Onboard Robust Nonlinear Control for Multiple Multirotor UAVs

Onboard Robust Nonlinear Control for Multiple Multirotor UAVs

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

MSc. Yuyi Liu

aus Shanghai, China

Tübingen

2017

Tag der mündlichen Qualifikation:	26.06.2018
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Berichterstatter:	Prof. Dr. rer. nat. Andreas Zell
2. Berichterstatter:	Prof. Dr.-Ing. Frank Allgöwer

Abstract

In this thesis, we focus on developing and validating onboard robust nonlinear control approaches for multiple multirotor unmanned aerial vehicles (UAVs), for the promise of achieving nontrivial tasks, such as path following with aggressive maneuvers, navigation in complex environments with obstacles, and formation in group. To fulfill these challenging missions, the first concern comes with the stability of flight control for the aggressive UAV maneuvers with large tilted angles. In addition, robustness of control is highly desired in order to lead the multirotor UAVs to safe and accurate performance under disturbances. Furthermore, efficiency of control algorithm is a crucial element for the onboard implementation with limited computational capability. Finally, the potential to simultaneously control a group of UAVs in a stable fashion is required. All of these concerns motivate our work in this thesis in the following aspects.

We first propose the problem of robust control for the nontrivial maneuvers of a multirotor UAV under disturbances. A complete framework is developed to enable the UAV to achieve the challenging tasks, which consists of a nonlinear attitude controller based on the solution of global output regulation problems for the rigid body rotations SO(3), a backstepping-like position controller, a six-dimensional (6D) wrench observer to estimate the unknown force and torque disturbances, and an online trajectory planner based on a model predictive control (MPC) method. We prove the strong convergence properties of the proposed method both in theory and via intensive real-robot experiments of aggressive waypoint navigation and large-tilted path following tasks in the presence of external disturbances, e.g. wind gusts.

Secondly, we propose the problem of autonomous navigation of a multirotor UAV in complex scenarios. We present an effective and robust control approach, namely a fast MPC method with the inclusion of nonlinear obstacle avoiding constraints, and implement it onboard the UAV at 50Hz. The developed approach enables the navigation for a multirotor UAV in 3D environments with multiple obstacles, by autonomously deciding to fly over or around the randomly located obstacles.

The third problem that is addressed in our work is formation control for a group of multirotor UAVs. We solve this problem by proposing a distributed formation control algorithm for multiple UAVs based on the solution of retraction balancing problem. The algorithm brings the whole group of UAVs simultaneously to a prescribed submanifold that determines the formation shape in an asymptotically stable fashion in 2D and 3D environments. We validate our proposed algorithm via a series of hardware-in-the-loop simulations and real-robot experiments in various formation cases of arbitrary time-varying (e.g. expanding, shrinking or moving) shapes. In the actual experiments, up

to 4 multirotors have been implemented to form arbitrary triangular, rectangular and circular shapes drawn by the operator via a human-robot-interaction device. We have also carried out virtual tests using up to 6 onboard computers to achieve a spherical formation and a formation moving through obstacles.

Kurzfassung

In dieser Arbeit konzentrieren wir uns auf die Entwicklung und Validierung von robusten nichtlinearen On-Bord Steuerungsansätzen für mehrere unbemannte Multirotor-Luftfahrzeuge (UAVs), mit dem Ziel, nicht triviale Aufgaben zu erledigen wie z.B. Wegfolge mit aggressiven Manövern, Navigation in komplexen Umgebungen mit Hindernissen und Formationsflug in einer Gruppe. Um diese anspruchsvollen Missionen zu erfüllen liegt unser Hauptaugenmerk bei der Stabilität der Flugsteuerung für aggressive UAV Manöver mit steilen Lagewinkeln. Des weiteren ist Kontroll-robustheit sehr wünschenswert, um die Multirotor-UAVs unter Beeinflussung sicher und genau zu steuern. Darüber hinaus ist die Effizienz des Kontrollalgorithmus ein wichtiges Element für die Onboard-Implementierung mit eingeschränkter Rechenfähigkeit. Abschliessend ist das Potenzial, gleichzeitig eine Gruppe von UAVs in stabiler Weise zu kontrollieren, erforderlich. All dies motiviert uns zur Arbeit an den folgenden Aspekten:

Zuerst behandeln wir das Problem der robusten Steuerung nichttrivialer Manöver eines Multirotor UAV unter Störeinfluss. Ein komplettes Framework wird entwickelt, welches dem UAV ermöglicht diese anspruchsvollen Aufgaben zu bewältigen. Es beinhaltet einem nichtlinearen Lageregler, basierend auf der Lösung von globalen Ausgangsregelungsproblemen für Starrkörperrotationen SO(3), einem backstepping basierten Positionsregler, einen sechsdimensionalen (6D) wrench observer um die unbekannten Kraft- und Drehmomenteinflüsse zu schätzen, sowie einem Online-Trajektorienplaner basierend auf Model Predictive Control (MPC). Wir weisen die starken Konvergenzcharakteristiken der vorgeschlagenen Methode nach, sowohl in der Theorie als auchmittels intensiver Realroboter-Experimente, mit aggressiver Wegpunktnavigation und Wegfindungsaufgaben in extremer Fluglage in Gegenwart externer Einflüsse, z.B. Windböen.

Als nächstes bearbeiten wir das Problem der autonomen Navigation eines Multirotor UAV in komplexen Szenarien. Wir stellen einen effektiven und robusten Steuerungsansatz dar, nämlich eine schnelle MPC-Methode mit der Einbeziehung von nichtlinearer Einschränkungen zur Hindernisvermeidung, und implmenetieren diese an Bord des UAV mit 50Hz. Der entwickelte Ansatz ermöglicht die Navigation eines Multirotor UAVs in 3D-Umgebungen mit mehreren Hindernissen, wobei autonom entschieden wir, über oder um die zufällig gelegenen Hindernisse zu fliegen.

Das dritte Problem, das in unserer Arbeit angesprochen wird, ist die Bildungssteuerung für eine Gruppe von Multirotor UAVs. Wir lösen dieses Problem, indem wir einen verteilten Formationskontrollalgorithmus für mehrere UAVs auf der Grundlage der Lösung des Retraction Balancing Problems vorschlagen. Der Algorithmus bringt die ganze Gruppe von UAVs gleichzeitig auf eine vorgeschriebene Untermanigfaltigkeit, welche die

Formation in asymtotisch stabiler Weise in 2D- und 3D-Umgebungen bestimmt. Wir validieren unseren vorgeschlagenen Algorithmus über eine Reihe von Hardware-in-the-Loop-Simulationen und Real-Roboter-Experimente mit verschiedenen Formationsvarianten in beliebigen zeitveränderlichen (z. B. expandierenden, schrumpfenden oder bewegten) Formen. In den eigentlichen Experimenten wurden bis zu 4 Multirotoren eingesetzt, um beliebige dreieckige, rechteckige und kreisförmige Formen zu bilden, die vom Bediener über eine Mensch-Roboter-Interaktionsvorrichtung vorgezeichnet wurden. Wir haben auch virtuelle Tests mit bis zu 6 Onboard-Computern durchgeführt, um eine sphärische Formation und eine Formation zu erreichen, die sich durch Hindernisse bewegt.

Acknowledgments

First of all, I am most grateful to my supervisor, Prof. Andreas Zell, for his support and supervision on my research and study towards this thesis. He has provided me the scholarship for my whole study in Tübingen. In addition, he has not only provided me freedom in deciding research topics and carrying out experiments, but also given me many helpful academic suggestions, and furthermore, a cheerful study environment with beautiful landscape outside the office in Sand.

I would like to gratefully acknowledge Dr. Jan Maximilian Montenbruck, who is the co-author of most of my research projects. I have greatly benefited from the collaborations with him. He has provided me novel ideas on control fields, and has helped me a lot on the stability proof of nonlinear control approaches.

I am deeply indebted to Dr. Paolo Stegagno, and Dr. Aamir Ahmad, the two postdoc research fellows in Max-Planck institute for Biological Cybernetics. They have cooperated with me and supported me on my research projects. Additionally, many interesting ideas on my research have been inspired from the warm brainstorms and discussions with them.

Furthermore, I really appreciate the contributions from other co-authors of my publications: Sujit Rajappa, and Marcin Odelga. I have benefited quite a lot from the collaborations with them. Without the supports from them on both software and hardware sides, the implementation of the control algorithms on overall five quadrotor UAVs would be way more time-consuming. In addition, I would like to thank Prof. Daniel Zelazo for his informative suggestions on my formation control research, as well as his contribution on the writing of the publication.

I would like to sincerely thank my second supervisor, Prof. Frank Allgöwer for his valuable comments to this thesis. I would also like to deeply thank Prof. Heinrich Bülthoff for additionally providing me with a nice study environment with advanced experimental equipment, which plays a major role in verifying the proposed control algorithms in this thesis.

I would like to show my gratitude to all the colleagues and former colleagues in both our groups in University of Tübingen and in Max-Planck institute for Biological Cybernetics, especially to Dr. Shaowu Yang, Prof. Huimin Lu, Dr. Sebastian Scherer, Dingsheng Sun, Isabel Patino, and Dr. Yann Berquin, for their helps on my research, teaching activities and daily life. Special thanks should be given to Klaus Beyreuther and Vita Serbakova who offer assistance to our work.

I would like to thank all my friends in Germany and France who have shared the happy time with me during my study.

Finally, my deepest and most sincere appreciation goes to my parents for their infinite love and support in the last seven years since I have been to Europe. Thank you, indeed.

Contents

Notation

The list below contains the symbols and abbreviations that are most frequently used in this thesis. Furthermore, all vectors are denoted by bold lower-case characters (e.g. $\boldsymbol{x}$). Matrices are expressed by upper-case characters (e.g. K).

m	mass of quadrotor UAV
g	gravitational acceleration
$\boldsymbol{x}$	translational position of quadrotor in inertial frame
F	forces generated in body frame of the quadrotor by the rotor aerodynamics
τ	moments generated in body frame of the quadrotor by the rotor aerodynamics
b_F	external force disturbance acting in inertial frame
b_τ	external torque disturbance acting in body frame
J	inertial matrix of the quadrotor in body frame
ω	angular velocity of the quadrotor in body frame
$Q(\omega)$	skew-symmetric matrix of the angular velocity
R	rotation matrix that represents the quadrotor attitude w.r.t. inertial frame
T	thrust generated in the quadrotor along its z axis
c_T	propeller thrust coefficient
c_Q	propeller drag coefficient
d	length of the quadrotor arm
x_{desired}	desired position of the quadrotor
x_{ref}	exogenous reference position of the quadrotor
K_0	diagonal, positive gain matrix for the position cost
e_v	linear velocity tracking error in inertial frame
K	diagonal, positive gain matrix for the tracking error
b_{est}	external force disturbance estimate in the inertial frame
e_b	external force disturbance error
K_b	diagonal gain matrix for the disturbance observer
R_{ref}	rotation matrix that represents the reference attitude w.r.t. inertial frame
E	attitude tracking error
e_ω	angular velocity tracking error in the body frame
K_{rot}	positive gain for the rigid body rotation cost

K_ω	positive gain for the angular velocity tracking error
ψ	yaw attitude of the quadrotor in body frame
ψ_{ref}	reference yaw attitude of quadrotor in body frame
$C(\zeta)$	coriolis-centrifugal matrix
Λ	input wrench generated by the rotor-propeller setup
Λ_{ext}	six dimensional external force/torque disturbance wrench
$\boldsymbol{J}_\Theta$	Jacobian relating the translational and rotational dynamics with the angular velocity
Q	generalized momenta of the quadrotor
K_I	diagonal gain matrix for the residual
$\widehat{\Lambda}_{\text{ext}}$	estimated generalized external force vector
$\widehat{b}_F$	estimated external force in inertial frame
$\widehat{b}_\tau$	estimated external toeque in body frame
$\boldsymbol{\xi}$	state vector of the trajectory planner
$\boldsymbol{u}$	nominal input (jerk) for the trajectory planner
Δt	sampling time for state $\boldsymbol{\xi}$
A	state matrix in the state space model of optimal control problem
B	input matrix in the state space model of optimal control problem
J_{OCP}	quadratic cost function of optimal control problem
N	receding horizon in trajectory planner
P	weights of stage input cost
L_{s}	stage state cost
L_{t}	terminal state cost

The list below contains the abbreviations that are most frequently used in this thesis.

UAV	Unmanned Aerial Vehicle
MAV	Micro Aerial Vehicle
SLAM	Simultaneous Localization and Mapping
GPS	Global Positioning System
CAD	Computer-Aided Design
3D	three dimensional
6D	six dimensional
DOF	Degrees of Freedom
IMU	Inertial Measurement Unit
RMSE	Root-Mean-Square Error
OCP	Optimal Control Problem
MPC	Model Predictive Control
VO	Valid Obstacle
HIL	Hardware In the Loop
HRI	Human-Robot Interaction

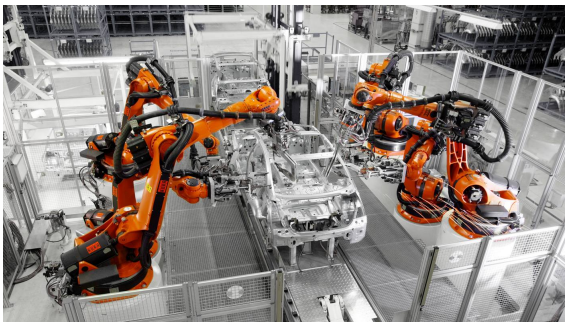
Chapter 1

Introduction

1.1 Motivation

Robotics has been an area of keen interests in scientific community since the last decades, because the development of robots is expected to provide human beings with a promising way to extend our physical capability and potential, to get rid of lifelong repeated work, and to make our childhood dreams true. Nowadays, robotics topic has even been a popular issue in daily life, of which the news can be found on most media over the world. Research on industrial robots, namely the manipulators/robot arms with multiple degrees of freedoms mounted on the fixed ground platforms, which has been widely employed for industrial manufacturing on assembly lines worldwide (as shown in Fig. 1.1a and 1.1b), or served as the assistant for doctors during the surgeries (as shown in Fig. 1.1c), or utilized as recreational facilities (as in Fig. 1.1d), has been carried out over decades. Meanwhile, research on mobile robots is recently becoming another attractive field in robotics community, which seeks the alternatives with mobility (i.e. automatic machines that can move) for human beings to discover the unknown and risky environments. The capability of locomotion provides the robots with broad range of applications, such as service, rescue, transportation, surveillance, exploration, etc. Simultaneously, the application of robots is therefore not restricted to ground, instead being extended to sky and sea.

Many impressive developments have been witnessed in the field of mobile robots Siegwart *et al.* (2011). So far, different types of mobile robots have been designed and served for various applications ranging from micro-scale robots, humanoid robots, to underwater and aerial robots, as depicted in Fig. 1.2: Researchers from John Hopkins University have designed and tested miscroscopic star-shaped grippers to grasp tissue samples for biopsies Gultepe *et al.* (2013) (as is shown in Fig. 1.2a). The humanoid robot Pepper (Softbank, 2017) (see Fig. 1.2b), produced by SoftBank to assist people in their daily life via oral and touch interactions, is able to respond to the questions from humans with informative and emotional answers. The autonomous driving cars designed by Google (Fig. 1.2c), which enable to achieve self-driving tasks on public roads without maneuvering from a human driver, have carried out over 2 million kilometers' test on the road (Waymo, 2017). The humanoid robotic diver from Stanford University (depicted in



(a)

(b)

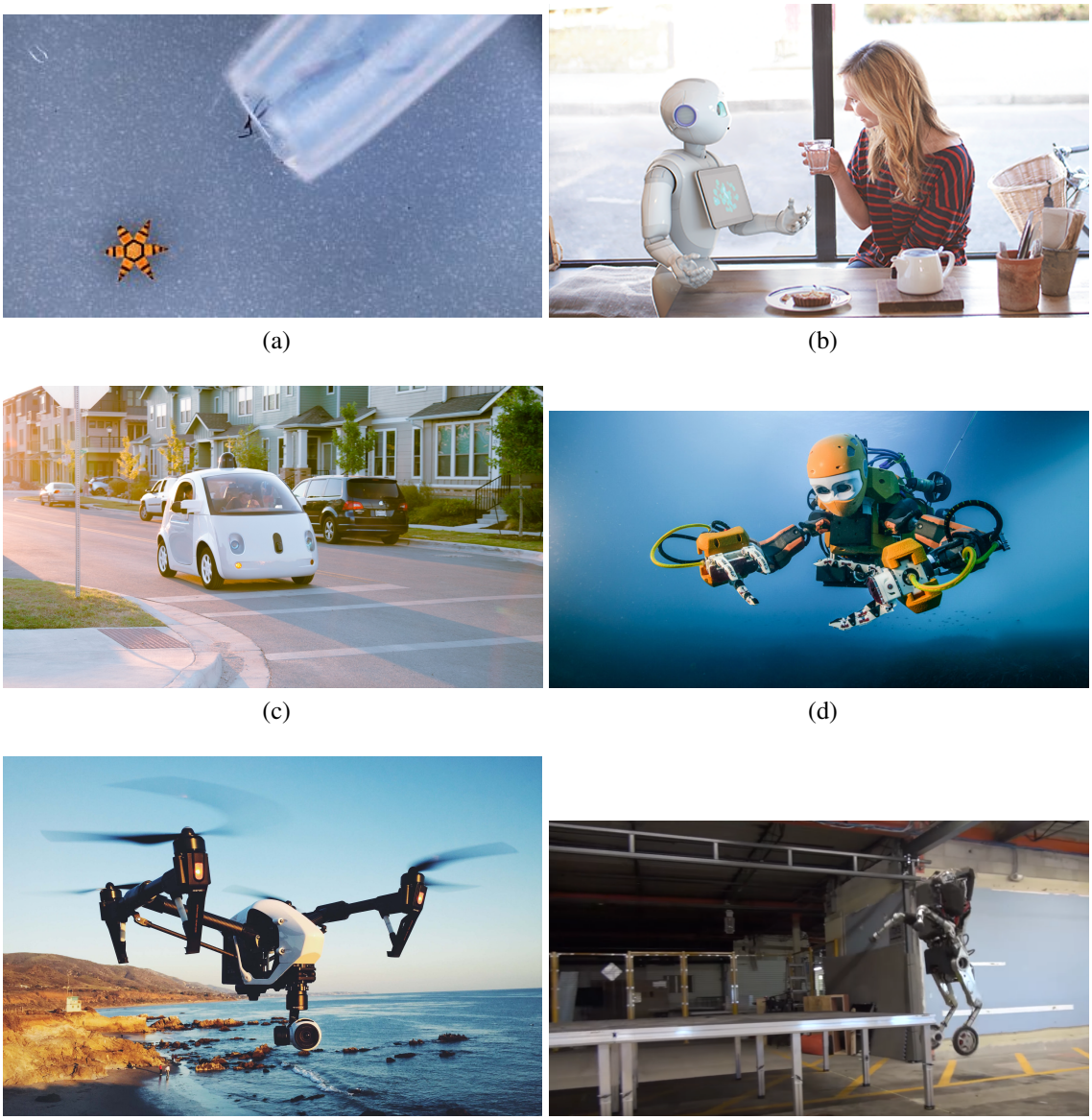
(c)

(d)

Figure 1.1: State-of-the-art robotic manupulator platforms. (a) Multiple KUKA industrial robots cooperatively work at a car assembly line. ©KUKA AG. (b) ABB Yumi dual-arm industrial robot. ©ABB Group. (c) The robot-assisted minimally invasive surgical platform SurgiBot. ©TransEnterix International, Inc. (d) The Kuka Coaster passenger-carrying robot for amusement ride. ©KUKA AG.

Fig. 1.2d), called OceanOne (Stanford Robotics Lab, 2017), is powered by artificial intelligence and haptic feedback systems, allowing human pilots an unprecedented ability to explore the depths of the oceans in high fidelity. Meanwhile, the unmanned aerial vehicles (UAVs), for example the Inspire 2 (see Fig. 1.2e) from dji (DJI, 2017) designed with person tracking and obstacle avoidance techniques, have been widely applied on film industry for the purpose of aerial photography. Fig. 1.2f demonstrates the two-legged wheeled robot with a payload over 45kg, called Handle from Boston Dynamics, which can jump over 1m height and travel at speeds up to 14km/h (Boston Dynamics, 2017).

Among more and more popular mobile robots research in the robotics community, UAV has been an area of keen interests since the last decade. In the variety of different types of UAVs, the multirotor UAVs is recently becoming a topic that grows rapidly.



(a) (b)

(c) (d)

(e) (f)

Figure 1.2: Examples of the state-of-the-art mobile robot platforms. (a) A star-shaped microgripper used to take biopsies. ©Gracias Lab, Johns Hopkins University. (b) A humanoid robot Pepper interacts with human. ©SoftBank Robotics. (c) A prototype of Google self-driving car. ©Waymo, Alphabet Inc. (d) The humanoid robotic diver OceanOne explors a 17th century shipwreck. ©Stanford Robotics Lab, Stanford University. (e) The UAV Inspire 2 from DJI is used for aerial photography ©Da-Jiang Innovations Science and Technology Co., Ltd. (f) The two-legged wheeled robot Handle is jumpping onto a table at high speed. ©Boston Dynamics Co., Ltd.

The dramatic development is potentially due to the versatility of multirotors, which can be utilized efficiently, with the implementation of reliable sensors, to explore complex 3-dimensional (3D) environments that are inaccessible to ground vehicles or large-size fixed-wing UAVs. With onboard visual solutions, e.g. simultaneous localization and mapping (SLAM) technique Heng *et al.* (2011), multirotor UAVs can autonomously navigate and map the unknown scenarios, such as a earthquake-damaged building Michael *et al.* (2012); With the implementation of mobile manipulators, multirotor UAVs are able to transport goods Palunko *et al.* (2012b) and achieve construction tasks Augugliaro *et al.* (2014), etc; With the application of trajectory planning techniques, a group of multirotors succeeded in performing swarm demonstrations above audience in the amusement park Intel (2017b). In order to achieve the variety of applications, control techniques are essential for the multirotor UAVs.

The control problem is one of the most fundamental problems in robotics. It makes the basic task for each mobile robot, *"how to move from A to B"*, possible. Without this basic task, the high-level applications presented above can never be realized properly. The answer of this question (*"how to"*) can be found through the application of the existing control algorithms that have been listed in the control textbooks, from the classical model-free patterns to the recently developed nonlinear or optimal approaches. From the perspective of application on multirotor UAVs, the development of flight control is so far becoming more and more matured: many consumer-level products on the market have impressively stable flight performance and user-friendly interfaces to the operators. However, the stable maneuvering is usually under trivial, mild maneuvers with restricted tilted angle and horizontal acceleration. This arouses us a new brainstorm, namely *"how to fly aggressively from A to B"*.

Although with the application of several advanced control techniques, multirotors can perform aerial acrobatics Lupashin *et al.* (2010); Mellinger *et al.* (2012), the answer to large-tilted maneuvers is still not complete for multirotor UAVs. To achieve these aggressive, large-tilted maneuvers, classical linear control algorithms are usually not the perfect solution because the linearization of the flight attitude of the UAV is only working properly under a small angle assumption, i.e. with tilted angle less than 20°. Thus the control algorithm should be based on one of the nonlinear patterns, and should be with strong convergence properties.

Therefore, the first problem to be solved is: ***Can we develop a nonlinear control algorithm for a multirotor UAV?*** This is the main motivation for the first part of our work in Chapter 3. The solution to this first specific problem addressed in this thesis is a nonlinear control approach to nontrivial/aggressive multirotor maneuvers. This can be nicely found by solving the problems for rigid body rotations SO(3). Meanwhile, we expect our proposed approach to be versatile, which should support a multirotor UAV to achieve different types of tasks, e.g. waypoint navigation, and path following, etc. To respond to this concern, a model predictive control (MPC) method is applied. Our work presented in Chapter 3 will reflect this solution through both mathematical derivations and experimental validations.

The control problem for UAVs should not be restricted to *"how to fly from A to B"*, and *"how to fly aggressively"*. It also includes reliable maneuvers in complex environments with disturbances, which can be stated as: *"how to fly aggressively under disturbances"*. Despite that we have experimentally tested several linear and nonlinear control algorithms on our multirotor UAV platform, the control performance is not as we expected due to the model uncertainties and external disturbances, e.g. wind gusts. This leaves us two potential directions: to perform improvement from the aspect of carrying out model identifications, or from the aspect of designing robust control approaches. Between the two ways, the latter is chosen in the thesis.

Hence, the second problem to be solved, on the basis of the solution to the first problem that we just addressed, is: ***Can we develop a robust nonlinear control structure for a multirotor UAV under disturbances?***, where the term *"robust" or "robustness"* throughout this thesis is mainly regarded as the capability of meeting high control accuracy under disturbances, instead of the response to sudden, unexpected hardware failure on sensors or onboard processors. This becomes the motivation for the second part of our work in Chapter 3, the solution of which is either to modify the nonlinear control algorithm with an additional robustness design, or to apply certain observation technique to estimate the unknown disturbances and further compensate against them. Both potential solutions have been attempted, through the design of convergent Lyapunov function and through the inclusion of a nonlinear observer. Again, our work in Chapter 3 will demonstrate that our solution towards the proposed problem is competitive via theoretical proofs and intensive real-robot experiments.

Then the next question directly comes, since the term "complex environments" can be explained in another extensible way, namely "with disturbances *and* obstacles". Much research has been carried out on the trajectory planning algorithms in the envrionments with obstacles. However, the majority proposes offline trajectory planning methods in 2D scenarios. One difference, between the problem we encounter with for a multirotor UAV and the obstacle avoidance problems for ground robots, is that we expect a trajectory planning algorithm, or even a direct control algorithm that enables the multirotor to navigate smartly in 3D environments, instead of only to fly around the detected obstacles with a kept altitude. Meanwhile, we also require the solution to be updated in real-time, so that the multirotor UAV could deal with sudden situations in complex environments.

These two major concerns lead to the third problem, as a extension from the second problem, to be solved: ***Can we develop a robust nonlinear control framework for a multirotor UAV in complex environments with obstacles and disturbances?*** This third problem addressed in the thesis motivates our work in Chapter 4, where we try to answer the following questions: *How to make a multirotor UAV autonomously avoid obstacles in 3D environment? How to speed up the computation of the trajectory planning algorithm so that it can be executed online?* We reply to these questions by developing a fast, nonlinear MPC method for the multirotor UAV and implementing it onboard with limited computational capability after optimizing the computational cost.

In case that we can solve the three problems addressed above, is it possible to ex-

tend the control framework from a single multirotor UAV to a group of UAVs, so that a networked system of UAVs can fly in formation and achieve commanded high-level missions? So far, much research on formation control relies on an offline, prescribed trajectory for each agent in the group. Meanwhile, many existing formation algorithms that output online trajectories for agents cannot lead the whole group of agents to the target positions simultaneously. Furthermore, since much research has been carried out on human-robot interaction, is it possible for an operator to interact with multiple UAVs at the same time, for instance *by drawing a shape*, etc.?

All these concerns turn to the forth problem to be solved: ***Can we develop a formation control algorithm for a group of multirotor UAVs that is suitable for simple human-robot interactions?*** By employing the work in Chapter 4, we address the problem of extending the flight control to a whole group of multirotor UAVs simultaneously in a stable fashion. We also have to bridge the interaction between the operator and all the UAVs in a simple way. These are the two motivations for our work in Chapter 5, which additionally includes our multirotors with a formation control algorithm based on a solution for retraction balancing problem, arouses a "draw and fly" concept between the operator and multiple UAVs via the "drawings" from finger motions.

1.2 Contributions

The research presented in this thesis largely focuses on developing and validating efficient and robust onboard flight control systems for the nontrivial tasks for multirotor UAVs, e.g. aggressive path following, waypoint navigation with obstacle avoidance, formation, etc. The specific contributions made in each of the three technical chapters can be described as follows:

Chapter 3:

- A nonlinear attitude controller based on the solution of rigid body rotations SO(3) is proposed, which is with strong convergence properties for almost all initial attitude conditions, and suitable for aggressive maneuvers for multirotor UAVs.
- A robust backstepping-like position controller is proposed, which is able to compensate against the force disturbances during the flight of UAV.
- A 6D nonlinear wrench observer is presented, which estimates the unknown force and torque disturbances and is implemented onto the flight control system for robustness.
- A versatile trajectory planning algorithm is developed based on a MPC method, which online updates the reference states for the multirotor UAV to track during both waypoint navigation and path following tasks.

- The asymptotic stability of the complete control-observation system is theoretically proved and experimentally validated.

- The analyses and discussions on a series of intensive real-robot experiments carried out in the scenarios with model uncertainties and wind gusts are presented.

Large parts of this work have been pre-published in the following paper:

1. Liu, Y., Montenbruck, J. M., Stegagno, P., Allgöwer, F., and Zell, A. (2015). A Robust Nonlinear Controller for Nontrivial Quadrotor Maneuvers: Approach and Verification. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 600-606, 2015.

2. Liu, Y., Rajappa, S., Montenbruck, J. M., Stegagno, P., Bülthoff, H., Allgöwer, F., and Zell, A. (2017b). Robust Nonlinear Control Approach to Nontrivial Maneuvers and Obstacle Avoidance for Quadrotor UAV under Disturbances. *Robotics and Autonomous Systems*, vol. 98, pp. 317-332, 2017.

Chapter 4:

- A tube-based MPC framework is designed for multirotor UAVs, which provides accurate control performance under model uncertainties.

- A MPC-observer framework is presented for multirotor UAVs in order to further compensate against external forces and torques generated from unknown disturbances.

- A MPC method with nonlinear obstacle avoiding constraints is developed, which enables the multirotor UAV to avoid the randomly located obstacles in 3D environment by autonomously deciding to fly over or fly around the detected obstacles.

- The proposed nonlinear MPC method with obstacle avoidance is modified into an online trajectory planner and is implemented onboard together with the robust nonlinear control approach presented in Chapter 3.

Large parts of the work presented in Chapter 3 and 4 have been together pre-published in the following paper:

2. Liu, Y., Rajappa, S., Montenbruck, J. M., Stegagno, P., Bülthoff, H., Allgöwer, F., and Zell, A. (2017b). Robust Nonlinear Control Approach to Nontrivial Maneuvers and Obstacle Avoidance for Quadrotor UAV under Disturbances. *Robotics and Autonomous Systems*, vol. 98, pp. 317-332, 2017.

Chapter 5:

- A distributed formation control approach is presented, which brings the whole group of UAVs simultaneously to a prescribed submanifold that determines the formation shape in an asymptotically stable fashion.
- The formation control algorithm is extended to be feasible for circular, triangular, rectangular, and spherical formations for a group of multirotor UAVs.
- A human-swarm interaction strategy is proposed for the multirotor UAVs to converge to a formation with a human operator "drawing" the target shape via a finger tracking device.
- By combining with an onboard MPC method, the distributed formation control algorithm is experimentally validated through a series of real-robot experiments and hardware-in-the-loop simulations.

Large parts of this work have been pre-published in the following paper:

3. Liu, Y., Montenbruck, J. M., Odelga, M., Zelazo, D., Rajappa, S., Bülthoff, H., Allgöwer, F., and Zell, A. (2017a). A Distributed Control Approach to Formation Balancing and Maneuvering of Multiple Multirotor UAVs. *IEEE Transactions on Robotics*, Special Issue: Swarm Aerial Robotics (to appear), 2018.

The structure of this thesis is given in the following way: the motivations and contributions of this thesis have first been presented from a general view in this chapter. The background related to multirotor UAVs, including the flight control methods we used for comparison, and the configurations of real-robot experiments, is presented in Chapter 2. The following three technical chapters demonstrate the main scientific work of this thesis, of which the highlight has been enumerated earlier in this section. The specific literature review on each topic is given individually in each technical chapter which proposes the corresponding problem. In Chapter 6, we finally conclude this thesis by summarizing the our solutions to the proposed problems and further raising open questions for future research.

Chapter 2

Background

In this chapter, we present background principles and related work on the flight control of multirotor UAVs. The outline of this chapter is as follows. In Sec. 2.1, we first introduce the dynamic model of a multirotor UAV and the basic estimation of the related physical parameters. We then briefly present an overview of widely-used control methods, e.g. PID, geometric tracking algorithm that are used for benchmark in the technical chapters, as well as their implementations on multirotor UAVs in Sec. 2.2. Finally in Sec. 2.3, we demonstrates the specific configurations of the multirotor UAV platforms on which we have implemented the control and trajectory planning algorithms we propose in this thesis.

2.1 Dynamics and Modelling of Multirotor UAV

In this section, we introduce the dynamic model of the multirotor UAV (e.g. quadrotor and hexarotor) and how we estimate the related physcial parameters, e.g. inertial and aerodynamic coefficients, that are required for the flight control algorithms proposed later in Sec. 3, 4 and 5 in this thesis.

2.1.1 Dynamic Model of Multirotor UAV

A multirotor UAV can be modelled as a rigid body with a multirotor body having the propeller arms and a group of propellers. The propeller connected to the motor rotors causes the spinning actuation. In most cases, these multirotors posses limited mobility because of the underactuation, i.e. the availability of only decoupled 4 control inputs in comparison to the 6 DoFs required to position/orient the UAVs in free space.

The dynamic model of the multirotor UAV is written in simplified form using Euler-Lagrangian generalized force formulation with the system states ζ for the translational and the rotational dynamics as

$$M\dot{\zeta} + C(\zeta)\,\zeta + G = \Lambda + \Lambda_{\text{ext}}, \tag{2.1}$$

where

$$M = \begin{bmatrix} m\boldsymbol{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & J \end{bmatrix} \in \mathbb{R}^{6\times6} \tag{2.2}$$

is the diagonal, positive definite inertial matrix with J the diagonal multirotor body inertia matrix, $\boldsymbol{I} \in \mathbb{R}^{3\times3}$ the identity matrix and $\mathbf{0} \in \mathbb{R}^{3\times3}$ the zero matrix;

$$C(\zeta) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \begin{bmatrix} 0 & I_{zz}r & -I_{yy}q \\ -I_{zz}r & 0 & I_{xx}p \\ I_{yy}q & -I_{xx}p & 0 \end{bmatrix} \end{bmatrix} \in \mathbb{R}^{6\times6} \tag{2.3}$$

expresses the coriolis and centrifugal terms with I_{xx}, I_{yy} and I_{zz} being the diagonal components of J; while G is the Gravitational vector given by

$$G = \begin{bmatrix} 0 & 0 & mg & 0 & 0 & 0 \end{bmatrix}^\top \in \mathbb{R}^6. \tag{2.4}$$

Here the system states ζ are defined using the translational and rotational velocities as

$$\zeta = \begin{bmatrix} \dot{\boldsymbol{x}} & \boldsymbol{\omega} \end{bmatrix}^\top = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & p & q & r \end{bmatrix}^\top \tag{2.5}$$

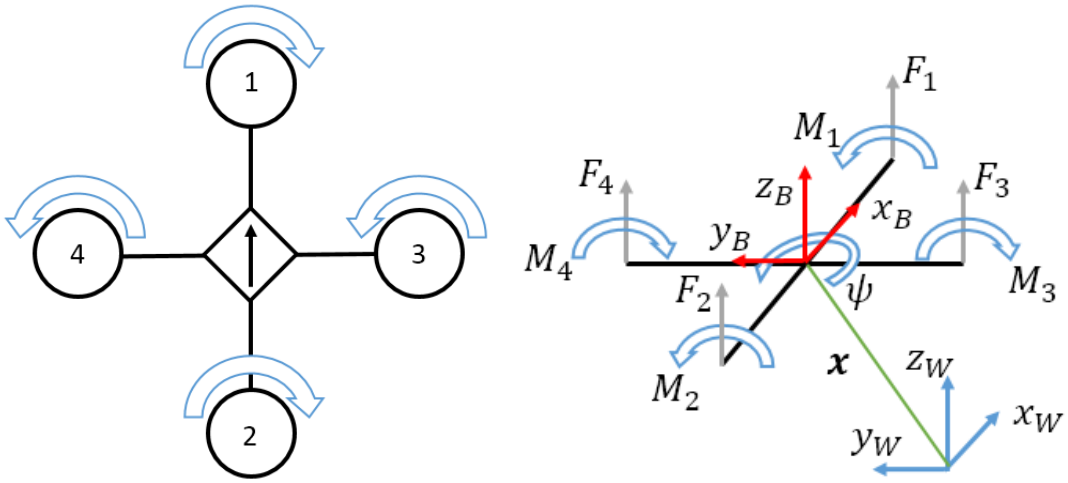
with $\boldsymbol{x}$ denoting the 3D position in the inertial frame and $\boldsymbol{\omega} = (p,q,r)^\top$ denoting the angular velocity in the body frame.

