \\ & \mathbf{R}_b^i \begin{bmatrix} -N_r r d_y - M_{pq}(T + d_z) \\ N_r r d_x + L_{pq}(T + d_z) \\ M_{pq}d_x - L_{pq}d_y + Z_w\ddot{w} \end{bmatrix} + \\ & \overline{A}(\mathbf{x}, \mathbf{d}) \begin{bmatrix}\delta_{lat} + d_{lat} & \delta_{lon} + d_{lon} & \ddot{\delta}_{col} & \delta_{ped} + d_{ped}\end{bmatrix}^T\end{aligned} \tag{7.20}$$

where

$$\overline{A}(\mathbf{x},\mathbf{d}) = \mathbf{R}_b^i \begin{bmatrix} M_{lat}(T+d_z) & M_{lon}(T+d_z) & 0 & -N_{ped}d_y \\ -L_{lat}(T+d_z) & -L_{lon}(T+d_z) & 0 & N_{ped}d_x \\ -M_{lat}d_x + L_{lat}d_y & -M_{lon}d_x + L_{lon}d_y & Z_{col} & 0 \end{bmatrix} \tag{7.21}$$

At this stage, the control inputs explicitly appear in (7.20). Therefore, the vector relative degree for the helicopter is $\rho = [\ 4 \quad 4 \quad 4 \quad 2\ ]$. Note that in the formulation of (7.20) $\ddot{\delta}_{col}$ is the new control input, whereas δ_{col} and $\dot{\delta}_{col}$ are treated as the states which can be obtained by adding integrators. This procedure is known as achieving relative degree through dynamics extension [45].

By invoking (7.14) -(7.19), it now can construct matrix $\bar{Y}_i$, $i = 1,2,3,4$. However, in order to find the elements in $\tilde{Y}_i$, $i = 1,2,\ldots,r+1$, further manipulations are required. By combining (7.17) and (7.20) and utilizing the Lie notation [45], one has:

$$\tilde{Y}_1 = \begin{bmatrix} y_1^{[\rho_1]} \\ y_2^{[\rho_2]} \\ y_3^{[\rho_3]} \\ y_4^{[\rho_4]} \end{bmatrix} = \begin{bmatrix} x^{[4]} \\ y^{[4]} \\ z^{[4]} \\ \psi^{[2]} \end{bmatrix} = \begin{bmatrix} L_f^{\rho_1} h_1(\mathbf{x},\mathbf{d}) \\ L_f^{\rho_2} h_2(\mathbf{x},\mathbf{d}) \\ L_f^{\rho_3} h_3(\mathbf{x},\mathbf{d}) \\ L_f^{\rho_4} h_4(\mathbf{x},\mathbf{d}) \end{bmatrix} + A(\mathbf{x},\mathbf{d})\tilde{\mathbf{u}} \tag{7.22}$$

where $\tilde{\mathbf{u}} = [\ \delta_{lat}+d_{lat} \quad \delta_{lon}+d_{lon} \quad \ddot{\delta}_{col} \quad \delta_{ped}+d_{ped}\ ]$; nonlinear terms $L_f^{\rho_i} h_i(\mathbf{x},\mathbf{d})$, $i = 1,2,3,4$, can be found in the previous derivation, and

$$A(\mathbf{x},\mathbf{d}) = \left[\begin{array}{ccc} L_{g_1}L_f^{\rho_1-1}h_1 & \cdots & L_{g_4}L_f^{\rho_1-1}h_1 \\ L_{g_1}L_f^{\rho_1-1}h_2 & \cdots & L_{g_4}L_f^{\rho_1-1}h_2 \\ \cdots & \cdots & \cdots \\ \hdashline L_{g_1}L_f^{\rho_1-1}h_4(\mathbf{x}) & \cdots & L_{g_4}L_f^{\rho_1-1}h_4(\mathbf{x}) \end{array}\right] = \begin{bmatrix} \overline{A}(\mathbf{x},\mathbf{d}) \\ \underline{A}(\mathbf{x},\mathbf{d}) \end{bmatrix}. \tag{7.23}$$

where $\overline{A}(\mathbf{x},\mathbf{d})$ is given in Eq.(7.21) and

$$\underline{A}(\mathbf{x},\mathbf{d}) = \begin{bmatrix} L_{lat}\frac{\sin\phi}{\cos\theta} & L_{lon}\frac{\sin\phi}{\cos\theta} & 0 & N_{ped}\frac{\cos\phi}{\cos\theta} \end{bmatrix}. \tag{7.24}$$

Differentiating (7.22) with respect to time together with substitution of the system's dynamics gives

$$\tilde{Y}_2 = \begin{bmatrix} y_1^{[\rho_1+1]} \\ y_2^{[\rho_2+1]} \\ y_3^{[\rho_3+1]} \\ y_4^{[\rho_4+1]} \end{bmatrix} = \begin{bmatrix} L_f^{\rho_1+1} h_1(\mathbf{x}) \\ L_f^{\rho_2+1} h_2(\mathbf{x}) \\ L_f^{\rho_3+1} h_3(\mathbf{x}) \\ L_f^{\rho_4+1} h_4(\mathbf{x}) \end{bmatrix} + A(\mathbf{x}, \mathbf{d})\tilde{\mathbf{u}}^{[1]} + p_1(\mathbf{x}, \tilde{\mathbf{u}}) \tag{7.25}$$

where $p_1(\mathbf{x}, \tilde{\mathbf{u}})$ is a nonlinear vector function of $\mathbf{x}$ and $\tilde{\mathbf{u}}$. By repeating this procedure, the higher derivatives of the output and $\tilde{Y}_i$, $i = 1, 2, \ldots, r$, can be calculated as

$$\tilde{Y}_{r+1} = \begin{bmatrix} y_1^{[\rho_1+r]} \\ y_2^{[\rho_2+r]} \\ y_3^{[\rho_3+r]} \\ y_4^{[\rho_4+r]} \end{bmatrix} = \begin{bmatrix} L_f^{\rho_1+r} h_1(\mathbf{x}) \\ L_f^{\rho_2+r} h_2(\mathbf{x}) \\ L_f^{\rho_3+r} h_3(\mathbf{x}) \\ L_f^{\rho_4+r} h_4(\mathbf{x}) \end{bmatrix} + A(\mathbf{x}, \mathbf{d})\tilde{\mathbf{u}}^{[r]} + p_r(\mathbf{x}, \tilde{\mathbf{u}}, \tilde{\mathbf{u}}^{[1]}, \ldots, \tilde{\mathbf{u}}^{[r]}) \tag{7.26}$$

So far by exploiting the helicopter model, the elements to construct $\bar{Y}$ and $\tilde{Y}$ in Eq.(7.9) are available. Therefore, the output of the helicopter in the future horizon $\mathbf{y}(t + \tau)$ can be expressed by its Taylor expansion in a generalized linear form with respect to the prediction time τ and current states as shown in Eq.(7.9).

In the same fashion as in Eq.(7.9), the reference in the receding horizon $\mathbf{w}(t+\tau)$, $0 \leq \tau \leq T$ can also be approximated by:

$$\mathbf{w}(t+\tau) = \begin{bmatrix} w_1(t+\tau) \\ w_2(t+\tau) \\ w_3(t+\tau) \\ w_4(t+\tau) \end{bmatrix} = \begin{bmatrix} T_f & T_s \end{bmatrix} \begin{bmatrix} \bar{W}_1(t)^T & \cdots & \bar{W}_4(t)^T | \tilde{W}_1(t)^T & \cdots & \tilde{W}_{r+1}(t)^T \end{bmatrix}^T \tag{7.27}$$

where

$$T_f = \begin{bmatrix} \bar{\tau}_1 & \cdots & 0_{1\times\rho_4} \\ \vdots & \ddots & \vdots \\ 0_{1\times\rho_1} & \cdots & \bar{\tau}_4 \end{bmatrix} \tag{7.28}$$

and

$$T_s = \begin{bmatrix} \tilde{\tau}_1 & \cdots & \tilde{\tau}_{r+1} \end{bmatrix} \tag{7.29}$$

and the construction of $\bar{W}_i(t)$, $i = 1, 2, 3, 4$, and $\tilde{W}_i$, $i = 1, \ldots, r+1$, can refer to the structure of $\bar{Y}_i(t)$ and $\tilde{Y}_i$, respectively.

7.3.2 Explicit nonlinear MPC solution

The conventional MPC needs to solve a formulated optimisation problem to generate the control signal, where the control performance index is minimised with respect to the future control input over the prediction horizon. In this chapter, after the output is approximated by its Taylor expansion, the control profile can be defined as

$$\tilde{\mathbf{u}}(t+\tau) = \tilde{\mathbf{u}}(t) + \tau\tilde{\mathbf{u}}^{[1]}(t) + \cdots + \frac{\tau^r}{r!}\tilde{\mathbf{u}}^{[r]}(t), 0 \le \tau \le T \tag{7.30}$$

Thereby, the helicopter outputs depend on the control variables $\bar{\mathbf{u}} = \{\tilde{\mathbf{u}}, \tilde{\mathbf{u}}^{[1]}, \ldots, \tilde{\mathbf{u}}^{[r]}\}$.

By recalling the performance index (7.6), the output and reference approximation (7.9) and (7.27), one has:

$$J = \frac{1}{2}(\bar{Y}(t) - \bar{W}(t))^T \begin{bmatrix} \mathcal{T}_1 & \mathcal{T}_2 \\ \mathcal{T}_2^T & \mathcal{T}_3 \end{bmatrix} (\bar{Y}(t) - \bar{W}(t)) \tag{7.31}$$

where

$$\bar{Y}(t) = \begin{bmatrix} \bar{Y}_1(t)^T & \cdots & \bar{Y}_4(t)^T | \tilde{Y}_1(t)^T & \cdots & \tilde{Y}_r(t)^T \end{bmatrix}^T, \tag{7.32}$$

$$\bar{W}(t) = \begin{bmatrix} \bar{W}_1(t)^T & \cdots & \bar{W}_4(t)^T | \tilde{W}_1(t)^T & \cdots & \tilde{W}_{r+1}(t)^T \end{bmatrix}^T, \tag{7.33}$$

$$\mathcal{T}_1 = \int_0^T T_f^T Q T_f d\tau, \tag{7.34}$$

$$\mathcal{T}_2 = \int_0^T T_f^T Q T_s d\tau, \tag{7.35}$$

and

$$\mathcal{T}_3 = \int_0^T T_s^T Q T_s d\tau. \tag{7.36}$$

Therefore, instead of minimising the performance index (7.6) with respect to control profile $\mathbf{u}(t+\tau)$, $0 < \tau < T$ directly, it is able to minimise the approximated index (7.31) with respect to $\bar{\mathbf{u}}$, where the necessary condition for the optimality is given by

$$\frac{\partial J}{\partial \bar{\mathbf{u}}} = 0 \tag{7.37}$$

After solving the nonlinear equation (7.37), it is able to obtain the optimal control variables $\bar{\mathbf{u}}^*$ to construct the optimal control profile defined by Eq.(7.30). As in MPC only the current control in the control profile is implemented, the explicit solution is $\tilde{\mathbf{u}}^* = \tilde{\mathbf{u}}(t+\tau)$, for $\tau = 0$. The resulting controller is given by

$$\tilde{\mathbf{u}}^* = -A(\mathbf{x}, \mathbf{d})^{-1}(KM_\rho + M_1) \tag{7.38}$$

where $K \in \mathbb{R}^{4\times(\rho_1+\cdots+\rho_4)}$ is the first 4 row of the matrix $\mathcal{T}_3^{-1}\mathcal{T}_2^T \in \mathbb{R}^{4(r+1)\times(\rho_1+\cdots+\rho_4)}$ where the ijth block of $\mathcal{T}_2$ is of $\rho_i \times 4$ matrix, and all its elements are zeros except the ith column is given by

$$\begin{bmatrix} q_i \frac{T^{\rho_i+j}}{(\rho_i+j-1)!(\rho_i+j)} & \cdots & q_i \frac{T^{2\rho_i+j-1}}{(\rho_i+j-1)!(\rho_i-1)!(2\rho_i+j-1)} \end{bmatrix}^T \tag{7.39}$$

for $i = 1,2,3,4$ and $j = 1,2,\ldots,r+1$, and ijth block of $\mathcal{T}_3$ is given by

$$\mathrm{diag}\left\{ q_1 \frac{T^{2\rho_1+i+j-1}}{(\rho_1+i-1)!(\rho_1+j-1)!(2\rho_1+i+j-1)}, \cdots, q_4 \frac{T^{2\rho_4+i+j-1}}{(\rho_4+i-1)!(\rho_4+j-1)!(2\rho_4+i+j-1)} \right\} \tag{7.40}$$

for $i,j = 1,2,\ldots,r+1$; the matrix $M_\rho \in \mathbb{R}^{\rho_1+\cdots+\rho_4}$ and matrix $M_i \in \mathbb{R}^4$ are defined as:

$$M_\rho = \begin{bmatrix} \bar{Y}_1(t)^T \\ \vdots \\ \bar{Y}_4(t)^T \end{bmatrix} - \begin{bmatrix} \bar{W}_1(t)^T \\ \vdots \\ \bar{W}_4(t)^T \end{bmatrix} \tag{7.41}$$

and

$$M_i = \begin{bmatrix} L_f^{\rho_1+i-1}h_1(t) \\ L_f^{\rho_2+i-1}h_2(t) \\ \vdots \\ L_f^{\rho_4+i-1}h_4(t) \end{bmatrix} - \tilde{W}_i(t)^T, i = 1,2,\ldots,r+1. \tag{7.42}$$

The detailed derivation and closed-loop stability can refer to [20]. The overall controller structure is shown in Fig.7.1.