Note that in the above equation (2.1), the terms related to the propeller dynamics, e.g. hub forces, the terms from IMU, e.g. gyro effects, and the non-diagonal components of the inertial matrix J, are all neglected because those static and viscous friction terms, and the coupling physical parameters in the model, which are part of the standard Lagrangian formulation, are usually small with respect to the other terms and are combined as part of the external disturbance.

Looking at the right-hand side of (2.1), $\Lambda \in \mathbb{R}^6$ is the input wrench (3D force F and 3D torque τ) that is applied on the multirotor as a means of control input.

The control input of force and torque is unified for most types of the multirotor UAVs, e.g. quadrotors and hexarotors. While it can be related to the rotational speed of the propellers, depending on the type of multirotor UAVs, and the mapping of these control inputs onto each motor on the top of the arms. We here give two configurations of quadrotor and hexarotor UAVs. In this thesis, we demonstrate the versatility of our proposed control and trajectory planning algorithms on different types of multirotor UAVs, by employing qauadrotor UAVs as the platform for real-robot experiments, while utilizing hexarotor UAVs as the platform for desktop-based and hardware-in-the-loop (HIL) simulations.

A common configuration of quadrotor UAVs can be illustrated in Fig. 2.1a, which is



(a) (b)

Figure 2.1: (a) Top view of rotation directions of the four rotors on a quadrotor UAV. The generated yawing moments acting are in the counter directions to motor rotation. (b) Forces and moments acting on the quadrotor frame, with a depiction of the principle of yaw control to anti-clockwise rotation.

with four fixed propellers mounted on the brushless rotors configured in a symmetric cross. The angle between two neighboring arms is fixed as 90°. This configuration applies to all the quadrotors we used for the real-robot experiments.

Meanwhile, the simplified dynamic model of a quadrotor represented in (2.1) can be illustrated in Fig. 2.1b. The quadrotor body frame B is assumed to have the coordinate center coinciding with the geometric center of a quadrotor. x_B coincides with the preferred forward ("heading") direction along one arm on the quadrotor (the arm that rotor 1 is mounted), while z_B points to the upward direction perpendicular to the plane where all four quadrotor arms are located. The rotor 1 and 2 rotates along z_B direction (anti-clockwise), while rotor 3 and 4 rotates along $-z_B$ direction. The rotation of each propeller generates a lift force perpendicular to the rotation plane of the propeller due to aerodynamics. In addition, the rotation produces a yawing moment acting on the rotation plane, whose direction is always in the counter directions to motor rotation. The forces and moments generated from the rotation of propeller at different speeds lead to the translational and rotational motion of the quadrotor UAV. Consequently, we control the attitude and position of the quadrotor by properly adjusting the rotation speed of each rotor mounted on the quadrotor Nonami *et al.* (2010).

The rotation of a rigid body can be expressed as a rotation matrix R, which transform a vector from the body frame B, with z_B pointing upward, to the world frame W, with z_W pointing upward. We apply $Z-Y-X$ Euler angles Schilling (1990) to represent the rotation of the quadrotor. This means that when rotating W to body frame B, we rotate the

coordinate first around z_W by the yaw angle ψ, then rotate it around y axis by the pitch angle θ, and finally rotate it around x axis by the roll angle ϕ. The resulting rotation matrix can be represented as

$$R = \begin{bmatrix} c\psi c\theta & c\psi s\phi s\theta - c\phi s\psi & s\phi s\psi + c\phi c\psi s\theta \\ c\theta s\psi & c\phi c\psi + s\phi s\psi s\theta & -c\phi s\psi s\theta - c\psi s\phi \\ -s\theta & c\theta s\phi & c\phi c\theta \end{bmatrix}, \tag{2.6}$$

where $s\phi$ and $c\phi$ represent $\sin(\phi)$ and $\cos(\phi)$, respectively. Similarly for θ and ψ.

The relation between the lift force generated from each propeller, $F_i \; \forall \; i = 1, \ldots, 4$, and the control input of force and torque can be represented as

$$\begin{bmatrix} f_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & -d & d \\ -d & d & 0 & 0 \\ k_m & k_m & -k_m & -k_m \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}, \tag{2.7}$$

where d, k_m denote length of the propeller arms and moment coefficient, respectively.

Therefore, the final mapping from the control input of force and torque into the rotational speed of each motor on a quadrotor UAV, $\omega_{t,i} = \sqrt{F_i/c_T}$, $\forall \; i = 1, \ldots, 4$, is given by

$$\Lambda = \begin{bmatrix} F \\ \tau \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ f_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \boldsymbol{J}_\Theta \begin{bmatrix} 0 \\ 0 \\ c_T(\omega_{t,1}^2 + \omega_{t,2}^2 + \omega_{t,3}^2 + \omega_{t,4}^2) \\ c_T d(\omega_{t,4}^2 - \omega_{t,3}^2) \\ c_T d(\omega_{t,2}^2 - \omega_{t,1}^2) \\ c_Q(\omega_{t,1}^2 + \omega_{t,2}^2 - \omega_{t,4}^2 - \omega_{t,3}^2) \end{bmatrix} \tag{2.8}$$

with c_T and c_Q being the propeller lift coefficient and propeller drag coefficient, respectively, having $c_T = c_Q/k_m$. The identification of these coefficients will be briefly introduced in Sec. 2.1.2. The $\boldsymbol{J}_\Theta$ in (2.8) is the Jacobian that relates the translational dynamics in inertial frame and the rotational dynamics in body frame with the angular velocity $\omega_{t,i}$ of each propeller whereas Λ_{ext} accounts for the external force/torque disturbance wrench and the unmodeled effects acting on the quadrotor.

Similarly, a common configuration of hexarotor UAVs is illustrated in Fig. 2.2. Compared to a quadrotor UAV, the major difference on dynamics is that the angle between two neighboring arms on a hexarotor is fixed as 60°.

Following the notations in the quadrotor dynamics we introduced above, the relation between the lift force generated from each propeller, $F_i \; \forall \; i = 1, \ldots, 6$, and the control

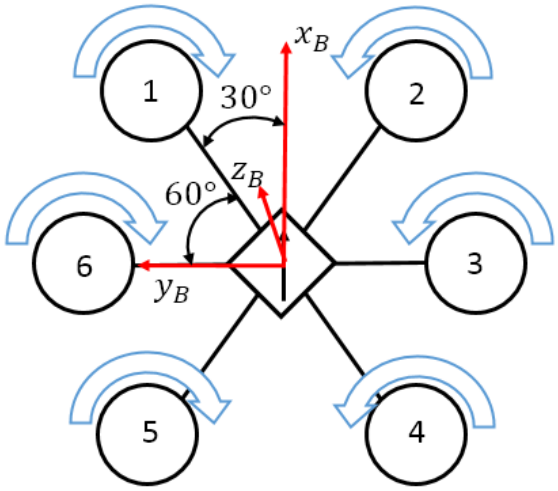


Figure 2.2: Top view of rotation directions of the six rotors on a hexarotor UAV. The generated yawing moments acting are in the counter directions to motor rotation. The angle between two neighboring arms is 60°.

input of force and torque now is given by

$$\begin{bmatrix} f_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ ds_{30} & -ds_{30} & -d & -ds_{30} & ds_{30} & d \\ -dc_{30} & -dc_{30} & 0 & dc_{30} & dc_{30} & 0 \\ k_m & -k_m & -k_m & -k_m & k_m & k_m \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{bmatrix}, \tag{2.9}$$

where s_{30}, c_{30} denote $\sin(30^\circ)$ and $\cos(30^\circ)$, respectively.

Therefore, the final mapping from force and torque into the rotational speed of each motor, $\omega_{t,i}$, $i = 1,\ldots,6$, can be described as

$$\Lambda = \begin{bmatrix} F \\ \tau \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ f_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \boldsymbol{J}_\Theta \begin{bmatrix} 0 \\ 0 \\ c_T(\omega_{t,1}^2 + \omega_{t,2}^2 + \omega_{t,3}^2 + \omega_{t,4}^2 + \omega_{t,5}^2 + \omega_{t,6}^2) \\ c_T d(\omega_{t,1}^2 s_{30} + \omega_{t,5}^2 s_{30} + \omega_{t,6}^2 - \omega_{t,2}^2 s_{30} - \omega_{t,3}^2 s_{30} - \omega_{t,4}^2) \\ c_T d(\omega_{t,4}^2 c_{30} + \omega_{t,5}^2 c_{30} - \omega_{t,1}^2 c_{30} - \omega_{t,2}^2 c_{30}) \\ c_Q(\omega_{t,1}^2 + \omega_{t,5}^2 + \omega_{t,6}^2 - \omega_{t,2}^2 - \omega_{t,3}^2 - \omega_{t,4}^2) \end{bmatrix} \tag{2.10}$$

The desired angular velocity of each motor, which is assumed as the control input on a hexarotor UAV by neglecting the error from the motor controller, therefore, can be obtained via solving the inverse of (2.10).

There have been various regulation approaches developed to compute the desired non-conservative forces and moments on the multirotor UAV. We will give an overview of several widely-applied methods in Sec. 2.2.

2.1.2 Parameter Identification

We here briefly introduce how the physical parameters required in the modelling of multirotor UAV are identified. The mass of a multirotor platform can be directly measured by a scale, meanwhile the arm length can also be simply measured. Therefore, the rest parameters include the inertial matrix J, the propeller lift coefficient c_T, and the propeller drag coefficient c_Q.

A widely-employed approach to identify the inertial matrix J of a multirotor UAV is to estimate all the inertial coefficient components of a full-scaled multirotor model virtually built in the Computer-Aided-Design (CAD) software.

This estimation via CAD modeling can be achieved by the following procedure:

1. Measuring the geometric size of each component on a multirotor UAV platform, e.g. metal frame, arm, rotor, propeller, landing gear, onboard computer, etc.;

2. Building the model of each component in the CAD software based on the geometric size of the parts on the real experimental platform;

3. Measuring the mass of each component in real-world and add the mass property onto the CAD model;

4. Assembling all component models together with proper constraints as a complete product/assembly, and computing the rigid-body inertial matrix J.

In this thesis, we employ the commercial CAD software CATIA V5 (Systems, 2017) to build the virtual multirotor UAV model and to estimate all the iertial coefficient components. For the sake of visualization, we here display a rendering photo of the CAD model of our QuadroXL platform in Fig. 2.3. In order to reduce the risk of damaging the onboard equipments, we removed the cameras and additionally mounted a two-layer plastic-polymer case to protect the onboard computer during real-robot experiments.

The mass, arm length, and inertial coefficients of all the multirotor UAV platforms we employed to validate our proposed control and trajectory planning algorithms, in the simulation and in real-robot experiments, are listed in the following tables 2.1 and 2.2:

The propeller lift coefficient c_T and the propeller drag coefficient c_Q can be estimated via a series of hardware tests. During the tests, a rotor, mounted one the multirotor arm and fixed on a frame, is required to get connected with a closed-loop-fashion brushless motor controller, which has the embedded sensor so that the rotational speed of the controlled rotor is able to be read out. In addition, a 1D force sensor is required to measure the lift force during the rotation of the propeller.

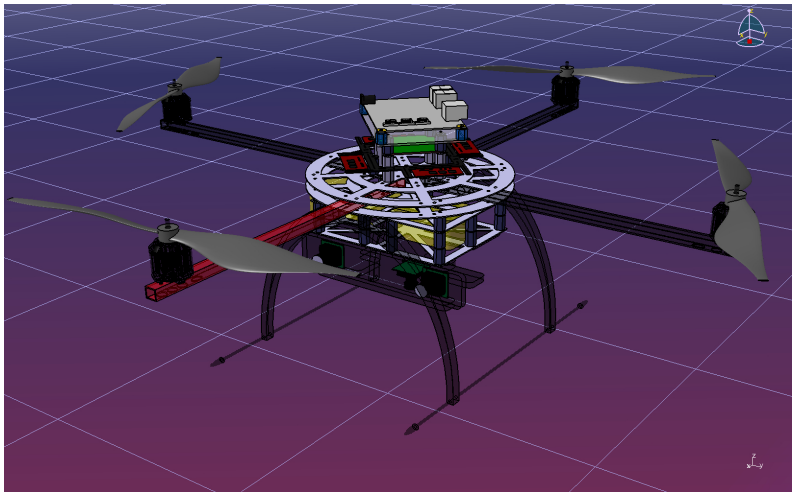


Figure 2.3: The CAD model of the QuadroXL (with camera) quadrotor UAV platform.

Platform	**mass**	**arm length**
QuadroXL (with camera)	1.715[kg]	0.29[m]
QuadroXL	1.605[kg]	0.29[m]
QuadroL	1.055[kg]	0.235[m]
Hexarotor (simulation)	1.588[kg]	0.215[m]

Table 2.1: Physical parameters of multirotor UAV platform: mass and arm length.

Considering that the lift force $F_i = c_T \omega_{t,i}^2$ from a single propeller, one can record the lift forces under different conditions of motor speed, and consequently generate a look-up table of measured lift force against the square of motor rotational speed. One can then directly calculate the estimate of c_T from the slope of the fitted function.

Analogous to the test of lift coefficient, the propeller drag coefficient can be estimated via similar tests with the help of a 1D torque sensor. Considering that the torque due to drag $\tau_z = c_Q \omega_{t,i}^2$ from a single propeller, a look-up table of measured torque against the square of motor rotational speed can be recorded. Thus the drag coefficient c_Q can be estimated from the slope of the fitted function.

The tests on the identification of propeller aerodynamic coefficients have been carried out in previous research Nonami *et al.* (2010). For the sake of brevity, we here directly list the estimates of lift coefficient and drag coefficient of 8inch and 10inch enhanced-plastic-polymer (EPP) propellers in the following table 2.3.

Platform	I_{xx}	I_{yy}	I_{zz}
QuadroXL (with camera)	0.028[kg·m^2]	0.027[kg·m^2]	0.049[kg·m^2]
QuadroXL	0.027[kg·m^2]	0.026[kg·m^2]	0.047[kg·m^2]
QuadroL	0.022[kg·m^2]	0.021[kg·m^2]	0.042[kg·m^2]
Hexarotor (simulation)	0.035[kg·m^2]	0.046[kg·m^2]	0.098[kg·m^2]

Table 2.2: Physical parameters of multirotor UAV platform: inertial coefficients.

Platform	lift coefficient c_T	drag coefficient c_Q
Propeller EPP 10"4.5′	$1.938e^{-5}$	$2.688e^{-7}$
Propeller EPP 8"4.5′	$8.549e^{-6}$	$1.368e^{-7}$

Table 2.3: Aerodynamic coefficients of multirotor propellers.

2.2 Flight Control of Multirotor UAV

This section gives an overview of the flight control of multirotor UAVs. Since a couple of control approaches to the multirotor UAVs have been proposed in this thesis, we here generally introduce the control architecture and a couple of existing flight control methods that have been widely implemented on the multirotor UAVs. For the purpose of comparison, we have also applied the methods introduced in this section in the real-robot experiments.

2.2.1 Multirotor UAV Control Architecture

It can be found from the Eq. (2.1) that the translational dynamics and rotational dynamics of the multirotor UAV are decoupled. Therefore, an efficient way of multirotor flight control is to regulate the position and attitude of the UAV in a cascaded feedback system.

A block diagram of the UAV flight control system can be illustrated in Fig. 2.4. More specifically, a position controller is applied in a higher level, which, in most cases, outputs the sum of lift forces (thrust) generated from the propellers, $T = f_z$ in (2.1), as well as the desired attitude, i.e. roll, pitch and yaw angles. The desired attitude is then passed into a lower-level controller, which is designed to track the error between current and desired attitudes. The output of the attitude controller is usually the regulated torque τ_x, τ_y, τ_z on body frame. Finally, the regulated force and torque are transferred into the mapping of motor speeds via (2.8), or (2.10).

Apart from the relevant parameters introduced in Sec 2.1.2, a closed-loop control system required an update of the states of UAV position and attitude at every moment.

The current attitude data (rotational angle and angular velocity) are usually comput-

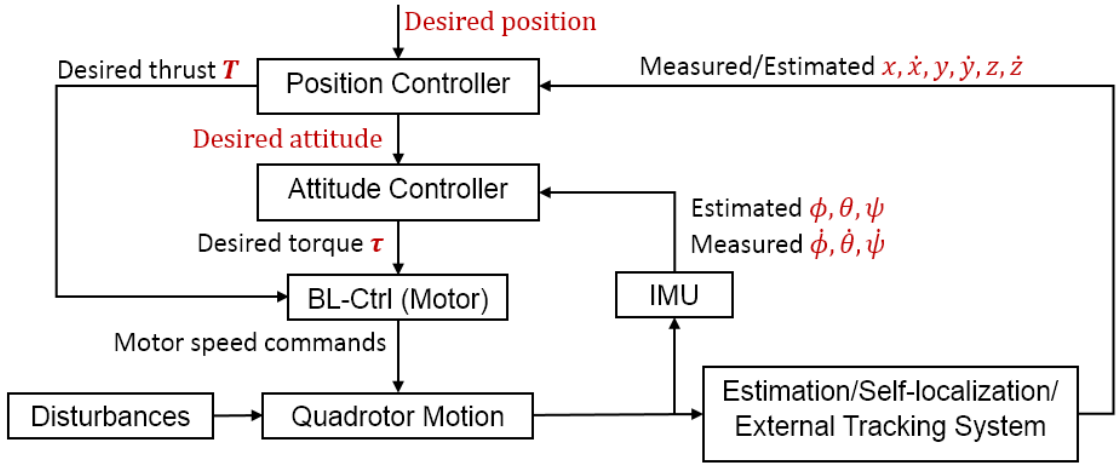


Figure 2.4: The diagram of the casacaded UAV flight control system.

ed/filtered based on the record from an embedded inertia-measurement-unit (IMU) on the multirotor UAV platform, which is able to collect real-time angular velocity and linear acceleration. These data collected from an onboard IMU sensor are usually with bias and are quite noisy, thus the fusing approaches, e.g. a Kalman filter Kalman (1960), or a particle filter Moral (1996), are additionally implemented in order to provide unbiased and smooth angular velocity and roll, pitch, yaw angle at each moment.

The current position data (position, linear velocity, and linear acceleration) can be collected or estimated with the help of onboard or offboard computer vision techniques. For instance, the external motion capture system, i.e. OptiTrack (Naturalpoint, 2017) or Vicon (Vicon, 2017), is able to provide accurate position of the tracked objects at 120Hz or faster. The linear velocity can then be differentiated from the position messages. Other onboard-camera-based vision techniques, e.g. optical flow Cesetti *et al.* (2010), simultaneous localization and mapping (SLAM) Bailey and Durrant-Whyte (2006), parallel tracking and mapping (PTAM) Klein and Murray (2007), also enable the multirotor UAV to estimate its current position in real-time. With the usage of onboard cameras and IMU sensor, the full autonomy of the multirotor UAV is feasible in a GPS-denied environment without the help of external tracking systems.

2.2.2 Proportional-Integral-Derivative Control

One of the existing control approaches that is dominant in engineering fields is the proportional-integral-derivative (PID) Control. Although it has been developed since 1920s, PID control is still widely used because of its simplicity, reliability and model-free characteristic.

A nested flight control framework, as illustrated in Fig. 2.5, has been developed and experimentally validated in Michael *et al.* (2010), which consists of a cascaded atti-

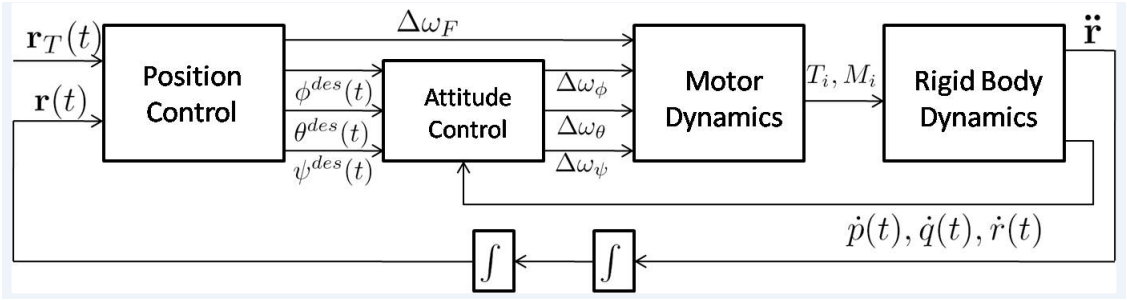


Figure 2.5: The casacaded UAV flight control system in Michael *et al.* (2010).

tude controller plus near-hovering position controller structure, with both controllers in linear PID style. Additionally, a 3D trajectory planing algorithm has been developed together with the PID feedback control system, which enables the agile quadrotor UAV, with accurate model identification, to achieve precise maneuvers (at 200Hz or faster frequency). Note that these controllers rely on small angle assumptions, e.g. the tilted angle of quadrotor should not be larger than 20°

The attitude controller is in a proportional-derivative (PD) type in form of

$$\Delta\omega_\phi = k_{p,\phi}\left(\phi^{des} - \phi\right) + k_{d,\phi}\left(p^{des} - p\right), \tag{2.11a}$$

$$\Delta\omega_\theta = k_{p,\theta}\left(\theta^{des} - \theta\right) + k_{d,\theta}\left(q^{des} - q\right), \tag{2.11b}$$

$$\Delta\omega_\psi = k_{p,\psi}\left(\psi^{des} - \psi\right) + k_{d,\psi}\left(r^{des} - r\right), \tag{2.11c}$$

where $\Delta\omega_\phi$, $\Delta\omega_\theta$, and $\Delta\omega_\psi$ produce moments leading to the rotation of roll, pitch and yaw, respectively.

The desired quadrotor motor speeds can be calculated as

$$\begin{bmatrix}\omega_1^{des}\\ \omega_2^{des}\\ \omega_3^{des}\\ \omega_4^{des}\end{bmatrix} = \begin{bmatrix}1 & 0 & 1 & -1\\ 1 & -1 & 0 & 1\\ 1 & 0 & -1 & -1\\ 1 & 1 & 0 & 1\end{bmatrix}\begin{bmatrix}\omega_h + \Delta\omega_F\\ \Delta\omega_\phi\\ \Delta\omega_\theta\\ \Delta\omega_\psi\end{bmatrix}, \tag{2.12}$$

where ω_h denotes the nominal rotor speed of a steady hovering state, and $\Delta\omega_F$ leads to a net force on z_B axis. In addition, $\Delta\omega_\phi$, $\Delta\omega_\theta$, and $\Delta\omega_\psi$ are the deviations of attitude.

At a higher position control level, a near-hovering controller is applied on the quadrotor to reach the desired position and yaw angle with zero linear and angular velocities. Denoting $\vec{r}_T$ and ψ_T the trajectory and yaw angle to be tracked by the position controller, and the position error $e_i = (\vec{r}_{T,i} - \vec{r}_i)$, the desired acceleration $\ddot{\vec{r}}_i^{des}$ at ith time step can be

calculated from a proportional-integral-derivative (PID) feedback system as

$$\ddot{\bar{r}}_i^{des} = k_{p,i}e_i + k_{i,i}\int e_i \,\mathrm{d}t + k_{d,i}\dot{e}_i. \tag{2.13}$$

Considering that the thrust produced by the propellers has the relation with the motor speed, $T = fz = c_T\omega^2$. The desired roll, pitch angles and the derivation of thrust can be calculated and passed to the attitude controller via

$$\phi^{des} = \frac{1}{g}(\ddot{\bar{r}}_1^{des}\sin(\psi_T) - \ddot{\bar{r}}_2^{des}\cos(\psi_T)), \tag{2.14a}$$

$$\theta^{des} = \frac{1}{g}(\ddot{\bar{r}}_1^{des}\cos(\psi_T) + \ddot{\bar{r}}_2^{des}\sin(\psi_T)), \tag{2.14b}$$

$$\Delta\omega_F = \frac{m}{8k_F\omega_h}\ddot{\bar{r}}_3^{des}. \tag{2.14c}$$

We briefly introduce this PID control framework here, because we have implemented this control frame on our QuadroXL platform during the validation tests of aggressive maneuvers. We will introduce the detail of experiments later in Chapter 3.

2.2.3 Geometric Tracking Approach

Apart from the classical linear PID control approach, several nonlinear methods based on Lyapunov function, e.g. backstepping and sliding modes Khalil (2002) have been developed and implemented on the multirotor UAV platforms. The major pro of these nonlinear approaches is that they can get rid of the small angle assumptions and enable the multirotor UAVs to achieve the nontrivial maneuvers with large tilted angles.

We here introduce one kind of backstepping controller, the geometric tracking approach Lee *et al.* (2010b), which is one of the nonlinear control approaches that have been successfully implemented on the multirotor UAVs. The architecture of the geometric tracking approach is also in a cascaded fashion, which indicates that a high level position controller is designed to compute the desired thrust and the desired attitude via tracking the reference trajectory and desired yaw angle. The desired attitude is then passed into a lower level attitude controller so that the regulated torque can be calculated. Finally, the regulated thrust and torque are transformed to the angular velocities of each motor via Eq. (2.8).

The major difference between the geometric tracking approach and the classical linear methods is that such controller is developed on the special Euclidean group SE(3), which is the semidirect product of $\mathbb{R}^3$ and the special orthogonal group $\mathrm{SO}(3) = \{R \in \mathbb{R}^3 | \, R^\top R = I, \det R = 1\}$. In Lee *et al.* (2010a), technical details of the algorithm as well as a rigorous proof of asymptotic stability of this nonlinear tracking method has been demonstrated. Therefore, we here only introduce the core concept of this method and several relevant, key equations, e.g. the control laws.

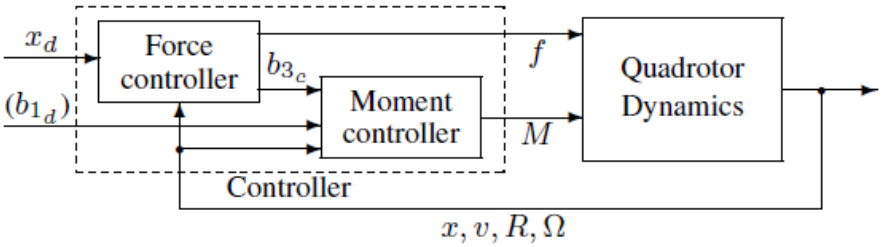


Figure 2.6: Control structure for geometric tracking controller in Lee *et al.* (2010a).

The geometric tracking controller is developed on the nonlinear configuration Lie group and thus it avoids the singularities in local coordinates. The general control structure of the approach is illustrated in Fig. 2.6.

Let the tracking errors between the current state and the position tracking command x_d for the position and velocity be given by:

$$e_x = x - x_d, \tag{2.15a}$$
$$e_v = v - \dot{x}_d. \tag{2.15b}$$

Following the notations in (2.8), the expression of the thrust magnitude for the high level position controller is represented as

$$f_z = (-k_x e_x - k_v e_v + mge_3 - m\ddot{x}_d) \cdot Re_3, \tag{2.16}$$

where k_x, k_v are positive constants, while R denotes the rotation matrix that expresses the current attitude of the quadrotor.

Considering that the desired attitude $R_d \in \mathrm{SO}(3)$ and the desired angular velocity Ω_d are given by

$$R_d = [b_{1_d};\ b_{3_d} \times b_{1_d};\ b_{3_d}],\ \Omega_d = R_d^\top \dot{R}_d, \tag{2.17}$$

where b_{3_d} is defined by

$$b_{3_d} = \frac{-k_x e_x - k_v e_v + mge_3 - m\ddot{x}_d}{\| - k_x e_x - k_v e_v + mge_3 - m\ddot{x}_d \|}, \tag{2.18}$$

and b_{1_d} is selected to be orthogonal to b_{3_d}, for instance

$$b_{1_d} = -\frac{b_{3_d} \times \left(b_{3_d} \times (\cos\psi, \sin\psi, 0)^\top \right)}{\| b_{3_d} \times (\cos\psi, \sin\psi, 0)^\top \|}, \tag{2.19}$$

with the desired yaw angle ψ.

A geometric attitude controller is further developed in order to track the desired atti-

tude calculated in (2.17). Following the derivation in Lee *et al.* (2010a), a real-valued error function is chosen as:

$$\Psi(R,R_d) = \frac{1}{2}\mathrm{tr}[I - R_d^\top R]. \tag{2.20}$$

By denoting the variation of a rotation matrix $\delta R = R\widehat{\eta}$ for $\eta \in \mathbb{R}^3$, the derivative of the error function is given by

$$\mathbf{D}_R\Psi(R,R_d)\cdot R\widehat{\eta} = -\frac{1}{2}\mathrm{tr}[R_d^\top R\widehat{\eta}] = e_R\cdot\eta, \tag{2.21}$$

where the attitude tracking error $e_R \in \mathbb{R}^3$ is chosen as

$$e_R = \frac{1}{2}(R_d^\top R - R^\top R_d)^\vee, \tag{2.22}$$

while the tracking error for the angular velocity is given by

$$e_\Omega = \Omega - R^\top R_d\Omega_d. \tag{2.23}$$

The vee map $\vee$: $\mathrm{SO}(3) \to \mathbb{R}^3$ denotes the inverse of the hat map, a skew symmetric matrix, for instance:

$$\widehat{\Omega}^\vee = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ q & p & 0 \end{bmatrix}^\vee = (p,q,r)^\top. \tag{2.24}$$

The nonlinear geometric tracking controller for the attitude is, therefore, described as:

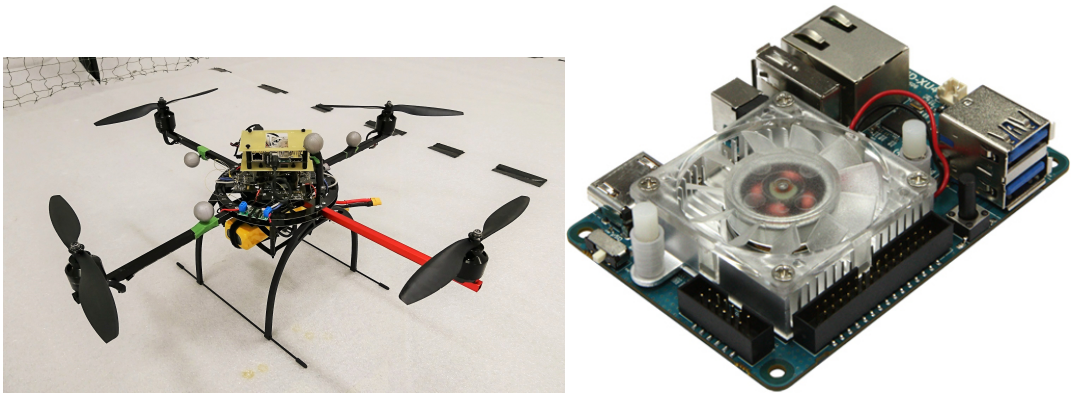
$$M = -k_R e_R - k_\Omega e_\Omega + \Omega\times J\Omega - J(\widehat{\Omega}R^\top R_d\Omega_d - R^\top R_d\dot{\Omega}_d), \tag{2.25}$$

where k_R, k_Ω are positive constant diagonal matrices.

This geometric tracking approach, with the control law (2.16) and (2.24), has also been implemented on our multirotor UAV platform QuadroXL to achieve the nontrivial path-following and waypoint navigation tasks. A comparison among this approach and other proposed methods will be demonstrated in the experiments in Chapter 3.

2.3 Multirotor UAV Platforms

In this thesis, we select two quadrotor configurations to be our multirotor UAV platforms in a series of real-robot experiments. Based on the various mission requirements, one medium-size (about 1.6kg) quadrotor as well as four relatively light-weight (about 1kg) quadrotor UAVs are built. The quadrotor platforms provide us with the benefits such as being simple in mechanism, having sufficient payload for onboard equipments and be-



(a) (b)

Figure 2.7: (a) Side view of the QuadroXL quadrotor UAV platform, with 10-inch propellers and onboard computational unit implemented. (b) Odroid-XU4 onboard computer implemented on our QuadroXL platform, which is used for all the computation on flight control and remote communication. ©Hardkernel co., Ltd.

ing suitable for indoor low-speed flights, thus the quadrotors enable us to experimentally validate our proposed control and trajectory planning algorithms in the indoor environments.

The medium-size quadrotor UAV platform is displayed in Fig. 2.7. The quadrotor mechanical frame is based on a frame from the MikroKopter QuadroXL (HiSystems, 2017), which is made of aluminum material and is thus durable. Later in this thesis, we may call it "QuadroXL" in some cases. The original quadrotor is equipped with four motors and 12-inch propellers, with an capability to carry a payload up to 1.0 kg except battery. The total weight of the quadrotor is approximately 1.605kg, including a 5000mAh 4-cell LiPo battery. The arm length of this quadrotor is 29cm. In order to achieve a flight with high autonomy, an onboard computational unit is implemented on the quadrotor. The onboard computer is one Odroid-XU4 board (Hardkernel, 2017) featuring Samsung Exynos5542 ARM® Cortex™-A15 2GHz and Cortex™-A7 Octa core CPUs, 2G LPDDR3 RAM and 64G SSD. Compared to other available onboard computers, e.g. Intel Nuc (Intel, 2017a), AMD Zotac (AMD, 2017), etc., Odroid-XU4 is with relatively lower cost but less powerful, with 5V input voltage. Hence, we additionally mount a Pololu 5V, 6A step-down voltage regulator chip (Pololu, 2017) on the quadrotor.

The inertial measurement unit (IMU), which includes the inertial sensors e.g. a tri-axes accelerometer, a tri-axes gyroscope, and an air pressure sensor, is embedded on an intermediate printed circuit board (PCB) with original flight control programmings from MikroKopter. However, since in this thesis we focus on the flight control and its implementation, we re-flash the flight control board and use it only as an interface to the readout of raw IMU data and to the transfer of computed motor speed messages to the four brushless motor controllers. The accelerometer and the gyroscope embedded

Figure 2.8: The four quadrotor UAVs for the formation missions.

on IMU provide a 3-dimensional (3D) linear acceleration ($\pm 8g$) and a 3D angular velocity ($\pm 500 deg/s$) measurements of the quadrotor UAV. The IMU board is connected to the onboard computer via two FTDI usb-to-serial cables, one for data readout, and another for publishing control commands. The transfer speed of IMU is theoretically up to 300Hz, while in the real-robot experiments we speed it down to 120Hz, in order to match the frequency of subscribing position data. The brushless controllers work in a open-loop fashion, which might lead to problematic mismatches between the desired motor speeds computed from the control algorithm and the actual motor speeds.

This quadrotor is employed as the experimental platform for the validation of robust control approach to the aggressive maneuvers under disturbances, and the predictive control for the maneuvers with obstacle avoidance, as in Chapter 3 and 4. There may be slight changes on the quadrotor platform during the real-robot experiments. For those specific configurations, we will detail them later in the relevant sections in the technical chapters.

We extend the research direction to the control of multi-agent systems in Chapter 5. In order to experimentally validate the proposed formation control approach, we built another four relatively light-weight quadrotors and command them to simultaneously fly in formation.

An assembly of the four quadrotor UAV platforms for the formation missions is shown in Fig. 2.8. Different from the QuadroXL platform we have introduced, each quadrotor UAV for the formation missions is with the weight of approximately 1.05kg including a 2700mAh 4-cell LiPo battery. Four 10-inch propellers are mounted with (smaller)

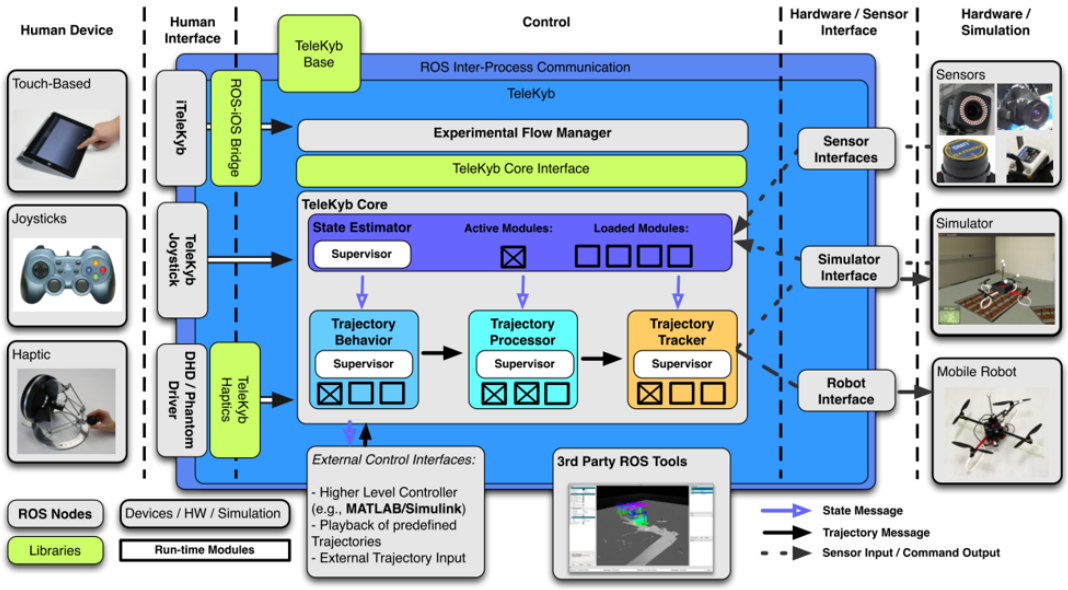


Figure 2.9: The Telekyb Framework for multirotor UAV teleoperation and control in Grabe *et al.* (2013).

motors on the arms of 23.5cm on the quadrotor. We still implement Odroid-XU4 on each quadrotor in order to execute all the computations onboard. The hardware settings in other aspects, i.e. the brushless controllers, the IMU board, the voltage regulator, etc., are kept with no change.

The group of the four quadrotor UAVs are employed for the experimental evaluation of the distributed control approach to the shape-driven formation, as in Chapter 5.

2.3.1 Experimental Configuration

In this section, we generally introduce the configurations of simulations and real-robot experiments. In most cases, the control and trajectory algorithms presented in this thesis have been validated step-by-step, from virtual stage to actual experiment stage.

First we carry out a series of numerical tests in MATLAB for the purpose of sanity check. ROS-based simulations are then carried out via Gazebo multi-robot simulator. We finally set up real-robot experiments by employing the hardware platforms that we just introduced. In order to develop a flexible software environment to experimentally validate our proposed algorithms, we bridged a modified interface between (onboard) computers, Gazebo physical simulator, and the quadrotor platforms on the basis of Telekyb Grabe *et al.* (2013) (as illustrated in Fig. 2.9). This interface enables us to run the exact packages of flight control (written in $C++$ language) for both the virtual simulation and real-robot experiments. In most of the situations, we tuned the relevant parameters (e.g. control gains, weight functions for optimization problems, etc.) in the

Figure 2.10: The Leap Motion gesture tracking device. ©LEAP MOTION, INC

Gazebo simulations and inherited them to the real quadrotors.

Since this thesis focuses on the control aspect, we rely on the external tracking system consisting of a group of infrared cameras on the wall to estimate the current position data of the quadrotors. The experiments we demonstrate in this thesis have been carried out in an indoor environment equipped with Vicon (Vicon, 2017) tracking system. Meanwhile, we have also validated the proposed algorithms using the same quadrotor in an indoor laboratory with OptiTrack (Naturalpoint, 2017) motion capture system.

During the real-robot experiment, the operator uses a Logitech Gamepad F710 joystick to remotely send the commands to the quadrotor. The commands include takeoff, landing, hovering, path-following, and waypoint navigation. The quadrotors are set to be completely autonomous in the whole flight duration. The communication between the joystick and the quadrotors is through the 2.4GHz wireless network.

Apart from the basic operational devices like a joystick, we additionally employ a Leap Motion 3D gesture tracking device (Leap Motion, 2017) for the purpose of achieving formation via human-swarm interaction, as we will later introduce in Chapter 5. The Leap Motion device is shown in Fig. 2.10, which consists of two infrared cameras to track the motions of palms and fingers. In this thesis, we utilize it to track the finger motions of the operator and distinguish the drawn “shape”. The detected shape will be

the formation shape for a group of quadrotors.

Chapter 3

Robust Nonlinear Control for Nontrivial Multirotor Maneuvering

In this chapter, we propose the problem of robust control of multirotor Unmanned Aerial Vehicles (UAVs) for nontrivial/aggressive maneuvers. We present two onboard robust nonlinear control approaches for the quadrotor UAVs in the environment with unknown disturbances.

The first approach consists of an attitude controller based on the solution of rigid body rotations SO(3), a robust position controller for the system with bounded disturbances, and an online trajectory planner based on a model predictive control (MPC) method. While the second approach is a modification on the basis of the first robust control approach. The new control structure is with the combination of the first approach and a 6-dimensional (6D) wrench (force/torque) observer to estimate the unknown force and torque disturbances. The performance of the two proposed algorithms have been validated through intensive experiments and compared with another nonlinear control method on the aggressive waypoint navigation and path following tasks in the presence of external disturbances, e.g. wind gusts.

Large parts of this work have been pre-published in Liu *et al.* (2015, 2017).

3.1 Introduction

Unmanned Aerial Vehicles (UAVs) have been an area of keen interest in the robotics community since the last decade. Among the variety of UAVs, quadrotor Micro Aerial Vehicles (MAVs) have becomed popular because of their mechanical simplicity, the capability of vertical takeoff and landing (VTOL), as well as the potential to low-speed flight. With the application of advanced control and computer vision techniques, MAVs are recently able to achieve versatile tasks, from autonomous navigation and mapping Heng *et al.* (2011); Fraundorfer *et al.* (2012), rescue searching Mueggler *et al.* (2014), to goods transportation Palunko *et al.* (2012b), construction Augugliaro *et al.* (2014), and aerial acrobatics Lupashin *et al.* (2010); Ritz *et al.* (2012); Mellinger *et al.* (2012), etc.

From the control perspective, research has been carried out on the real implementation of advanced control theories Mellinger *et al.* (2012), on the development of the algo-

rithms that enable quadrotors to reach exceptional maneuverability Lee *et al.* (2010b), and on the improvement of MAV robustness Alexis *et al.* (2011). The application on MAVs of various control theories has been exploited, including the Lyapunov-based nonlinear control approaches such as backstepping and sliding mode Bouabdallah and Siegwart (2005, 2007), as well as the relatively novel control algorithms, e.g. super twisting Derafa *et al.* (2012) and model predictive control Raffo *et al.* (2010), among others. In Raffo *et al.* (2010), a cascaded linear MPC and nonlinear H_∞ control framework has been proposed and numerically verified. However, the classical linear control patterns, such as proportional-derivative controllers Hoffmann *et al.* (2007); Bouabdallah and Siegwart (2007) and Linear Quadratic Regulators (LQR) Bouabdallah and Siegwart (2007), are still dominant in real implementations. A novel geometric tracking method on the special Euclidean group SE(3) for quadrotor motion has been proposed in recent research Lee *et al.* (2010b), and implemented in Mellinger *et al.* (2012) with impressive performances under accurate model identification. A switching explicit MPC has been validated by experimental results in Alexis *et al.* (2011), while a learning-based MPC for quadrotor translational motion has been proposed and tested in Bouffard *et al.* (2012). Although predictive control techniques have been successfully tested on FPGAs Hartley *et al.* (2014), they have been seldom implemented on onboard computational units due to the expensive computational cost.

In this chapter, we first propose a robust nonlinear control method, the Robust Output Regulation (RobOR) approach, henceforth called the RobOR method. we also focused on its onboard implementation for a heavier and less agile quadrotor UAV with model mismatches, in order to enable the quadrotor to achieve waypoint navigation and path following tasks with aggressive, large-tilt maneuvers. Technically, the RobOR method is a combination of a robust backstepping-like controller for a system with bounded disturbances, and a nonlinear attitude controller that is suitable for almost all initial attitudes, which is based on the solution of a certain class of global output regulation problems for a nonlinear rigid-body system on Lie groups and is endowed with strong convergence properties. An online trajectory planner based on a linear MPC method, which considers the trajectory planning problem as three decoupled local Optimal Control Problems (OCPs), is additionally developed and comined with the proposed nonlinear control algorithm, for the purpose of achieving both waypoint navigation and path-following tasks smoothly.

In order to further improve the performance of the RobOR method presented in this chapter, we extend the control architecture into a modified approach with the inclusion of a nonlinear wrench observer. Many interesting approaches have been proposed recently in the context of aerial robots and interaction. In Hacksel and Salcudean (1994), nonlinear observers for velocity and force estimation on a rigid body are presented. An unscented Kalman Filter for the linearized model of a quadrotor is proposed in Augugliaro and D'Andrea (2013). Recently, adaptive control methods have been employed to counteract external disturbances Roberts and Tayebi (2009); Palunko *et al.* (2012a); Antonelli *et al.* (2013). In Bellens *et al.* (2012), external wrench estimation for flying robots has

been investigated in the context of hybrid pose/wrench control, where they performed an offline measurement of the forces and torques generated by the UAV. However, the UAV was fixed to a base during the experiments. A force control approach with an external feedforward signal has been utilized in Albers *et al.* (2010). A force sensor is used as an estimator in Nguyen and Lee (2013). An alternative Lyapunov-based nonlinear observer for estimating the external forces applied has been proposed in Yüksel *et al.* (2014) and numerically validated. However, it can not precisely track a rapidly varying wrench. In our control algorithm we employ a nonlinear 6 dimensional (6D) force/torque wrench observer with the concept of a momenta-based residual technique. This residual technique has been earlier used in Collision/Fault Detection and Identification (FDI) methodology for robotic manipulator arms Takakura *et al.* (1989); De Luca and Mattone (2003) and recently been exploited for other robotic platforms Tomic and Haddadin (2015).

In summary, we develop an overall system including a robust nonlinear flight controller with a 6D force/torque estimator, as well as an online MPC-based trajectory planner, which considers the trajectory planning problem as a coupled 3D local OCP. The complete system is implemented onboard on a low-power computational unit. We call the complete control framework the RobOR+ method henceforth in this chapter. Technically, the RobOR+ method includes the same nonlinear backstepping-like controllers we developed in the RobOR method for position and attitude control. The control performance is additionally improved with the use of the proposed online nonlinear disturbance observer plus the trajectory planner. This framework provides the multirotor UAVs, even ones heavier, less agile and with uncertainties, with robust performance when achieving aggressive waypoint navigation and path following tasks in a scenario with wind gusts. In order to demonstrate the robust control algorithm intuitively, we tested the state-of-the-art geometric tracking method as introduced in Lee *et al.* (2010b) and tested in Mellinger and Kumar (2011); Mellinger *et al.* (2012) (we have briefly introduce this method in Sec. 2.2.3, henceforth it is called the GeoTrack method), the RobOR method we first proposed, and the improved RobOR+ method on our quadrotor platform on the exact same tasks for the purpose of comparison.

The outline of this chapter is as follows. Sec. 3.2 describes the structure of our proposed control framework. A nonlinear approach for the position and attitude control for the quadrotor UAVs, including the proofs revealing that our algorithms are asymptotically stable, are introduced as the main structure of both RobOR and RobOR+ methods. In Sec. 3.3, we demonstrate the robustness design applied in our first proposed RobOR method. We then present a nonlinear 6D wrench observer, which is used in RobOR+ method, for unknown disturbances and prove the asymptotic stability of the complete control-observation system in Sec. 3.4. An online trajectory planner using a MPC method for waypoint navigation or path following tasks is introduced in Sec. 3.5. Sec. 3.6 demonstrates the configuration and the results of the experimental evaluations we have carried out to validate our proposed control approach, as well as the comparisons with the GeoTrack method. Finally, we conclude this chapter in Sec. 3.7.

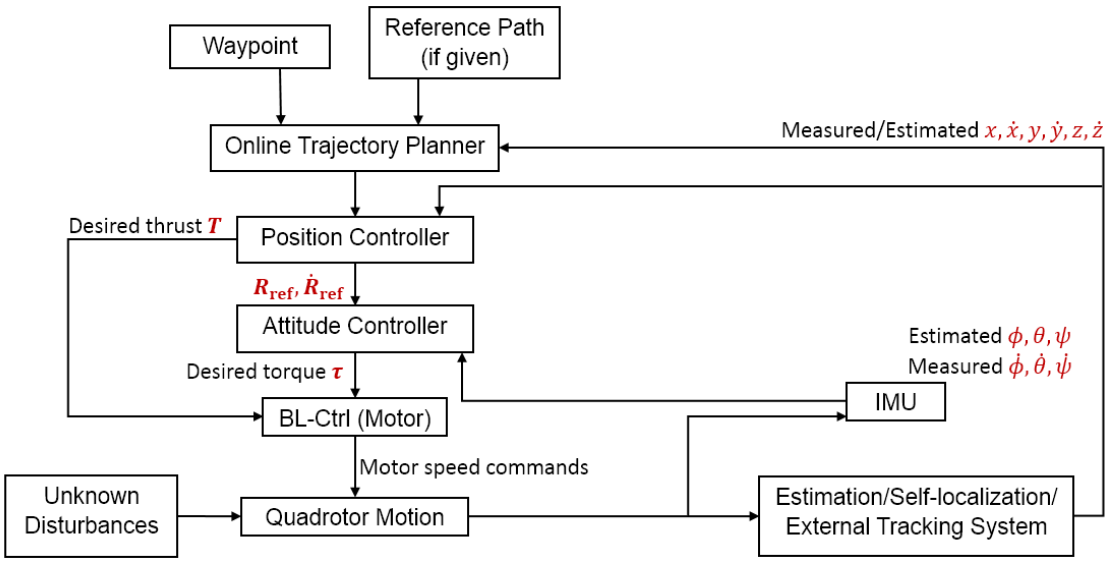


Figure 3.1: The diagram of a feedback system for a quadrotor UAV.

3.2 Control Approach

We here start the technical details on the two nonlinear control approaches.

Fig. 3.1 depicts the block diagram of a closed-loop control system for a quadrotor UAV. Although it is possible to develop a single-layer control approach to the position and attitude of the UAV, a cascaded system is more popular for the multirotor UAV because of the decoupled characteristic of the multirotor dynamics. More specifically, in case the model-based control methods are applied, only the rotational dynamic model used in the attitude controller is nonlinear. Thus the position controller is based on a linear translational model, which is more effective and leads to lower computational cost.

In our proposed system, we develop a cascaded feedback with both position and attitude controller designed based on the dynamic model of the multirotor.

For the control theorem derivation and discussion, the dynamic equations of the multirotor UAV (2.1) are rewritten simplified in their Newton-Euler formulation as

$$\dot{\boldsymbol{x}} = v, \tag{3.1a}$$
$$m\dot{v} = -mge_3 + RF + b_F, \tag{3.1b}$$
$$\dot{R} = RQ(\omega), \tag{3.1c}$$
$$J\dot{\omega} = -\omega \times J\omega + \tau + b_\tau, \tag{3.1d}$$

where m is the mass of the quadrotor, $\boldsymbol{x}$ denotes the translational position, g is the scalar value of the gravity acceleration, $e_3 = (0,0,1)^\top$, F is the nonconservative force generated in the body frame of the quadrotor, τ is the nonconservative moments generated

in the body frame of the quadrotor by the aerodynamics of the rotors, b_F is the disturbance due to the external force and unstructured dynamics in the inertial frame. b_τ is the disturbance due to the external torque and unstructured dynamics in the body frame. The inertial matrix J in the body frame is computed on a simple CAD model of the QuadroXL platform used in the real experiment (as introduced in Chapter 2), with unknown mismatch to the actual values. $Q(\omega)$ is the skew-symmetric matrix form of the angular velocity ω in the body frame cross product such that

$$Q(\omega) = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}. \tag{3.2}$$

The rotation matrix R represents the attitude of the quadrotor with respect to the inertial frame, which is an element of the special orthogonal group SO(3)$= \{R \in \mathbb{R}^{3\times 3} | R^{-1} = R^\top, \det R = 1\}$.

On the quadrotor UAV, the direction of the thrust T is along the Z axis of the body frame and therefore $F = (f_x, f_y, f_z)^\top = (0,0,T)^\top$. The desired angular velocity $\omega_{t,i}$ of each motor, which is assumed as the control input on a quadrotor UAV by neglecting the error from the motor controller, therefore, can be obtained via solving the inverse of (2.8). The regulation approach to compute the desired nonconservative forces and moments on the quadrotor is discussed further in this section.

On the promise of fulfilling various tasks, one can pre-describe a set of references, which can either be a path or simply a waypoint far from the quadrotor. An online trajectory planner (which we will introduce later) generates a trajectory so that later the quadrotor moves via tracking these references. The reference information, as well as the position and attitude data collected or estimated via reliable onboard hardware, algorithms or external motion capture systems, are passed to a control block, which outputs the desired rotation speed of each motor on the quadrotor via a cascaded position and attitude control algorithm and finally the transformation through (2.8). The details of the regulation algorithms for the quadrotor position and attitude are introduced in Sec. 3.2.1 and 3.2.2, respectively.

3.2.1 Position Control

First, a position controller based on a nonlinear backstepping approach is introduced here, which tracks the desired translational state and generates a desired thrust T. The desired attitude matrix and its derivative are further calculated and passed into the attitude controller .

Subject to the first two equations in (3.1) but starting with an assumption of the full knowledge of external disturbances, we define the strongly convex cost function

$$P_1(x - x_{\text{ref}}) = (x - x_{\text{ref}})^\top K_0 (x - x_{\text{ref}}), \tag{3.3}$$

in order to specify the desired differential equation

$$\dot{x}_{\text{desired}} - \dot{x}_{\text{ref}} = -\nabla P_1(x - x_{\text{ref}}), \tag{3.4}$$

where x_{desired} and x_{ref} represents the desired quadrotor position resulting from the above gradient system and the exogenous reference position to track, respectively, while $K_0 = \text{diag}(k_{0x}, k_{0y}, k_{0z})$ is a positive gain matrix (containing the diagonal elements denoting positive gains in each axis) for the position cost. The tracking error for the velocity and its derivative are accordingly given by

$$e_v = \dot{x} - \dot{x}_{\text{desired}} = v - \dot{x}_{\text{ref}} + \nabla P_1(x - x_{\text{ref}}), \tag{3.5a}$$

$$\dot{e}_v = \dot{v} - \ddot{x}_{\text{ref}} + \nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}). \tag{3.5b}$$

Using a backstepping-like approach, we choose a Lyapunov function candidate

$$V_{\text{pos}}(e_v, x - x_{\text{ref}}) = \frac{1}{2} e_v^\top e_v + P_1(x - x_{\text{ref}}), \tag{3.6}$$

whose Lie derivative is rendered negative definite (under the assumption of full disturbance knowledge as we will see later) if R equals R_{ref} subject to

$$\begin{aligned} R_{\text{ref}} F \triangleq\ & mge_3 - b_F - m\nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}) \\ & + m\ddot{x}_{\text{ref}} - mKe_v - m\nabla P_1(x - x_{\text{ref}}) =: F', \end{aligned} \tag{3.7}$$

where $K = \text{diag}(k_x, k_y, k_z)$ is a positive gain matrix (in the same structure as K_0) for the tracking error.

Considering that R_{ref} in (3.7) is orthonormal and only the third entry of F, i.e. the thrust T, is nonzero, one must choose $T = \|F'\|$ in order to suffice (3.7). Therefore, the third column of $R_{\text{ref}} = \begin{bmatrix} r^1_{\text{ref}} & r^2_{\text{ref}} & r^3_{\text{ref}} \end{bmatrix}$ can be solved from

$$r^3_{\text{ref}} = \frac{F'}{\|F'\|}, \tag{3.8}$$

and the other two columns of R_{ref} can be filled orthonormally, for instance by a Gram-Schmidt process with candidates r^3_{ref} from the previous equation and r^1, r^2 from the actual rotation $R = \begin{bmatrix} r^1 & r^2 & r^3 \end{bmatrix}$, i.e.

$$r^{2\prime}_{\text{ref}} = r^2 - \frac{r^2 \cdot r^3_{\text{ref}}}{r^3_{\text{ref}} \cdot r^3_{\text{ref}}} r^3_{\text{ref}}, \tag{3.9a}$$

$$r^2_{\text{ref}} = \frac{r^{2\prime}_{\text{ref}}}{\|r^{2\prime}_{\text{ref}}\|}, \tag{3.9b}$$

$$r^{1\prime}_{\text{ref}} = r^1 - \frac{r^1 \cdot r^3_{\text{ref}}}{r^3_{\text{ref}} \cdot r^3_{\text{ref}}} r^3_{\text{ref}} - \frac{r^1 \cdot r^2_{\text{ref}}}{r^2_{\text{ref}} \cdot r^2_{\text{ref}}} r^2_{\text{ref}}, \tag{3.9c}$$

$$r^1_{\text{ref}} = \frac{r^{1\prime}_{\text{ref}}}{\|r^{1\prime}_{\text{ref}}\|}, \tag{3.9d}$$

such that we obtain both the desired total thrust T and the desired attitude R_{ref} for the quadrotor control.

Here we prove the asymptotic stability of this position controller for $R = R_{\text{ref}} \triangleq \left[r^1_{\text{ref}}\, r^2_{\text{ref}}\, r^3_{\text{ref}}\right]$. By sending RF to F', the closed-loop dynamics under the full disturbance knowledge assumption is given by

$$\begin{aligned}
\dot{e}_v &= \dot{v} - \ddot{x}_{\text{ref}} + \nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{ref}) \\
&= -ge_3 + \frac{b_F}{m} + \frac{1}{m} RF - \dot{v}_{\text{ref}} + \nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}) \\
&\triangleq -ge_3 - \dot{v}_{\text{ref}} + \nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}) + ge_3 + \dot{v}_{\text{ref}} \\
&\quad - \nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}) - Ke_v - \nabla P_1(x - x_{\text{ref}}) \\
&= -Ke_v - \nabla P_1(x - x_{\text{ref}}).
\end{aligned} \tag{3.10}$$

The (directional) derivative of the chosen Lyapunov function candidate (3.6) is

$$\begin{aligned}
\dot{V}_{\text{pos}} &= \nabla P_1(x - x_{\text{ref}}) \cdot (\dot{x} - \dot{x}_{\text{ref}}) + e_v \cdot \dot{e}_v \\
&= \nabla P_1(x - x_{\text{ref}}) \cdot (e_v - \nabla P_1(x - x_{\text{ref}})) + e_v \cdot (-Ke_v - \nabla P_1(x - x_{\text{ref}})) \\
&= -\nabla P_1(x - x_{\text{ref}})^\top \nabla P_1(x - x_{\text{ref}}) - Ke_v^\top e_v \\
&< 0,
\end{aligned} \tag{3.11}$$

hence (3.11) is negative definite.

3.2.2 Attitude Control

In this section, we introduce a control method for the tracking of the attitude R of the quadrotor with strong convergence properties. The specific approach is based on the solution of a class of output regulation problems which contains the rotational motion for a rigid body. The tracking error is given by $E = RR^\top_{\text{ref}}$, where R corresponds to the rotation matrix describing the current attitude, and R_{ref}, computed via the approach introduced in Section 3.2.1, represents the desired attitude. We also assume the error between the current and a desired angular velocity in body frame as

$$e_\omega = Q(\omega) - Q(\omega_{\text{desired}}). \tag{3.12}$$

The goal of the attitude controller is to regulate the output of the quadrotor attitude $(E, e_\omega) \to (I, 0)$ for $t \to \infty$, which implies $R \to R_{\text{ref}}$ and $Q(\omega) \to Q(\omega_{\text{desired}})$ for $t \to \infty$.

We employ a backstepping method analogous to the position controller for the attitude regulation. Based on the proof in Schmidt *et al.* (2013a), a cost function $P_2(E) = \frac{1}{2}\mathrm{tr}((E-I)^\top(E-I)) = n - \mathrm{tr}(E)$, which has

$$\mathrm{grad}\, P_2(E) = \frac{1}{2}(E - E^\top)E, \tag{3.13}$$

can be chosen, where $\mathrm{grad}\, P_2$ is the projection of ∇P_2 onto tangent spaces of SO(3). Here we assume a cost function for the quadrotor rotation in form of

$$P_2(E) = 3 - \mathrm{tr}(E). \tag{3.14}$$

The derivative of the rotational error E becomes

$$\dot{E} = \dot{R}R_{\mathrm{ref}}^\top + R\dot{R}_{\mathrm{ref}}^\top \triangleq -K_{\mathrm{rot}}\,\mathrm{grad}\, P_2(E), \tag{3.15}$$

where K_{rot} is a positive gain for the cost of rigid body rotation.

By employing (3.1c), the desired angular velocity of the quadrotor is given by

$$Q(\omega_{\mathrm{desired}}) = -\frac{K_{\mathrm{rot}}}{2}(R_{\mathrm{ref}}^\top R - R^\top R_{\mathrm{ref}}) - \dot{R}_{\mathrm{ref}}^\top R_{\mathrm{ref}}, \tag{3.16}$$

which is obtained from (3.15):

$$RQ(\omega_{\mathrm{desired}})R_{\mathrm{ref}}^\top \triangleq -K_{\mathrm{rot}}\,\mathrm{grad}\, P_2(E) - R\dot{R}_{\mathrm{ref}}^\top \tag{3.17a}$$

$$\begin{aligned} Q(\omega_{\mathrm{desired}}) &\triangleq R^\top(-K_{\mathrm{rot}}\,\mathrm{grad}\, P_2(E))R_{\mathrm{ref}} - R^\top R\dot{R}_{\mathrm{ref}}^\top R_{\mathrm{ref}} \\ &= -K_{\mathrm{rot}}R^\top\frac{1}{2}(RR_{\mathrm{ref}}^\top - R_{\mathrm{ref}}R^\top)RR_{\mathrm{ref}}^\top R_{\mathrm{ref}} - \dot{R}_{\mathrm{ref}}^\top R_{\mathrm{ref}}. \end{aligned} \tag{3.17b}$$

The derivative of the desired angular velocity computed from (3.16) is

$$\begin{aligned} Q(\dot{\omega}_{\mathrm{desired}}) &= \frac{d}{dt}\left\{-\frac{K_{\mathrm{rot}}}{2}(R_{\mathrm{ref}}^\top R - R^\top R_{\mathrm{ref}}) - \dot{R}_{\mathrm{ref}}^\top R_{\mathrm{ref}}\right\} \\ &= -\frac{K_{\mathrm{rot}}}{2}(\dot{R}_{\mathrm{ref}}^\top R + R_{\mathrm{ref}}^\top\dot{R} - \dot{R}^\top R_{\mathrm{ref}} - R^\top\dot{R}_{\mathrm{ref}}) - \ddot{R}_{\mathrm{ref}}^\top R_{\mathrm{ref}} - \dot{R}_{\mathrm{ref}}^\top\dot{R}_{\mathrm{ref}}. \end{aligned} \tag{3.18}$$

Considering Eqs. (3.1d) and (3.18), we obtain the derivative of the angular velocity error represented by

$$\begin{aligned} \dot{e}_\omega &= Q(\dot{\omega}) - Q(\dot{\omega}_{\mathrm{desired}}) \\ &= Q(J^{-1}(-\omega\times J\omega + \tau + b_\tau)) + \ddot{R}_{\mathrm{ref}}^\top R_{\mathrm{ref}} + \dot{R}_{\mathrm{ref}}^\top\dot{R}_{\mathrm{ref}} \\ &\quad + \frac{K_{\mathrm{rot}}}{2}(\dot{R}_{\mathrm{ref}}^\top R + R_{\mathrm{ref}}^\top\dot{R} - \dot{R}^\top R_{\mathrm{ref}} - R^\top\dot{R}_{\mathrm{ref}}). \end{aligned} \tag{3.19}$$

For a backstepping approach for the rigid body rotation, we choose a Lyapunov function candidate

$$V_{\text{att}}(E, e_\omega) = \frac{1}{2} Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega) + P_2(E), \tag{3.20}$$

whose Lie derivative, similar as the position controller we introduced in Sec. 3.2.1, is rendered negative definite if τ equals τ_{desired} subject to

$$\begin{aligned} \tau_{\text{desired}} = JQ^{-1} \Bigg(& -K_\omega e_\omega - \frac{K_{\text{rot}}}{2} \Big[\dot{R}_{\text{ref}}^\top R + R_{\text{ref}}^\top \dot{R} - \dot{R}^\top R_{\text{ref}} \\ & - R^\top \dot{R}_{\text{ref}} \Big] - \ddot{R}_{\text{ref}}^\top R_{\text{ref}} - \dot{R}_{\text{ref}}^\top \dot{R}_{\text{ref}} - 2R_{\text{ref}}^\top R \Bigg) + \omega \times J\omega - b_\tau, \end{aligned} \tag{3.21}$$

where K_ω is a positive gain for the tracking error of the quadrotor angular velocity. The derivative $\dot{R}$ can be obtained by introducing the current attitude R into (3.1c), while R_{ref} inherits from the results of (3.8) and (3.9). We first compute the derivative of quadrotor desired attitude $\dot{R}_{\text{ref}}$ from (3.7) through the following process:

The (closed) derivative of F' in (3.7) is

$$\begin{aligned} \dot{R}_{\text{ref}} F \triangleq & \frac{d}{dt} \big(mge_3 - b_F - m\nabla^2 P_1 (x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}) + m\ddot{x}_{\text{ref}} \\ & - mKe_v - m\nabla P_1 (x - x_{\text{ref}}) \big) \\ = & -m(2K_0 + K)(\ddot{x} - \ddot{x}_{\text{ref}}) - 2mK_0 K(\dot{x} - \dot{x}_{\text{ref}}) + m\dddot{x}_{\text{ref}} - \dot{b}_F \\ =: & \ \dot{F}', \end{aligned} \tag{3.22}$$

from which the reference angular velocity is computed using (3.1c),

$$Q(\omega_{\text{ref}}) F = R_{\text{ref}}^\top \dot{F}'. \tag{3.23}$$

Recalling the third column entry of $Q(\omega)$ in (3.2), one can obtain the reference roll and pitch rate in form of

$$p_{\text{ref}} = -\frac{(R_{\text{ref}}^\top \dot{F}')^2}{\|F'\|}, \tag{3.24a}$$

$$q_{\text{ref}} = \frac{(R_{\text{ref}}^\top \dot{F}')^1}{\|F'\|}. \tag{3.24b}$$

Considering that in most cases the yaw attitude is required as the camera front direction (or zero for the quadrotor with no vision-based equipment) and no specific desired yaw rate is required for the quadrotor flight, one can constantly define the value of $r_{\text{ref}} = \frac{1}{2}(\psi_{\text{ref}} - \psi_{\text{current}})$ for the reference and current yaw attitude ψ at every moment, and $\dot{R}_{\text{ref}}$, therefore, can be solved via (3.1c).

Analogous to the derivations for the position controller, we prove here the asymptotic stability of the attitude controller for $\tau = \tau_{\text{desired}}$ under the assumption of full knowledge

of torque disturbance. By sending τ to (3.1d), the closed loop dynamics for rigid body rotation is given by

$$\begin{aligned}
\dot{e}_\omega \triangleq\ & Q\Big(J^{-1}\Big(-\omega\times J\omega+\omega\times J\omega+JQ^{-1}\Big(-2R_{\text{ref}}^\top R \\
& -K_\omega e_\omega-\frac{K_{\text{rot}}}{2}\left[\dot{R}_{\text{ref}}^\top R+R_{\text{ref}}^\top\dot{R}-\dot{R}^\top R_{\text{ref}}-R^\top\dot{R}_{\text{ref}}\right] \\
& -\ \ddot{R}_{\text{ref}}^\top R_{\text{ref}}-\dot{R}_{\text{ref}}^\top\dot{R}_{\text{ref}}\Big)+b_\tau-b_\tau\Big)\Big)+\ddot{R}_{\text{ref}}^\top R_{\text{ref}}+\dot{R}_{\text{ref}}^\top\dot{R}_{\text{ref}} \\
& +\frac{K_{\text{rot}}}{2}\left[\dot{R}_{\text{ref}}^\top R+R_{\text{ref}}^\top\dot{R}-\dot{R}^\top R_{\text{ref}}-R^\top\dot{R}_{\text{ref}}\right] \\
=\ & -K_\omega e_\omega-2R_{\text{ref}}^\top R.
\end{aligned} \tag{3.25}$$

The (directional) derivative of the chosen Lyapunov function candidate (3.20), is

$$\begin{aligned}
\dot{V}_{\text{att}}(E,e_\omega) &= Q^{-1}(e_\omega)^\top Q^{-1}(\dot{e}_\omega)+\operatorname{grad}P_2(E)\cdot\dot{E} \\
&= Q^{-1}(e_\omega)^\top Q^{-1}(-K_\omega e_\omega-2R_{\text{ref}}^\top R)+\operatorname{grad}P_2(E)\cdot\dot{E} \\
&= -Q^{-1}(e_\omega)^\top Q^{-1}(2R_{\text{ref}}^\top R) \\
&\quad -K_\omega Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega)+\operatorname{grad}P_2(E)\cdot\dot{E}.
\end{aligned} \tag{3.26}$$

Considering the property $x^\top y=-\frac{1}{2}\operatorname{tr}[Q(x)Q(y)]$ for any two vectors, we can reach

$$Q^{-1}(e_\omega)^\top Q^{-1}(2R_{\text{ref}}^\top R)=-\frac{1}{2}\operatorname{tr}(e_\omega 2R_{\text{ref}}^\top R), \tag{3.27}$$

by applying (3.15) and (3.27), (3.26) is given by

$$\begin{aligned}
\dot{V}_{\text{att}}(E,e_\omega) &= -K_\omega Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega)+\frac{1}{2}\operatorname{tr}(e_\omega 2R_{\text{ref}}^\top R)-\operatorname{tr}(\dot{E}) \\
&= -K_\omega Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega)+\frac{1}{2}\operatorname{tr}(e_\omega 2R_{\text{ref}}^\top R-2\dot{E}) \\
&=: \dot{V}_{\text{att},1}+\dot{V}_{\text{att},2},
\end{aligned} \tag{3.28}$$

where $\dot{V}_{\text{att},1}=-K_\omega Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega)<0$.