The system has a trivial zero dynamics as $\rho_1+\rho_2+\rho_3+\rho_4 = 14$, which is the order of the helicopter dynamics plus the dynamic extension. If the disturbance terms are set to zero, the controller is equivalent to that designed using the nominal model. The information of disturbances are hold in the controller to eliminate their influences.

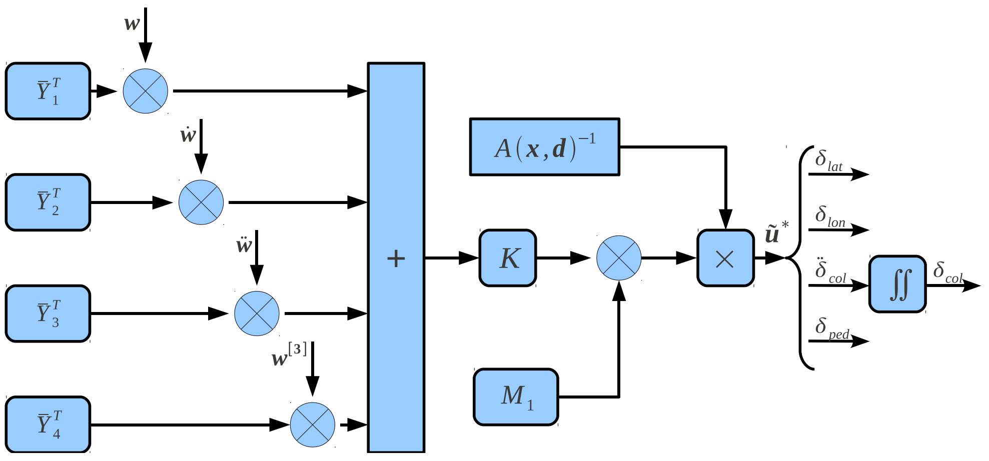


Figure 7.1: ENMPC structure

7.3.3 Command prefilter

When the ENMPC is applied for trajectory tracking of autonomous helicopters, not only the reference trajectory is required, but the higher derivatives of the reference trajectory with respect to time are also needed in the prediction. Although this can be achieved by using various modern path planning algorithms, there are still some applications where the dedicated path generator is not available. In these cases the reference is more likely to be designed comprising only the demanded helicopter position and the associated heading angle. To address this problem, a simple but effective method of low-pass prefilter (7.43) is adopted as shown in Fig. 7.2.

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{7.43}$$

Given the appropriate parameters ζ and ω, the command prefilter can provide first and second derivatives of the original reference, which is adequate for a smooth trajectory tracking [15].

In order to implement the ENMPC strategy the disturbances must be available, which is unrealistic for helicopter flight. Next section will introduce a nonlinear disturbance observer to estimate these unavailable disturbances.

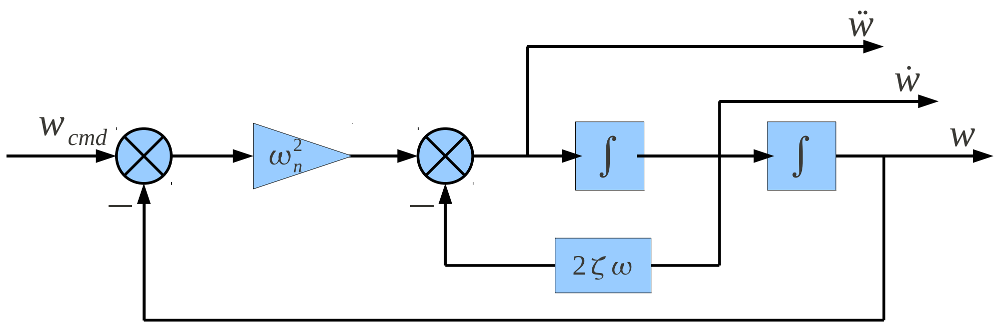


Figure 7.2: Command prefilter

7.4 Disturbance observer based control

7.4.1 Disturbance observer

For a system like a small-scale helicopter, precisely modelling its dynamics or directly measuring the disturbances acting on it is very challenging. However, the disturbance observer technique provides an alternative way to estimate them. In this section, a nonlinear disturbance observer is introduced to estimate the lumped unknown disturbances d in the general form of helicopter model (7.5). The disturbance observer [17] is given as follows (scalar variables are used for the sake of simplicity),

$$\begin{aligned} \hat{d} &= z + p(x) \\ \dot{z} &= -l(x)g_2(x)z - l(x)(g_2(x)p(x) + f(x) + g_1(x)u) \end{aligned} \tag{7.44}$$

where $\hat{d} = [\ \hat{d}_x \quad \hat{d}_y \quad \hat{d}_z \quad \hat{d}_{lat} \quad \hat{d}_{lon} \quad \hat{d}_{ped}\ ]^T$ is the estimation of disturbances; z is the internal state of the nonlinear observer, $p(x)$ is a nonlinear function to be designed, and $l(x)$ is the nonlinear observer gain given by

$$l(x) = \frac{\partial p(x)}{\partial x} \tag{7.45}$$

In this observer, the estimation error is defined as $e_d = d - \hat{d}$. Under the assumption that the disturbance is slowly varying compared to the observer dynamics and by combining Eq.(7.44)-Eq.(7.45) and Eq.(7.5), it can be shown that the estimation error has the following property:

$$\begin{aligned} \dot{e}_d &= \dot{d} - \dot{\hat{d}} \\ &= -\dot{z} - \frac{\partial p(x)}{\partial x}\dot{x} \\ &= -l(x)g_2(x)e_d \end{aligned} \tag{7.46}$$

Therefore, $\hat{d}(t)$ approaches $d(t)$ exponentially if $p(x)$ is choose such that Eq.(7.46) is globally exponentially stable for all $x \in \mathbb{R}^n$.

The design of a disturbance observer essentially is to chose an appropriate gain $l(x)$ and associated $p(x)$ such that the convergence of estimation error is guaranteed. Thereby, there exist a considerable degree of freedom. Since the disturbance input matrix $g_2(x)$ for the helicopter model is a constant matrix as:

$$g_2(x) = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} & 0_{3\times3} \\ 0_{3\times3} & \begin{matrix} L_{lat} & L_{lon} & 0 \\ M_{lat} & M_{lon} & 0 \\ 0 & 0 & N_{ped} \end{matrix} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}, \tag{7.47}$$

It is possible to choose $l(x)$ as a constant matrix such that all the eigenvalues of matrix $-l(x)g_2(x)$ have negative real part. Next, integrating $l(x)$ with respect to the helicopter state x yields $p(x) = l(x)x$. The observer gain matrix $l(x)$ corresponding to g_2 is designed in the form:

$$l(x) = \begin{bmatrix} 0_{3\times3} & L_1 & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & L_2 & 0_{3\times3} \end{bmatrix} \tag{7.48}$$

where matrix $L_1 = \text{diag}\{l_1, l_2, l_3\}$ and

$$L_2 = \text{diag}\{l_4, l_5, l_6\} \begin{bmatrix} L_{lat} & L_{lon} & 0 \\ M_{lat} & M_{lon} & 0 \\ 0 & 0 & N_{ped} \end{bmatrix}^{-1} \tag{7.49}$$

for $l_i > 0$, $i = 1, \ldots, 6$. Thereby, $-l(x)g(x) = -\text{diag}\{l_1, \ldots, l_6\}$. Form the above analysis, it can be seen that the convergence of the disturbance observer is guaranteed regardless of the helicopter state.

7.4.2 Composite controller

External force and torque disturbances generated by wind, turbulences and other factors coupled with modelling errors and uncertainties may significantly degrade the helicopter's tracking performance and may even cause instability unless their influence has been properly taken into account in the system design. It shall be noted that in the previous derivation of the ENMPC, the lumped disturbances appear in the control law. Therefore, once the disturbance observer provides the estimation of disturbances, the ENMPC controller takes into account the disturbances by replacing the disturbance by their estimation and achieves desired tracking performance. Let $d_f = [d_x \quad d_y \quad d_z]^T$ and $d_e = [d_{lat} \quad d_{lon} \quad d_{ped}]^T$. The composite controller law using the estimated disturbances is given in

$$\tilde{\mathbf{u}} = -A(\mathbf{x}, \hat{d}_f)^{-1}(K\hat{M}_\rho + \hat{M}_1) \tag{7.50}$$

where, the hatted variables denote the estimated values. If the trim errors is considered in the helicopter dynamics, the composite control becomes

$$\mathbf{u} = \tilde{\mathbf{u}} - \hat{\mathbf{u}}_0 \tag{7.51}$$

where $\hat{\mathbf{u}}_0 = [\hat{d}_{lat} \quad \hat{d}_{lon} \quad 0 \quad \hat{d}_{ped}]^T$ is the control trim error estimated by the disturbance observer. The composite controller structure is illustrated in Fig.7.3.

7.5 Stability analysis

The stabilities of the ENMPC and the disturbance observer are guaranteed in their design procedures outlined in Section 7.3 and 7.4, respectively. However, the stability of the closed-loop system still needs to be examined, because the true disturbances are replaced by their estimation in the composite controller (7.51), and there are interactions among the ENMPC, the disturbance observer and the helicopter dynamics.

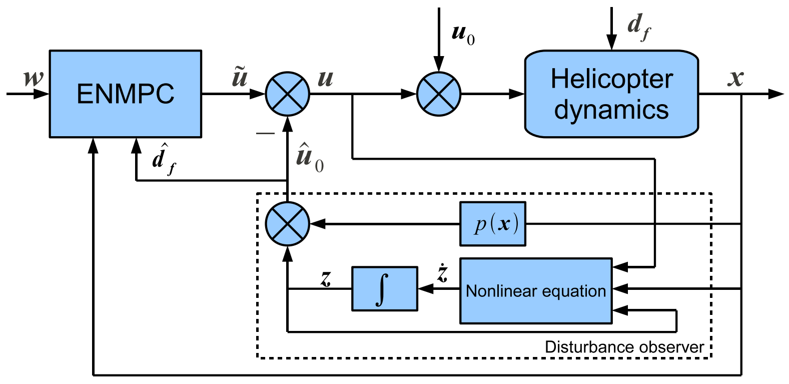


Figure 7.3: Composite controller structure

The closed-loop dynamics under the composite control law can be examined by applying Eq.(7.51) into helicopter model (7.5). Since the resulting system is too complicated, it is worth to define a new coordinate to describe the closed-loop system. First, let position tracking error defined as:

$$z_p^0 = [x - w_1 \quad y - w_2 \quad z - w_3]^T \tag{7.52}$$

then it first derivative can be defined as:

$$\dot{z}_p^0 = z_p^1 = [\dot{x} - \dot{w}_1 \quad \dot{y} - \dot{w}_2 \quad \dot{z} - \dot{w}_3]^T \tag{7.53}$$

where the expressions of $\dot{x}$, $\dot{y}$ and $\dot{z}$ are given in Eq.(7.14). Since the real disturbances are replaced by their estimations in the closed-loop system, the next state can be defined as:

$$z_p^2 = \mathbf{R}_b^i \begin{bmatrix} \hat{d}_x \\ \hat{d}_y \\ T + \hat{d}_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \begin{bmatrix} \ddot{w}_1 \\ \ddot{w}_2 \\ \ddot{w}_3 \end{bmatrix}, \tag{7.54}$$

By following the same procedure as Eq.(7.14) and (7.16), combining Eq(7.53) and

(7.54) gives $\dot{z}_p^1 = z_p^2 + \mathbf{R}_b^i \cdot e_{d_f}$, where $e_{d_f} = d_f - \hat{d}_f$. Similarly, z_p^3 is defined as:

$$z_p^3 = \mathbf{R}_b^i \hat{\omega} \begin{bmatrix} \hat{d}_x \\ \hat{d}_y \\ T + \hat{d}_z \end{bmatrix} + \mathbf{R}_b^i \begin{bmatrix} 0 \\ 0 \\ Z_w \dot{w} + Z_{col} \dot{\delta}_{col} \end{bmatrix} - \begin{bmatrix} \dddot{w}_1 \\ \dddot{w}_2 \\ \dddot{w}_3 \end{bmatrix} \tag{7.55}$$

From Eq.(7.54) and (7.55) and recalling observer dynamics (7.46), it can be observed that

$$\begin{aligned} \dot{z}_p^2 &= z_p^3 - \mathbf{R}_b^i \dot{e}_{d_f} \\ &= z_p^3 + \mathbf{R}_b^i L_1 e_{d_f} \end{aligned} \tag{7.56}$$

Repeat this procedure, z_p^4 is defined from Eq.(7.20) by using estimated disturbances, such that

$$\dot{z}_p^3 = z_p^4 + \mathbf{R}_b^i \hat{\omega} \cdot L_1 e_{d_f} \tag{7.57}$$

In addition, the heading tracking error and its derivatives are defined as $z_\psi^0 = \psi - w_4$, $z_\psi^1 = \dot{\psi} - \dot{w}_4$, where $\dot{\psi}$ is given in Eq.(7.15) and $z_\psi^2 = \ddot{\psi} - \ddot{w}_4$, where $\ddot{\psi}$ is provided in Eq(7.17).