We therefore only focus on the rest term $\dot{V}_{\text{att},2}$ in (3.28) since $\dot{V}_{\text{att},1}$ is negative definite

(proved in Schmidt *et al.* (2013b)),

$$\begin{aligned}
\dot{V}_{\text{att},2} &= \frac{1}{2}\operatorname{tr}(e_\omega 2R_{\text{ref}}^\top R - 2\dot{E}) \\
&= \frac{1}{2}\operatorname{tr}\Bigg(\Bigg(R^\top\dot{R} + \frac{K_{\text{rot}}}{2}(R_{\text{ref}}^\top R - R^\top R_{\text{ref}}) \\
&\quad + \dot{R}_{\text{ref}}^\top R_{\text{ref}}\Bigg) 2R_{\text{ref}}^\top R - 2\dot{R}R_{\text{ref}}^\top - 2R\dot{R}_{\text{ref}}^\top\Bigg) \\
&= \frac{1}{2}\operatorname{tr}(K_{\text{rot}}(R_{\text{ref}}^\top R R_{\text{ref}}^\top R - I)) \\
&= \frac{K_{\text{rot}}}{2}\operatorname{tr}((E^\top - E)E^\top) = \frac{K_{\text{rot}}}{2}\operatorname{tr}(E(E - E^\top)) \\
&= \frac{K_{\text{rot}}}{4}\operatorname{tr}(E(E - E^\top) + (E^\top - E)E^\top) \\
&= \frac{K_{\text{rot}}}{4}\operatorname{tr}((E - E^\top)^2) \\
&< 0,
\end{aligned} \tag{3.29}$$

hence (3.26) is negative definite.

3.3 Robust Design for Position Control

An assumption of full disturbance knowledge is not always practical for real implementations, especially under the condition that the reliable model identification or accurate estimate of external influence is not available. We, therefore, in the following do not impose the assumption of no disturbance for the purpose of robustness.

In our case, we regard the sum of the unstructured dynamics, internal disturbance and the external forces as the disturbance b from the exogenous system, and we need an estimate of such disturbance, b_{est}, such that the disturbance error $e_b = \frac{1}{m}(b_{\text{est}} - b_F) \to 0$ for $t \to \infty$. Although still not in perfect practice, we further choose a Lyapunov function candidate

$$V_{\text{pos},1}(e_v, x - x_{\text{ref}}) = V_{\text{pos}}(e_v, x - x_{\text{ref}}) + \frac{K_b^{-1}}{2} e_b^\top e_b, \tag{3.30}$$

under the assumption of a bounded external disturbance whose first derivative is negligible, which still works for a dynamic disturbance with trivial changes. K_b denotes the gain diagonal matrix of the disturbance observer. The Lie derivative of $V_{\text{pos},1}(e_v, x - x_{\text{ref}})$ is rendered negative definite (as we will see later) subject to

$$\begin{aligned}
\dot{b}_{\text{est}} &= mK_b(\dot{x} - \dot{x}_{\text{ref}}) + 2mK_bK_0(x - x_{\text{ref}}) \\
b_{\text{est}} &= \int_{-\infty}^{t} \dot{b}_{\text{est}}(s)ds,
\end{aligned} \tag{3.31}$$

where b_{est} can replace the external force disturbance term b_F in (3.7) in the closed-loop feedback system.

In the next step, we consider the real environment without full knowledge of the disturbance generated from the unstructured dynamics and exogenous system. Still by sending RF to F', but replacing b_F with b_{est}, the closed-loop dynamics is now given by

$$\begin{aligned}\dot{e}_v =& -\frac{1}{m}(b_{\text{est}} - b_F) - Ke_v - \nabla P_1(x - x_{\text{ref}}) \\ =& -Ke_v - \nabla P_1(x - x_{\text{ref}}) - e_b.\end{aligned} \tag{3.32}$$

The (directional) derivative of the Lyapunov function candidate (3.6) now turns to be

$$\dot{V}_{\text{pos}} = -\nabla P_1(x - x_{\text{ref}})^\top \nabla P_1(x - x_{\text{ref}}) - Ke_v^\top e_v - e_v^\top e_b, \tag{3.33}$$

which is no longer negative definite for arbitrary disturbance error.

We here naively assume that the change of external force disturbances is very slow, thus $\dot{b}_F$ is negligible. Now we consider the new Lyapunov function candidate (3.30) extended with a term of disturbance error, whose derivative, subject to a chosen derivative of disturbance as in (3.31), is given by

$$\begin{aligned}\dot{V}_{\text{pos},1} &= \dot{V}_{\text{pos}} + \frac{K_b^{-1}}{2}\left(\dot{e}_b^\top e_b\right) \\ &= \dot{V}_{\text{pos}} + \frac{K_b^{-1}}{m} e_b^\top \dot{b}_{\text{est}} \\ &= -\nabla P_1(x - x_{\text{ref}})^\top \nabla P_1(x - x_{\text{ref}}) - Ke_v^\top e_v \\ &\quad - e_v^\top e_b + \frac{K_b^{-1}}{m} e_b^\top \dot{b}_{\text{est}} \\ &= -\nabla P_1(x - x_{\text{ref}})^\top \nabla P_1(x - x_{\text{ref}}) - Ke_v^\top e_v \\ &< 0,\end{aligned} \tag{3.34}$$

Therefore, (3.34) is negative definite, which proves the asymptotical stability of the proposed position control algorithm.

The design of RobOR method (above) is to some extent robust, since the modified position controller is able to reduce the tracking error and thus compensates for the unknown force disturbances. However, it is with limit since there is no compensation for the torque disturbances. Meanwhile, the assumption we proposed in the proof that $\dot{b}_F = 0$ is quite strong and not always practical, thus such a robust design is not perfect in real world.

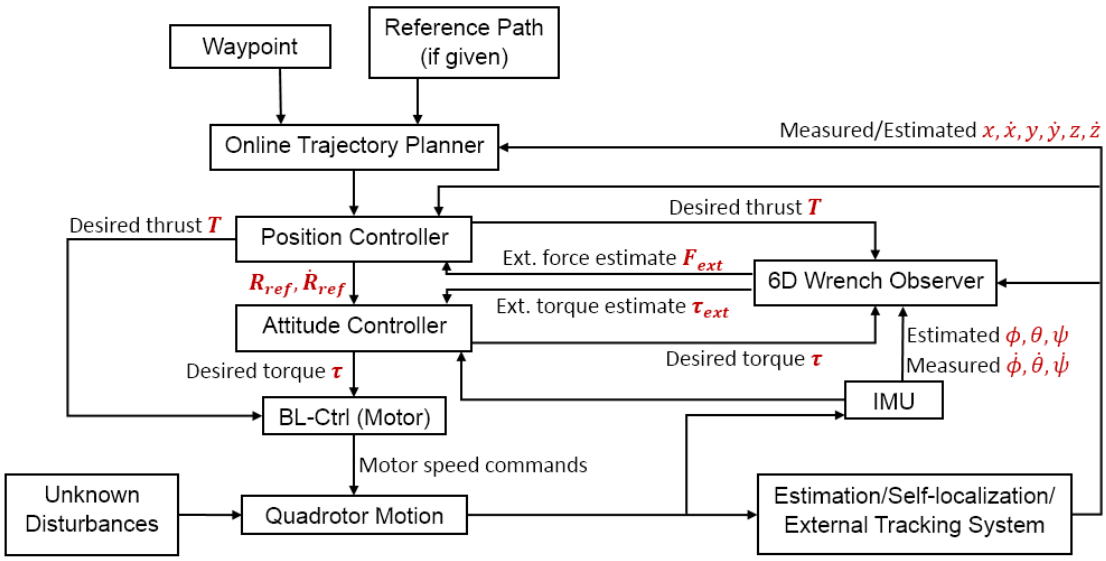


Figure 3.2: The control structure of the proposed RobOR+ method.

3.4 Disturbance Observation

In this section, we extend the backstepping-like regulator discussed in Sec. 3.2 with the inclusion of a nonlinear force/torque external wrench observer using the Fault Detection and Isolation (FDI) method which has been used earlier in manipulator arms Takakura *et al.* (1989); De Luca and Mattone (2003). The observer estimates all the system offsets, parameter uncertainties and external disturbances, and passes the estimate to the controller for compensation.

The modified complete control-observation system, as our proposed RobOR+ method, can be illustrated in Fig. 3.2.

Denoting the external forces acting on the quadrotor in the inertial frame with $b_F \in \mathbb{R}^3$ and the external torque acting on the quadrotor in the body frame with $b_\tau \in \mathbb{R}^3$, the external disturbance Λ_{ext} in (2.1) can be defined as

$$\Lambda_{\text{ext}} = \begin{bmatrix} b_F \\ b_\tau \end{bmatrix}, \tag{3.35}$$

The external disturbance may be variable (e.g. wind) or permanent (e.g. due to uncertainty in the dynamical parameters). Therefore, the FDI technique needs to generate an asymptotically stable residual vector signal, to make sure that the disturbance recovery is occuring and that each scalar value of the residual is decoupled from each other. The FDI design is based on the simple but powerful idea of the generalized momenta $Q = M\zeta$. In fact, one can write the following first-order dynamic equation for the momentum as

$$\dot{Q} = \Lambda + \Lambda_{\text{ext}} + C^\top(\zeta)\zeta - G, \tag{3.36}$$

which is obtained from (2.1) and the knowledge that the coriolis term $C^{\top}(\zeta)$ involves the partial differentiation Khalil and Dombre (2004) of the mass matrix M w.r.t. to the system state ζ in (2.5).

Let the residual vector $\boldsymbol{r} \in \mathbb{R}^6$ for the disturbance estimation of the quadrotor be defined as De Luca *et al.* (2006)

$$\boldsymbol{r}(t) = K_I\left(Q - \int_0^t \left(\Lambda + C^{\top}(\zeta)\zeta - G + \boldsymbol{r}\right) ds\right), \tag{3.37}$$

where $Q = M\zeta$ is the momentum of the quadrotor and $K_I > 0$ is the diagonal gain matrix. The dynamic evolution of residual $\boldsymbol{r}$ satisfies

$$\dot{\boldsymbol{r}} = K_I\left(\Lambda_{\text{ext}} - \boldsymbol{r}\right), \quad \text{when } \boldsymbol{r}(0) = 0, \tag{3.38}$$

which has an exponentially stable equilibrium at $\boldsymbol{r} = \Lambda_{\text{ext}}$. For "sufficiently" large gains the dynamic residual in (3.38) becomes

$$\boldsymbol{r} \simeq \Lambda_{\text{ext}}. \tag{3.39}$$

This approach provides a model-based estimate of the external torque Λ_{ext} resulting from the force/torque disturbances that is acting on the quadrotor.

The residual vector gives deeper knowledge about the disturbance components that are affecting the quadrotor. If a particular component of external disturbance is zero then the scalar residual value corresponding to that component converges to zero.

Hence we can write the estimated external wrench as

$$\widehat{\Lambda_{\text{ext}}} = \begin{bmatrix} \widehat{b_F} \\ \widehat{b_\tau} \end{bmatrix} = \boldsymbol{r}, \tag{3.40}$$

where $\wedge$ indicates the estimated value.

When no external disturbance is acting on the quadrotor, the whole residual vector $\boldsymbol{r}$ is practically zero. In presence of Γ_{ext}, one or more residuals rise above the threshold corresponding to the disturbance. In particular, the larger the value of $K_I > 0$ is, the faster and more accurately the residual in (3.37) will converge to the external force/torque disturbance. On the other hand, too large values of K_I will result in noisy estimates. Hence, the gain matrix K_I must be tuned taking into account those two aspects.

With the inclusion of the external wrench estimate, the position control law in (3.7) is modified as

$$\begin{aligned} RF \triangleq & -\widehat{b}_F + mge_3 - m\nabla^2 P_1(x - x_{\text{ref}})(\dot{x} - \dot{x}_{\text{ref}}) \\ & + m\ddot{x}_{\text{ref}} - mKe_v - m\nabla P_1(x - x_{\text{ref}}), \end{aligned} \tag{3.41}$$

while the attitude control law in (3.21) becomes

$$\begin{aligned}\tau = &-\widehat{b}_\tau + JQ^{-1}\left(-K_\omega e_\omega - \frac{K_{\text{rot}}}{2}\left[\dot{R}_{\text{ref}}^\top R + R_{\text{ref}}^\top \dot{R} - \dot{R}^\top R_{\text{ref}} - R^\top \dot{R}_{\text{ref}}\right]\right.\\ &\left.-\ddot{R}_{\text{ref}}^\top R_{\text{ref}} - \dot{R}_{\text{ref}}^\top \dot{R}_{\text{ref}} - 2R_{\text{ref}}^\top R\right) + \omega \times J\omega.\end{aligned} \tag{3.42}$$

3.4.1 Control-Observation Stability

In this section, we consider a more practical environment without full knowledge of the disturbance generated from the unstructured dynamics and exogenous system. Instead, we employ the disturbance estimate proposed in the previous section into the position and attitude control law. This indicates that there exist the error between external force disturbance and its estimate $e_{b_F} = b_F - \widehat{b}_F$, as well as the error between external torque disturbance and its estimate $e_{b_\tau} = b_\tau - \widehat{b}_\tau$, respectively.

For the sake of completeness, we hereby prove the asymptotic stability of the whole control-observation system that gets rid of the full disturbance knowledge assumption ($\widehat{b}_F = b_F, \widehat{b}_\tau = b_\tau$).

We first prove the asymptotic stability of the position control algorithm. Still by sending RF to F', but replacing b_F with $\widehat{b}_F$ (applying the control law in (3.41)), the closed-loop dynamics is now given by

$$\begin{aligned}\dot{e}_v =& \frac{1}{m}(b_F - \widehat{b}_F) - Ke_v - \nabla P_1(x - x_{\text{ref}})\\ =& -Ke_v - \nabla P_1(x - x_{\text{ref}}) + \frac{1}{m}e_{b_F}.\end{aligned} \tag{3.43}$$

The (directional) derivative of the Lyapunov function candidate (3.6) now turns

$$\dot{V}_{\text{pos}} = -\nabla P_1(x - x_{\text{ref}})^\top \nabla P_1(x - x_{\text{ref}}) - Ke_v^\top e_v + \frac{1}{m}e_v^\top e_{b_F}, \tag{3.44}$$

which is no longer negative definite for arbitrary disturbance error.

We therefore choose a new Lyapunov function candidate (3.45) extended with a term of disturbance error

$$V_{\text{pos,dis}}(e_v, x - x_{\text{ref}}) = V_{\text{pos}}(e_v, x - x_{\text{ref}}) + \frac{1}{2m}e_{b_F}^\top e_{b_F}, \tag{3.45}$$

Since the force estimate satisfies (3.38), we can obtain the relationship between the derivative of force estimate and the error of force estimate $\dot{\widehat{b}}_F = K_I(b_F - \widehat{b}_F) =: K_I e_{b_F}$.

The derivative of (3.45) is thus given by

$$\begin{aligned}\dot{V}_{\text{pos,dis}} &= \dot{V}_{\text{pos}} + \frac{1}{2m}\left(e_{b_F}^{\top}\dot{e}_{b_F}\right)\\ &= \dot{V}_{\text{pos}} + \frac{1}{m}e_{b_F}^{\top}\dot{e}_{b_F}\\ &= -\nabla P_1(x-x_{\text{ref}})^{\top}\nabla P_1(x-x_{\text{ref}}) - Ke_v^{\top}e_v + \frac{K_I^{-1}}{m}e_v^{\top}\dot{\widehat{b}}_F + \frac{K_I^{-1}}{m}\dot{\widehat{b}}_F^{\top}\dot{e}_{b_F}\\ &= -\nabla P_1^{\top}\nabla P_1 - Ke_v^{\top}e_v + \frac{K_I^{-1}}{m}(e_v\cdot\dot{\widehat{b}}_F + \dot{\widehat{b}}_F\cdot\dot{e}_{b_F}).\end{aligned} \tag{3.46}$$

We here write

$$\begin{aligned}\dot{V}_{\text{pos,dis}} &= -\|\nabla P_1\|^2 - K\|e_v\|^2 + \frac{K_I^{-1}}{m}\left((e_v+\dot{e}_{b_F})\cdot\dot{\widehat{b}}_F\right)\\ &= -(1-\theta_{\text{pos}})\left(\|\nabla P_1\|^2 + K\|e_v\|^2\right)\\ &\quad -\theta_{\text{pos}}\left(\|\nabla P_1\|^2 + K\|e_v\|^2\right) + \frac{K_I^{-1}}{m}\left((e_v+\dot{e}_{b_F})\cdot\dot{\widehat{b}}_F\right),\end{aligned} \tag{3.47}$$

where $\theta_{\text{pos}} \in (0,1)$.

Since $-(1-\theta_{\text{pos}})\left(\|\nabla P_1\|^2 + K\|e_v\|^2\right) < 0$, as long as we can find a condition that suffices to

$$\left|\frac{K_I^{-1}}{m}\left((e_v+\dot{e}_{b_F})\cdot\dot{\widehat{b}}_F\right)\right| < \theta_{\text{pos}}\left(\|\nabla P_1\|^2 + K\|e_v\|^2\right), \tag{3.48}$$

we can prove that (3.46) is negative definite.

According to the Cauchy-Schwarz inequality and absolute homogeneity, (3.48) will certainly be the case if

$$\frac{K_I^{-1}}{m}\|e_v+\dot{e}_{b_F}\|\left\|\dot{\widehat{b}}_F\right\| < \theta_{\text{pos}}\left(\|\nabla P_1\|^2 + K\|e_v\|^2\right), \tag{3.49}$$

which holds, by the triangle inequality, whenever

$$\begin{aligned}&\frac{K_I^{-1}}{m}\left(\|e_v\| + \|\dot{e}_{b_F}\|\right)\left\|\dot{\widehat{b}}_F\right\| = \frac{1}{m}\left(\|e_v\| + \|\dot{e}_{b_F}\|\right)\|e_{b_F}\|\\ &< \theta_{\text{pos}}\left(\|\nabla P_1\|^2 + K\|e_v\|^2\right).\end{aligned} \tag{3.50}$$

Therefore, as long as e_v, e_{b_F} and $\dot{e}_{b_F}$ are bounded, we can always make K_0 and K large enough to have the inequality (3.50), and hence (3.48), satisfied on any compact set. So far, we have proved that the position control algorithm together with force estimation is asymptotically stable when K_0 and K are chosen so as to guarantee the overestimate (3.50).

Analogous to the derivations for the position controller, we prove here the asymptotic stability of the attitude controller without the full torque disturbance knowledge assumption.

Similarly as the force disturbance observation, the torque estimate suffices to (3.38), we have $\dot{\widehat{b}}_\tau = K_I(b_\tau - \widehat{b}_\tau) =: K_I e_{b_\tau}$. By sending τ in attitude control law (3.42) to (3.1d), the closed-loop rotational dynamics becomes

$$\dot{e}_\omega = -K_\omega e_\omega - 2R_{\text{ref}}^\top R + Q(J^{-1}e_{b_\tau}). \tag{3.51}$$

Hence the derivative of the Lyapunov function candidate (3.20) is given by

$$\begin{aligned}\dot{V}_{\text{att}}(E,e_\omega) &= Q^{-1}(e_\omega)^\top Q^{-1}(\dot{e}_\omega) + \operatorname{grad} P_2(E)\cdot\dot{E} \\ &= -K_\omega Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega) + \frac{K_{\text{rot}}}{4}\operatorname{tr}((E-E^\top)^2) + Q^{-1}(e_\omega)^\top J^{-1}e_{b_\tau},\end{aligned} \tag{3.52}$$

which is no longer negative definite for arbitrary disturbance error.

Therefore, we choose a new Lyapunov function candidate (3.53) extended with a term of disturbance error

$$V_{\text{att,dis}}(E,e_\omega) = V_{\text{att}}(E,e_\omega) + \frac{J^{-1}}{2}e_{b_\tau}^\top e_{b_\tau}, \tag{3.53}$$

and its derivative now turns to be

$$\begin{aligned}\dot{V}_{\text{att,dis}} &= Q^{-1}(e_\omega)^\top J^{-1}e_{b_\tau} + J^{-1}e_{b_\tau}^\top \dot{e}_{b_\tau} \\ &\quad - K_\omega Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega) + \frac{K_{\text{rot}}}{4}\operatorname{tr}((E-E^\top)^2).\end{aligned} \tag{3.54}$$

We here write

$$\begin{aligned}\dot{V}_{\text{att,dis}} &= -K_\omega\|Q^{-1}(e_\omega)\|^2 - \frac{K_{\text{rot}}}{2}\|E-E^\top\|^2 \\ &\quad + J^{-1}\left((Q^{-1}(e_\omega)+\dot{e}_{b_\tau})\cdot e_{b_\tau}\right) \\ &= -(1-\theta_{\text{att}})(K_\omega\|Q^{-1}(e_\omega)\|^2 + \frac{K_{\text{rot}}}{2}\|E-E^\top\|^2) \\ &\quad - \theta_{\text{att}}(K_\omega\|Q^{-1}(e_\omega)\|^2 + \frac{K_{\text{rot}}}{2}\|E-E^\top\|^2) \\ &\quad + J^{-1}\left((Q^{-1}(e_\omega)+\dot{e}_{b_\tau})\cdot e_{b_\tau}\right),\end{aligned} \tag{3.55}$$

where $\theta_{\text{att}} \in (0,1)$.

Considering that $-(1-\theta_{\text{att}})(K_\omega\|Q^{-1}(e_\omega)\|^2 + \frac{K_{\text{rot}}}{2}\|E-E^\top\|^2) < 0$, we can prove

that (3.54) is negative definite, if we can find a condition under which

$$\left|J^{-1}\left((Q^{-1}(e_\omega)+\dot{e}_{b_\tau})\cdot e_{b_\tau}\right)\right| < \theta_{\text{att}}\left(K_\omega\|Q^{-1}(e_\omega)\|^2+\frac{K_{\text{rot}}}{2}\|E-E^\top\|^2\right). \tag{3.56}$$

By the Cauchy-Schwarz inequality and absolute homogeneity, (3.56) will hold if

$$J^{-1}\|Q^{-1}(e_\omega)+\dot{e}_{b_\tau}\|\,\|e_{b_\tau}\| < \theta_{\text{att}}\left(K_\omega\|Q^{-1}(e_\omega)\|^2+\frac{K_{\text{rot}}}{2}\|E-E^\top\|^2\right). \tag{3.57}$$

While by the triangle inequality, (3.57) holds whenever

$$J^{-1}\left(\|Q^{-1}(e_\omega)\|+\|\dot{e}_{b_\tau}\|\right)\|e_{b_\tau}\| < \theta_{\text{att}}\left(K_\omega\|Q^{-1}(e_\omega)\|^2+\frac{K_{\text{rot}}}{2}\|E-E^\top\|^2\right). \tag{3.58}$$

Hence, as long as e_ω, e_{b_τ} and $\dot{e}_{b_\tau}$ are bounded, we can always make K_{rot} and K_ω sufficiently large to have the inequality (3.58), and therefore (3.56) satisfied on any compact set. At this stage, we have proved that the proposed attitude control algorithm together with torque estimation, provided that K_{rot} and K_ω are chosen sufficiently large for guaranteeing (3.58), is stable.

So far, we have proved the asymptotic stability of the complete control-observation system for sufficiently large control gains. We further experimentally validated the accuracy and convergence of the nonlinear disturbance observer, the details of which will be introduced in Sec. 3.6.2.

3.5 Trajectory Planning

In this section, we further propose an online trajectory planning algorithm based on an MPC method. This planner is used in both RobOR and RobOR+ methods. The online planner is able to subscribe both a single waypoint and an array of prescribed path, and then generate a trajectory prediction for the quadrotor via solving a 3D Optimal Control Problem (OCP) at each step. The planner sets the nominal jerk $\mathbf{u}$ (the 3rd order derivative of position) as the input vector for a coupled OCP describing the translational motion of the quadrotor. The discrete time state ξ of the OCP consists of the position, velocity and linear acceleration along the three axes in the inertial frame. In general, the discrete time equations of state at the k^{th} step with the sampling time Δt are given by

$$\mathbf{u}[k] = \dddot{\mathbf{x}}[k] \tag{3.59a}$$

$$\boldsymbol{\xi}[k] = [x[k]\ \dot{x}[k]\ \ddot{x}[k]\ y[k]\ \dot{y}[k]\ \ddot{y}[k]\ z[k]\ \dot{z}[k]\ \ddot{z}[k]]^\top \tag{3.59b}$$

$$\boldsymbol{\xi}[k+1] = A\boldsymbol{\xi}[k]+B\mathbf{u}[k] \tag{3.59c}$$

$$A_{(9\times9)} = \begin{bmatrix} A_b & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & A_b & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & A_b \end{bmatrix}, \tag{3.59d}$$

$$\text{where } A_b = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.59e}$$

$$B_{(9\times3)} = \begin{bmatrix} B_b & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & B_b & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & B_b \end{bmatrix}, \tag{3.59f}$$

$$\text{where } B_b = [\frac{1}{6}\Delta t^3 \ \frac{1}{2}\Delta t^2 \ \Delta t]^\top. \tag{3.59g}$$

The OCP is with a convex quadratic cost function

$$\begin{aligned} \min J_{\text{OCP}} = \sum_{k=0}^{N} \Big(& \mathbf{u}[k]^\top P \mathbf{u}[k] \\ & + (\boldsymbol{\xi}[k] - \boldsymbol{\xi}_{\text{ref}}[k])^\top L_{\text{s}} (\boldsymbol{\xi}[k] - \boldsymbol{\xi}_{\text{ref}}[k]) \Big) \\ & + (\boldsymbol{\xi}[N+1] - \boldsymbol{\xi}_{\text{ref}}[N+1])^\top L_{\text{t}} (\boldsymbol{\xi}[N+1] - \boldsymbol{\xi}_{\text{ref}}[N+1]), \end{aligned} \tag{3.60}$$

subject to the dynamics explained above and the state and input constraints

$$\mathbf{u}_{\min} \leq \dddot{\mathbf{x}}[k] \leq \mathbf{u}_{\max}, \tag{3.61a}$$

$$\boldsymbol{\xi}_{\min} \leq \boldsymbol{\xi}[k] \leq \boldsymbol{\xi}_{\max}, \tag{3.61b}$$

where N represents the receding horizon, and P, L_{s} and L_{t} are the weight matrices of input cost, stage state cost and terminal state cost, respectively. The described OCP is solved via CVXGen Mattingley *et al.* (2011) using the interior point method. We have also attempted to decouple the 3D OCP into three independent 1D OCPs via a similar linearization approach. Both configurations are feasible for the quadrotors without big difference on performance.

The online planner is with an adaptive cost weight setting. For a waypoint navigation task in which only one waypoint reference, instead of a group of references, is passed into the planner, the weight of the terminal state cost is automatically set dominant, while the weight off all states in the stage cost is set to zero. In contrast, for a path following task, the weight of the terminal cost is set to zero. Differently from a normal MPC approach where the first step control input $\mathbf{u}[0]$ is used as the input for the system, the first step predicted state (as $\boldsymbol{\xi}[1]$) as well as $\mathbf{u}[0]$ are passed into the controller introduced in Sec. 3.2.1. The nominal jerk $\mathbf{u}[0]$ is utilized in (3.22) as $\dddot{\mathbf{x}}_{\text{ref}}$. In case that no feasible locally optimal solution is available, the planner automatically loads the last previous feasible solution to avoid an "unstable" trajectory.

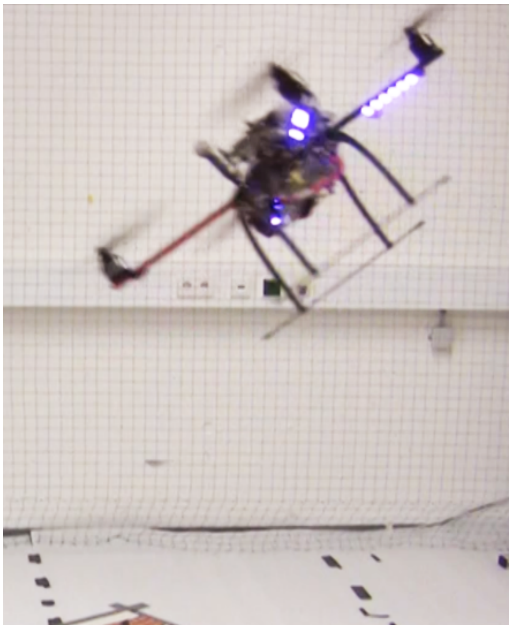


Figure 3.3: "Large Tilt Maneuvers" experiment. The quadrotor is tracking a tilted eight-shape trajectory with a maximum tilt of over 40°.

3.6 Verification

In order to validate both RobOR and RobOR+ approaches introduced herein, we have step-by-step carried out a series of experiments via numerical tests in MATLAB, physical simulations in ROS/Gazebo and finally multiple trials with real implementations. However, for the sake of brevity, we report here only the results of the actual experiments. Initially, we have validated the force/torque residual based wrench estimator with the quadrotor in hovering mode subject to external forces and torques.

The performance of our proposed control method has then been assessed via a waypoint navigation test, a large tilt maneuver test (Fig. 3.3) and a hovering with wind gusts test (Fig. 3.4), through the actual experiments in our lab. The experimental results have been compared with the results obtained using the geometric tracking control (GeoTrack) algorithm in Lee *et al.* (2010b). The GeoTrack method (as we introduced in Sec. 2.2.3) is chosen because it has been verified to be ideal for aggressive quadrotor maneuvers and is widely used in the robotics community.

Other conventional control approach, such as PID (in Sec. 2.2.2), has also been tested initially. However the linear PID controller is with poor performance for aggressive quadrotor maneuvers with large tilt or high acceleration. Therefore, we do not report the results obtained using PID controller.

We carried out each set of experiments with multiple trials using all three control approaches (RobOR+, RobOR, GeoTrack) so that the influence on the results due to

Figure 3.4: "Hovering with wind gusts" experiment. The quadrotor with the proposed robust output regulation approach plus the nonlinear observer (RobOR+) is hovering with the wind gusts generated by the ventilation system and two 40 W electric fans.

random factors could be largely ruled out.

3.6.1 Experimental Configuration

Although we have introduced the hardware platform in 2.3, we here briefly recap the technical details for the sake of brevity.

A middle-size quadrotor UAV (our QuadroXL) with a capability of up to 1.0 kg payload is set as the platform for a series of our experiments. The quadrotor is equipped with an Odroid-XU4 board with Cortex^{TM} *A*15 quad-core CPUs and all the computations are carried out onboard.

We have implemented our proposed control approach within the ROS-based TeleKyb framework Grabe *et al.* (2013), which provides an interface with the brushless motor controllers on the quadrotor, so that the desired motor speeds regulated via the given control approaches can be executed. An external motion capture system is employed to record the position and the orientation of the quadrotor. The linear velocity is then estimated from the position. These data, plus the other essential data collected via a

low-cost on-board Inertial Measurement Unit (IMU), i.e. the linear acceleration and the angular velocity, are passed into the on-board computational unit at a sampling frequency of 120 Hz over a wireless channel. The on-board computer filters the collected data, performs all the computation of the control algorithms and generates the regulated motor commands for the quadrotor at 100 Hz. The 6D wrench observer is also run at the frequency of 100 Hz. Meanwhile, the online MPC-based trajectory planner is run at 20 Hz with a receding horizon of 1 s due to its computational time consumption of 24 ms on average. A faster frequency of the trajectory planner up to 35 Hz is feasible, however, a shorter sampling time would lead to more aggressive trajectories for the quadrotor. In our tests, we conservatively keep the trajectory planner at a sampling time of 0.05 s.

The MAV is equipped with a 5000 mAh LiPo battery, which allows a 15 min intensive flight test without recharge. The overall weight of the quadrotor platform, including a battery, an onboard computer, communication devices and all sensors except cameras, reaches 1.605 kg. However, the nominal mass is manually set to 1.45 kg in the controller throughout the experiments to intentionally generate a mass mismatch. The inertial coefficients are estimated via a simple CAD model, without further advanced system identification approaches. In addition, the mass distribution is centered away from the actual center of the quadrotor. These parametric mismatches lead to the steady hovering state (with no use of robustness designs) at a tilt of approximately 3° on pitch and -1° on roll in air. The motors on the MAV are controlled in an open-loop fashion, which is problematic when the LiPo battery turns low because the actual currents sent to the motors would be lower than the required. After multiple flight tests, the actual rotation of the motors would become slower than the given motor commands. This disturbance due to the hardware affects the control performance to a large extent (we will discuss it later). The lab where we have carried out our experiments is equipped with a ventilation system in the ceiling. The constant winds from the vents and the airflow generated by the propellers become the external disturbances acting on the quadrotor. All these parametric mismatches and disturbances, plus the other unknown internal and external disturbances, e.g. wind gusts from electrical fans, are added in order to check the robustness properties of our proposed approach.

During the validation of the external wrench observer, we used known suspended weights which in turn produce known forces or torques. These weights were transferred to the desired axes by means of a rope-pulley setup. Frictionless pulleys were used so that the losses due to friction are negligible.

During the validation of the whole control framework, we chose relatively conservative control gain settings on all three controllers for the purpose of more stable aggressive maneuvers throughout the experiments. With the RobOR and geometric tracking controller, a more aggressive gain setting can reduce the translational offsets on hovering but may lead to a risk of instability when executing aggressive waypoint navigation and path following tasks.

3.6.2 Experimental Results

Experiment 1: ***Validation of Wrench Observer***

During this experiment, we validated the 6D wrench observer by subjecting the quadrotor to known external forces and torques at its CoM while it was in hovering. This series of experiments were operated by the coauthor of Liu *et al.* (2017), Sujit Rajappa.