Finally, by invoking Eq(7.22) and the definitions of z_p^4 and z_ψ^2, one has

$$\begin{aligned} \begin{bmatrix} z_p^4 \\ z_\psi^2 \end{bmatrix} &= \hat{M}_1 + A(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f})(\mathbf{u} + \mathbf{u}_0) \\ &= \hat{M}_1 + A(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f})(-A(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f})^{-1}(K\hat{M}_\rho + \hat{M}_1) - \hat{\mathbf{u}}_0 + \mathbf{u}_0) \\ &= -K\hat{M}_\rho + A(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f}) e_{u_0} \end{aligned} \tag{7.58}$$

where, $e_{u_0} = \mathbf{u}_0 - \hat{\mathbf{u}}_0$ and K has the form:

$$K = \begin{bmatrix} k_{11} \cdots k_{14} & 0_{1\times4} & 0_{1\times4} & 0_{1\times2} \\ 0_{1\times4} & k_{21} \cdots k_{24} & 0_{1\times4} & 0_{1\times2} \\ 0_{1\times4} & 0_{1\times4} & k_{31} \cdots k_{34} & 0_{1\times2} \\ 0_{1\times4} & 0_{1\times4} & 0_{1\times4} & k_{41} \cdots k_{42} \end{bmatrix} \tag{7.59}$$

By recalling the definition of $\hat{M}_\rho$ in Eq.(7.41), Eq.(7.58) can be further written as:

$$\begin{bmatrix} \dot{z}_p^3 \\ \dot{z}_\psi^1 \end{bmatrix} = \begin{bmatrix} K_1 z_p^0 + K_2 z_p^1 + K_3 z_p^2 + K_4 z_p^3 \\ k_{41} z_\psi^0 + k_{42} z_\psi^1 \end{bmatrix} + \begin{bmatrix} \mathbf{R}_b^i \hat{\omega} \cdot L_1 e_{d_f} \\ 0 \end{bmatrix} + A(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f}) e_{u_0} \tag{7.60}$$

where $K_i = \mathrm{diag}(k_{1i}, k_{2i}, k_{3i})$, for $i = 1, 2, 3, 4$.

Summarizing Eq.(7.52)-(7.60) gives a linear form of the closed-loop system:

$$\begin{bmatrix} \dot{z}_p^0 \\ \dot{z}_p^1 \\ \dot{z}_p^2 \\ \dot{z}_p^3 \\ \dot{z}_\psi^0 \\ \dot{z}_\psi^1 \end{bmatrix} = \underbrace{\begin{bmatrix} 0_{3\times3} & \mathbf{I}_3 & 0_{3\times3} & 0_{3\times3} & 0 & 0 \\ 0_{3\times3} & 0_{3\times3} & \mathbf{I}_3 & 0_{3\times3} & 0 & 0 \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & \mathbf{I}_3 & 0 & 0 \\ K_1 & K_2 & K_3 & K_4 & 0 & 0 \\ 0_{1\times3} & 0_{1\times3} & 0_{1\times3} & 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0_{1\times3} & 0_{1\times3} & 0_{1\times3} & k_{41} & k_{41} \end{bmatrix}}_{A_z} \underbrace{\begin{bmatrix} z_p^0 \\ z_p^1 \\ z_p^2 \\ z_p^3 \\ z_\psi^0 \\ z_\psi^1 \end{bmatrix}}_{z} + \underbrace{\begin{bmatrix} 0_{3\times1} \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ 0_{1\times1} \\ \epsilon_5 \end{bmatrix}}_{\epsilon} \tag{7.61}$$

or, compactly

$$\dot{z} = A_z z + \epsilon \tag{7.62}$$

where $\epsilon_1 = \mathbf{R}_b^i \cdot e_{d_f}$, $\epsilon_2 = \mathbf{R}_b^i \cdot L_1 e_{d_f}$, $\epsilon_3 = \mathbf{R}_b^i \hat{\omega} \cdot L_1 e_{d_f} + \overline{A}(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f}) e_{u_0}$ and $\epsilon_5 = \underline{A}(\mathbf{x}, \hat{\mathbf{d}}_\mathbf{f}) e_{u_0}$. All these terms depend on the helicopter states and estimation errors e_d.

System (7.61) can be classified as a cascade system:

$$\begin{aligned} \dot{z} &= f_1(z) + \epsilon(x, e_d) e_d \\ \dot{e}_d &= f_2(e_d) \end{aligned} \tag{7.63}$$

where the upper system is Eq(7.61) and the lower system is the observer dynamics (7.46).

When estimation errors are zeros, the upper system $\dot{z} = f_1(z)$ reduces to a linear system $\dot{z} = A_z z$. In this case, its globally asymptotic stability can be guaranteed by proper choosing MPC gain K such that A_z is Hurwitz, which is assured in the ENMPC design. On the other hand, the globally asymptotic stability of the lower system is guaranteed during the design of disturbance observer. Therefore, the closed-loop system under the composite control law is at least locally asymptotic stable according to [45]. Moreover, it is able to extend the above result further by introducing the following lemma.

Lemma 7.1 ([90]). *If assumptions A7.1-A7.3 below are satisfied then the cascaded system (7.63) is globally uniformly asymptotically stable.*

Assumption 7.1. The system $\dot{z} = f_1(z)$ is globally uniformly asymptotically stable with a Lyapunov function $V(z)$, $V : \mathbb{R}^n \to \mathbb{R}$ positive definite (that is

$V(0) = 0$ and $V(z) > 0$ for all $z \neq 0$) and proper which satisfies

$$\left\| \frac{\partial V}{\partial z} \right\| \|z\| \leq c_1 V(x), \forall \|z\| \geq \eta \tag{7.64}$$

where $c_1 > 0$ and $\eta > 0$.

Assumption 7.2. The function $\epsilon(x, e_d)$satisfies

$$\|\epsilon(x, e_d)\| \leq \alpha_1(\|e_d\|) + \alpha_2(\|e_d\|) \|z\| \tag{7.65}$$

where $\alpha_1, \alpha_2 : \mathbb{R} \to \mathbb{R}$ are continuous.

Assumption 7.3. Equation $\dot{e}_d = f_2(e_d)$ is globally uniformly asymptotically stable and for all $t \geq t_0$

$$\int_{t_0}^{\infty} \|e_d(t)\| \, \mathrm{d}t \leq \beta(\|e_d(t_0)\|) \tag{7.66}$$

where function β is a class $\mathcal{K}$ function.

The rigorous proof of lemma 7.1 is given in [90]. The basic idea is first to show that the upper system of cascade system does not escape in finite time and are bounded for $t > t_0$ with the condition that the input vector $\epsilon(x, e_d)$ grows linearly and at the fastest in the state z. Then it needs to show that as $t \to \infty$, estimation error $e_d \to 0$ and $z \to 0$ due to the global asymptotic stability of $\dot{z} = f_1(z)$.

Theorem 7.2. *Given that the reference trajectory* $\mathbf{w}$*, its first* ρ_i *derivatives, and disturbance* $\mathbf{d}$ *are bounded, the closed-loop system (7.58) under the composite control law (7.51) is globally asymptotically stable.*

Proof. By using Lemma 7.1, for closed-loop system (7.58) in the cascade form (7.63), if assumptions A7.1-A7.3 are satisfied, the proof will then be completed.

First, A7.1 is satisfied due to the fact that $\dot{z} = f_1(z) = A_z z$ and A_z is Hurwitz. Then, it needs to investigate the boundness on $\epsilon(x, e_d)$ in terms of $\|z\|$ and $\|e_d\|$.

From their definitions, there are

$$\|\epsilon_1\| \leq \|e_d\| \tag{7.67}$$
$$\|\epsilon_2\| \leq \|L_1\| \, \|e_d\| \tag{7.68}$$
$$\|\epsilon_3\| \leq \|\hat{\omega}\| \, \|L_1\| \, \|e_{d_f}\| + \|\overline{A}(\cdot,\cdot)\| \, \|e_d\| \tag{7.69}$$
$$\|\epsilon_5\| \leq \|\underline{A}(\cdot,\cdot)\| \, \|e_d\| \tag{7.70}$$

The skew-matrix $\hat{\omega}$ can be seen consisted of nominal state decided by the reference command and the error state, i.e. $\hat{\omega} = \hat{\omega}_c + \hat{\omega}_e$. The former one is bounded as the bounded command, and the latter one is bounded by tracking error $\|z\|$. Therefore, there exist two constant $b_1 > 0$ and $b_2 > 0$, such that $\|\hat{\omega}\| \leq b_1 + b_2 \|z\|$. Moreover, $\|\overline{A}(\cdot,\cdot)\|$ linearly depends on $\hat{d}$ and state T. Due to d is bounded and disturbance observer is globally exponentially stable, $\hat{d}$ is also bounded. On the other hand, T is bounded by $\|z\| + \|d\| + g$ using Eq(7.16). Hence, there is $\|\overline{A}(\cdot,\cdot)\| \leq b_3 + b_4 \|z\|$, for some $b_3 > 0$ and $b_4 > 0$. Then the bound on ϵ_3 can be write as $\|\epsilon_3\| \leq \beta_1 \|e_d\| + \beta_2 \|e_d\| \, \|z\|$, for some $\beta_1 > 0$ and $\beta_2 > 0$. At last, following the same fashion $\epsilon_5 \leq b_5 \|e_d\|$, for some $b_5 > 0$ if pitch angle $\theta \neq \pm\pi/2$. Combining bounds on $\|\epsilon_i\|$, $i = 1, \ldots, 5$ gives

$$\begin{aligned} \|\epsilon\| &\leq \|\epsilon_1\| + \cdots + \|\epsilon_5\| \\ &\leq \gamma_1 \|e_d\| + \gamma_2 \|e_d\| \, \|z\| \end{aligned} \tag{7.71}$$

where $\gamma_1 > 0$ and $\gamma_2 > 0$. Thus, A7.1 is satisfied.

Finally, as lower system $\dot{e}_d = f_2(e_d)$ is globally exponentially stable, A7.3 is satisfied. □

7.6 Simulation

Several numerical simulations have been carried out to verify the proposed control framework. Firstly, a comparison between the ENMPC and the MPC using online optimisation from Chapter 5 is carried out. Secondly, tracking performance of ENMPC and DOBC is studied under the uncertainties and constant wind gust. Thirdly, the capability of disturbance rejection of the proposed control scheme is investigated. The ENMPC is designed with the prediction horizon $T = 4s$, control order $r = 4$ and $Q = \text{diag}\{1, 1, 1, 1\}$. The disturbance observer gain is simply set to $L_1 = \text{diag}\{10, 10, 10\}$, and $L_2 = \text{diag}\{5, 5, 5\}$. The command prefilter parameters are chosen as $\zeta = 0.7$ and $\omega_b = 10rad/s$.

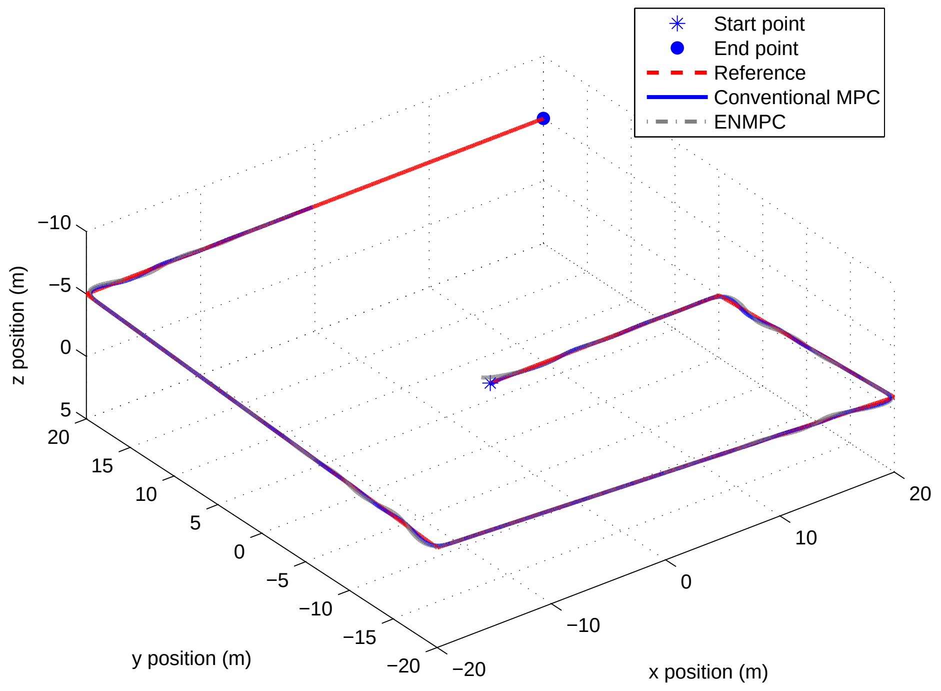


Figure 7.4: Tracking performance under different MPC schemes (Sim 1)

In the first simulation, the helicopter needs to track a multi-section reference connected by abrupt turns. The helicopter tracking performance is given in Fig.7.4. It can be seen that the helicopter under both conventional MPC and ENMPC is able to track the reference with very satisfactory performance. However, in the conventional MPC the average time for solving the formulated OP is about 0.2s, which restricts the control bandwidth to 5Hz. The ENMPC tackles this problem by directly using the explicit solution, therefore it can easily reach the required control bandwidth.

In the second simulation, it is assumed that there are 20% uncertainties on the model parameters. Moreover, there is a constant wind disturbance with speed of $5m/s$ acting on the helicopter. Disturbance forces caused by wind are calculated using velocity damping coefficients X_u and Y_v, such that $d_x = X_u d_u$

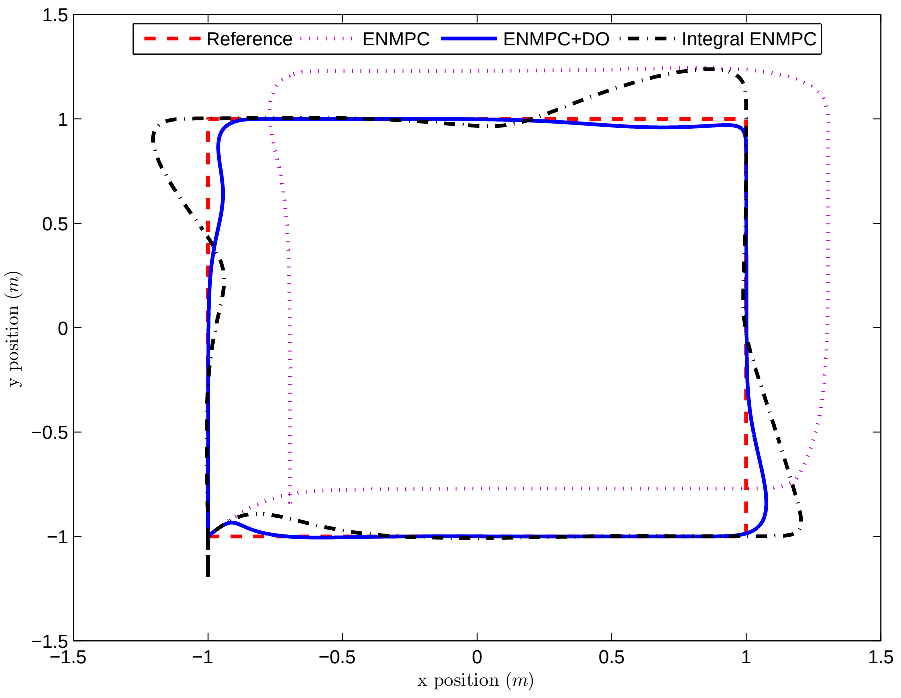


Figure 7.5: Tracking performance with uncertainties and constant wind (Sim 2)

and $d_y = Y_v d_v$, where d_u and d_v are wind compoents along helicopter axes. The helicopter is required to track a square trajectory under the control of the original ENMPC, an ENMPC with integral action and the composite ENMPC with DOBC. The tracking results are illustrated in Fig.7.5 with the control signals given in Fig.7.6.

It can be seen from the tracking results that the ENMPC is able to deal with uncertainties and achieve satisfactory tracking, but it cannot compensate the steady state error mainly caused by the wind disturbance. In contrast, the ENMPC with integral action cancels the steady state error, but it has side-effects like overshoot and aggressive control commands (see Fig.7.5). Obviously, the ENMPC augmented by DOBC outperforms the other two control strategies to a large extent in delivering accurate tracking and smooth control signals. This is because the disturbances is taken into account in the control scheme.

Although the disturbance observer is designed under the assumption of "slow" disturbance, i.e. $\dot{d} \approx 0$, it can be shown that DOBC is able to handle the random turbulence as it can be designed quick enough for estimation. In the

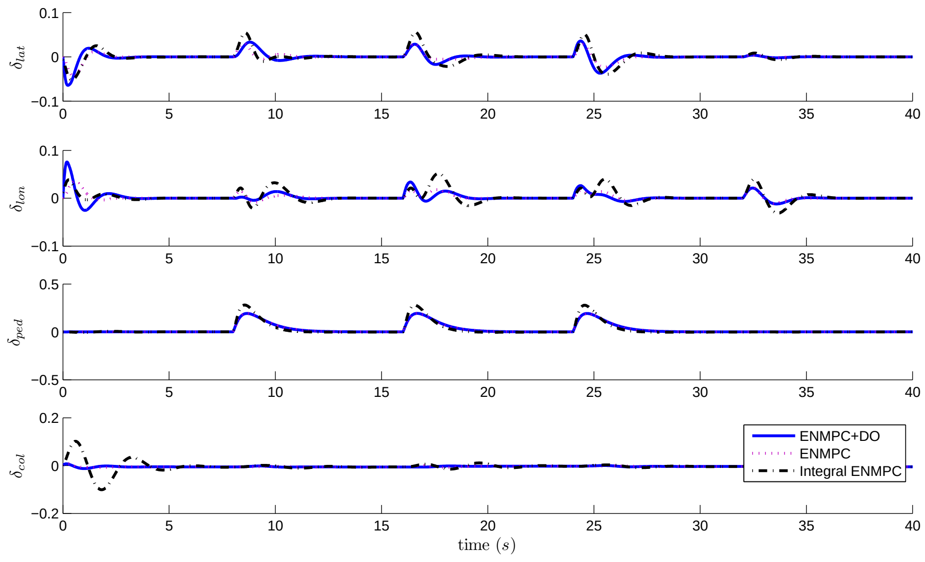


Figure 7.6: Control signals (Sim 2)

third simulation, a wind turbulence is introduced with its components along the fuselage axes are independently excited by correlated Gauss Markov processes [32]:

$$\begin{bmatrix} \dot{d}_u \\ \dot{d}_v \end{bmatrix} = \begin{bmatrix} -1/\tau_c & 0 \\ 0 & -1/\tau_c \end{bmatrix} \begin{bmatrix} d_u \\ d_v \end{bmatrix} + \rho B_w \begin{bmatrix} \mu_u \\ \mu_v \end{bmatrix} \tag{7.72}$$

where μ_u and μ_v are independent white noise with zero mean, τ_c is the correlation time of the wind $\mu_u = \mu_v = 6m/s$, B_w is the turbulence input identity matrix, and $\rho = 1/2$ is the scalar weighting factor. The disturbance fed into the helicopter is plotted in Fig.7.7 with respect to time.

The third simulation contains a forward step movement at $10s$ and a left step at $25s$. The tracking results are presented in Fig.7.8 along with the control signals in Fig.7.9. Again, the ENMPC with DOBC has a much better performance against that of ENMPC which is compromised by wind turbulence. On the other hand, the disturbance observer also works well to estimate the random disturbance. As shown in Fig.7.10 the estimated disturbance is very close to the artificial disturbance fed into the helicopter.

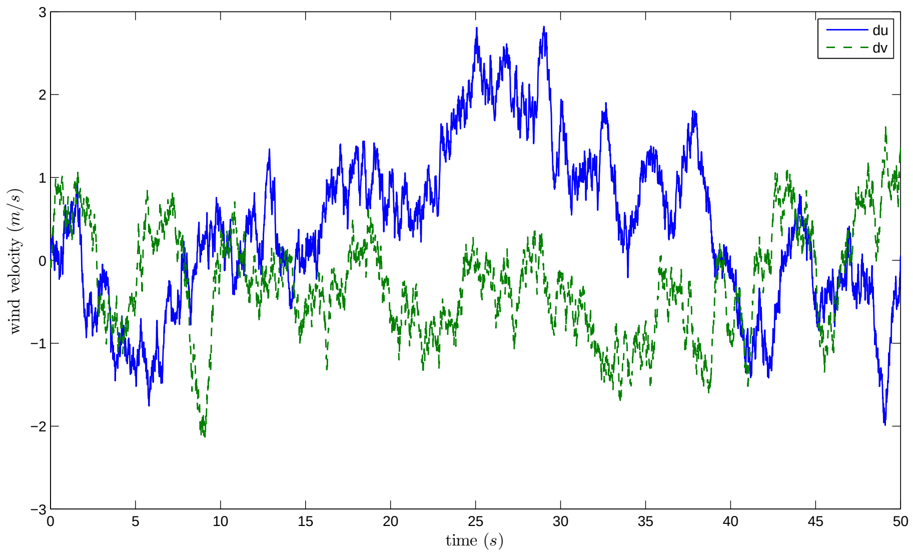


Figure 7.7: Random turbulence in fuselage axes

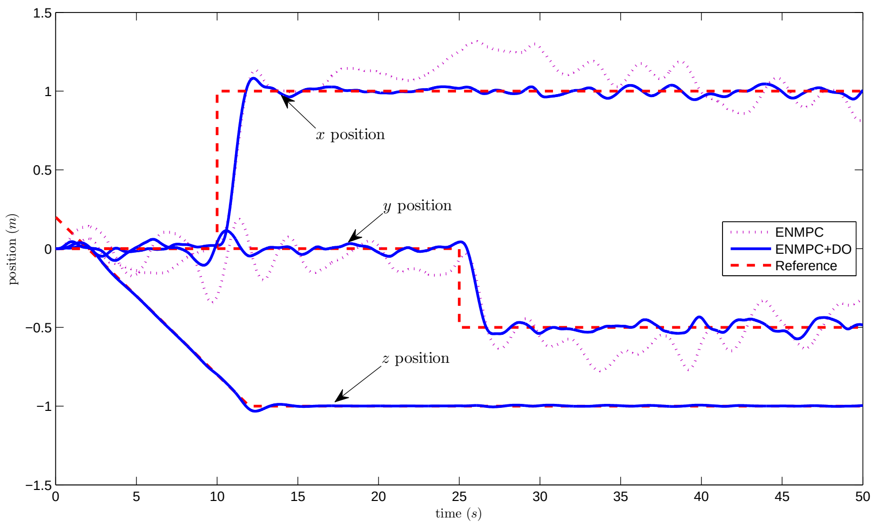


Figure 7.8: Tracking performance under random turbulence (Sim 3)

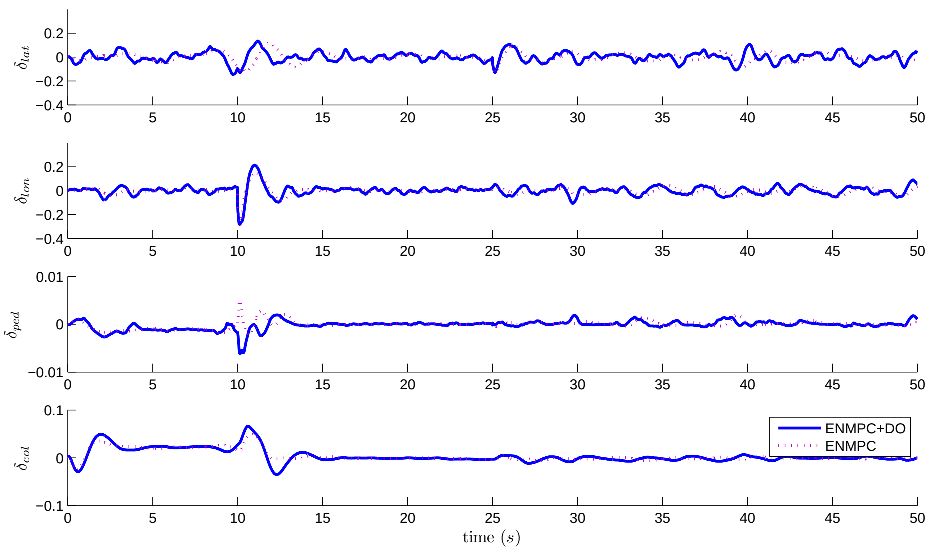


Figure 7.9: Control signals (Sim 3)

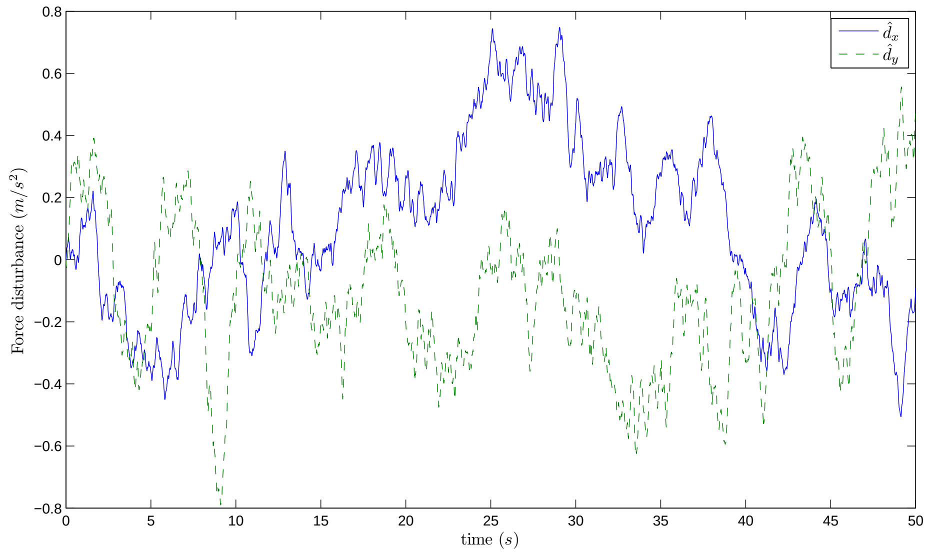


Figure 7.10: Estimated disturbance in fuselage axes

7.7 Flight test

After the initial verification using simulations, a number of flight tests have been conducted to investigate the performance of the proposed control scheme on real helicopters, where the uncertainties and unmodelled dynamics naturally exist. The flight tests presented here include a hovering and perturbation test and a pirouette manoeuvre.