The forces were applied in steps, to test different forces acting on the quadrotor. As shown in Fig. 3.5, at the time instants 35 s, 46 s and 57 s respectively forces of 0.85 N, 2.17 N and 4.35 N were applied along the Z axis by adding weights of 87 g, 221 g and 443 g. The estimated external force along the Z axis, shown in blue, is correctly estimated by the observer.

At time 69 s a weight of 222 g acting along the Z axis was removed and then at times 97 s, 107 s and 119 s respectively, forces of 0.22 N, 1.19 N and 2.53 N were added along a vector laying on the XY-plane and having an azimuth of -60°. These forces were successively removed in the same order in which they were added. As shown in Fig. 3.5 the external forces F_{ext} were correctly estimated for F_{ext_x} (red), F_{ext_y} (green) and F_{ext_z} (blue).

Like the force estimation validated above, a similar experiment was conducted separately for the torque estimation with the same setup. In the first torque experiment the weights were suspended at the end of the arm of the quadrotor along the $-Y$ axis. This creates are torque w.r.t. to X axis, namely τ_{ext_x}. During the experiment, a weight of 27 g was suspended at a distance of 37.5 cm from the quadrotor center in the $-Y$ axis. As shown in Fig. 3.6, a torque of 0.099 Nm was applied. The Figure also shows the estimated torque τ_{ext_x} (red). A small constant τ_{ext_y} (green) is also estimated during the whole experiment, and it is due to a non perfect balance of the weights on the quadrotor.

In the second torque experiment, a force was applied through our rope-pulleys setup in order to create a torque τ_{ext_z} around the Z axis. Similarly to the first experiment, a weight of 27 g was suspended at a distance of 37.5 cm from the quadrotor center. As we show in Fig. 3.7, τ_{ext_z} of 0.099 Nm was correctly estimated (blue in Fig. 3.7). Overall, these experiments prove the effectiveness of the external force/torque estimator.

Experiment 2: ***Waypoint Navigation***

After the validation of the wrench observer, we first evaluated the performance of our two proposed approaches by commanding the quadrotor to achieve a navigation task by given a waypoint 2.5 m far from the quadrotor. We started the first test with a full battery and repeatedly carried out 13 trials until the voltage of the onboard LiPo battery turned very low. For a fair comparison, we also performed 13 trials for the same task using each of the other two control algorithms. In each trial, the quadrotor took off and kept hovering at a height of 1 m. It was then commanded to repeatedly fly forward between the waypoints $(-0.65,-1,1)^\top$ and $(-0.65,1.5,1)^\top$ every 5 s for four times, and was finally commanded to land.

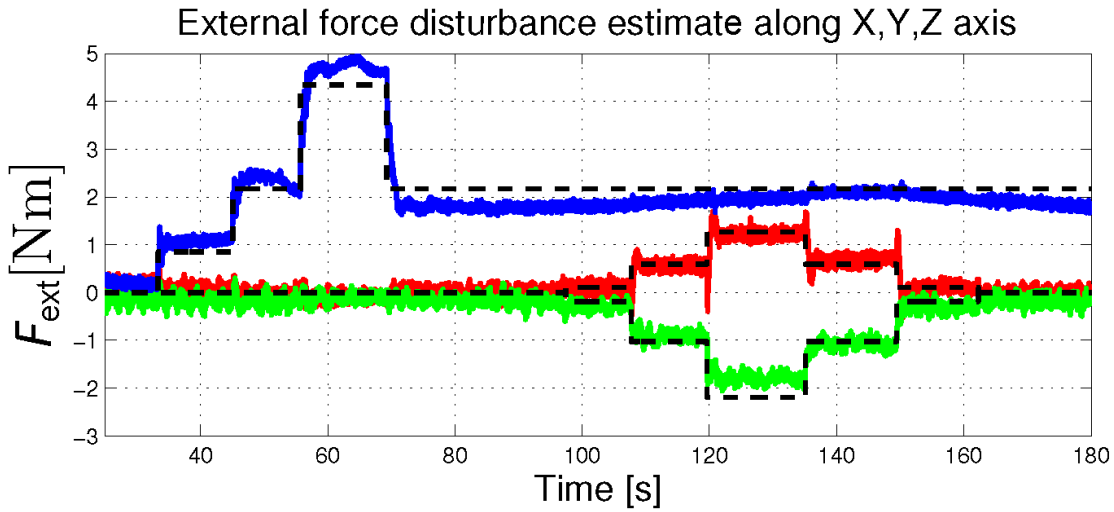


Figure 3.5: Results of the experiment with the external force disturbance along X, Y and Z axes. Applied disturbance (black dashed lines), estimated disturbance F_{ext_x} along X (red solid line), estimated disturbance F_{ext_y} along Y (green solid line) and estimated disturbance F_{ext_z} along Z (blue solid line).

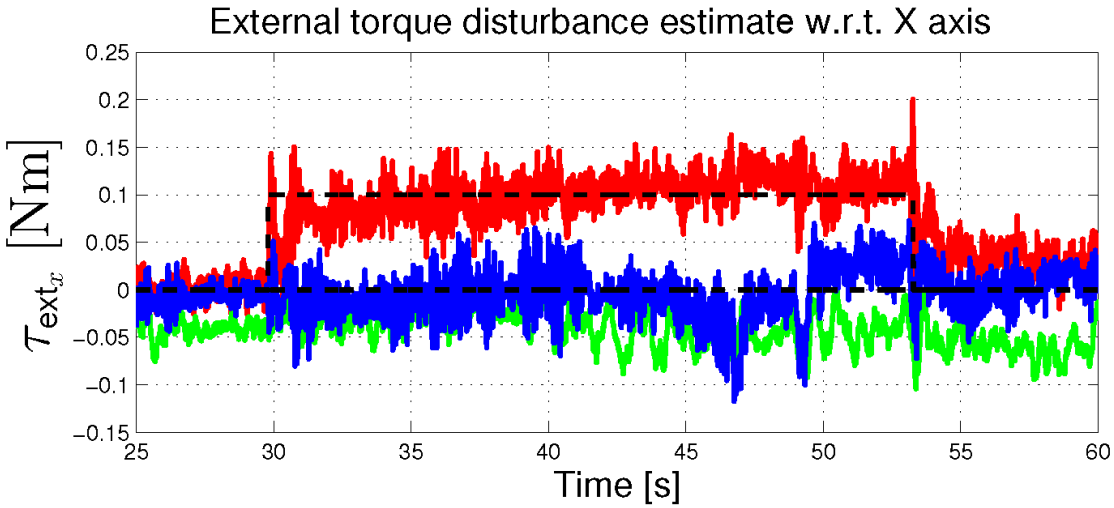


Figure 3.6: Results of the experiment with the external torque disturbance around X. Applied disturbance (black dashed lines), estimated disturbance τ_{ext_x} (red solid line), estimated disturbance τ_{ext_y} (green solid line) and estimated disturbance τ_{ext_z} (blue solid line).

Since directly sending the waypoint which is far from the current position to the controller would lead to an unstable flight, we employed the online trajectory planner introduced in Sec. 3.5 to balance the aggressiveness and stability. The aggressiveness of

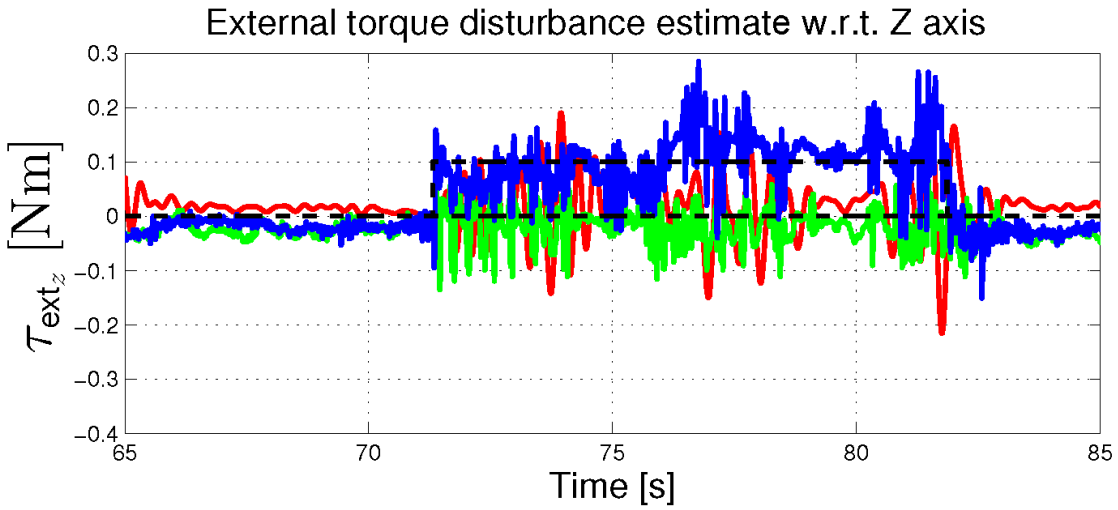


Figure 3.7: Results of the experiment with the external torque disturbance around Z. Applied disturbance (black dashed lines), estimated disturbance τ_{ext_x} (red solid line), estimated disturbance τ_{ext_y} (green solid line) and estimated disturbance τ_{ext_z} (blue solid line).

the trajectory can be modified via changing the weight matrices within the cost function. In our first experiment, we set the parameters so that the maximum horizontal acceleration reached 25.23 m/s^2 and the maximum magnitude of the quadrotor velocity along the y direction arrived at approximately 3 m/s. For all three control approaches, it is also possible to track an offline, prescribed trajectory instead of using our proposed online MPC-based planner. However, tracking the reference generated from the online trajectory planner outperformed the direct tracking of an offline trajectory in most instances during our experiments. We, therefore, kept the online trajectory planner active for all the trials.

In order to demonstrate the hardware disturbances due to low battery levels more intuitively, we show the translational trajectories of the quadrotor on each axis in the 1st and 13th trial, respectively, in Fig. 3.8 and in Fig. 3.9. The solid green line represents the quadrotor trajectory with the geometric tracking approach (GeoTrack). The solid blue line represents the quadrotor trajectory with the robust output regulation approach (RobOR). The solid red line represents the quadrotor trajectory with our proposed robust output regulation approach plus the nonlinear observer (RobOR+).

Due to the parametric mismatches as well as unknown internal and external disturbances, there existed a translational offset up to 40 cm with the GeoTrack method at all times, whether the quadrotor was at full or low battery. Meanwhile, with a fully charged onboard LiPo, the RobOR method to a large extent reduced the translational errors by compensating for the offsets due to the external disturbances and model mismatches. However, when comparing the results of the two trials, we can read out that the control

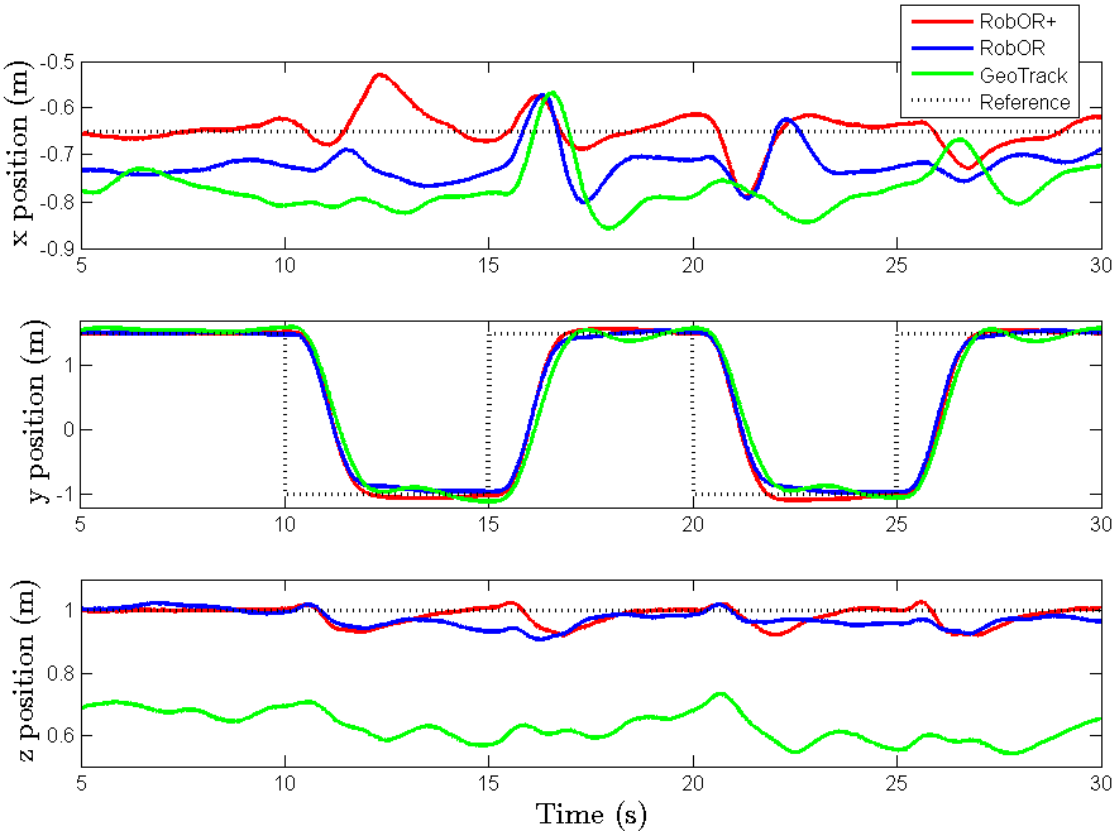


Figure 3.8: Resulting quadrotor response for the 1st trial of the repeated waypoint navigation task. The reference position is switched from $(-0.65, -1, 1)^\top$ to $(-0.65, 1.5, 1)^\top$ every 5 s. The solid red line represents the quadrotor trajectory with our proposed robust output regulation approach plus the nonlinear observer (RobOR+). The solid blue line represents the quadrotor trajectory with the robust output regulation approach (RobOR). The solid green line represents the quadrotor trajectory with the geometric tracking approach (GeoTrack). The dotted black line represents the waypoint reference.

performance with RobOR method dropped when the battery turned low. Meanwhile, the use of the 6D nonlinear observer helped the RobOR+ method to improve the robustness against the unknown disturbances, such as that due to low battery.

We compare the control performance of all three control approaches by calculating the resulting Root-Mean-Square Error (RMSE) between the quadrotor position and the waypoint reference from 1 s after the new waypoint is given to the controller until the next waypoint is subscribed in every trial, as is shown in Fig. 3.10. Among the 13 trials using the RobOR+ method, the RMSE is in the interval of $[4.730\,\text{cm}, 5.838\,\text{cm}]$, with the average RMSE equal to 5.127 cm. Meanwhile, due to the influences of low battery and other disturbances, the maximum RMSE with the RobOR method reaches 19.078 cm, although the minimum RMSE, occurring in the 2nd trial, is 8.327 cm. The

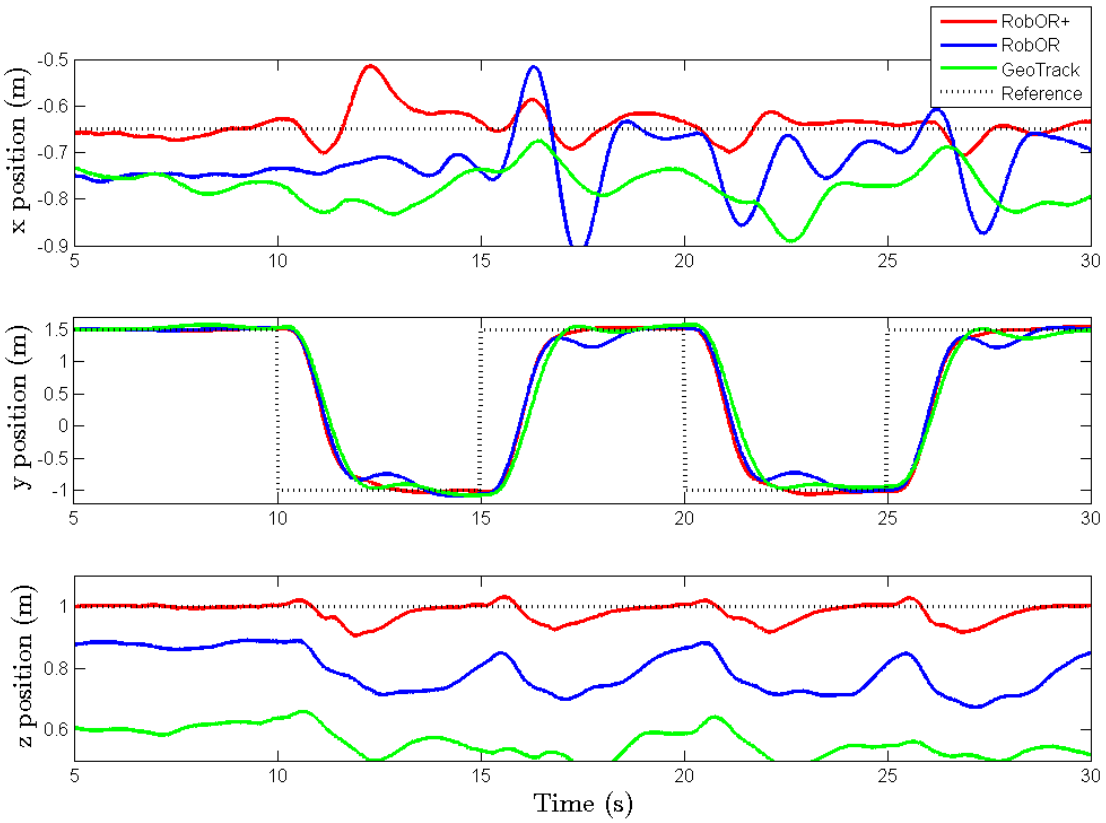


Figure 3.9: Resulting quadrotor response for the 13th trial of the repeated waypoint navigation task. The reference position is switched from $(-0.65, -1, 1)^\top$ to $(-0.65, 1.5, 1)^\top$ every 5 s. The solid red line represents the quadrotor trajectory with our proposed robust output regulation approach plus the nonlinear observer (RobOR+). The solid blue line represents the quadrotor trajectory with the robust output regulation approach (RobOR). The solid green line represents the quadrotor trajectory with the geometric tracking approach (GeoTrack). The dotted black line represents the waypoint reference.

average RMSE with the RobOR method equals 13.935 cm. Without any robust design, the average RMSE with the GeoTrack controller among 13 trials reaches 22.162 cm. The minimum RMSE with the GeoTrack method, at 17.453 cm, occurs in the 1st trial, while the 13th trial is with the maximum RMSE at 28.359 cm.

Experiment 3: ***Aggressive Path Following***

In the third experiment, we demonstrated the capability of the controllers to nontrivial large tilt maneuvers (as is shown in Fig. 3.3). More specific, the quadrotor was commanded to hover at 1.6 m and then to track a prescribed tilted eight-shape path repeatedly for 24 s. We evaluated the quadrotor with the same eight-shape path we employed

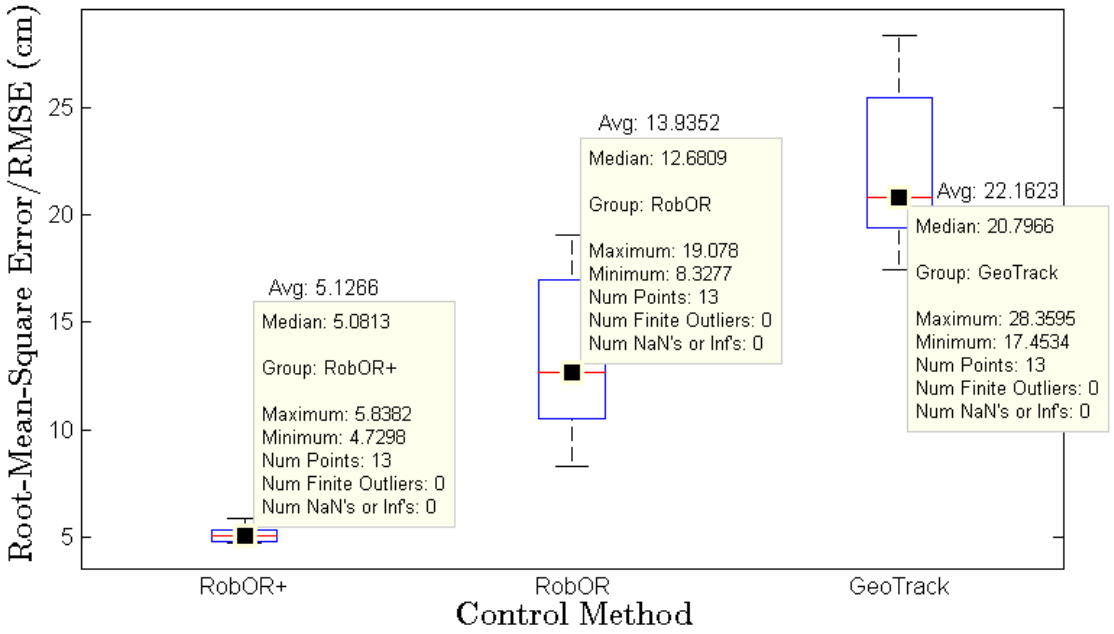


Figure 3.10: Resulting Root-Mean-Square Error (RMSE) on position for 13 trials of the repeated waypoint navigation task with three control algorithms. The left group is with our proposed robust output regulation approach plus the nonlinear observer (RobOR+). The middle group is with the robust output regulation approach (RobOR). The right group is with the geometric tracking approach (GeoTrack).

in Liu *et al.* (2015), which was parameterized as

$$P_d(t) = R_x(\frac{\pi}{6})R_z(\frac{\pi}{12}) \begin{bmatrix} \frac{\sin(\sigma(t))\cos(\sigma(t))}{\sin(\sigma(t))^2+1} \\ \frac{\cos(\sigma(t))}{\sin(\sigma(t))^2+1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1.2 \end{bmatrix}, \tag{3.62}$$

where R_x and R_z denote the rotation matrices along the x and z axis, respectively, while $\sigma(t)$ is a function whose derivative is given by

$$\dot{\sigma}(t) = V\sqrt{\sin^2 t + 1}. \tag{3.63}$$

The prescribed path, as the dotted black line in Fig. 3.11, provides the quadrotor with a tilted eight-shape flight at an average speed of V m/s. In our experiment, we set $V = 1.25$ m/s.

By deriving the position reference in (3.62), we can obtain the analytic solutions of the velocity and acceleration references, so that an offline trajectory can be tracked by the controllers. However, similarly to the first experiment, tracking the trajectory online outperforms tracking the offline trajectory. Therefore, during all the trials (we have 4 trials

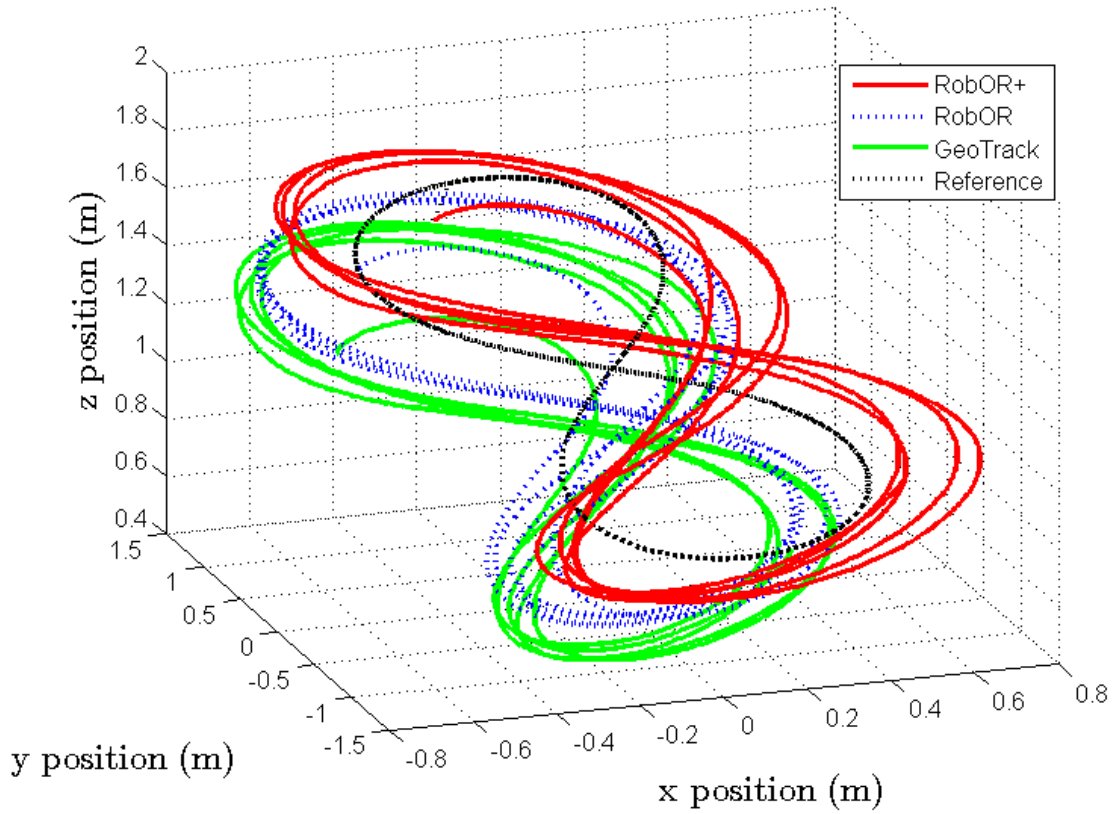


Figure 3.11: Resulting quadrotor trajectories when following a tilted eight-shape path with the use of three control approaches. The solid red line represents the quadrotor trajectory with our proposed robust output regulation approach plus the nonlinear observer (RobOR+). The solid blue line represents the quadrotor trajectory with the robust output regulation approach (RobOR). The solid green line represents the quadrotor trajectory with the geometric tracking approach (GeoTrack). The dotted black line represents the waypoint reference.

with each control approach) in the second experiment, we only passed the position reference (3.62) into the onboard computer and utilized the online planner to automatically generate full required state references at each step.

The maximum tilt of the quadrotor during the flight reaches 40.3°. Fig. 3.12 shows a comparison on the translational error among the three algorithms for the eight-shape path following task. From the results, we can find that the error in z direction, mainly due to the mass mismatch, is largely compensated by the RobOR+ method and partly compensated by the RobOR method. On x, y axes, the errors are relatively close, however, the magnitude of the errors are not small because the quadrotor platform is not light weighed and agile, but with payload and thus larger inertia.

A more intuitive comparison is illusrtated in Fig. 3.13. The average RMSE among

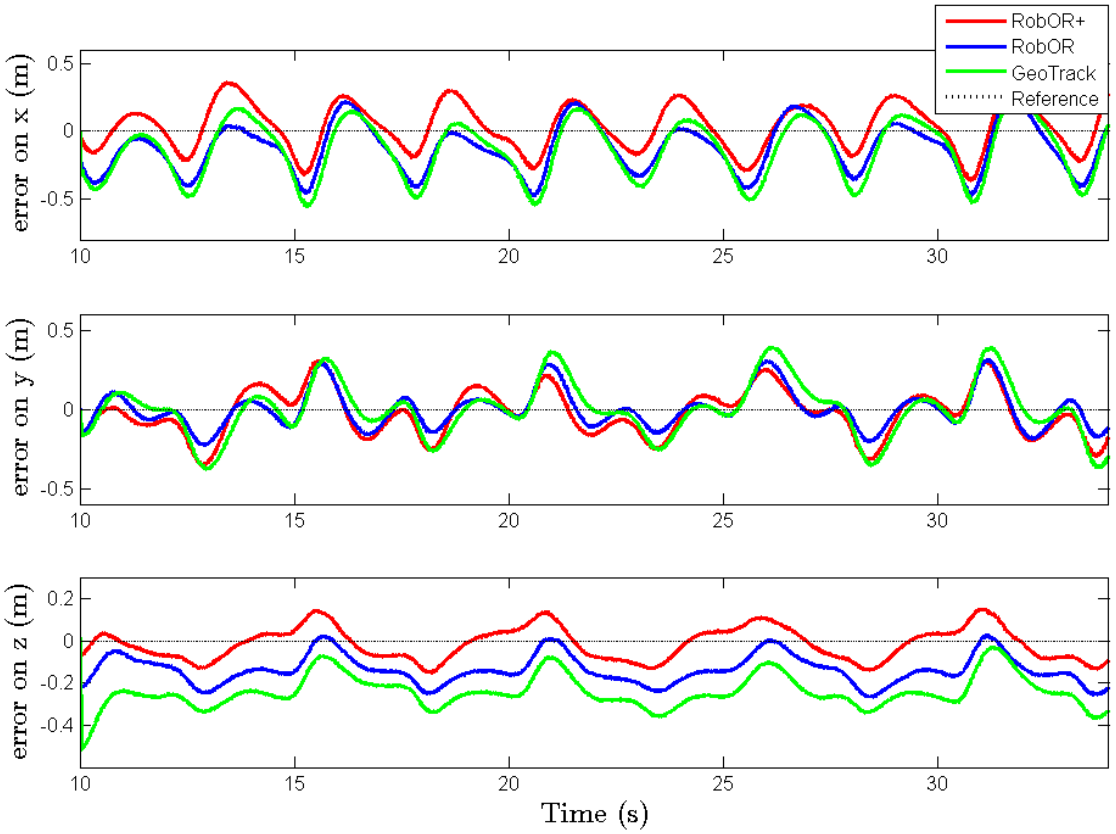


Figure 3.12: Resulting translational error plot for following a tilted eight-shape path. The solid red line represents the quadrotor trajectory with our proposed robust output regulation approach plus the nonlinear observer (RobOR+). The solid blue line represents the quadrotor trajectory with the robust output regulation approach (RobOR). The solid green line represents the quadrotor trajectory with the geometric tracking approach (GeoTrack). The dotted black line represents the zero reference.

the 4 trials with the RobOR+ method is 11.905 cm, while all the values are between $[10.746\text{ cm}, 13.178\text{ cm}]$. The RobOR method is with an average RMSE equal to 13.559 cm, which is close to the RobOR+ method. The GeoTrack method is with the average RMSE reaching 22.873 cm.

Experiment 4: ***Hovering with Wind Gusts***

In the last experiment, we validated the robustness of our proposed RobOR+ method by commanding the quadrotor hovering in an environment with strong wind gusts. Apart from the constant winds from the vents in the ceiling in our lab, we additionally deployed two 40 W electric fans close to the hovering point, both with a 0.7 m horizontal distance at approximately 1 m height. The quadrotor was commanded to hover at $(0, 0, 1)^\top$ after it

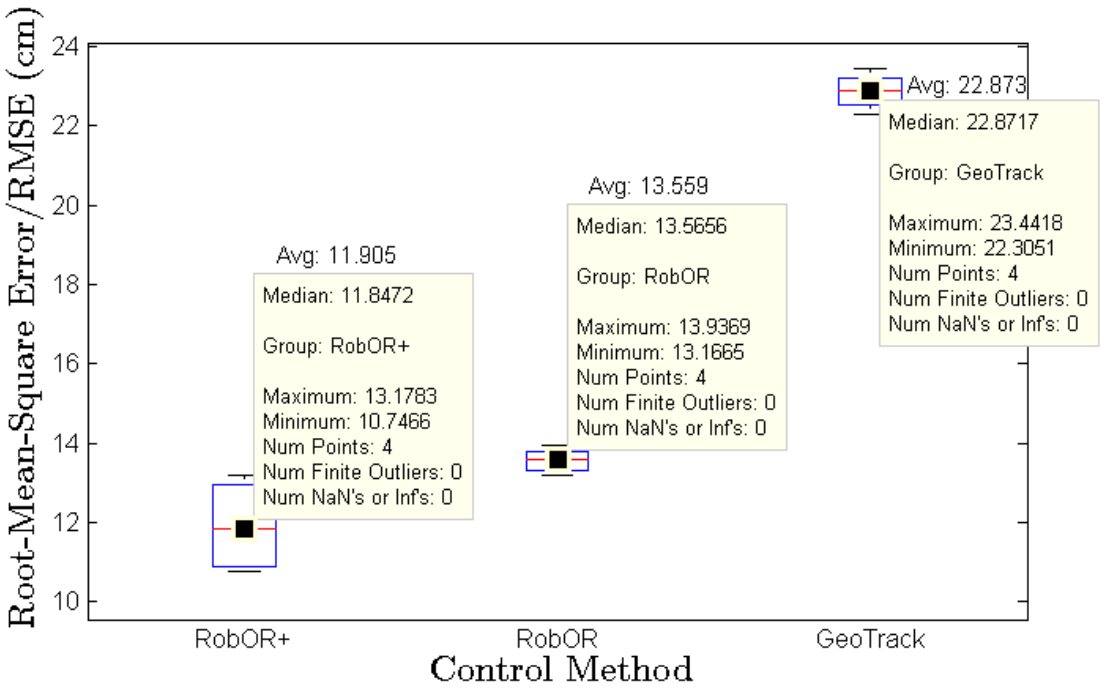


Figure 3.13: Resulting Root-Mean-Square Error (RMSE) on position for 4 trials of the repeated eight-shape path following task with three control algorithms. The left group is with our proposed robust output regulation approach plus the nonlinear observer (RobOR+). The middle group is with the robust output regulation approach (RobOR). The right group is with the geometric tracking approach (GeoTrack).

took off, with both electric fans running at their maximum power (Fig. 3.4). The electric fans provided wind gusts at approximate 2 m/s while the vents provided a 1 m/s wind. In addition, the quadrotor itself with four 10 inch propellers generated a strong wind flow at around 5.5 m/s. The overall wind flow influenced the quadrotor strongly since it was commanded to hover in the space blocked by two boxes that were closely located.

We carried out 6 trials for the wind gust experiment. In each trial, we once turned the two electric fans on and off during the quadrotor hovering instead of keeping the fans on, so that we could evaluate the performance of our proposed method under a sudden wind gust. The maximum RMSE occurs in the 3rd trial, which reaches 3.039 cm, while the minimum RMSE is 1.541 cm in 1st trial. The average RMSE among all 6 trials under wind gusts equals 2.09 cm. The translational errors during the hovering in the 1st and 3rd trial are shown in Fig. 3.14. Our proposed controller demonstrates its robustness against sudden strong wind gusts during the experiment.