In the first test, the helicopter was required to take-off and hover at the origin at height of 0.5m. A wind perturbation was then applied on the helicopter by posing a fan in front of the helicopter. The test result is given in Fig.7.11. In the test, the helicopter was first under the control of ENMPC to perform take-off and hovering. It can be seen that the ENMPC stabilised the helicopter but with a steady state error due to the mismatch between the model used for ENMPC design and the real helicopter dynamics. At 25 seconds of the test, the disturbance observer switched on and the composite controller took action to bring the helicopter to the setpoint. At 60 seconds, the fan was turned on to generate the wind gust. The average wind speed is about 3m/s, which is significant strong for the Trex-250 test helicopter with a small dimension. This can be observed from the attitude history in Fig.7.12, where the magnitude of pitch and roll angles of the helicopter dramatically surges after wind gust is applied. However, the composite controller exhibited an outstanding performance under the wind gust and maintained the helicopter position very well. The force disturbances estimated by disturbance observer are given in Fig.7.13, and the control signals are illustrated in Fig.7.14

It is also interesting to see where the disturbances come from without external wind gust, and this will also explain why ENMPC based on the nominal model cannot achieve asymptotic tracking if the helicopter is not trimmed properly. By recalling helicopter dynamics model (7.1) and considering steady-state model, we have

$$\begin{aligned} 0 &= -g\sin\theta_0 + d_x \\ 0 &= g\cos\theta_0\sin\phi_0 + d_y \end{aligned} \tag{7.73}$$

where ϕ_0 and θ_0 are the trim attitude (also known as sitting angle) depending on the particular helicopter. The trim attitude may be attributed to asymmetrical structure and model uncertainties. Their values are small so that they usually

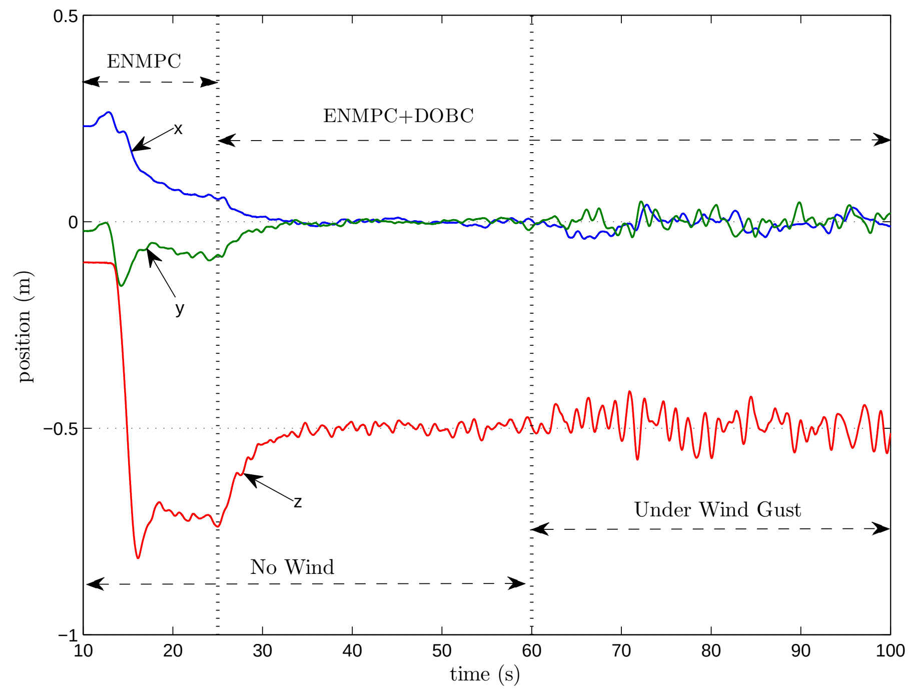


Figure 7.11: Helicopter position (Flight test 1)

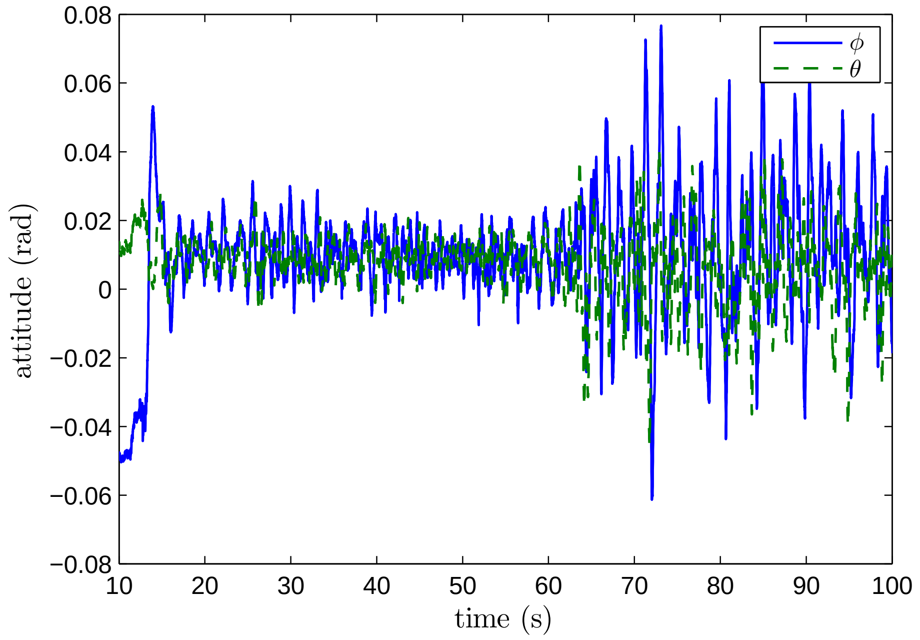


Figure 7.12: Helicopter attitude (Flight test 1)

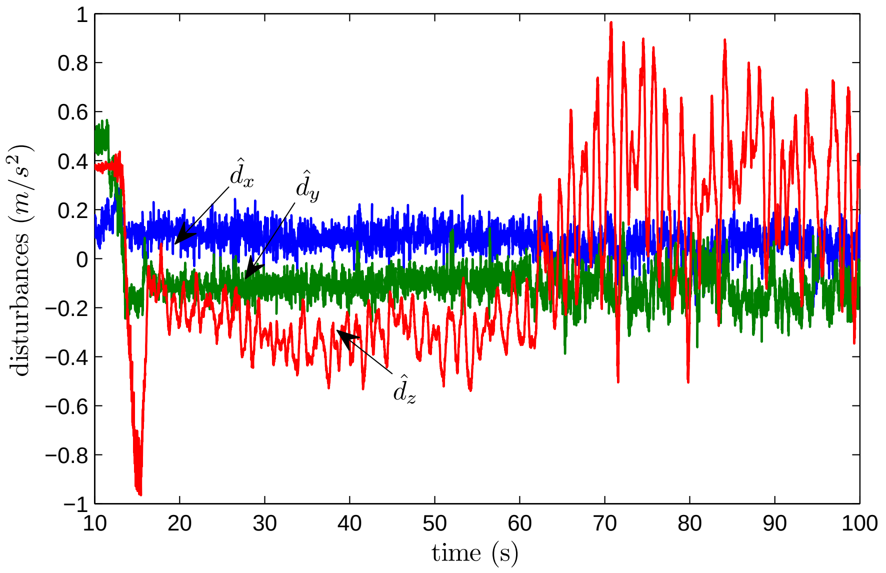


Figure 7.13: Disturbance observer outputs (Flight test 1)

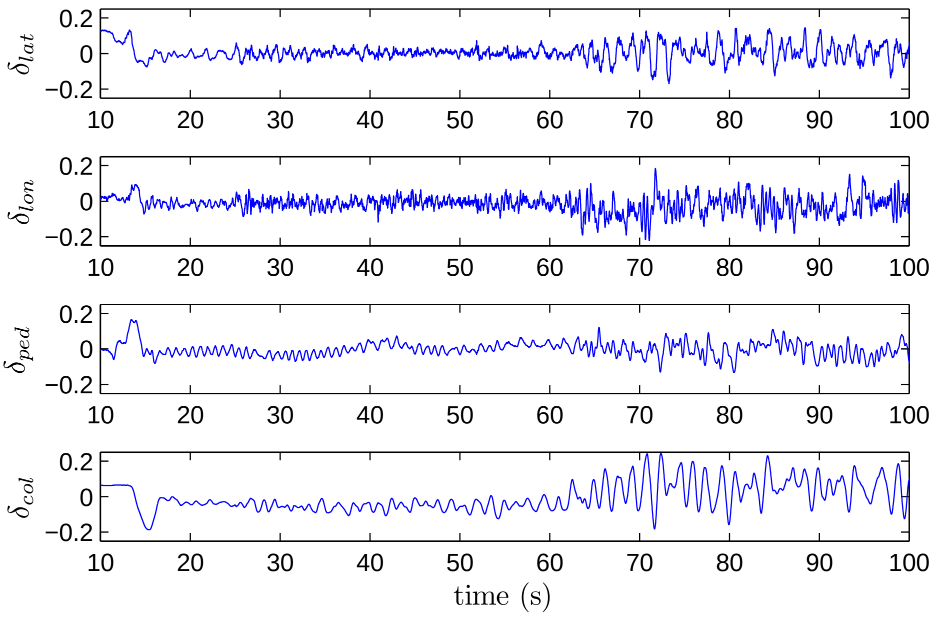


Figure 7.14: Control signals (Flight test 1)

are ignored in the theoretical analysis, but they do affect the actual control performance as they project vertical lift to longitudinal and lateral directions. This phenomena can be further explained by carefully examining the measurement from the flight test. Observing the attitude measurement in Fig.7.12 reveals the average roll and pitch angles are about 0.01rad, which contribute $0.1m/s^2$ and $-0.1m/s^2$ to d_x and d_y according to Eq.(7.73), respectively. The estimated d_x from observer is very close to our rough calculation, whereas the estimated d_y is smaller than what we expected. This is because that the tail rotor also generates lateral thrust that has not been taken into account in the nominal model. The above quantitative analysis gives a good confidence on the proposed disturbance observer.

The second flight test aimed to examine the capability of aerobatic manoeuvre. Unlike the conventional MPC being restricted to a low control bandwidth, the high bandwidth that ENMPC can achieve makes it a suitable candidate for controlling helicopter to perform complicated manoeuvres. In the flight test, the helicopter was controlled to perform a pirouette manoeuvre, in which helicopter started from the hovering position and flew along a straight line while pirouetting at a yaw rate of 120 deg/s. This is an extremely challenging flight pattern, because the lateral and longitudinal controls are strongly coupled by the rotation, and they have to coordinate with each other to achieve a straight progress. Besides, the varying position of the tail rotor with respect to the progress direction poses severe disturbances on the forward flight. The result from the flight test is shown in Fig.7.15 and the control signals are provided in Fig.7.16. It can be seen that the helicopter under the control of ENMPC executed the manoeuvre with a very high quality.

7.8 Summary

This chapter describes a composite control framework for trajectory tracking of autonomous helicopters. The nonlinear tracking control is achieved by an explicit MPC algorithm, which eliminates the computationally intensive online optimisation in the traditional MPC. On the implementation side, the introducing of disturbance observer solves the difficulties of applying model based control technique into the practical environment. The design of ENMPC provides a seamless way of integrating the disturbance information. On the other hand the

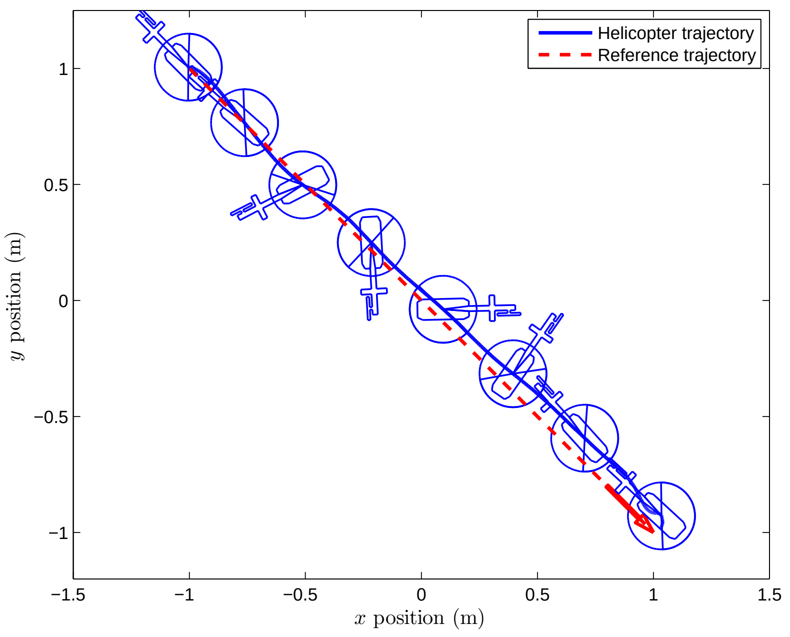


Figure 7.15: Pirouette manoeuvre results (Flight test 2)

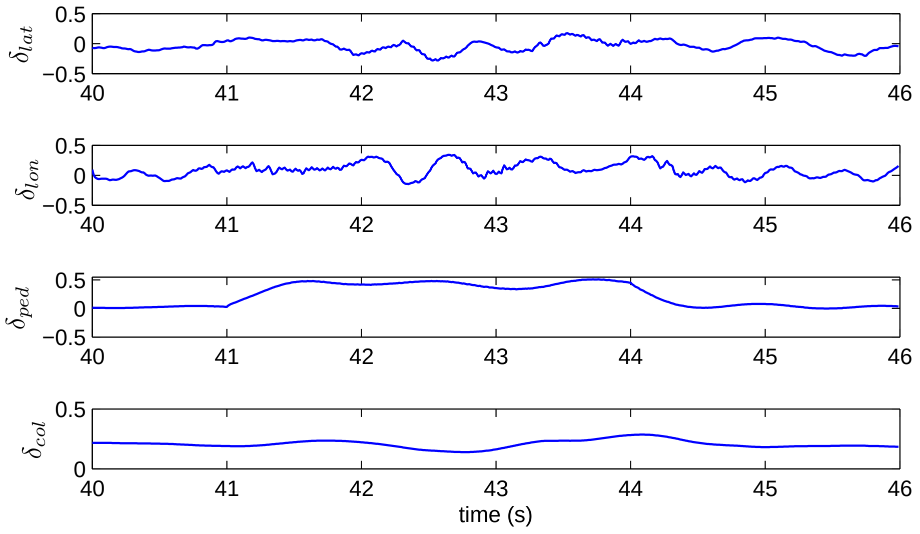


Figure 7.16: Control signals (Flight test 2)

robustness and disturbance attenuation of the controller are enhanced by the nonlinear disturbance observer.

Simulation and experiment results show promising performance of the combination of the ENMPC and DOBC. Apart from the reliable tracking that the proposed controller guarantees, it also has the ability of estimating the helicopter trim condition during the flight which helps controller to deal with the variation of the helicopter status like payload changing and component upgrades.

Chapter 8

Conclusions

8.1 Summary

This thesis presents the research work of applying advanced control to a miniature helicopter. Multiple disciplines are involved in this research, including mathematical analysis, software/hardware integration, and flight test operation, to achieve the following objects:

- to establish a systematic process of flight control design for miniature helicopters, including modelling, control analysis and design, and verification using flight tests.
- to demonstrate the practical feasibility of applying computationally intensive MPC algorithms into systems with fast dynamics such as helicopters.
- to improve the dynamic performance, e.g. trajectory tracking, aggressive manoeuvre, and disturbance attenuation, of an autonomous helicopter by using advanced control algorithms.

The research outcomes towards the study on the small-scale helicopter lie in two aspects: synthesis of the flight control system and development of advanced control algorithms. These contributions are summarised as follows:

Firstly, an indoor testbed for ground vehicles and miniature helicopters has been constructed by adopting commercial-off-the-shelf components. This testbed is able to observer the flight state of helicopters, carry out control algorithms and real-time control of small aerial/ground vehicles. All these functions are integrated in a Matlab/Simulink environment to facilitate the software development allowing the rapid prototyping of new control algorithms. The testbed has played a fundamental role in the research work because all the other activities, including system identification, control implementation and flight tests, are established upon it.

Secondly, a general model of a miniature helicopter has been developed that has a simple structure but can capture the main characteristic of helicopter dynamics. A specific system identification process has also been developed to identify the unknown parameters in the model for the Trex-250 helicopter. The proposed process exploits the model breakdown in conjunction with the corresponding flight patterns to improve the identification results. The resulting helicopter model has successfully served the following model based control design.

Thirdly, a MPC based control framework has been developed where the intensive computation burden of online optimisation is mitigated by the piecewise constant scheme and the control bandwidth is increased by the two-level control structure with a local linear controller. This framework can adopt the full dynamic model of a helicopter in the online prediction, so that it takes the advantages of general nonlinear MPC that tackles the complicated helicopter dynamics subject to state and control constraints. In particular, it integrates the outer-loop and inner-loop of a helicopter to deliver an integrated guidance and control fashion, and takes into account future reference to achieve smooth tracking. The stability analysis has also been carried out, which shows that the composite control framework consisting of the MPC and low-level controller is stable under mild condition.

Next, the local path planning and the corresponding tracking control have been solved in a hierarchical framework for miniature helicopters. In the planning part, the same two-level MPC based framework is employed. The MPC planning algorithm as the high-level controller is designed based on the helicopter's kinematic model and the potential field method, so that it can generate the kinematically feasible and obstacle-free trajectory and provide the corresponding guidance command. This guidance command is compensated by the low-level controller (guidance compensator) before propagated to the flight controller to follow. The flight controller is designed using the constrained linear MPC. It is able to track the guidance command without steady state error while maintaining the helicopter attitude in a linear range, which in turn guarantees the effectiveness of the planning using kinematic model.

Finally, a composite controller of ENMPC and DOBC has been developed for trajectory tracking of autonomous helicopters under disturbances and uncertainties. The controller is designed on a modified helicopter model where the unknown forces and torques caused by uncertainties and external factors are treated as lumped disturbances. An explicit nonlinear MPC is designed based the modified model by assuming the lumped disturbances are measurable. Then, a disturbance observer is designed to estimate these known disturbances and feeds them back to the MPC. The control design enhances the model by explicitly accounting for disturbances, which naturally leads to a better performance, especially in practice. On the theoretical side, the analysis has shown that the closed-loop system under the composite controller is globally asymptotically stable given that the disturbances and command reference are bounded.

8.2 Discussion and perspective

The indoor testbed proposed in this thesis provides an alternative way in delivering advanced flight control algorithms to UAV systems. Performing comprehensive simulations is a standard process in synthesis and verification of flight control design, but it will be more elaborate if flight tests can be conducted to provide a more realistic assessment of any new algorithms. This rapid prototype feature allows researchers to obtain more realistic performance of the newly developed algorithm, to build up confidence and reduce risks when deploying them on real UAVs. The future work related to the testbed is to expand its functions, potentially in the following two directions: hardware-in-the-loop simulation and automatic code generation. The first function is used to test the embedded systems for onboard calculation of the proposed algorithms, whereas the second function is to investigate the ability of converting Simulink programs into C codes so that they can be implemented on embedded computers rapidly.

The works on modelling and system identification have solved the fundamental problem in model based control design. In turn, the advanced control designed based on the identified model have successfully deployed on the target helicopter. The future work in this area is to take into account the main rotor speed of the helicopter to generate a more accurate model that can account for a wider flight envelop.

After the preparation works in the flight control development have been done, three advanced control algorithms based on MPC techniques are developed for miniature helicopters focusing on different aspects in operations of autonomous helicopters.

In applying MPC based algorithms to UAV applications, no matter for path planning or flight control, the most critical conflict is between the computational load and performance. High performance control can be achieved by adopting more elaborate vehicle models or including more environment information, but these will cause computational delay and consequently a low control bandwidth that cannot accommodate UAV's fast dynamics.

The MPC based control framework proposed in Chapter 5 solves this problem to a certain extend by using the combination of a piecewise constant scheme and a two-level control structure. This is a general framework that can adapt to different MPC algorithms that need to be implemented on systems with fast dynamics. As demonstrated in the thesis, this framework is able to control miniature helicopters with complicated nonlinear dynamics.

When introducing environment information into the online optimisation, the MPC framework using full dynamic models may become intractable. Hence, a hierarchical framework for the path planning and the following tracking control is adopted, which account for outer-loop kinematics and inner-loop dynamics, respectively. Although the interactions between the outer-loop and inner-loop dynamics are ignored, the planning and control separation is efficient for the normal flight.

For the case of trajectory tracking, the composite controller combining an explicit MPC and a nonlinear disturbance observer provides a promising solution. This controller generates control signals based on a MPC criterion and takes into account disturbances to improve the practical performance in various application environment. It has shown this ability in the daily operation of the testbed, where helicopters with different payloads and trim conditions can all perform flight in the desired quality. The disturbance observer based control also has the potential to combine with constrained MPC algorithms leading to a more promising control scheme.

It can be seen that although some problems in applying MPC based algorithms into UAV systems have been solved in this thesis, there are broader potentials that can be explored to further improve the performance of UAVs by using advance control. Here, two interesting directions are listed that can be considered as the future work based upon this thesis.

One interesting topic is to combine the local path planning and tracking control in a more tight manner. This is very useful for helicopters flying in cluttered area, which requires a faster response to the external environment and needs to exploit the full dynamic ability of a vehicle in the planning to perform aggressive dodging manoeuvre. Along this line, a simplified representation of helicopter dynamics like kinematic model may be used in the MPC framework to reduce the computation time, but the constraints on the kinematics have to be updated based on the current state of the helicopter to reflect the helicopter's manoeuvrability.

The multi-vehicle coordination is also a potential area to be explored by using MPC techniques. Employing a group of UAV can provide unique features such as efficiency and redundancy. In cooperative control of multiple vehicles, the path planning on each individual vehicle needs to consider the other vehicle's movements to avoid collisions. In this case, MPC techniques are naturally suited because that a in MPC framework not only the current position of each vehicle can be obtained, the future trajectory can also be predicted and to be known a prior by other vehicles in the group through communications.

At last, the proposed MPC based algorithms described in the thesis, along with the way of designing and synthesising them into real helicopters, are hoped to contribute to the UAV development and encourage the future work in related area.

Appendix A

Linear dynamic model

The linear dynamic model can be linearised from the full dynamic model developed in Chapter 4 around the hovering state and by combining the flapping angle approximation (5.2).

For the planner movement, i.e. for describing longitudinal/lateral motions, it has

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{p} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} X_u & 0 & 0 & -g & 0 & 0 \\ 0 & Y_v & g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & \tau L_b & \tau L_a \\ 0 & 0 & 0 & 0 & \tau M_b & \tau M_a \end{bmatrix} \begin{bmatrix} u \\ v \\ \phi \\ \theta \\ p \\ q \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ L_{lat} & L_{lon} \\ M_{lat} & M_{lon} \end{bmatrix} \begin{bmatrix} \delta_{lat} \\ \delta_{lon} \end{bmatrix} \tag{A.1}$$

where $L_{lat} = L_a A_{lat} + L_b B_{lat}$, $L_{lon} = L_a A_{lon} + L_b B_{lon}$, $M_{lat} = M_a A_{lat} + M_b B_{lat}$ and $M_{lon} = M_a A_{lon} + M_b B_{lon}$. By defining state variables $\mathbf{x}_1 = [u, v, \phi, \theta, p, q]^T$ and control inputs $\mathbf{u}_1 = [\delta_{lat}, \delta_{lon}]^T$, a compact form of Eq.(A.1) can be expressed as $\dot{\mathbf{x}}_1 = A_1\mathbf{x}_1 + B_1\mathbf{u}_1$.