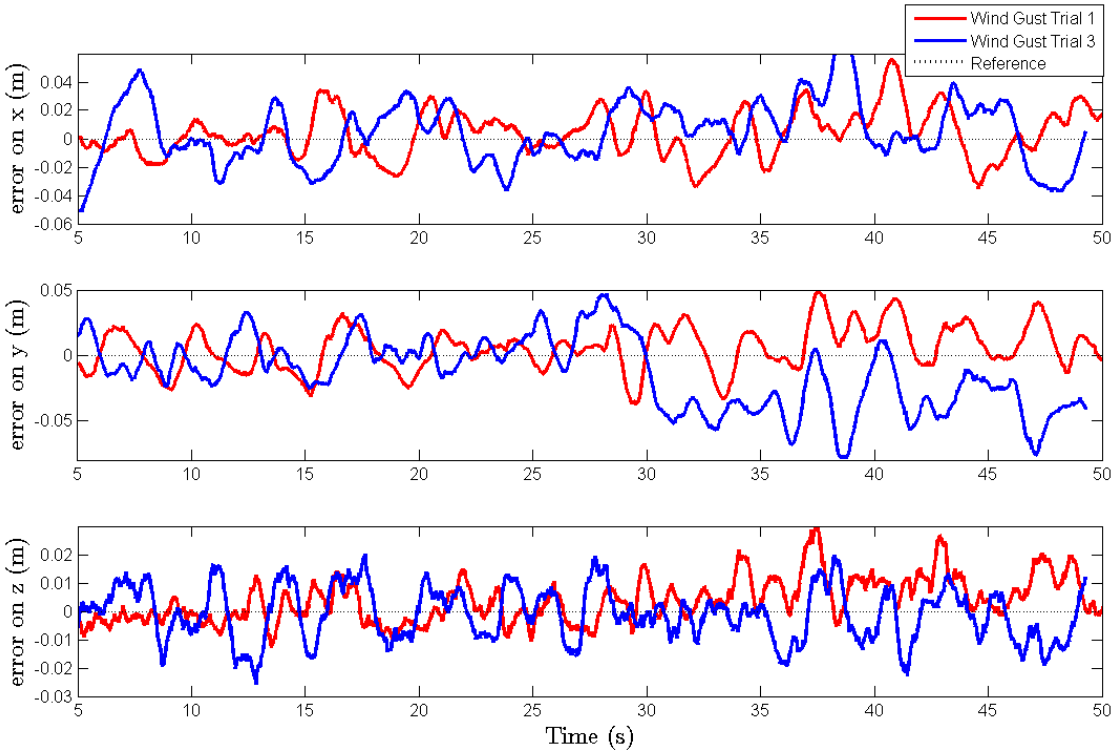


Figure 3.14: Resulting translational errors during quadrotor hovering under strong wind gusts. The solid red line represents the errors in the 1st trial. The solid blue line represents the errors in the 3rd trial. The dotted black line represents the zero reference.

3.7 Conclusions

In this chapter, we have proposed two complete, robust nonlinear control approaches for quadrotor MAVs, so that the quadrotors, even with uncertainties, can achieve challenging tasks that require aggressive maneuvers. We have not only validated the proposed methods in theory, but also verified it intensively (with more than 70 trials) via actual experiments such as waypoint navigation, path following with large tilt maneuvers, and stabilization in an environment with strong wind gusts.

A supplementary video demonstration of this work can be found online[1].

Future work could include a combination of the proposed robust control framework with pose estimation approaches via reliable onboard vision-based algorithms, so that the flying robots can exploit unknown environments without support of the external tracking systems, e.g. GPS or wall cameras. Another potential improvement is to combine the proposed approach with formation control and obstacle avoiding algorithms, in order to enable one or multiple UAVs to achieve tasks in complex scenarios autonomously.

[1] `https://drive.google.com/open?id=0B4QuufsZAdQacjJ3SVpNLXRUcHc`, accessed March-2017

Chapter 4

Fast Model Predictive Control with Obstacle Avoidance

In this chapter, we propose the problem of robust control of multirotor unmanned aerial vehicles (UAVs) in the complex scenarios. We present an effective and robust control approach to the waypoint navigation for the multirotor UAVs in 3D environments in presence of multiple obstacles.

The control architecture is based on a model predictive control (MPC) method with nonlinear constraints for obstacle avoidance, while the nonlinear attitude controller and the 6D nonlinear force/torque observer are both combined for the purpose of achieving robust and agile multirotor maneuvers. The fast nonlinear MPC framework is onboard implemented and is executed at 50Hz. The multirotor is able to autonomously decide to fly over or around the obstacles based on the online update of obstacle locations. The proposed control approach has been experimentally verified by commanding a quadrotor whose model is known only up to a certain degree of uncertainty to achieve waypoint navigation missions with obstacle avoidance. In addition, we have validated the robustness by comparing with other MPC methods (e.g. a nominal MPC and a tube MPC) on a waypoint navigation task in a collision-free environment.

The design of fast MPC with nonlinear obstacle avoiding constraint and the relevant configurations in this work have been modified as an online trajectory planning algorithm. These parts of work have been pre-published in Liu *et al.* (2017).

4.1 Introduction

Multirotor UAVs have become an active area of research within both control and robotics community, due to their potential to versatile tasks, e.g. aerial transportation Michael *et al.* (2011), building constructions Augugliaro *et al.* (2014) and mapping Fraundorfer *et al.* (2012), etc.

With the current trend of hardware development and miniaturization and algorithm optimization, those robust but computational costly control approaches, such as MPC methods, have been implemented onboard Aswani *et al.* (2012); Hartley *et al.* (2014). A learning-based MPC was proposed and tested in Bouffard *et al.* (2012) with low-power

onboard computational units. A fast nonlinear MPC using a particle swarm optimization approach has been recently proposed and implemented on FPGA Xu *et al.* (2016).

Apart from research on control theory and implementation Lee *et al.* (2010b); Raffo *et al.* (2010); Mellinger and Kumar (2011), much research has also been carried out on robustness Alexis *et al.* (2011); Bouffard *et al.* (2012) and trajectory generation with obstacle avoidance Heng *et al.* (2011) in order to help the multirotor UAVs to autonomously achieve complicated tasks in complex environments. An obstacle avoidance method for UAVs that is based on chaos trajectory surface was simulated in Bae and Kim (2004). A novel algorithm using a modified Grossberg neural network (GNN) Wang *et al.* (2007) is proposed for obstacle/collision avoidance in scenario of static obstacle. A gradient search algorithm for real-time obstacle avoidance and target tracking in UAVs is proposed in Zengin and Dogan (2007). An algorithm based on predictive control scheme with a decentralized control methodology has been designed in Boivin *et al.* (2008). In Paul *et al.* (2008), a potential field based collision and obstacle avoidance strategy has been followed. A robust control algorithm and a higher level path generation method based on graph theoretic considerations have been proposed in Regula and Lantos (2014). Recently, an algorithm based on measurements from an RGB-D camera and bin-occupancy filter along with a MPC approach has been proposed in Odelga *et al.* (2016). A technique based on a transformation of a variable constraint into an input saturation to ensure a safe trajectory around the obstacles along with a RGB-D sensor is proposed in Chauffaut *et al.* (2016). A novel biomimetic algorithm that uses optical flow data generated from the on-board camera to avoid obstacle has been proposed in Simpson and Sabo (2016). An improved method based on optical flow for obstacle avoidance is discussed in Wang *et al.* (2015). A novel monocular vision-based real time obstacle detection and avoidance for UAVs in GPS denied environment has been proposed in Saha *et al.* (2014). Stereo vision based obstacle avoidance algorithms have been proposed in Park and Kim (2012); Yuan-yan and Ying-xun (2011). The development and implementation of an obstacle avoidance controller for UAVs using four ultra-sound sensors is proposed in Bouabdallah *et al.* (2007).

In this chapter, we focus on a robust MPC method with nonlinear constraints for shortest path problems with obstacle avoidance. The proposed approach, along with a combination of the nonlinear attitude controller proposed in Liu *et al.* (2015), enables the multirotor UAV to achieve waypoint navigation tasks in complex environments with multiple obstacles, even in case of model uncertainties and unknown disturbances. Based on the online update of obstacle positions, the proposed MPC approach allows the multirotor to avoid the detected obstacles. Therefore, the main contribution of this chapter is a fast implementation of MPC with the formulation of nonlinear soft constraints on a quadrotor UAV to avoid obstacles in 3-dimensional scenarios. Simply adding nonlinear constraints into MPC is too computationally expensive to run the algorithm on our onboard computer. Thus we have also designed and integrated an algorithm to detect valid obstacles (those actively obstructing the UAV path), and to decide when the multirotor should fly over or around the obstacles. The effectiveness of this approach is proved through experi-

ments in which the MPC is performed onboard at 50Hz in an environment with randomly located box obstacles. We have also evaluated the robustness of the proposed controller through the waypoint navigation tasks in a collision-free environment, comparing the results with a linear nominal MPC method and a robust "tube MPC" method (see, e.g. Langson *et al.* (2004); Mayne *et al.* (2005)).

The outline of this chapter is as follows. We first present the background of model predictive control and the basic MPC formulation for the position control of multirotor UAVs in Sec. 4.2. Sec. 4.3 gives a brief description on the nonlinear attitude controller implemented on the multirotor. We then introduce the two proposed robust MPC design for the multirotor control: In Sec. 4.4, we introduce some technical details of robust MPC, including the design of a tube-based MPC for the multirotor position control; We demonstrate how to combine the MPC configuration with the nonlinear 6D force/torque external wrench observer in Sec. 4.5. The MPC architecture with obstacle avoidance is described in Sec. 4.6, where we detail a method of MPC formulation with nonlinear constraints for obstacle avoidance, and how the multirotor UAV decides to fly over or around the obstacles. Sec. 4.7 demonstrates the experimental results of waypoint navigation tasks in environments with obstacles, and the results of the comparison on the obstacle-free tests. We conclude the chapter in Sec. 4.8.

4.2 Model Predictive Control Architecture

We develop a MPC-based control framework in order to maneuver the multirotors to achieve the waypoint navigation tasks in the environments with obstacles.

MPC is an advanced feedback control method that predicts the change in the dependent variables of the modeled system. It first found broad early application in the process industry, where the typically longer time scales were compatible with the required time to solve the optimization problem. Since last decades, MPC has become feasible for control of systems with faster dynamics, thanks to both the development of more efficient solution techniques and the exponential increase in computing power. The controller minimizes a cost function by computing a sequence of optimal control inputs and the resulting optimal trajectory for a UAV model with constraints over a finite horizon. Only the first control input of the sequence is then applied to the UAV. At the next time step, the calculations are repeated starting from the current state, obtaining new control inputs as well as a new predicted trajectory.

The final design of our proposed framework can be illustrated in the block diagram in Fig. 4.1. For the purpose of fitting to the onboard implementation, the MPC is used as a position controller in the complete control framework, while a nonlinear backstepping-like controller is used to regulate the attitude of a multirotor. A 6-dimensional nonlinear wrench observer is further applied to estimate the unknown disturbances as external force and torque terms, and pass the estimates to the controllers for compensation. Since a MPC is used in the closed-loop system, no additional trajectory planner is required

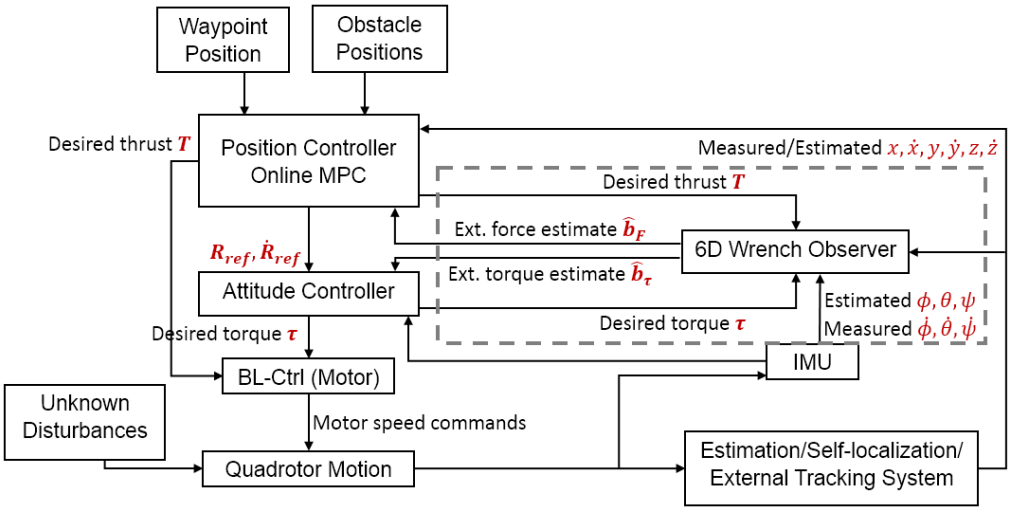


Figure 4.1: Block diagram of the MPC-based control framework for multirotor waypoint navigation with obstacle avoidance. The gray dotted block denotes the inclusion of a 6D nonlinear observer for unknown force and torque disturbances.

during the flight.

In this chapter, for the purpose of benchmark, we also employ a linear MPC and a tube-based MPC onto a quadrotor. In these two cases, the 6D nonlinear observer is not activated, thus the gray dotted block in Fig. 4.1 is omitted.

In actual experiments, the positions of the target waypoint and obstacles, as well as the current translational and attitude data of a quadrotor are updated online and passed to the proposed MPC and then attitude controller, which finally outputs the desired rotational speed of each motor on the quadrotor.

For the sake of completeness, we here first briefly recap the dynamic model used in the design of multirotor control. The mapping from the desired force and torque to the motor speeds can be found in detail in Sec. 2.1.1.

Considering the aerial vehicle as rigid body (as in Sec. 2.1.1), the dynamic equations are given by

$$\dot{x} = v, \tag{4.1a}$$
$$m\dot{v} = -mge_3 + RF + b_F, \tag{4.1b}$$
$$\dot{R} = RQ(\omega), \tag{4.1c}$$
$$J\dot{\omega} = -\omega \times J\omega + \tau + b_\tau, \tag{4.1d}$$

where m is the mass of a multirotor, x denotes the translational position, g is the scalar value of the gravity acceleration, and $e_3 = (0,0,1)^\top$. The nonconservative forces and moments in the body frame generated by the rotor propellers on the multirotors are represented by $F \in \mathbb{R}^3$ and $\tau = (\tau_x, \tau_y, \tau_z)^\top$. Here b_F, considered in the form of an external

force in the inertial frame, denotes the additional forces due to external disturbances and unmodeled dynamics, and b_τ is considered as an external torque in the body frame. The inertia matrix J in the body frame, which is decided via a simple CAD model of the multirotor UAV for computation. $Q(\omega)$ is the skew-symmetric matrix form of the angular velocity $\omega = (\omega_1, \omega_2, \omega_3)^\top$ in the body frame.

The rotation matrix R, which is an element of the special orthogonal group SO(3)= $\{R \in \mathbb{R}^{3\times3} | R^{-1} = R^\top,\ \det R = 1\}$, represents the attitude of the quadrotor with respect to the inertial frame.

4.2.1 MPC Formulation

In this section, we first present a general structure of the MPC for the position control of a multirotor. By modifying the translational motion (4.1a) and (4.1b) into a discrete-time model, the MPC can be formulated as a general optimal control problem (OCP):

$$\min_{\bar{u},\varepsilon} J(\bar{\xi}(\cdot), \bar{u}(\cdot), \varepsilon), \tag{4.2a}$$

$$\text{s.t. } \bar{\xi}[k+1] = A\bar{\xi}[k] + B(\bar{u}[k] - ge_3), \tag{4.2b}$$

$$a_{\min} \leq \bar{u}[k] \leq a_{\max}, \tag{4.2c}$$

$$\xi_{\min} \leq \bar{\xi}[k] \leq \xi_{\max}, \tag{4.2d}$$

where $a_{\min}, a_{\max} \in \mathbb{R}^3$ represent the lower and upper constraints of quadrotor accelerations along three axes. Similarly, $\xi_{\min}, \xi_{\max} \in \mathbb{R}^6$ denote the lower and upper bounds of the position and velocity. ε represents the parameter to bound the soft constraints for obstacle avoidance, which is introduced later.

The discrete-time model of the multirotor can be derived from Eq.(4.1). The mathematical model sets the acceleration as the inputs for the OCP describing the multirotor motion. The state $\bar{\xi}$ consists of the position and velocity along three axes, namely $(x, \dot{x}, y, \dot{y}, z, \dot{z})^\top$. The state $\bar{\xi}$ and nominal input $\bar{u}$ at k^{th} step with a sampling time Δt in Eq.(4.2b) are given by

$$\bar{\xi}[k] = \begin{bmatrix} x(k\Delta t) \\ \dot{x}(k\Delta t) \\ y(k\Delta t) \\ \dot{y}(k\Delta t) \\ z(k\Delta t) \\ \dot{z}(k\Delta t) \end{bmatrix}, \bar{u}[k] = \begin{bmatrix} \ddot{x}(k\Delta t) \\ \ddot{y}(k\Delta t) \\ \ddot{z}(k\Delta t) \end{bmatrix}, \tag{4.3a}$$

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.3b}$$

$$B = \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 & 0 \\ \Delta t & 0 & 0 \\ 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & \Delta t \end{bmatrix}. \tag{4.3c}$$

Therefore, in a collision-free assumption, the OCP for waypoint navigation task of a multirotor can be described as a quadratic cost function

$$\min_{\bar{u}} J = \sum_{k=0}^{N} \|\bar{u}[k]\|_P^2 + \|\bar{\xi}[N+1] - \xi_{\text{ref,N+1}}\|_L^2, \tag{4.4}$$

where N represents the receding horizon, $\xi_{\text{ref,N+1}}$ denotes the terminal reference (i.e. waypoint position and terminal velocity). P and L are the weights of input cost and terminal state cost, respectively.

4.2.2 MPC for Position Control

The major cost of OCP computation is on the problem formulation and matrix generation. We thus generate in advance the matrices that are used for numerical computation. The computed first step control input $\bar{u}[0]$ is used in the control law. The described convex OCP is solved via CVXGen Mattingley *et al.* (2011) using the interior point method.

Substituting the computed nominal input to the multirotor model (4.1) and assuming that R in (4.1b) equals R_{ref}, the control law for the multirotor position can be stated as

$$R_{\text{ref}} F \triangleq m\bar{u}[0] := F'. \tag{4.5}$$

Considering that R_{ref} in (4.5) is orthonormal and only the third entry of F, i.e. the thrust T, is nonzero, one must choose $T = \|F'\|$ in order to suffice Eq. (4.5). Therefore, the third column of $R_{\text{ref}} = \begin{bmatrix} r_{\text{ref}}^1 & r_{\text{ref}}^2 & r_{\text{ref}}^3 \end{bmatrix}$ can be solved from

$$r_{\text{ref}}^3 = \frac{F'}{\|F'\|}, \tag{4.6}$$

and the other two columns of R_{ref} can be filled orthonormally, for instance by a Gram-

Schmidt process with candidates r^3_{ref} from the previous equation and r^1, r^2 from the actual rotation $R = \begin{bmatrix} r^1 & r^2 & r^3 \end{bmatrix}$, i.e.

$$r^{2\prime}_{\text{ref}} = r^2 - \frac{r^2 \cdot r^3_{\text{ref}}}{r^3_{\text{ref}} \cdot r^3_{\text{ref}}} r^3_{\text{ref}}, \tag{4.7a}$$

$$r^2_{\text{ref}} = \frac{r^{2\prime}_{\text{ref}}}{\| r^{2\prime}_{\text{ref}} \|}, \tag{4.7b}$$

$$r^{1\prime}_{\text{ref}} = r^1 - \frac{r^1 \cdot r^3_{\text{ref}}}{r^3_{\text{ref}} \cdot r^3_{\text{ref}}} r^3_{\text{ref}} - \frac{r^1 \cdot r^2_{\text{ref}}}{r^2_{\text{ref}} \cdot r^2_{\text{ref}}} r^2_{\text{ref}}, \tag{4.7c}$$

$$r^1_{\text{ref}} = \frac{r^{1\prime}_{\text{ref}}}{\| r^{1\prime}_{\text{ref}} \|}, \tag{4.7d}$$

such that we obtain both the desired total thrust T and the desired attitude R_{ref} for the attitude control.

4.3 Attitude Control

The control method we have implemented for the tracking of the attitude R of a multirotor UAV was proposed in Chapter 3. We here only report the control law that is employed on the multirotor for the sake of brevity.

The specific approach is based on the solution of a class of output regulation problems which contains the rotational motion for a rigid body. The tracking error is given by $E = RR^\top_{\text{ref}}$, where R corresponds to the rotation matrix describing the current attitude, and R_{ref}, computed via the approach introduced in Sec. 4.2.2, represents the desired attitude. We also assume the error between the current and a desired angular velocity in body frame $e_\omega = Q(\omega) - Q(\omega_{\text{d}})$.

The goal of the attitude controller is to regulate the output of the attitude $(E, e_\omega) \to (I, 0)$ for $t \to \infty$, which implies $R \to R_{\text{ref}}$ and $Q(\omega) \to Q(\omega_{\text{d}})$ for $t \to \infty$. For the sake of brevity, the full derivation as well as the proof of asymptotic stability can be found in Sec. 3.2.2., is not reported in this chapter.

Based on Liu *et al.* (2015), the control law for the multirotor UAV's attitude is given by

$$\begin{aligned} \tau_{\text{d}} = JQ^{-1} \Bigg(& -k_\omega e_\omega - \frac{k_{\text{rot}}}{2} \Big[\dot{R}^\top_{\text{ref}} R + R^\top_{\text{ref}} \dot{R} - \dot{R}^\top R_{\text{ref}} \\ & - R^\top \dot{R}_{\text{ref}} \Big] - \ddot{R}^\top_{\text{ref}} R_{\text{ref}} - \dot{R}^\top_{\text{ref}} \dot{R}_{\text{ref}} - 2R^\top_{\text{ref}} R \Bigg) + \omega \times J\omega, \end{aligned} \tag{4.8}$$

where k_ω is a positive gain for the tracking error of the multirotor angular velocity. The derivative $\dot{R}$ can be obtained by introducing the current attitude R into (4.1c), while R_{ref}

inherits from the results of (4.6) and (4.7). We choose the value $\omega_{1,\text{ref}}, \omega_{2,\text{ref}} = 0$ and $\omega_{3,\text{ref}} = \psi_{\text{ref}} - \psi_{\text{current}}$ for the reference and current yaw attitude ψ at every moment, and $\dot{R}_{\text{ref}}$, hence, can be solved via Eq.(4.1c).

4.4 Tube-based Model Predictive Control

We have introduced a model predictive control framework for the flight control of multirotor UAVs. We next introduce two different ways to improve the robustness of the MPC frame. In this section, we modify the general MPC configurations into a "tube" MPC for the multirotor position control. We will give several background definitions on robust MPC, followed by the computations of the Minimally Robust Positive Invariant set (MRPI) for the robust MPC design, and then show the control law of a "tube" MPC in Section 4.4.2.

The general idea of a tube MPC is to use a control law in the form of

$$u = \bar{u} + K(\xi - \bar{\xi}), \tag{4.9}$$

where $\bar{u}$ and $\bar{\xi}$ denote the nominal control input and the system state under no disurbance assumption (which have been introduced in Sec. 4.2.1), while u and ξ denote the control input and state in the actual system with disturbances. K represents a linear stabilizing state feedback gain for the error between the actual system states and the predicted states. Using a predefined K, we can compute an MRPI for the error system, so that the nominal system, which optimizes the quadrotor translational motion, can maintain exponential stability with tightened state and input constraints.

In general, a discrete-time, time-invariant linear system to be controlled can be described by

$$\xi_{k+1} = A\xi_k + Bu_k + w, \tag{4.10}$$

where ξ_k, u_k and w represent the state, input and disturbance vectors. The matrix pair (A, B) is assumed controllable. The system is subject to the constraints

$$\xi \in \Xi \subseteq \mathbb{R}^n,\ u \in \mathcal{U} \subseteq \mathbb{R}^m,\ w \in \mathcal{W} \subseteq \mathbb{R}^n, \tag{4.11}$$

where $\Xi, \mathcal{U}$ and $\mathcal{W}$ are compact, convex polytopes that contain the origin in their interiors.

Under no disturbance assumption, we can define the nominal system corresponding to (4.10) as

$$\bar{\xi}_{k+1} = A\bar{\xi}_k + B\bar{u}_k. \tag{4.12}$$

The computed first step control input $\bar{u}[0]$ is used in the control law. In additional, the predicted state at the first step (as $\bar{\xi}[1]$ in the nominal OCP) is passed into the controller as the nominal state used to calculate the error between the current quadrotor state and the predicted state in next time step (we record it as $\bar{\xi}_{\text{prev}}$ in (4.16)).

4.4.1 Robust MPC and Robustly Positively Invariant Set

In this section, we introduce how to set up the "tightened" constraints for nominal states and control inputs so that the configuration of a tube MPC maintains stable. We here start from certain background definitions widely applied in robust control research.

For two given sets $X \subseteq \mathbb{R}^n$ and $Y \subseteq \mathbb{R}^n$, the Minkowski set addition $\oplus$ is defined by $X \oplus Y \triangleq \{x+y | x \in X, y \in Y\}$. The Pontryagin difference $\ominus$ is defined by $X \ominus Y \triangleq \{x \in \mathbb{R}^n | x+y \in X, \forall y \in Y\}$.

The one-step robust reachable set for the system (4.10) subject to (4.11) is defined as

$$\begin{aligned} \text{Reach}(\Xi, \mathcal{U}, \mathcal{W}) \subseteq \{\xi_1 \in \mathbb{R}^n | \exists \xi_0 \in \Xi,\ \exists u \in \mathcal{U}, \\ \exists w \in \mathcal{W}, s.t. \xi_1 = A\xi_0 + Bu + w\}. \end{aligned} \tag{4.13}$$

A set $\mathcal{Z} \subseteq \Xi$ is defined as an MRPI for the system (4.10) subject to the constraints in (4.11), if initial state $\xi_0 \in \mathcal{Z}$, then $\xi_k \in \mathcal{Z}, \forall w_k \in \mathcal{W}, \forall k \geqslant 0$.

Now we assume that $\mathcal{Z}$ is an MRPI for the system (4.10). If $\xi_k \in \{\bar{\xi}_k\} \oplus \mathcal{Z}$, then $\xi_{k+i} \in \{\bar{\xi}_{k+i}\} \oplus \mathcal{Z}$, for all $i > 0$ and $w \in \mathcal{W}$, where ξ_k and $\bar{\xi}_k$ are the states at k^{th} step for the actual and nominal system respectively.

This proposition suggests that if $\xi_0 \in \{\bar{\xi}_0\} \oplus \mathcal{Z}$, then $\xi_k \in \{\bar{\xi}_k\} \oplus \mathcal{Z}$, for all $k > 0$ and $w \in \mathcal{W}$, as well as that the control law in (4.9) can satisfy the constraints for the system with disturbance as long as there is a feasible solution for the nominal system (4.12) subject to the tightened constraints

$$\bar{\Xi} = \Xi \ominus \mathcal{Z}, \quad \bar{\mathcal{U}} = \mathcal{U} \ominus K\mathcal{Z}. \tag{4.14}$$

$K \in \mathbb{R}^{m \times n}$ is a gain (also to be used in (4.9)) such that $(A+BK)$ is Hurwitz for (A,B) in the nominal system (4.12) (these have been proved in Mayne *et al.* (2005)).

The state feedback gain K used for the tube MPC design is chosen from a infinite horizon continuous linear quadratic regulator (LQR) for the nominal system (A,B), denoted as K_{LQR}. Therefore, $u = -K_{\text{LQR}}\xi$ in (4.13). The MRPI $\mathcal{Z}$ is computed offline associated with this gain. Instead of satisfying an exact convergence condition of MRPI, which is with no guarantee of a termination within finite iterations, previous work provides methods Rakovic and Baric (2010) to compute an approximate of MRPI. We use a similar strategy to compute MRPI with an approximate convergence condition, as is described by the following procedure.

Starting with an initial approximation of the set $\Omega_0 = \{0\}$, we repeat the computation that $\Omega_i = \text{Reach}(\Omega_{i-1}, \mathcal{U}, \mathcal{W}) \cup \Omega_{i-1}$ until the set satisfies the approximate convergence condition $\mu(\Omega_i \setminus \Omega_{i-1}) < \varepsilon$, where μ denotes a positive parameter and ε denotes a bound set. Therefore, $\mathcal{Z} = \Omega_i$ at the end of the iterations is the MRPI for the system (4.10), and the state and input constraints (4.14) in the nominal OCP become

$$\bar{\Xi} = \Xi \ominus \mathcal{Z}, \quad \bar{\mathcal{U}} = \mathcal{U} \ominus K_{\text{LQR}}\mathcal{Z}, \tag{4.15}$$

where a larger K_{LQR} will results in a smaller $\mathcal{Z}$ but a larger $K_{\mathrm{LQR}}\mathcal{Z}$ so that the input actions are further tightened.

4.4.2 Tube MPC Design

The output of the proposed tube MPC method at each time step is written as

$$u_{\mathrm{pos}} = \bar{u}[0] + K_{\mathrm{LQR}}(\xi - \bar{\xi}_{\mathrm{prev}}), \tag{4.16}$$

where $u_{\mathrm{pos}} \subseteq \mathbb{R}^3$ denotes the first step control input computed from our proposed tube MPC method. Assuming that R in (4.1b) equals R_{ref}, the complete control law for the quadrotor position is given by

$$R_{\mathrm{ref}}F \triangleq mu_{\mathrm{pos}} = m\bar{u}[0] + mK_{\mathrm{LQR}}(\xi - \bar{\xi}_{\mathrm{prev}}) := F'. \tag{4.17}$$

Despite the fact that the mismatch of mass would still lead to the offsets to reference, such offsets are expected to get largely compensated by K_{LQR}. To make the solution practical, the unknown disturbance is assumed within the boundary between $(-3N, -3N, -4N)$ and $(3N, 3N, 4N)$.

The desired attitude R_{ref} for the attitude control can be then calculated via the Gram-Schmidt process. Since we have introduced the technical details once in Sec 4.2.2, we do not repeat the computation.

4.5 Nonlinear Observer

In this section, we extend the MPC approach discussed in previous sections with the inclusion of a nonlinear force/torque external wrench observer using the Fault Detection and Isolation (FDI) method Takakura *et al.* (1989). The observer estimates all the system offsets, parameter uncertainties and external disturbances, and passes the estimate to the controller for compensation. The addition of such a nonlinear observer greatly improves the robustness and the accuracy of our MPC method. The technical details, i.e. the full definitions and the proof of asymptotic stability, and the experimental validation of the observer on a multirotor UAV can be found in Chapter 3 in this thesis.

For the sake of completeness we briefly summarize its working principle. The technical details can be found in 3.4. Considering the system states $\zeta \triangleq [\dot{x}\ \dot{y}\ \dot{z}\ \omega_1\ \omega_2\ \omega_3]^\top$, the dynamic model of the quadrotor in (4.1) can be expressed conveniently following the Lagrangian formulation as

$$M\dot{\zeta} + C(\zeta)\zeta + G = \Lambda + \Lambda_{ext} \tag{4.18}$$

where $M = \begin{pmatrix} m\boldsymbol{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & J \end{pmatrix} \in \mathbb{R}^{6\times 6}$ is the positive definite inertial matrix, $C(\zeta) = \begin{pmatrix} m\mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -S(J\omega) \end{pmatrix} \in \mathbb{R}^{6\times 6}$ expresses the coriolis and centrifugal terms, $G = [0\ 0\ mg\ 0\ 0\ 0]^\top$ is the gravita-

tional vector, $\Lambda = [RF^\top \ \tau^\top]^\top \in \mathbb{R}^6$ is the nominal wrench due to the control input and $\Lambda_{\text{ext}} = [b_F{}^\top \ b_\tau{}^\top]^\top \in \mathbb{R}^6$ is the external wrench acting on the quadrotor.

The residual based observer design is based on the simple but powerful idea of the generalized momenta $Q_w = M\zeta$. In fact, one can write the following first-order dynamic equation for the momentum as

$$\dot{Q}_w = \Lambda + \Lambda_{\text{ext}} + C^\top(\zeta)\zeta - G. \tag{4.19}$$

We define the residual vector $\boldsymbol{r} \in \mathbb{R}^6$ for the disturbance estimation of the multirotor as

$$\boldsymbol{r}(t) = K_I\left(Q_w - \int_0^t \left(\Lambda + C^\top(\zeta)\zeta - G + \boldsymbol{r}\right)ds\right), \tag{4.20}$$

where $K_I > 0$ is the diagonal gain matrix. The dynamic evolution of residual $\boldsymbol{r}$ satisfies

$$\dot{\boldsymbol{r}} = K_I\left(\Lambda_{\text{ext}} - \boldsymbol{r}\right), \quad \text{when } \boldsymbol{r}(0) = 0, \tag{4.21}$$

which has an exponentially stable equilibrium at $\boldsymbol{r} = \Lambda_{\text{ext}}$. For "sufficiently" large gains the dynamic residual in (4.21) becomes

$$\boldsymbol{r} \simeq \Lambda_{\text{ext}}. \tag{4.22}$$

Being a model-based estimating approach, if a particular component of Λ_{ext} is zero then the scalar $\boldsymbol{r}$ corresponding to that component converges to zero. Hence we can write the estimated external wrench as

$$\widehat{\Lambda}_{\text{ext}} = \begin{bmatrix} \widehat{b}_F \\ \widehat{b}_\tau \end{bmatrix} = \boldsymbol{r}, \tag{4.23}$$

where $\wedge$ indicates the estimated value of the variable.

With the inclusion of the external wrench estimate, the position control law in (4.5) is given by

$$R_{\text{ref}}F \triangleq -\widehat{b}_F + F' = -\widehat{b}_F + m\bar{u}[0], \tag{4.24}$$

while the attitude control law in (4.8) becomes

$$\begin{aligned} \tau_{\text{d}} = &-\widehat{b}_\tau + JQ^{-1}\left(-K_\omega e_\omega - \frac{K_{\text{rot}}}{2}\left[\dot{R}_{\text{ref}}^\top R + R_{\text{ref}}^\top \dot{R} - \dot{R}^\top R_{\text{ref}} - R^\top \dot{R}_{\text{ref}}\right]\right. \\ &\left. -\ddot{R}_{\text{ref}}^\top R_{\text{ref}} - \dot{R}_{\text{ref}}^\top \dot{R}_{\text{ref}} - 2R_{\text{ref}}^\top R\right) + \omega \times J\omega. \end{aligned} \tag{4.25}$$

The Eq.(4.24) and Eq.(4.25) provide the multirotor with a robust MPC method for the aggressive waypoint navigation tasks. The asymptotic stability of the attitude control-observation system has been proved in 3.4.1, while the position control-observation sys-

tem can not be directly proved stable. However, we experimentally validate the robustness of our proposed control approach in 4.7.2.

4.6 Predictive Control with Obstacle Avoidance

We have demonstrated the general framework of a fast, cascaded MPC method for multirotor UAVs and its two extensions with robustness design, a "tube" MPC and an observer combined MPC, in the last couple of sections. The latter two methods are both feasible for the robust control of the multirotor under disturbance in collision-free environment. In this section, we extend our proposed robust MPC method with obstacle avoidance behaviors by adding constraints into the cost function in the OCP.

The main objective of the obstacle avoidance constraints is to keep the multirotor with no collision even in a complex environment. We hereby detail the constraints imposed on the multirotor states to guarantee safe maneuvers.

Although the translation model of the multirotor is linear in the OCP, the new MPC frame we here propose is *nonlinear* since it includes nonlinear constraints to avoid obstacles. Instead of directly using hard constraints like state bounds, we set the obstacle avoidance constraint as a series of soft constraints in the cost function, in order to reduce the possibility that the solution of the OCP is infeasible.