For the heaving/heading motions it has

$$\begin{bmatrix} \dot{z} \\ \dot{w} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ z_w & 0 & 0 \\ 0 & 0 & N_r \end{bmatrix} \begin{bmatrix} z \\ w \\ r \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ Z_{col} & 0 \\ N_{col} & N_{ped} \end{bmatrix} \begin{bmatrix} \delta_{col} \\ \delta_{ped} \end{bmatrix} \tag{A.2}$$

Again, by defining $\mathbf{x}_2 = [z, w, r]^T$ and $\mathbf{u}_2 = [\delta_{col}, \delta_{ped}]^T$, Eq.(A.2) can be written as $\dot{\mathbf{x}}_2 = A_2\mathbf{x}_2 + B_2\mathbf{u}_2$.

Given a discretisation time ΔT, the continuous models developed in the last section can be transferred into a discrete form for the MPC design. By assuming the control input is constant during the time step, the transferring can use the following relation ship.

$$\begin{aligned} \mathbf{x}_i(k+1) &= e^{A_i\Delta T}\mathbf{x}_i(k) + A_i^{-1}(e^{A_i\Delta T} - \mathbf{I})\mathbf{u}_i(k) \\ &= A_i^d\mathbf{x}_i(k) + B_i^d\mathbf{u}_i(k) \end{aligned} \tag{A.3}$$

where index $i = 1, 2$ indicates the longitudinal/lateral or heave/heading subsystem, and $k+1$ indicates the next time step.

Appendix B

Publications

B.1 Paper under review

1. Cunjia Liu, Wen-Hua Chen and John Andrews. Trajectory tracking of autonomous helicopters using explicit nonlinear MPC augmented with disturbance observers. *Control engineering practice*, 2011. Resubmission invited.

B.2 Published paper

1. Cunjia Liu, Wen-Hua Chen and John Andrews. Explicit Nonlinear Model Predictive Control for Autonomous Helicopters. *Proceedings of the Institution of Mechanical Engineers, Part G, Journal of Aerospace Engineering*, 2011. To appear.

2. Cunjia Liu, Wen-Hua Chen and John Andrews. Trajectory tracking of small helicopters using explicit nonlinear MPC and DOBC. *Proceedings of the 18th World Congress, Milano, Italy*, 2011.

3. Cunjia Liu, Wen-Hua Chen and John Andrews. Piecewise constant model predictive control for autonomous helicopters. *Robotics and Autonomous Systems*, 59[7-8]:571 - 579, 2011.

4. Cunjia Liu, Jonathan Clarke, Wen-Hua Chen, and John Andrews. Rapid prototyping flight test environment for autonomous unmanned aerial vehi-

cles. *International Journal of Modelling, Identification and Control*, 12[3]:200–209, 2011.

5. Cunjia Liu, Wen-Hua Chen and John Andrews. Model predictive control for autonomous helicopters with computational delay. *Proceedings of UKACC International Conference on Control 2010*, 2010.

6. Cunjia Liu, Wen-Hua Chen and John Andrews. Experimental tests of autonomous ground vehicles with preview. *International Journal of Automation and Computing*, 7[3]:342-348, 2010.

7. Cunjia Liu, Wen-Hua Chen and John Andrews. Optimisation based control framework for autonomous vehicles: Algorithm and experiment. *Proceedings of Mechatronics and Automation (ICMA), 2010 International Conference on*, 1030-1035, 2010.

8. Cunjia Liu, Jonathan Clarke, Wen-Hua Chen, and John Andrews. Modeling and identification of a miniature helicopter. *Proceedings of Workshop on Human Adaptive Mechatronics (HAM)*, 2010.

References

[1] M. Alighanbari and J. How. Cooperative task assignment of unmanned aerial vehicles in adversarial environments. In *Proceedings of the 2005 American Control Conference*, pages 4661 – 4666, 2005. 13

[2] M. Alighanbari, Y. Kuwata, and J. How. Coordination and control of multiple uavs with timing constraints and loitering. In *Proceedings of the 2003 American Control Conference*, volume 6, pages 5311 – 5316, 2003. 13

[3] F. Allgöwer, R. Findeisen, and Z. Nagy. Nonlinear Model Predictive Control: From Theory to Application. *Journal of the Chinese Institute of Chemical Engineers*, 35(3):299–315, 2004. 10

[4] J. Bellingham, A. Richards, and J. How. Receding horizon control of autonomous aerial vehicles. In *Proceedings of the 2002 American Control Conference*, volume 5, pages 3741 – 3746, 2002. 13, 14

[5] J. Bellingham, M. Tillerson, M. Alighanbari, and J. How. Cooperative path planning for multiple uavs in dynamic and uncertain environments. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 3, pages 2816 – 2822, 2002. 13

[6] A. Berry, J. Howitt, D. Gu, and I. Postlethwaite. Enabling the Operation of Multiple Micro-Air-Vehicles in Increasingly Complex Obstacle-Rich Environments. In *AIAA Infotech@Aerospace 2010, Atlanta, Georgia*, 2010. 16

[7] A. Berry, J. Howitt, I. Postlethwaite, and D. Gu. Situation Aware Trajectory Tracking for Micro Air Vehicles in Obstacle-Rich Environments. In

2009 AIAA Guidance, Navigation, and Control Conference, Chicago, Illinois. American Institute of Aeronautics and Astronautics, 1801 Alexander Bell Dr., Suite 500 Reston VA 20191-4344 USA,, 2009. 16

[8] M. Bisgaard, A. la Cour-Harbo, and J. D. Bendtsen. Adaptive control system for autonomous helicopter slung load operations. *Control Engineering Practice*, 18(7):800 – 811, 2010. Special Issue on Aerial Robotics. 25

[9] A. Bogdanov and E. Wan. State-dependent riccati equation control for small autonomous helicopters. *Journal of Guidance, Control, and Dynamics*, 30(1):47–60, 2007. 23, 75, 127

[10] M. Campbell, J. Lee, E. Scholte, and D. Rathbun. Simulation and Flight Test of Autonomous Aircraft Estimation, Planning, and Control Algorithms. *Journal of Guidance Control and Dynamics*, 30(6):1597–1609, 2007. 18

[11] C. Castillo, W. Moreno, and K. Valavanis. Unmanned helicopter waypoint trajectory tracking using model predictive control. In *Mediterranean Conference on Control Automation*, pages 1 –8, 2007. 17

[12] M. Castillo-Effen, C. Castillo, W. Moreno, and K. P. Valavanis. Control fundamentals of small / miniature helicopters - a survey. In *Advances in Unmanned Aerial Vehicles*, volume 33, pages 73–118. Springer Netherlands, 2007. 22

[13] H. Chao, Y. Cao, and Y. Chen. Autopilots for small unmanned aerial vehicles: A survey. *International Journal of Control, Automation and Systems*, 8:36–44, 2010. 10.1007/s12555-010-0105-z. 17

[14] H. Chen and F. Allgwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205 – 1217, 1998. 9, 86

[15] W. Chen. Nonlinear disturbance observer-enhanced dynamic inversion control of missiles. *Journal of Guidance Control and Dynamics*, 26(1):161–166, 2003. 127, 138

[16] W. Chen, J. O'Reilly, and D. Ballance. On the terminal region of model predictive control for non-linear systems with input/state constraints. *International Journal of Adaptive Control and Signal Processing*, 17(3):195–207, 2003. 86

[17] W.-H. Chen. Disturbance observer based control for nonlinear systems. *Mechatronics, IEEE/ASME Transactions on*, 9(4):706 –710, dec. 2004. 139

[18] W.-H. Chen, D. Ballance, P. Gawthrop, and J. O'Reilly. A nonlinear disturbance observer for robotic manipulators. *Industrial Electronics, IEEE Transactions on*, 47(4):932 –938, aug 2000. 127

[19] W.-H. Chen, D. Ballance, and J. O'Reilly. Model predictive control of nonlinear systems: computational burden and stability. *IEE Proceedings - Control Theory and Applications*, 147(4):387–394, Jul 2000. 9, 102

[20] W.-H. Chen, D. J. Ballance, and P. J. Gawthrop. Optimal control of nonlinear systems: a predictive control approach. *Automatica*, 39(4):633 – 641, 2003. 127, 130, 137

[21] M. Cook. *Flight dynamics principles*. Elsevier aerospace engineering series. Elsevier/Butterworth-Heinemann, 2007. 49

[22] I. Cowling, O. Yakimenko, J. Whidborne, and A. Cooke. Direct method based control system for an autonomous quadrotor. *Journal of Intelligent & Robotic Systems*, 60:285–316, 2010. 16

[23] K. Culligan, M. Valenti, Y. Kuwata, and J. How. Three-dimensional flight experiments using on-line mixed-integer linear programming trajectory optimization. In *Proceedings of the 2007 American Control Conference*, pages 5322 –5327, 2007. 13

[24] L. Daga. Real-time blockset. `http://leonardodaga.insyde.it`, 2008. 36

[25] K. Dalamagkidis, K. P. Valavanis, and L. A. Piegl. Nonlinear model predictive control with neural network optimization for autonomous autorotation of small unmanned helicopters. *IEEE Transactions on Control Systems Technology*, 2010. 25

[26] J. F. Du, K. Kondak, Y. O. Zhang, and T. S. Lu. Modelling and control of a small-scale unmanned helicopter. *Proceedings of the institution of mechanical engineers part I - journal of systems and control engineering*, 222(I6):481–492, SEP 2008. 17

[27] W. Dunbar and R. Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 4, pages 4631 – 4636, 2002. 14

[28] P. Falcone, F. Borrelli, J. Asgari, H. Tseng, and D. Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on Control Systems Technology*, 15(3):566 –580, May 2007. 17

[29] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat. Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation. *International Journal of Robust and Nonlinear Control*, 18(8):862–875, 2008. 17

[30] H. Ferreau, H. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit mpc. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008. 119

[31] C. Floudas. *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press, New York, 1995. 13

[32] J. Gadewadikar, F. Lewis, K. Subbarao, and B. Chen. Structured H1 Command and Control-Loop Design for Unmanned Helicopters. *Journal of Guidance, Control, and Dynamics*, 31(4), 2008. 23, 109, 127, 149

[33] V. Gavrilets, I. Martinos, B. Mettler, and E. Feron. Control logic for automated aerobatic flight of miniature helicopter. In *AIAA Guidance, Navigation, and Control Conference and Exhibit, Monterey, CA*, 2002. 20

[34] V. Gavrilets, B. Mettler, and E. Feron. Dynamic model for a miniature aerobatic helicopter. In *AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada*, 2001. AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada. 20, 54

[35] J. C. Geromel and P. Colaneri. Robust stability of time varying polytopic systems. *Systems & Control Letters*, 55(1):81 – 85, 2006. 80

[36] C. Goerzen, Z. Kong, and B. Mettler. A survey of motion planning algorithms from the perspective of autonomous uav guidance. *Journal of Intelligent & Robotic Systems*, 57:65–100, 2010. 11, 13

[37] J. Grauer, J. Conroy, J. Hubbard Jr, and D. Pines. System Identification of a Miniature Helicopter. *Journal of Aircraft*, 46(4), 2009. 22

[38] Ĺ. Grman, D. RosinovÁ, V. Veselý, and A. KovÁ. Robust stability conditions for polytopic systems. *International Journal of Systems Science*, 36(15):961–973, 2005. 80, 94

[39] E. Gyurkovics and A. M. Elaiw. Stabilization of sampled-data nonlinear systems by receding horizon control via discrete-time approximations. *Automatica*, 40(12):2017 – 2028, 2004. 76

[40] U. Hald, M. Hesselbæk, J. Holmgaard, C. Jensen, S. Jakobsen, M. Siegumfeldt, A. la Cour-Harbo, and J. Thomsen. Autonomous Helicopter–Modelling and Control. Technical report, Aalborg university, 2005. 21

[41] J. Hedrick and Y. Kang. Linear tracking for a fixed-wing uav using nonlinear model predictive control. *IEEE Transactions on Control Systems Technology*, 17(5):1202–1210, Sept. 2009. 78

[42] R. Heffley and M. Mnich. Minimum-complexity helicopter simulation math model. Technical report, NASA, Moffett,CA, 1988. 20

[43] K. Holmström, A. Göran, and M. Edvall. Users Guide for TOMLAB/NPSOL. Technical report, TOMLAB Optimization Inc, 2008. 9, 95, 120

[44] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian. Real-time indoor autonomous vehicle test environment. *IEEE Control Systems Magazine*, 28(2):51 –64, 2008. 28

[45] A. Isidori. *Nonlinear control systems*. Springer Verlag, 1995. 134, 144

[46] A. Isidori, L. Marconi, and A. Serrani. Robust nonlinear motion control of a helicopter. *IEEE Transactions on Automatic Control*, 48(3):413 – 426, Mar. 2003. 24

[47] A. Jadbabaie. *Receding horizon control of nonlinear systems: A control Lyapunov function approach*. PhD thesis, California Institute of Technology, 2000. 14

[48] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776 –783, May 2001. 13

[49] E. Johnson and S. Kannan. Adaptive trajectory control for autonomous helicopters. *Journal of Guidance, Control, and Dynamics*, 28(3):524–538, 2005. 24, 127

[50] E. Johnson, D. Schrage, J. Prasad, and G. Vachtsevanos. UAV flight test programs at Georgia Tech. In *Proceedings of the 3rd AIAA Unmanned Unlimited Technical Conference, Workshop, and Exhibit*, 2004. AIAA-2004-6492. 28

[51] T. Keviczky and G. Balas. Software-Enabled Receding Horizon Control for Autonomous Unmanned Aerial Vehicle Guidance. *Journal of Guidance, Control, and Dynamics*, 29(3):680–694, 2006. 17

[52] T. Keviczky and G. J. Balas. Receding horizon control of an f-16 aircraft: A comparative study. *Control Engineering Practice*, 14(9):1023 – 1033, 2006. 17

[53] H. Kim, D. Shim, and S. Sastry. Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. In *Proceedings of the 2002 American Control Conference*, volume 5, pages 3576–3581, 2002. 18, 73

[54] H. J. Kim and D. H. Shim. A flight control system for aerial robots: algorithms and experiments. *Control Engineering Practice*, 11(12):1389 – 1400, 2003. Award winning applications-2002 IFAC World Congress. 18, 22

[55] S. Kim and D. Tilbury. Mathematical Modeling and Experimental Identification of an Unmanned Helicopter Robot with Flybar Dynamics. *Journal of Robotic Systems*, 21(3):95–116, 2009. 21

[56] Y. Kim, D.-W. Gu, and I. Postlethwaite. Real-time optimal mission scheduling and flight path selection. *IEEE Transactions on Automatic Control*, 52(6):1119 –1123, 2007. 13

[57] E. King, Y. Kuwata, M. Alighanbari, L. Bertuccelli, and J. How. Coordination and control experiments on a multi-vehicle testbed. In *Proceedings of the 2004 American Control Conference*, 2004. 13, 28

[58] T. Koo and S. Sastry. Output tracking control design of a helicopter model based on approximate linearization. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 4, pages 3635–3640, Dec 1998. 23, 129

[59] A. Krupadanam, A. Annaswamy, and R. Mangoubi. Multivariable adaptive control design with applications to autonomous helicopters. *Journal of Guidance, Control, and Dynamics*, 25(5):843–851, 2002. 127

[60] Y. Kuwata and J. P. How. Cooperative distributed robust trajectory optimization using receding horizon milp. *IEEE Transactions on Control Systems Technology*, 19(2):423 –431, 2011. 13

[61] Y. Kuwata, A. Richards, T. Schouwenaars, and J. How. Decentralized robust receding horizon control for multi-vehicle guidance. In *Proceedings of the 2006 American Control Conference*, pages 2047 – 2052, 2006. 13

[62] Y. Kuwata, A. Richards, T. Schouwenaars, and J. How. Distributed robust receding horizon control for multivehicle guidance. *IEEE Transactions on Control Systems Technology*, 15(4):627–641, July 2007. 13

[63] M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004. 8

[64] M. La Civita, W. Messner, and T. Kanade. Modeling of small-scale helicopters with integrated first-principles and system-identification techniques. In *Proceedings of the 58th Forum of the American Helicopter Society*, volume 2, pages 2505–2516, 2002. 22

[65] M. La Civita, G. Papageorgiou, W. Messner, and T. Kanade. Design and flight testing of an H8 controller for a robotic helicopter. *Journal of Guidance, Control, and Dynamics*, 29(2):485–494, 2006. 23, 127

[66] T. Lapp and L. Singh. Model predictive control based trajectory optimization for nap-of-the-earth (noe) flight including obstacle avoidance. In *Proceedings of the 2004 American Control Conference*, 2004. 15

[67] C.-T. Lee and C.-C. Tsai. Nonlinear adaptive aggressive control using recurrent neural networks for a small scale helicopter. *Mechatronics*, 20(4):474 – 484, 2010. 24

[68] C. Leung, S. Huang, N. Kwok, and G. Dissanayake. Planning under uncertainty using model predictive control for information gathering. *Robotics and Autonomous Systems*, 54(11):898 – 910, 2006. Planning Under Uncertainty in Robotics. 14

[69] L. Ljung. MATLAB System Identification Toolbox, Version 7.3, 2009. 46, 60, 67

[70] J. Maciejowski. *Predictive control: with constraints*. Pearson education, 2002. 8, 117

[71] L. Magni, D. Raimondo, and F. Allgöwer. *Nonlinear Model Predictive Control: Towards New Challenging Applications*. Lecture Notes in Control and Information Sciences. Springer, 2009. 9

[72] L. Magni and R. Scattolini. Model predictive control of continuous-time nonlinear systems with piecewise constant control. *IEEE Transactions on Automatic Control*, 49(6):900–906, June 2004. 73, 80

[73] L. Marconi and R. Naldi. Robust full degree-of-freedom tracking control of a helicopter. *Automatica*, 43(11):1909 – 1920, 2007. 24, 127, 129

[74] D. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, Jul 1990. 9

[75] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789 – 814, 2000. 9, 76

[76] T. McLain and R. Beard. Cooperative path planning for timing-critical missions. In *Proceedings of the 2003 American Control Conference*, volume 1, pages 296 – 301, 2003. 13

[77] D. McLean. *Automatic flight control systems*. Prentice Hall, 1990. 16

[78] B. Mettler. *Identification modeling and characteristics of miniature rotorcraft*. Kluwer Academic Pub, 2003. 21, 52, 53

[79] B. Mettler, N. Dadkhah, and Z. Kong. Agile autonomous guidance using spatial value functions. *Control Engineering Practice*, 18(7):773 – 788, 2010. Special Issue on Aerial Robotics. 14

[80] B. Mettler, C. Dever, and E. Feron. Scaling effects and dynamic characteristics of miniature rotorcraft. *Journal of guidance, control, and dynamics*, 27(3):466–478, 2004. 22

[81] B. Mettler, M. Tischler, and T. Kanade. System identification of small-size unmanned helicopter dynamics. In *Proceedings of the 55th Forum of the American Helicopter Society*, 1999. 21

[82] B. Mettler, M. B. Tischler, and T. Kanade. System identification modeling of a small-scale unmanned helicopter. *Journal of the American Helicopter Society*, October 2001. 21, 64, 67

[83] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar. The grasp multiple micro-uav testbed. *IEEE Robotics Automation Magazine*, 17(3):56 –65, 2010. 28

[84] H. Michalska and D. Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, Nov 1993. 9

[85] M. Milam, R. Franz, J. Hauser, and R. Murray. Receding horizon control of vectored thrust flight experiment. *IEE Proceedings - Control Theory and Applications*, 152(3):340 – 348, May 2005. 18

[86] J. Morris, M. van Nieuwstadt, and P. Bendotti. Identification and control of a model helicopter in hover. In *Proceedings of the 1994 American Control Conference*, volume 2, pages 1238 – 1242, 1994. 21

[87] A. Mujumdar and R. Padhi. Evolving philosophies on autonomous obstacle/collision avoidance of unmanned aerial vehicles. *Journal of Aerospace Computing, Information, and Communication*, 8(2):1542–9423, 2010. 13

[88] A. Ollero and L. Merino. Control and perception techniques for aerial robotics. *Annual Reviews in Control*, 28(2):167 – 178, 2004. 5, 16

[89] G. Padfield. *Helicopter flight dynamics: the theory and application of flying qualities and simulation modeling*. AIAA education series. American Institute of Aeronautics and Astronautics, 2007. 52

[90] E. Panteley and A. Loria. On global uniform asymptotic stability of nonlinear time-varying systems in cascade. *Systems & Control Letters*, 33(2):131 – 138, 1998. 144, 145

[91] Y. C. Paw and G. J. Balas. Development and application of an integrated framework for small uav flight control development. *Mechatronics*, In Press, Corrected Proof:–, 2010. 27

[92] K. Peng, G. Cai, B. M. Chen, M. Dong, K. Y. Lum, and T. H. Lee. Design and implementation of an autonomous flight control law for a uav helicopter. *Automatica*, 45(10):2333 – 2338, 2009. 109

[93] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733 – 764, 2003. 5

[94] I. A. Raptis, K. P. Valavanis, and W. A. Moreno. A novel nonlinear backstepping controller design for helicopters using the rotation matrix. *IEEE Transactions on Control Systems Technology*, PP(99):1 –9, 2010. 129

[95] A. Richards and J. How. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *Proceedings of the 2002 American Control Conference*, 2002. 13

[96] A. Richards and J. How. Decentralized model predictive control of cooperating uavs. In *Proceedings of 43rd IEEE Conference on Decision and Control*, volume 4, pages 4286 – 4291, 2004. 13

[97] A. Richards, J. How, T. Schouwenaars, and E. Feron. Plume avoidance maneuver planning using mixed integer linear programming. In *2001 AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada*, 2001. 13

[98] A. Richards, Y. Kuwata, and J. How. Experimental demonstrations of real-time MILP control. In *2003 AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003. 13

[99] T. Samad and G. Balas. *Software-enabled control: information technology for dynamical systems*. Wiley-IEEE Press, 2003. 5

[100] E. Scholte and M. Campbell. Robust nonlinear model predictive control with partial state information. *IEEE Transactions on Control Systems Technology*, 16(4):636–651, 2008. 15, 18

[101] T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed Integer Programming For Multi-Vehicle Path Planning. In *Proceedings of the 2001 European Control Conference*, pages 2603 – 2608, 2001. 13

[102] T. Schouwenaars, J. How, and E. Feron. Receding horizon path planning with implicit safety guarantees. In *Proceedings of the 2004 American Control Conference*, 2004. 13

[103] T. Schouwenaars, M. Valenti, E. Feron, and J. How. Implementation and flight test results of milp-based uav guidance. In *2005 IEEE Aerospace Conference*, pages 1 –13, 2005. 13

[104] D. Shim, H. Chung, H. Kim, and S. Sastry. Autonomous Exploration in Unknown Urban Environments for Unmanned Aerial Vehicles. In *2005 AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 1–8, 2005. 28

[105] D. Shim, H. Chung, and S. Sastry. Conflict-free navigation in unknown urban environments. *IEEE Robotics Automation Magazine*, 13(3):27 –33, 2006. 19

[106] D. Shim, H. Kim, and S. Sastry. Decentralized nonlinear model predictive control of multiple flying robots. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, volume 4, pages 3621–3626, Dec. 2003. 18, 73

[107] D. Shim and S. Sastry. An evasive maneuvering algorithm for uavs in see-and-avoid situations. In *Proceedings of the 2007 American Control Conference*, pages 3886 –3891, 2007. 19

[108] J. Shin and H. Kim. Nonlinear model predictive formation flight. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 39(5):1116 –1125, 2009. 14

[109] J. Shin, K. Nonami, D. Fujiwara, and K. Hazawa. Model-based optimal attitude and positioning control of small-scale unmanned helicopter. *Robotica*, 23(01):51–63, 2005. 22, 23, 67

[110] L. Simon, Z. Nagy, and K. Hungerbuehler. Swelling constrained control of an industrial batch reactor using a dedicated nmpc environment: Optcon. In L. Magni, D. Raimondo, and F. Allgwer, editors, *Nonlinear Model Predictive Control*, volume 384 of *Lecture Notes in Control and Information Sciences*, pages 531–539. Springer Berlin / Heidelberg, 2009. 9

[111] L. Singh and J. Fuller. Trajectory generation for a uav in urban terrain, using nonlinear mpc. In *Proceedings of the 2001 American Control Conference*, 2001. 14

[112] L. Singh and T. Lapp. Model predictive control in nap-of-earth flight using polynomial control basis functions. In *Proceedings of the 2005 American Control Conference*, pages 840 – 845, 2005. 15

[113] T. Templeton, D. Shim, C. Geyer, and S. Sastry. Autonomous vision-based landing and terrain mapping using an mpc-controlled unmanned rotorcraft. In *2007 IEEE International Conference on Robotics and Automation*, pages 1349 –1356, 2007. 19

[114] J. Tisdale, Z. Kim, and J. Hedrick. Autonomous uav path planning and estimation. *IEEE Robotics Automation Magazine*, 16(2):35 –42, 2009. 14, 15

[115] A. Tsourdos, B. White, and M. Shanmugavel. *Cooperative Path Planning of Unmanned Aerial Vehicles*. Aerospace Series. John Wiley & Sons, 2011. 10

[116] G. Vachtsevanos, L. Tang, G. Drozeski, and L. Gutierrez. From mission planning to flight control of unmanned aerial vehicles: Strategies and implementation tools. *Annual Reviews in Control*, 29(1):101 – 115, 2005. 10

[117] W. Van Soest, Q. Chu, and J. Mulder. Combined Feedback Linearization and Constrained Model Predictive Control for Entry Flight. *Journal of Guidance, Control, and Dynamics*, 29(2):427–434, 2006. 17

[118] Vicon. Vicon motion capture system. http://www.vicon.com/, 2008. 29, 32

[119] A. Wächter and L. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006. 9

[120] F. Wang, A. Tsourdos, R. Zbikowski, and B. White. Robust polynomial eigenstructure assignment control of a class of nonlinear systems. In *15th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 13 –18, 2010. 114