In summary, the OCP (4.4) becomes a convex optimization problem with a set of soft constraints

$$\min_{\bar{u},\varepsilon} \sum_{k=0}^{N} \|\bar{u}[k]\|_P^2 + \|\bar{\xi}[N+1] - \xi_{\text{ref,N+1}}\|_L^2 + \lambda\varepsilon, \tag{4.26a}$$

$$\text{s.t. } f_{\text{obs}}(\bar{\xi}[k], p_{\text{obs,i}}) \leq \mathbf{1}\varepsilon,\ i = 1,\ldots,i_N, \tag{4.26b}$$

$$\varepsilon \geq 0, \tag{4.26c}$$

where $p_{\text{obs,i}}$ represents the 3D position of the i^{th} valid obstacle (identified from all detected obstacles). A set of i_N obstacle avoidance constraints are compactly written as $f_{\text{obs}}(\bar{\xi}, p_{\text{obs,i}})$, whose expressions are given later in equations (4.29) and (4.30), depending on the choice of the UAV to fly over or around the obstacle respectively. $\mathbf{1}$ is the 1-value vector with the dimension of i_N. ε represents a parameter to be minimized in the OCP, which is the upper bound of the soft constraints for obstacle avoidance. λ denotes a positive weight of a non-negative scalar ε.

To fit the limited onboard computational capability, we need to reduce the complexity of the OCP. One possible way is to reduce the number of obstacles considered in the OCP. Therefore, we perform two additional steps: selecting a group of valid obstacles that really block the flight line during navigation, and deciding the best way for the multirotor to avoid obstacles.

The algorithm to decide the obstacle avoidance constraints consists of the following

steps: i) detecting the real obstacles via sensors and model them into "obstacles"; ii) determining the "Valid Obstacles" (VOs) that block the multirotor to fly towards the waypoint in a shortest path; iii) flying over the VOs if possible; otherwise, determining a group of VOs to avoid by flying around, and generating a function to describe obstacle avoidance constraints.

Assuming that the 3D positions of obstacles are available, we simply regard an "obstacle" as one cylinder with a nominal blocking radius and height. Such a model is naive in practical applications since the obstacle shape can be arbitrary. To make the assumptions more realistic, we improve this by setting each corner (or a certain point on a long edge) of a real obstacle as the center of a nominal "obstacle". For instance, in case of a polygon-shaped obstacle (e.g. a box), we consider the top corners of the box as a group of 4 (overlapping) "obstacles" instead of one. In real experiments, we deploy the markers on the corners of arbitrary polygon-shape obstacles and obtain the locations of the markers via the external tracking system.

While we have specified step (i) in the previous paragraph, the remaining steps of the obstacle avoidance algorithm are described in Sec. 4.6.1 and 4.6.2. The way to identify valid obstacles that block the flight line is inspired from the idea to detect "blocking objects" in the subtarget method on soccer robots in Bruijnen *et al.* (2007). We extend it to fit 3D environments.

4.6.1 Identification of Valid Obstacles

An example of the obstacle map for the multirotor flight can be illustrated in Fig. 4.2.

We decompose all "obstacle" positions into horizontal (2D) and vertical (1D) components. Let o_r and h_r respectively be the horizontal and vertical components of the position of a multirotor UAV, o_t and h_t be the components of the target waypoint position, and o_i and h_i describe the position of an arbitrary "obstacle", respectively. To determine which "obstacle" is VO that really obstruct the flight path, the centers of all "obstacles" are projected onto the straight flight line between the multirotor and the waypoint. The projected distances are given by

$$a_i = \frac{(o_t - o_r) \cdot (o_i - o_r)}{\|o_t - o_r\|}, \tag{4.27a}$$

$$b_i = \frac{(o_t - o_r) \times (o_i - o_r)}{\|o_t - o_r\|}, \tag{4.27b}$$

$$\delta_i = \frac{a_i}{\|o_t - o_r\|}(h_t - h_r) + h_r - h_i, \tag{4.27c}$$

where a_i denotes the length of the projected vector on the horizontal plane, b_i and δ_i are the horizontal and vertical distance between an "obstacle" i to its projection point, respectively. We consider the set B of "obstacles" that are high enough and close to the

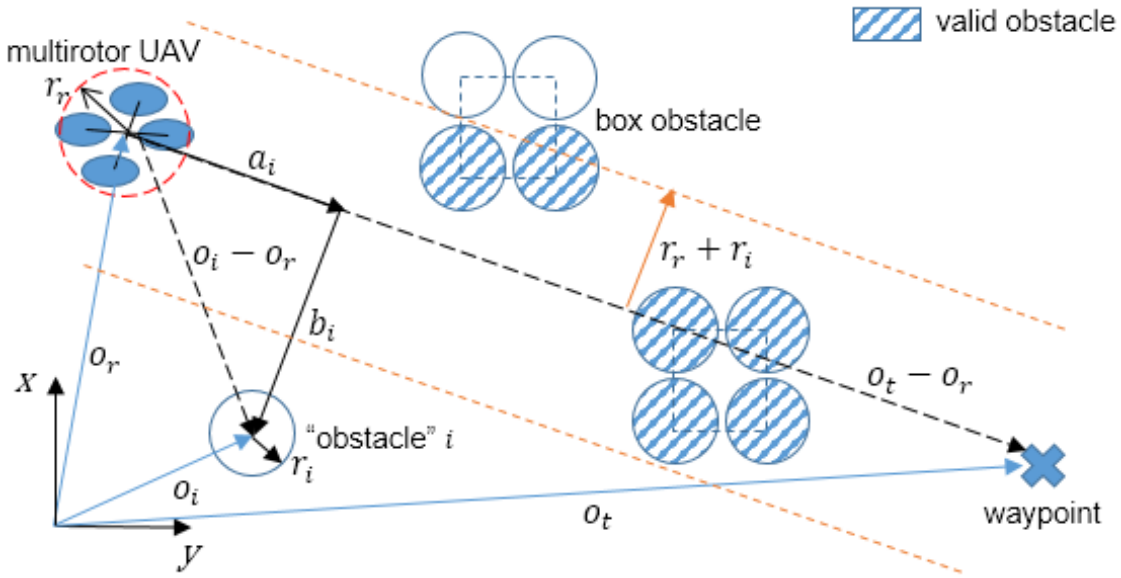


Figure 4.2: An example of an obstacle map for waypoint navigation with obstacle avoidance. The projection on horizontal plane.

flight line as VOs by computing as follows

$$\begin{aligned} B = &\{i \in \mathbb{N}^{n_b} | 0 < a_i < \|o_t - o_r\| \cap \|b_i\| < r_i + r_r \\ &\cap \delta_i < \delta h_{\text{safe}}\} \in \mathbb{N}^{m_b}, \end{aligned} \tag{4.28}$$

where n_b, m_b denote the amount of "obstacles" and VOs respectively, r_i is the nominal radius of each "obstacle" cylinder and r_r is the assumed safe radius around the multirotor during the flight. $\delta h_{\text{safe}} \in \mathbb{R}^+$ represents the safe height difference between the multirotor and an "obstacle", noting that the vertical distance δ_i will be negative if an "obstacle" is higher than the expected flight height at the projected point.

4.6.2 Flying Over or Around

Based on the valid obstacle detection algorithm that we described in the previous subsection, there are two possibilities for the multirotor to avoid the obstacles: either to *fly over*, or to *fly around* them. It is possible for the multirotor UAV to *fly over* the VOs that are not too tall, namely $\forall \delta_i \geq 0$. We conservatively design the lift height to fly over as δh_{safe}. Therefore, the obstacle avoidance constraints are given by

$$f_{\text{obs}}(\bar{\xi}, p_{\text{obs,i}}) = \delta h_{\text{safe}} + z_{\text{obs,i}} - z[k],\ k = 1, \ldots, N, \tag{4.29}$$

which is a i_N-D vector used to set the soft constraint. $z[k]$ and $z_{\text{obs,i}}$ denote the vertical position of k^{th} state, $\bar{\xi}[k]$, and i^{th} valid obstacle, respectively. In this case, the constraint

function is still linear.

Otherwise, in case that the multirotor is not able to fly over the (tall) obstacles, we set the constraint so that the UAV can fly around those VOs, which can be stated as

$$\begin{aligned} f_{\text{obs}}(\bar{\xi}, p_{\text{obs,i}}) = &- (x[k] - x_{\text{obs,i}})^2 - (y[k] - y_{\text{obs,i}})^2 \\ &+ (r_i + r_r)^2,\ k = 1, \ldots, N, \end{aligned} \tag{4.30}$$

where $x[k]$, $y[k]$ and $x_{\text{obs,i}}$, $y_{\text{obs,i}}$ denote the horizontal positions of k^{th} state, $\bar{\xi}[k]$, and i^{th} valid obstacle, respectively. The obstacle avoidance constraint thus becomes nonlinear.

By importing the obstacle avoidance constraint (4.29) or (4.30) into the OCP (4.26), we solve the OCP via interior point method via the nonconvex OCP solver IPOPT Wächter and Biegler (2006). The number of VOs fixed to i_N and all VOs are initially set to a nominal position way far from the multirotor.

4.7 Verification

The robust nonlinear MPC approach presented herein have been validated step-by-step via numerical case studies in MATLAB, ROS/Gazebo physical simulations of a multirotor model with parametric mismatch and disturbances, and finally through real robot tests. We do not report the results of the simulations for the sake of brevity.

4.7.1 Experiment Configuration

A series of experiments have been carried out on a quadrotor UAV with capable of carrying a payload up to 1.0 kg. The quadrotor is equipped with a low-power Odroid-XU4 board.

The control algorithm has been implemented within the ROS-based TeleKyb framework Grabe *et al.* (2013). The MAV position and the positions of obstacles are recorded via an external motion capture system and passed to the on-board computer at a sampling frequency of 120 Hz over a wireless channel. The other essential data i.e. linear acceleration, orientation and angular velocity of the quadrotor, are all measured by a low-cost onboard Inertial Measurement Unit (IMU) and filtered via an extended Kalman filter. The attitude controller is executed at 100 Hz with the use of the filtered received states. Due to an average computational cost of 16.2 ms, the robust MPC controller is executed at 50 Hz with a horizon of 12 steps.

Overall, the vehicle weight is 1.6 kg. However, we set the nominal mass to 1.4kg during the experiments in order to create a mismatch between the physical and nominal values. Similarly, we estimated the coefficients of the inertia matrix through a CAD model and we used some mismatched values. The model uncertainties (mass and inertial parameters) and unknown disturbances (e.g. wind from a/c system and motor control

in open loop fashion, etc.) have been introduced in order to test the robustness properties of our controller, and they lead to a steady hovering of the quadrotor at a tilt of approximately 3° on pitch and -1.5° on roll in air.

The assumed radius of each "obstacle", r_i, is set to 0.3 m and the safe quadrotor radius r_r is set to 0.5 m, while the safe height difference δh_{safe} is set to 0.5 m. In order to reduce the computational cost of MPC, the VOs were ordered based on a_i and i_N was limited to 5. Therefore, only the clostest 5 VOs were taken into consideration in MPC formulation.

4.7.2 Experimental Results

Initially, the performance of the proposed robust MPC (as in Fig 4.1) was compared with a nominal MPC method and a tube MPC method. To meet a relatively fair comparison, the terminal weight parameters were fixed in the OCP so that the aggressiveness of the quadrotor UAV did not change during all trials with three controllers. The nominal MPC method uses only the configuration proposed in 4.2, thus with no robustness design. The tube MPC is based on the computation of a positively robust invariant set and thus tightened state and input constraints. For the sake of brevity, the offline computation of the robust invariant set is omitted. The readers can refer to Langson *et al.* (2004); Mayne *et al.* (2005) for more details.

In the first experiment, we commanded the quadrotor to achieve a waypoint navigation task. The quadrotor flew a square trajectory by reaching a series of 2m-far waypoints, $(-1,-1,1)$, $(1,-1,1)$, $(1,1,1)$, and $(-1,1,1)$, one by one and finally flying back to the starting point. The waypoints were sent to the quadrotor every 5s.

The resulting trajectories have been displayed in Fig. 4.3. From the results it can be read that a nominal MPC method (solid cyan line in Fig.4.3) led to a large offset in vertical direction up to 30cm, since the quadrotor mass was mismatched by 200g in the model. In horizontal directions, the errors to the waypoints are relatively small because the weights in MPC is set to large values in order to increase the aggressiveness and accuracy of flight. With the additional LQR settings, the tube MPC method (solid blue line in Fig.4.3) to a large extent improves the control performance. The offset in z-direction reduces dramatically.

The resulting Root-Mean-Square Error (RMSE) on position, in the period from 1 s after the new waypoint is given until the next waypoint is subscribed, is computed and shown in Table 4.1. The proposed robust MPC method (solid red line in Fig.4.3) leads to a RMSE of 6.46 cm. For tube MPC method, the RMSE is 13.86 cm, while the RMSE with the nominal MPC algorithm reaches 32.52 cm. According to the experimental results, our proposed robust MPC outperforms the other two MPC methods.

In the second series of experiments, the quadrotor UAV executed a waypoint navigation task in the environment with arbitrary located box obstacles (Fig. 4.4). In order to arrive at the target waypoint, the quadrotor had to decide autonomously how to avoid the box obstacles. The results of 3 trials are shown in Fig. 4.5. During the experiments, reflective markers were set at each corner of the boxes. The position data of the markers

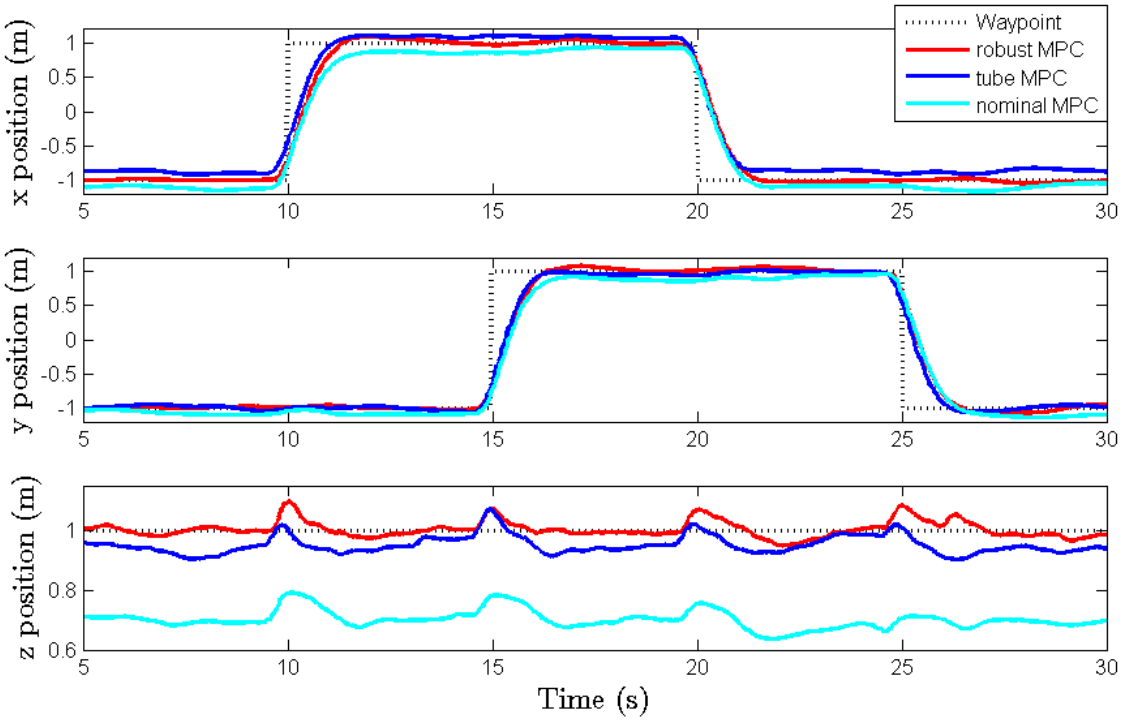


Figure 4.3: Resulting quadrotor response for a square waypoint navigation task. The reference was switched to a new 2m-far waypoint every 5s. The dotted black line represents the waypoint reference. The solid red line represents the results with the proposed robust MPC method. The solid blue line is the results with a tube MPC method. The solid cyan line shows the results with a nominal MPC method.

were detected through the external tracking system and passed to the onboard computer. Therefore, the quadrotor received the position of "obstacles" at around 100Hz.

In the first trial, the quadrotor was commanded to fly towards the target waypoint in the environment visualized in Fig. 4.5a. Two tall boxes obstructed the shortest-path flight line and thus the quadrotor autonomously decided to turn right to fly around the box obstacles. Then it detected another box obstacle that was not too tall. Hence, the quadrotor enabled to fly over the box and reached the target waypoint.

In the second trial, the boxes in the experimental area were relocated. We moved the two tall boxes slightly far from the shortest-path flight line. Still the quadrotor executed the same waypoint navigation task. Since the box obstructing the flight line were not too tall, the quadrotor succeeded in flying over it and arrived at the waypoint as shown in Fig. 4.5b.

In the third trial, three tall boxes were located between the quadrotor and the waypoint. We did not expect that the quadrotor could fly over or around all these obstacles and reach the final target. During the flight, the quadrotor first planned to turn right and fly around the box obstacles; however, it soon detected more tall obstacles that obstructed.

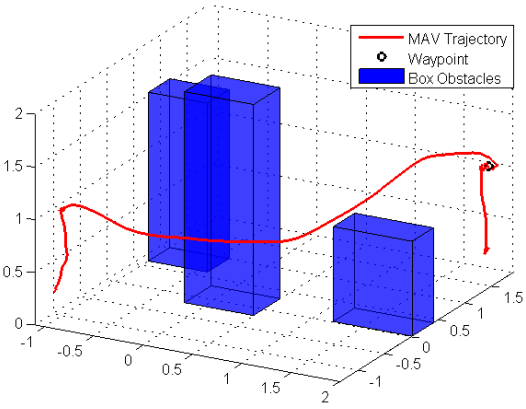
Methods	**RMSE** [**unit**]
Nominal MPC	32.52 [cm]
Tube MPC	13.86 [cm]
Proposed robust MPC	6.46 [cm]

Table 4.1: Root-Mean-Square Error (RMSE) for the quadrotor waypoint navigation task.

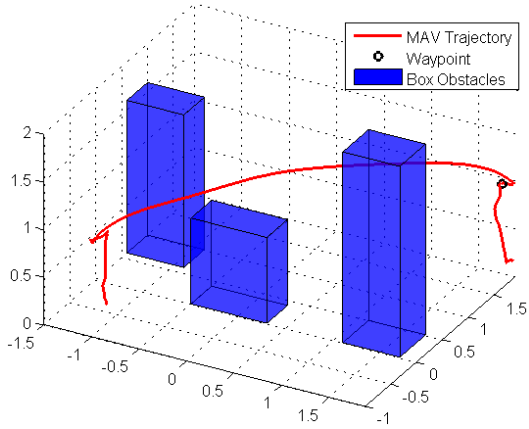
Figure 4.4: "Obstacle Avoidance" experiment. The quadrotor UAV executed a waypoint navigation task in an environment with obstacles.

The quadrotor UAV thus turned left to avoid the boxes but once again found another tall box obstacle on the way to waypoint (Fig. 4.5c). The quadrotor UAV was stuck and continuously turned left and right to find a feasible trajectory. In this case, the UAV, as expected, could not find a feasible trajectory until the end (that the quadrotor was commanded to land) since the indoor flight space was included as the hard constraints in MPC.

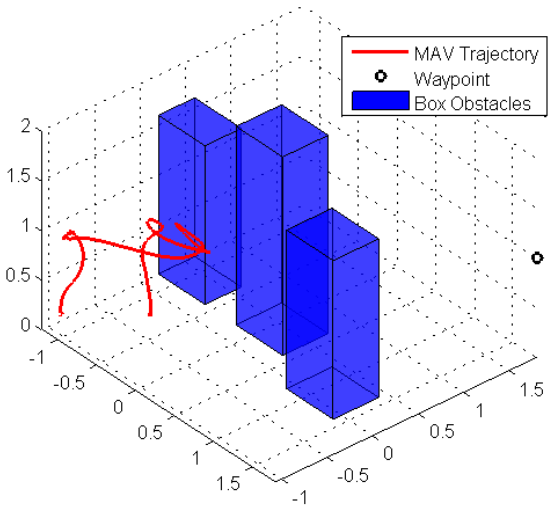
Due to the local information of valid obstacles (only closest 5 are taken into consideration), and the short receding horizon, the MPC approach we proposed is unable to overcome every complex environment. It is possible in the proposed MPC approach that the quadrotor gets into a local optimum flight trajectory, when several valid obstacles fully obstruct the potential flight path. This could be solved by implementing a MPC method, with a longer receding horizon and a more informative obstacle map, into a more powerful onboard computer.



(a) Scenario 1. The quadrotor UAV avoided the tall box obstacles by first turning right and flying around them. It then flew over the short box obstacle.



(b) Scenario 2. The quadrotor UAV avoided the box obstacles by flying over the short box obstacles.



(c) Scenario 3. The quadrotor UAV attempted to turn left and right to avoid 3 tall box obstacles, but it was unable to find a trajectory to reach the waypoint.

Figure 4.5: Results in "Obstacle Avoidance" experiments. The blue boxes represent the obstacles in the experimental area. The black circle denotes the target waypoint. The solid red line represents the resulting trajectory of the multirotor.

4.8 Conclusions

In this chapter, we have proposed a robust control approach based on a fast nonlinear model predictive control (MPC) method for a multirotor UAV. The presented MPC structure enables the multirotors to online determine the optimal trajectories for the waypoint navigation tasks in complex environments with multiple obstacles. We have detailed the onboard implementation of the proposed controller on a quadrotor with model uncertainties and disturbances. The experimental validations we have carried out and the comparison with other two MPC methods demonstrate the improved performance of our proposed robust MPC approach.

A supplementary video demonstration of this work can be found online[1].

The MPC method with obstacle avoidance proposed in this chapter has also been modified as a trajectory planning algorithm, by following the MPC formulation in Sec.3.5 in Chapter 3. The usage of fast MPC as a trajectory planner can be found online[2] in the last series of experiments.

Future work could include a combination of the proposed control framework and onboard cameras (with helps of reliable vision-based algorithms), so that the multirotor UAV is able to estimate its own pose and detect the obstacle positions without support from external tracking systems. Another potential work would be to solve the problems for obstacle avoidance due to the local optimum.

[1] `https://drive.google.com/open?id=0B4QuufsZAdQaR1c0bG85OFB6WEU`, accessed March-2017
[2] `https://drive.google.com/open?id=0B4QuufsZAdQacjJ3SVpNLXRUcHc`, accessed March-2017

Chapter 5

Distributed Control for Formation Balancing and Multi-UAVs Maneuvering

In this chapter, we propose a distributed formation control algorithm for a group of multirotor Unmanned Aerial Vehicles (UAVs). The algorithm brings the whole group of UAVs simultaneously to a prescribed submanifold that determines the formation shape in an asymptotically stable fashion in 2-dimensional (2D) and 3D environments. The complete distributed control framework is implemented with the combination of a robust model predictive control method on multiple low-power onboard computational units on multirotor UAVs and validated via a series of hardware-in-the-loop simulations and real-world experiments. The experiments are configured to study the control performance in various formation cases of arbitrary time-varying (e.g. expanding, shrinking or moving) shapes. In the actual experiments, up to 4 multirotors have been implemented to form arbitrary triangular, rectangular and circular shapes drawn by the operator via a human-robot-interaction device. We also carry out hardware-in-the-loop simulations using up to 6 onboard computers to achieve a spherical formation and a formation moving through obstacles.

Large parts of this work have been pre-published in Liu *et al.* (2018).

5.1 Introduction

Multi-agent systems have become one of the most active research topics within the robot control community. One of the fundamental problems studied for multi-agent systems is the *formation control* problem. In this problem, a team of agents is tasked with arranging into some pre-specified spatial configuration Oh *et al.* (2015). Often, formations are specified by certain relative state information that can be sensed between agents. These include position-based strategies Ren (2006), distance-based strategies Anderson *et al.* (2008); Krick *et al.* (2009), and bearing-based strategies Zelazo *et al.* (2015); Zhao and Zelazo (2015). The sensing capabilities of the vehicles will dictate which formation control strategy is most appropriate. More recently, a new approach to formation control was

proposed in Montenbruck *et al.* (2017) where the formation is specified by prescribing a *shape* (i.e., a circle, triangle, rectangle, etc.). Each agent then implements a decentralized controller that asymptotically stabilizes the agents to the desired shape while, simultaneously, a distributed controller balances their configuration on that shape.

As the theory of formation control has developed, so have the practical implementation of these strategies. Teams of Unmanned Aerial Vehicles (UAVs), for example, have the potential to perform versatile tasks, such as aerial transportation Michael *et al.* (2011), building constructions Augugliaro *et al.* (2014) and swarming above audience for entertainment D'Andrea (2016); Intel (2017b). Therefore, one interesting application of formation control research is on UAVs, e.g., fixed-wing drones and multirotors, due to their various potential applications.

Much research on formation control of UAVs has been carried out in both aeronautics and robotics communities. A collision-free control method based on the modified Grossberg neural network for a group of UAVs in square formation has been proposed in Wang *et al.* (2007). In Wang and Xin (2013), an integrated optimal formation method has been presented, which employs an inverse optimal control approach to achieve the formation of multiple UAVs with obstacle avoidance. A nonlinear model predictive control (MPC) method incorporating obstacle avoiding conditions using Karush Kuhn Tucker conditions has been proposed in Shin and Kim (2009) for formation flight. In Turpin *et al.* (2012), a decentralized formation control algorithm has been presented and tested on a group of quadrotors, which enables the robots to safely change the formation shape by following a specified group trajectory. A bearing formation control method based on relative angles has been proposed and tested on mutlirotor UAVs in real experiments in Franchi *et al.* (2012). A formation controller based on the virtual rigid body in SE(3) has been proposed and experimentally validated in Zhou and Schwager (2015). The latter method allows the quadrotors to simultaneously execute collision-free agile maneuvers as a group.

In this chapter, we propose a formation control algorithm which brings a group of systems to a specified shape simultaneously in a balanced and stable fashion. The algorithm is distributed and decentralized (we refer readers to Montenbruck *et al.* (2017) for the stability proofs of the proposed formation algorithm). The agents in the group exchange their position information only with their relative neighbors specified by a static and given information exchange network, and eventually arrange their positions in a specified shape.

Apart from the advantage of distributed fashion, the property of collision avoidance, and the capability of trajectory following, which have been included in some other formation approaches, the key novelties of our proposed approach are: i) This method is driven by a complete target shape, which is different from many existing formation approaches that lead the agents to the (absolute or relative) target positions of the target polygon vertices. More specifically, we define a desired formation by simply prescribing a shape in either $\mathbb{R}^2$, e.g. a circle, triangle or rectangle, or a shape in $\mathbb{R}^3$, e.g. a sphere, etc., plus the center and size of the desired formation. This is suitable for certain practi-

cal applications such as target capture, since we can simply define a moving target (not necessary an agent in the group of robots) as the center of a target shape. ii) The proposed method enables the agents to simultaneously reach to a balanced and converged condition. This allows the agents to start from most of the initial conditions, e.g. a line array, etc. The strong convergence property also ensures the group of agents to switch among various target (2D or 3D) formation shapes smoothly.

In summary, we address a series of balancing control problem for a practical application (formation) of a group of aerial robots. More specifically, we have proposed and implemented a novel formation algorithm, in a distributed fashion, onboard multiple multirotor UAVs with a combination of a fast MPC approach. In order to achieve a closer connection between the proposed algorithm and real-robot applications, we further develop the shape-driven property of our approach and implement the formations via the human-robot interaction of drawing the target shapes through a gesture tracking device (i.e. Leap Motion Leap Motion (2017)) during the experiments. The performance of the complete control framework has been validated via both a series of hardware-in-the-loop (HIL) simulations and real-world indoor experiments. During the experiments, multirotor UAVs were commanded to form an arbitrary, time-varying (e.g. expanding, shrinking or moving) circular, triangular, rectangular and spherical shape in 2D and 3D scenarios.

The outline of this chapter is as follows. In Sec. 5.2, we introduce the formation control algorithm that enables a group of agents to simultaneously reach balanced retractions. More specifically, we describe how to implement the distributed algorithm on multiple agents through case studies with various shapes. We then present a practical application of our proposed algorithm onto a group of multirotor UAVs in Sec. 5.3, where the complete control system of each multirotor UAV based on a MPC method is demonstrated in detail. Sec. 5.4 introduces the experimental validation of the complete distributed control framework and discussions, on both HIL simulation and real-world experimental stages. Sec. 5.5 concludes the technical chapter.

5.2 Formation Control and the Balancing Problem

In this section, we start from a brief recap of the solution (proposed in Montenbruck *et al.* (2017)) towards the retraction balancing problem. The solution leads to a distributed formation algorithm for the multi-agent system to retract and converge to a target shape with a smooth submanifold, e.g. circle, eclipse.

Compared to the previous work, we first extend this formation control algorithm with the targets in the shape of arbitrary convex polygon (e.g. triangle, rectangle, etc.), and make use of it in order to steer a group of multirotor UAVs towards a desired formation. In addition, we explore the possibilities of maneuvering the formation shape M through $\mathbb{R}^2$ and $\mathbb{R}^3$, i.e., to have our submanifold position and shape varying over time. These modifications were not addressed in Montenbruck *et al.* (2017) and are thus novel.

We consider n dynamical systems with positions x_i in $\mathbb{R}^2$ (or $\mathbb{R}^3$). Our goal throughout

the chapter is to eventually bring these positions x_i towards an evenly spaced, "balanced" (equidistant) configuration on a given shape $M \subset \mathbb{R}^2$ (or $M \subset \mathbb{R}^3$). Thereby, we assume that M is a smoothly embedded submanifold. In particular, M could be a circle, leading to a circular formation, a triangle (with smoothed out corners), leading to a triangular formation, or a portion of a line, leading to a collinear formation.

In the previous work Montenbruck *et al.* (2017), the authors introduced the scalar field

$$\phi(x) := \sum_{j>i} W_{ij} \ln(d(x_i, x_j)), \tag{5.1}$$

whereby x denotes the tuple of all positions x_i and $d\left(x_i, x_j\right)$ is the length of the shortest curve on M joining x_i with x_j, in resemblance to the potential whose maximizers are the so-called Fekete points known from mathematics Fekete (1923). Therein, the weights W_{ij} are determined by a weighted connected undirected graph, with W_{ij} denoting the weight of the edge joining i and j, and the convention that $W_{ij} = 0$ whenever there is no such edge. For most of the cases, we will take our graph to be the (undirected) cycle graph with $W_{ij} = W_{ji} = 1$ for $j = (i+1) \bmod n$ and $W_{ij} = W_{ji} = 0$ else. The scalar field (5.1) has the purpose of evenly spacing points on M through its gradient flow. If one, in addition, asymptotically stabilizes M, then one will eventually attain an evenly spaced configuration on M in a stable fashion. To this end, we studied the convergence properties of the formation control algorithm

$$\dot{x} = r(x) - x + \operatorname{grad}\phi(r(x)). \tag{5.2}$$

Therein, r is the smooth retraction onto M^n. Thus, the control action $r(x) - x$ asymptotically brings our positions x_i towards M. At the same time, $\operatorname{grad}\phi(r(x))$, the gradient vector field of ϕ evaluated at the retraction of x onto M^n, has the purpose of evenly spacing the points x_i on M. Together, these two controls thus bring the points x_i towards an evenly spaced configuration on M, as desired.

The differential equation (5.2) has quite strong convergence properties. In particular, all solutions initialized in the preimage of any regular superlevel set of the potential ϕ under the retraction converge to the desired formation in a stable fashion.

According to the derivation in our previous work Montenbruck *et al.* (2017), $\operatorname{grad}\phi(r(x))$ is computed via

$$\operatorname{grad}\phi(r(x)) = \sum_{j=1}^{n} \frac{W_{ij}}{d(r(x_i), r(x_j))} V_{ij}, \tag{5.3}$$

where $(x_i, x_j) \mapsto V_{ij}$ is the velocity vector of the unit speed geodesic (distance on submanifold) joining $r(x_i)$ and $r(x_j)$.

Therefore, the differential equation to regulate the motion of each agent in the group is given by

$$\dot{x}_i = r(x_i) - x_i + \sum_{j=1}^{n} \frac{W_{ij}}{d(r(x_i), r(x_j))} V_{ij}, \tag{5.4}$$

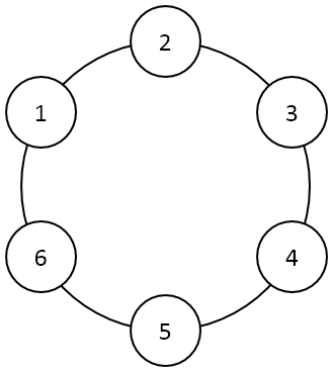


Figure 5.1: Cycle graph C_6 for a group of 6 agents.

In the following sections, we introduce three examples of prescribed shapes in 2D environment as well as an example of a formation shape in 3D environment.

5.2.1 $\mathbb{R}^2$ Formation: The Circle

In 2D formation cases, the agents in the group communicate through the unweighed cycle graph (e.g. an example of 6 agents is shown in Fig. 5.1). Let the formation M be an arbitrary circle embedded in $\mathbb{R}^2$, centered at c with the radius r_c, i.e.

$$M = \{x_i \in \mathbb{R}^2 | \, \|x_i - c\| = r_c\}. \tag{5.5}$$

The (smooth) retraction of a certain agent x_i from the tubular neighborhood of the circle thus is equal to the length r_c vector

$$r(x_i) = \frac{r_c}{\|x_i - c\|}(x_i - c) + c. \tag{5.6}$$

For arbitrary two retractions $r(x_i)$ and $r(x_j)$ on the circular formation, employing the canonical (Lie) group isomorphism

$$\begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix} \mapsto \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{5.7}$$

between the circle and the rotation group SO(2), the scaled velocity vector of the geodesic joining $r(x_i)$ and $r(x_j)$ can be computed by

$$-r_c d(x_i, x_j) V_{ij} = \log\left(\frac{1}{\|x_i - c\| \|x_j - c\|} \begin{bmatrix} x_i \cdot x_j & x_i \cdot \Omega x_j \\ x_j \cdot \Omega x_i & x_i \cdot x_j \end{bmatrix} \right) \frac{(x_i - c)}{\|x_i - c\|}, \tag{5.8}$$

where Ω denotes the infinitesimal rotation $\left(\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right)$.

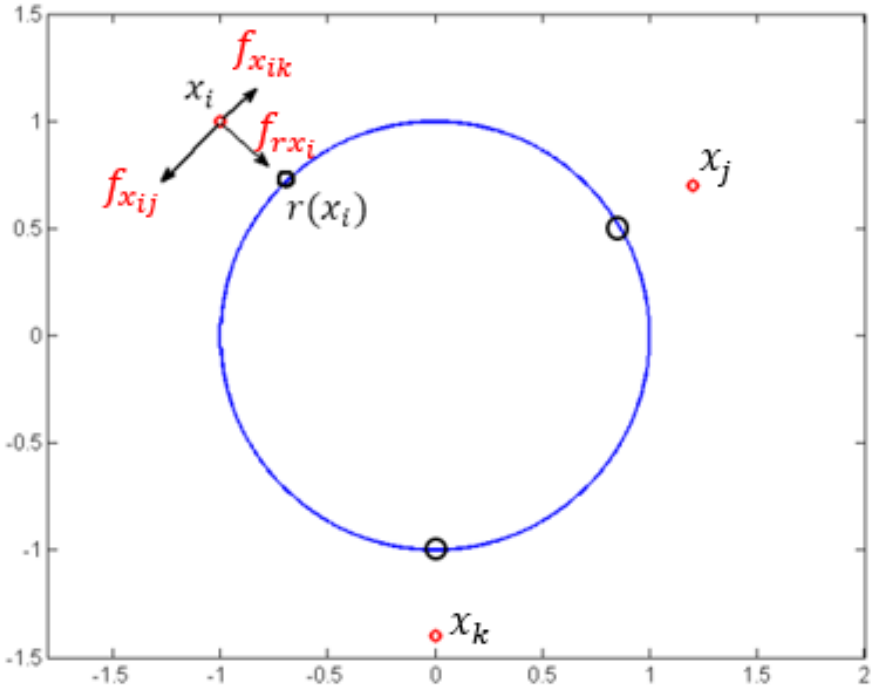


Figure 5.2: An illustration on a group of 3 agents to converge to a target circle.

Therefore, to form a prescribed circle shape, the control Eq. (5.4) is finally given by

$$\dot{x}_i = \left(\frac{r_c - \|x_i - c\|}{\|x_i - c\|}\right)(x_i - c) + \sum_{j=1}^{n} \frac{W_{ij}}{r_c \|x_i - c\|} \cdot (\log \left(\frac{1}{\|x_i - c\|\|x_j - c\|}\begin{bmatrix} x_i \cdot x_j & x_i \cdot \Omega x_j \\ x_j \cdot \Omega x_i & x_i \cdot x_j \end{bmatrix}\right)^{-1} (x_i - c). \tag{5.9}$$

The computed $\dot{x}_i$, is employed as the terminal velocity reference in the MPC method that controls the multirotor UAV (we will introduce the details in latter sections) at each time step.

For the sake of an intuitive understanding, we hereby give an illustration on the case of a group of 3 agents to generate a circular formation, as depicted in 5.2. We analyze the differential equation w.r.t. the agent x_i at the certain moment, where we can divide the motion trend of this agent into 3 parts, i.e. f_{rx_i}, $f_{x_{ij}}$, and $f_{x_{ik}}$.

From the illustration, we can simply consider f_{rx_i} as the "attractive trend" towards x_i from its retraction $r(x_i)$ on the submanifold, while $f_{x_{ij}}$ and $f_{x_{ik}}$ can be regarded as the "repulsive trends" onto x_i generated from its neighbors x_j and x_k, respectively. Note that the "repulsive trends" are always onto the tangent space of the "attractive trend" that leads the agent to the convergent retraction position $r(x_i)$. The sum of the 3 "trends" can be described as $\dot{x}_i$ in Eq. (5.9).

5.2.2 $\mathbb{R}^2$ Formation: The Triangle

Apart from the arbitrary circle shape case from the foregoing section, our algorithm fits for rather arbitrary polygonal formations, e.g. the triangle. Unlike a circular shape whose retraction can be represented explicitly, we compute the retraction point $r(x_i)$ on a triangular shape by the following procedure:

1. Locate an edge on the triangle that is closest to the position x_i;
2. Determine the shortest distance between x_i and any arbitrary point on the closest edge found;
3. Set the point with the shortest distance to x_i as the retraction $r(x_i)$.

The geodesics $d(x_i, x_j)$ and their velocity vectors V_{ij} between two agents depend on their retractions, namely $r(x_i)$ and $r(x_j)$.

If $r(x_i)$ and $r(x_j)$ are located on the same edge of the triangle,

$$d(x_i, x_j) = \|r(x_j) - r(x_i)\|, \tag{5.10a}$$

$$V_{ij} = -\frac{r(x_j) - r(x_i)}{\|r(x_j) - r(x_i)\|}. \tag{5.10b}$$

Meanwhile, if $r(x_i)$ and $r(x_j)$ are located on two edges that share a vertex V_s,

$$d(x_i, x_j) = \|r(x_j) - V_s\| + \|V_s - r(x_i)\|, \tag{5.10c}$$

$$V_{ij} = -\frac{V_s - r(x_i)}{\|V_s - r(x_i)\|}. \tag{5.10d}$$

By substituting the Eq. (5.10) into Eq. (5.4), the differential equation that represents the agent motion during the formation period can be computed. The resulting $\dot{x}_i$ of each agent will be passed into our proposed MPC controller respectively at every time step.

Similarly for an intuitive understanding, we give a second illustration on the case of a group of 3 agents to generate a triangular formation, as depicted in 5.3. Still, we analyze the differential equation w.r.t. the agent x_i at the certain moment, where we can divide the motion trend of this agent into 3 parts, i.e. f_{rx_i}, $f_{x_{ij}}$, and $f_{x_{ik}}$.

From the illustration, we still consider f_{rx_i} as the "attractive trend" towards x_i from its retraction $r(x_i)$ on the submanifold, while $f_{x_{ij}}$ and $f_{x_{ik}}$ can be regarded as the "repulsive trends" onto x_i generated from its neighbors x_j and x_k, respectively.

Differently from the circular case, we cannot always find the "repulsive trends" onto the tangent space of the "attractive trend" that leads the agent to the convergent retraction position $r(x_i)$ in polygonal cases, since an arbitrary polygon is not a continuous geometry in submanifold. Hence, the "repulsive trends" are defined in parallel to the edge where $r(x_i)$ is located at the certain moment. Similarly, the sum of the 3 "trends" is still described as $\dot{x}_i$ in Eq. (5.10).

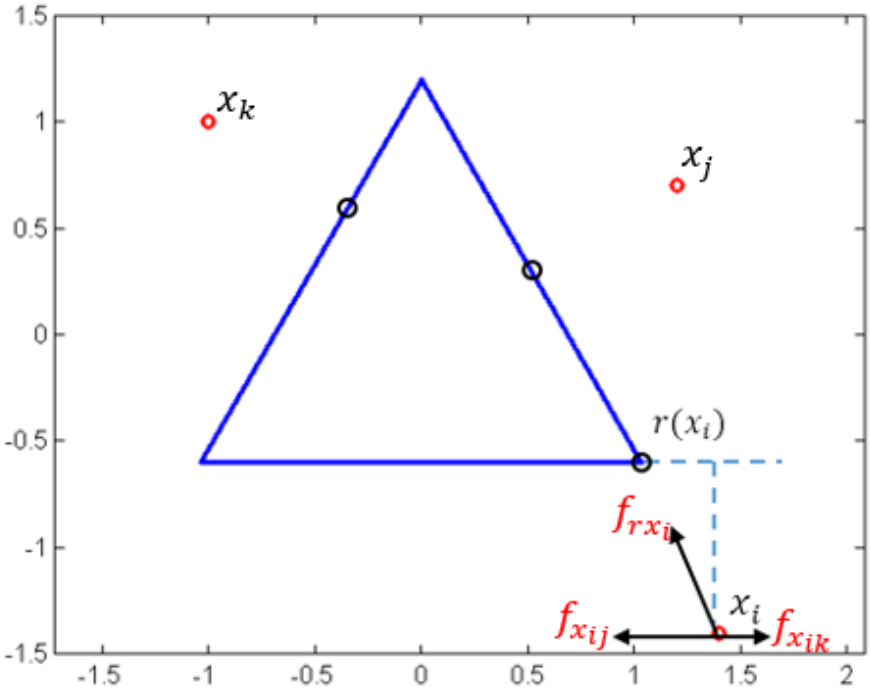


Figure 5.3: An illustration on a group of 3 agents to converge to a target triangle.

5.2.3 $\mathbb{R}^2$ Formation: The Rectangle

The formation of a rectangular or other polygonal shape can be implemented through a similar algorithm as what was shown for the triangle.

For a rectangle, the retraction $r(x_i)$ can first be computed by finding the edge that is closest to x_i and solving a minimization problem, same as in Sec. 5.2.2. The geodesics, arc lengths, and velocity vectors can be computed based on the relative distance between $r(x_i)$ and $r(x_j)$.

If $r(x_i)$ and $r(x_j)$ are located on the same edge or if $r(x_i)$ and $r(x_j)$ are located on two neighboring edges that share a vertex V_s, we employ (5.10).

Otherwise, if $r(x_i)$ and $r(x_j)$ are located on two edges with no shared vertex but linked by the edge whose vertices are V_{li} and V_{lj},

$$d(x_i,x_j) = \|r(x_j) - V_{lj}\| + \|V_{lj} - V_{li}\| + \|V_{li} - r(x_i)\|, \tag{5.11a}$$

$$V_{ij} = -\frac{V_{li} - r(x_i)}{\|V_{li} - r(x_i)\|}. \tag{5.11b}$$

Finally, the differential equation of motion can be computed by substituting Eq. (5.11) into Eq. (5.4). The output of Eq.(5.4), namely the computed velocity vector $\dot{x}_i$ of an agent in the group, will be employed as the terminal velocity reference in a MPC method to control the multirotor UAVs.

5.2.4 $\mathbb{R}^2$ Formation: The Convex Polygon

The two cases above from the foregoing sections, namely the triangular and rectangular formations, can be described as rather arbitrary polygonal formations. In this section, we demonstrate a general expression for this kind of formation. Unlike a circular shape whose retraction can be represented explicitly, we compute the retraction point $r(x_i)$ on a target polygonal shape by the following algorithm:

1. Locate an edge on a polygon with n vertice, which is closest to the position x_i;
2. Calculate the shortest distance, between x_i and any arbitrary point on the closest edge found;
3. Set the point with the shortest distance to x_i as the retraction $r(x_i)$.

Considering that the geodesic between two agents $d(x_i,x_j)$, and their relative velocity vectors V_{ij} depend on their retractions, namely $r(x_i)$ and $r(x_j)$, then $d(x_i,x_j)$ and V_{ij} can be computed via the general formulation displayed in the following algorithm.

1. For a target polygon with n vertice ($n \geq 3$)
2. If $r(x_i)$ and $r(x_j)$ are located on the same edge of polygon, then
3. $d(x_i,x_j) = \|r(x_j) - r(x_i)\|$,
4. $V_{ij} = -\frac{r(x_j)-r(x_i)}{\|r(x_j)-r(x_i)\|}$;
5. Else if $r(x_i)$ and $r(x_j)$ are located on two neighboring edges that share a vertex V_s
6. $d(x_i,x_j) = \|r(x_j) - V_s\| + \|V_s - r(x_i)\|$,
7. $V_{ij} = -\frac{V_s-r(x_i)}{\|V_s-r(x_i)\|}$;
8. For $n \geq 4$
9. If $r(x_i)$ and $r(x_j)$ are located on two edges with no shared vertex but linked by the edge whose vertice are $V_{l,i}$ and $V_{l,j}$, then
10. $d(x_i,x_j) = \|r(x_j) - V_{l,j}\| + \|V_{l,j} - V_{l,i}\| + \|V_{l,i} - r(x_i)\|$,
11. $V_{ij} = -\frac{V_{l,i}-r(x_i)}{\|V_{l,i}-r(x_i)\|}$;
12. For $n \geq 5$
13. If $r(x_i)$ and $r(x_j)$ are located on two edges with no shared vertex, and the bridge edges among these two edges are with the vertice $V_{l,i}$, $V_{l,j}$ and $V_{l,k_1}, V_{l,k_2}, ...V_{l,k_n}$, where $(1 \leq k_n \leq n-4)$, then

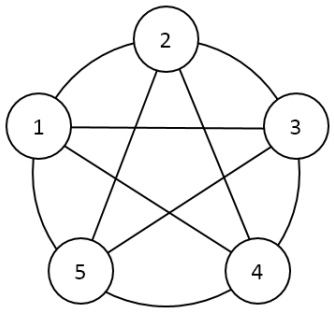


Figure 5.4: Complete graph K_5 for a group of 5 agents.

14. $d(x_i,x_j) = \|r(x_j) - V_{l,j}\| + \|V_{l,j} - V_{l,k_n}\| + \|V_{l,k_n} - V_{l,k_n-1}\| + ... + \|V_{l,k_2} - V_{l,k_1}\| + \|V_{l,k_1} - V_{l,i}\| + \|V_{l,i} - r(x_i)\|$,

15. $V_{ij} = -\frac{V_{l,i} - r(x_i)}{\|V_{l,i} - r(x_i)\|}$;

The final step to employ the computed velocity as the terminal velocity reference in a MPC method onto multirotor UAVs is kept identical as the expression in previous sections.

5.2.5 $\mathbb{R}^3$ Formation: The Sphere

Our distributed control algorithm is not only restricted to formations in $\mathbb{R}^2$. A group of multirotor UAVs is, e.g., able to form a spherical shape centered at arbitrary points in $\mathbb{R}^3$ from arbitrary initial positions. The communication among the group of agents in $\mathbb{R}^3$ is based on the complete graph (an example of 5 UAVs is depicted in Fig. 5.4).

Most considerations from the circular formation in $\mathbb{R}^2$ remain correct. Similarly as in (5.7), we employ the (Lie) group isomorphism

$$\begin{bmatrix} \sin(\beta)\cos(\alpha) \\ \sin(\beta)\sin(\alpha) \\ \cos(\beta) \end{bmatrix} \mapsto \begin{bmatrix} \cos(\beta)\cos(\alpha) & -\sin(\alpha) & \sin(\beta)\cos(\alpha) \\ \cos(\beta)\sin(\alpha) & \cos(\alpha) & \sin(\beta)\sin(\alpha) \\ -\sin(\beta) & 0 & \cos(\alpha) \end{bmatrix} \tag{5.12}$$

from the sphere onto $\mathrm{SO}(3)$. We denote the represenation (5.12) of a certain retracted x_i as a member of $\mathrm{SO}(3)$ by R_i. The retraction of a certain agent $x_i \in \mathbb{R}^3$ from the tubular neighborhood of the sphere, whose center locates at an arbitrary point c in $\mathbb{R}^3$ with the radius r_c, is still given by

$$r(x_i) = \frac{r_c}{\|x_i - c\|}(x_i - c) + c. \tag{5.13}$$

Thus we can apply the logarithmic map $\log : \mathrm{SO}(3) \to \mathfrak{so}(3)$ to $R_j^\top R_i$, in order to find the velocity vector of the geodesic joining R_i and R_j. Since in $\mathrm{SO}(3)$, the tangent space

is no longer 1D, we use the identity $2d(x_i,x_j)^2 = -\mathrm{tr}(\log(R_j^\top R_i)^2)$ and can finally obtain the differential equation of the velocity vector subject to an arbitrary sphere in $\mathbb{R}^3$

$$\begin{aligned}\dot{x}_i = &\left(\frac{r_c - \|x_i - c\|}{\|x_i - c\|}\right)(x_i - c) \\ &+ \sum_{j=1}^{n} \frac{2W_{ij}}{r_c\|x_i - c\|\mathrm{tr}(\log(R_j^\top R_i)^2)} \log(R_j^\top R_i)\,(x_i - c)\,.\end{aligned} \tag{5.14}$$

Similarly as in all formation cases in $\mathbb{R}^2$, we employ the velocity vector computed from Eq. (5.14) as the 3D terminal velocity reference in the MPC-based controller for the multirotor UAVs.

5.3 Model Predictive Control of Multirotors

For the onboard implementation of the formation control algorithm onto a multirotor UAV platform, we propose a framework based on a model predictive control method to control the position and attitude of the multirotors.

Following the algorithm we proposed in the last section, we develop a distributed framework as shown in the right part of Fig. 5.5. For the purpose of robustness and onboard implementation, a linear MPC method plus a nonlinear geometric approach are used hierarchically to regulate the translational motion and rotational motion, respectively. A nonlinear wrench observer is further implemented onboard to estimate the unknown disturbances as external force and torque terms, and pass the estimates to the controller for compensation. The onboard MPC block on each multirotor (without considering the formation algorithm) is similar (with minor modifications) to the robust MPC framework in Chapter 4 that combines a linear MPC, a nonlinear attitude controller, and a 6D force and torque observer.

Since we have introduced the technical details of the onboard MPC block in Chapter 4, we here only report the highlights (i.e. the core concepts, the implemented control laws, etc.) in the following technical sections. The overall control system is optimized to fit the real-time computations into low-power onboard computers. We will show the experimental verifications later in Sec. 5.4.

5.3.1 Dynamic Model

We still consider a multirotor UAV as a rigid body (as in (2.1)), the dynamic equations of which can be simplified in their Newton-Euler formulation as

$$\dot{x} = v, \tag{5.15a}$$

$$m\dot{v} = -mge_3 + RF + b_F, \tag{5.15b}$$

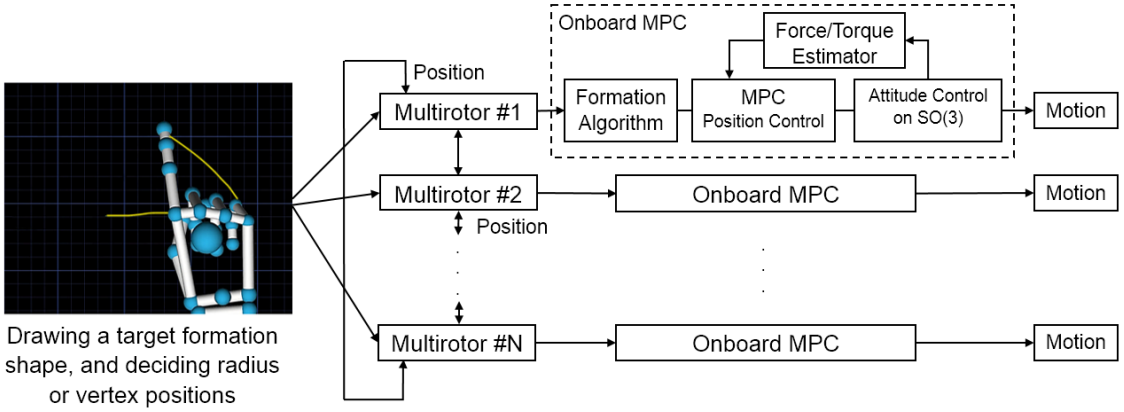


Figure 5.5: The block diagram of a distributed predictive control framework for the balancing formation of a group of multirotors.

$$\dot{R} = RQ(\omega), \tag{5.15c}$$
$$J\dot{\omega} = -\omega \times J\omega + \tau + b_\tau, \tag{5.15d}$$

where m and J denote the mass and the inertial matrix of a multirotor, x denotes the translational position, g is the gravitational acceleration, while $e_3 = (0,0,1)^\top$. The desired forces and moments to be controlled are denoted by $F \in \mathbb{R}^3$ and $\tau = (\tau_x, \tau_y, \tau_z)^\top$, while b_F, considered in the form of an external force in the inertial frame, represents the additional forces due to external disturbances and unmodeled dynamics, and b_τ is defined as the external torque in the body frame.

The rotation matrix R, which is an element of the special orthogonal group SO(3)$=\{R \in \mathbb{R}^{3\times3} | R^{-1} = R^\top,\ \det R = 1\}$, represents the attitude of the multirotor w.r.t. the inertial frame.

5.3.2 Position Control

Based on the dynamic model (5.15), whose translational and rotational motions are decoupled, we are able to build a discrete-time model of the translational motion of a multirotor under no disturbance assumption. The discrete-time model sets the acceleration in 3D as the inputs for an optimal control problem (OCP) describing the multirotor motion. The state $\bar{\xi}$ consists of the position and velocity along three axes.

In summary, the discrete-time state $\bar{\xi}$ and control input $\bar{u}$ at the k^{th} time step with a

sampling time Δt are given by

$$\bar{\xi}[k] = \begin{bmatrix} x(k\Delta t) \\ \dot{x}(k\Delta t) \\ y(k\Delta t) \\ \dot{y}(k\Delta t) \\ z(k\Delta t) \\ \dot{z}(k\Delta t) \end{bmatrix}, \bar{u}[k] = \begin{bmatrix} \ddot{x}(k\Delta t) \\ \ddot{y}(k\Delta t) \\ \ddot{z}(k\Delta t) \end{bmatrix}, \tag{5.16a}$$

$$\bar{\xi}[k+1] = A\bar{\xi}[k] + B(\bar{u}[k] - ge_3), \tag{5.16b}$$

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.16c}$$

$$B = \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 & 0 \\ \Delta t & 0 & 0 \\ 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & \Delta t \end{bmatrix}. \tag{5.16d}$$

The OCP can be described with a quadratic cost function

$$\begin{aligned} \min_{d_{\bar{\xi}}, \bar{u}} J = \sum_{k=0}^{N} & (\bar{u}[k]^\top P \bar{u}[k] + d_{\bar{\xi}}[k]^\top L_\mathrm{s} d_{\bar{\xi}}[k]) \\ & + d_{\bar{\xi}}[N+1]^\top L_\mathrm{t} d_{\bar{\xi}}[N+1], \end{aligned} \tag{5.17}$$

subject to the dynamics explained above and corresponding boundary conditions

$$a_{\min} \leq \bar{u}[k] \leq a_{\max}, \tag{5.18a}$$

$$\xi_{\min} \leq \bar{\xi}[k] \leq \xi_{\max}, \tag{5.18b}$$

where N represents the length of the receding horizon, $d_{\bar{\xi}}[k]$ denotes the error between state and reference, and P, L_s and L_t are the weights of input cost, stage state cost and terminal cost, respectively. The described convex OCP is solved via CVXGen Mattingley *et al.* (2011) using the interior point method. THe vectors $a_{\min}, a_{\max} \in \mathbb{R}^3$ represent the lower and upper constraints of multirotor accelerations along three axes. Similarly, $\xi_{\min}, \xi_{\max} \in \mathbb{R}^6$ denote the position and velocity boundaries. The computed first step control input $\bar{u}[0]$ is used in the control law at each time step.

The OCP is designed with adaptive cost weight settings, so that the controller can

predict a locally optimal trajectory and generate a 3D vector as the control input for a given path or simply a waypoint. For a waypoint navigation task, the weight of the terminal cost is automatically set dominant, while the weight of all stage state costs is set to zero. In contrast, for a path following task, the weight of the terminal cost is set to zero.

For the formation tasks of multiple multirotor UAVs, we keep the weight of the stage cost at zero. Differently from a waypoint task, we consider the output of (5.4), $\dot{x}$, as the terminal velocity reference in the OCP. Therefore, the translational control based on MPC of a multirotor becomes a "velocity control" method instead of a "position control" method.

Assuming that R in (5.15b) equals R_{ref}, the control law for the multirotor position is given by

$$R_{\text{ref}}F \triangleq m\bar{u}[0] := F'. \tag{5.19}$$

Considering that R_{ref} in (5.19) is orthonormal and only the third entry of F, i.e., the thrust T, is nonzero, one must choose $T = \|F'\|$ in order to suffice (5.19). Therefore, the third column of $R_{\text{ref}} = \begin{bmatrix} r^1_{\text{ref}} & r^2_{\text{ref}} & r^3_{\text{ref}} \end{bmatrix}$ can be solved from

$$r^3_{\text{ref}} = \frac{F'}{\|F'\|}, \tag{5.20}$$

and the other two columns of R_{ref} can be filled orthonormally, for instance by a Gram-Schmidt process with candidates r^3_{ref} from the previous equation and r^1, r^2 from the actual rotation $R = \begin{bmatrix} r^1 & r^2 & r^3 \end{bmatrix}$. To reduce the redundancy, we refer the readers to 4.2.2 for the derivation via the Gram-Schmidt process.

5.3.3 Attitude Control

In this section, we briefly introduce the nonlinear control method for the tracking of the attitude R of a multirotor. For the sake of brevity, we do not report the technical details in this chapter. The full derivation and the proof of asymptotic stability of this attitude controller based on the global output regulation can be found in 3.2.2. The tracking error is given by $E = RR^{\top}_{\text{ref}}$, where R corresponds to the rotation matrix describing the current attitude, and R_{ref}, computed via the approach introduced in Section 5.3.2, represents the desired attitude.

The goal of the attitude controller is to regulate the output of the attitude $(E, e_\omega) \to (I, 0)$ for $t \to \infty$, which implies $R \to R_{\text{ref}}$ and $Q(\omega) \to Q(\omega_{\text{d}})$ for $t \to \infty$.

We employ a backstepping-like method for attitude regulation. Following the deriva-

tion in 3.2.2, the control law for the multirotor UAV's attitude is given by

$$\begin{aligned}\tau_{\mathrm{d}} = JQ^{-1}\Bigg(-k_{\omega}e_{\omega} - \frac{k_{\mathrm{rot}}}{2}\Big[\dot{R}_{\mathrm{ref}}^{\top}R + R_{\mathrm{ref}}^{\top}\dot{R} - \dot{R}^{\top}R_{\mathrm{ref}} \\ -R^{\top}\dot{R}_{\mathrm{ref}}\Big] - \ddot{R}_{\mathrm{ref}}^{\top}R_{\mathrm{ref}} - \dot{R}_{\mathrm{ref}}^{\top}\dot{R}_{\mathrm{ref}} - 2R_{\mathrm{ref}}^{\top}R\Bigg) + \omega \times J\omega,\end{aligned} \tag{5.21}$$

where k_{ω} is a positive gain for the tracking error of the multirotor angular velocity. The derivative $\dot{R}$ can be obtained by introducing the current attitude R into (5.15c), while R_{ref} inherits from the results of the Gram-Schmidt process. We choose the value $\omega_{1,\mathrm{ref}}, \omega_{2,\mathrm{ref}} = 0$ and $\omega_{3,\mathrm{ref}} = \psi_{\mathrm{ref}} - \psi_{\mathrm{current}}$ for the reference and current yaw attitude ψ, and $\dot{R}_{\mathrm{ref}}$, hence, can be solved via Eq. (5.15c).

5.3.4 Disturbance Estimation

In this section, we extend the control approach discussed in the previous subsections with the inclusion of a nonlinear force/torque external wrench observer using the Fault Detection and Isolation (FDI) method Takakura *et al.* (1989). The observer estimates all the system offsets, parameter uncertainties and external disturbances, and passes the estimates to the controller for compensation. The detailed introduction and the experimental validation on a multirotor UAV can be found in our previous work Liu *et al.* (2017).

The technical details have been fully given in 3.4, the stability proof has been shown in 3.4.1, and the implementation of the nonlinear observer with MPC has been presented in 4.5. Therefore, we here only show the control laws of the complete control-observation system.

Following the notations in Chapter 3 and 4, we can write the estimated external wrench as

$$\widehat{\Lambda_{\mathrm{ext}}} = \begin{bmatrix}\widehat{b_F} \\ \widehat{b_{\tau}}\end{bmatrix} = \boldsymbol{r}, \tag{5.22}$$

where $\wedge$ indicates the estimated value of the variable.

With the inclusion of the external wrench estimate, the position control law in (5.19) is given by

$$RF \triangleq -\widehat{b}_F + F' = -\widehat{b}_F + m\bar{u}[0], \tag{5.23}$$

while the attitude control law in (5.21) becomes

$$\begin{aligned}\tau = -\widehat{b}_{\tau} + JQ^{-1}\Bigg(-K_{\omega}e_{\omega} - \frac{K_{\mathrm{rot}}}{2}\Big[\dot{R}_{\mathrm{ref}}^{\top}R + R_{\mathrm{ref}}^{\top}\dot{R} - \dot{R}^{\top}R_{\mathrm{ref}} - R^{\top}\dot{R}_{\mathrm{ref}}\Big] \\ -\ddot{R}_{\mathrm{ref}}^{\top}R_{\mathrm{ref}} - \dot{R}_{\mathrm{ref}}^{\top}\dot{R}_{\mathrm{ref}} - 2R_{\mathrm{ref}}^{\top}R\Bigg) + \omega \times J\omega.\end{aligned} \tag{5.24}$$

The Eq.(5.23) and Eq.(5.24), plus the final mapping from desired thrust/torque to mo-

tor speeds, provide the multirotor with a robust model predictive control method for high-speed waypoint navigation, path following, and formation tasks.

5.4 Verification

We carried out two stages of experiments, with a connection of human-swarm-interaction via the finger tracking device, to validate the performance of the proposed distributed control approach, including simulations and real-world experiments.

At the simulation stage, we validated the presented formation algorithm combined with the robust MPC method via ROS/Gazebo physical simulations of multiple (up to 6) multirotor UAV models with artificial parametric mismatch and disturbances. Since the computational cost of MPC is quite critical for the multirotor UAVs with limited computational capability in actual experiments, we have set up hardware-in-the-loop (HIL) simulations using low-power onboard computational units, which ensure that our proposed control approach can be executed properly at 50Hz within real-world experiments.

At the real-world experiment stage, we have implemented the complete formation control framework onto up to 4 quadrotor UAVs with onboard computational units (same as those used in HIL simulations) and carried out a series of experiments on triangular, rectangular and circular formation.

We tested our proposed approach via circular, triangular and rectangular formations in a 2D environment, as well as via a spherical formation in a 3D environment. In this section, we will introduce the detailed configuration of our experimental and HIL simulation platform, and then demonstrate the results for various shapes, respectively. For the sake of brevity, we hereby mainly report the results from real-world experiments if we have tested one formation in both experimental stages. We additionally report several selected results of the HIL simulation where more than 4 multirotor UAVs are required.

5.4.1 Human-Swarm Interaction

Since our proposed formation algorithm relies on a prescribed shape instead of relative distance of targets, which enables a human operator to decide a target formation by simply “drawing” a geometrical shape. To realize the interaction between human and multiple aerial robots, a Leap Motion 3D gesture device has been implemented into our formation control framework, so that the formation shapes for the multirotor UAVs can be generated via finger/palm motions (shown in Fig. 5.6). In order to command the UAVs to generate the required formation in a safe and practical way, an end-user (human) needs to draw the formation shape, e.g. circle, triangle, rectangle, sphere, etc., and also provide auxiliary information, i.e. the geometric center and the radius (when giving a circle shape) or the position of vertices (when providing a polygonal shape).

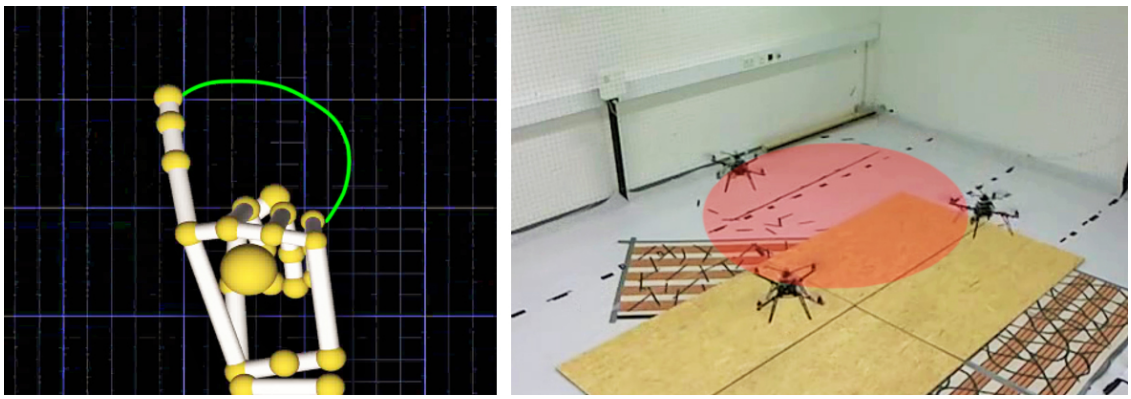


Figure 5.6: Screenshot from the progress of prescribing a formation shape via human-robot-interaction. Left: The user draws a circular trajectory via the Leap Motion device. Right: The trajectory of finger motions is then fitted and matched as a circular shape and mapped as the target formation for the group of multirotor UAVs.

More specifically, the human-swarm interaction between the operator and a group of UAVs can be achieved by the following procedures:

1. The "drawing" from the operator through the figure tracking device is first collected as a set of trajectory;

2. The trajectory is passed into a customized curve-fitting program, where the drawing from Leap Motion will be filtered and described in the form of one 2^{nd}-order polynomial, or a cluster of several 1^{st}-order polynomials;

3. The polynomials are paired with the geometric shapes: we naively map a 2^{nd}-order polynomial into a circle, a group of 3 1^{st}-order polynomials including one with near-zero slope as triangle, and a group of 4 1^{st}-order polynomials including two with near-zero slopes as rectangle, etc.;

4. The fitted shape is mapped to the experimental area with proper 3D positions and implemented onto the group of multirotor UAVs for the target formation.

5.4.2 Experimental Configuration

Up to 4 quadrotor UAVs with 10inch propellers have been set as the platform for a series of the real-world experiments. Each quadrotor is equipped with an Odroid-XU4 board with Cortex^{TM}-A15 CPUs. Thus all the computations are carried out onboard during the experiments. We also have carried out several HIL simulations using the Odroid boards in the same model as implemented on the quadrotors. The complete control approach we proposed in previous sections has been implemented within a ROS-based framework,

Figure 5.7: Screenshot on the real-world formation experiment. A group of 4 multirotor UAVs were commanded to generate a prescribed rectangular formation.

which continuously passes the required motor speed messages at each time step in the feedback system.

In the cases of real-world, indoor experiments (e.g. in Fig. 5.7), the position data of a multirotor are passed via the external motion capture system (e.g. Vicon) at 100Hz, while the attitude data are filtered using an extended Kalman filter from the raw angular velocity data collected by a low-cost onboard Inertial Measurement Unit (IMU). The linear acceleration data are also collected by the IMU and then filtered into linear velocity data at each time step. The motor speeds are computed and passed to the open-loop brushless motor controllers at the frequency of the proposed attitude controller and finally to the embedded motors.

Meanwhile, in HIL simulations, the motor speed messages obtained from the proposed controller are passed into the built multirotor models in Gazebo. Differently from the real experiments, we have employed the hexarotor models in the virtual tests, thus the desired thrust/torque computed from Eq.(5.23) and Eq.(5.24) are mapped into 6 motor speed values at each time step. The position data of each multirotor are generated with no noise, since in real experiments we regard the position data from the motion capture system as "ground truth". The IMU data are generated with artificial gaussian noises, drift and initial bias to simulate a practical onboard IMU. In the simulations of multiple hexarotor UAVs, each UAV is able to obtain its own position, (raw) linear acceleration and (raw) angular velocity data, as well as the position data of its two neighbors at 100Hz.

The average computational cost of our robust MPC approach is approximate 0.0149s. Therefore, each Odroid board is set to execute the MPC position controller at 50Hz, with a receding horizon of 15 steps. The attitude controller computes the motor speed commands at 100Hz with the usage of collected and filtered data. The decision on the present formation is also made at a 50Hz frequency. All the control parameters on each multiro-

tor are kept identical and fixed with no change during the whole series of experiments, i.e. the real-world experiments and simulations.

In the real-world experiments, the disturbances due to the hardware, the delay on wireless communication, and the external disturbances, e.g. winds generated from the group of multirotors and from the ventilation system, highly affect the control performance. The resulting trajectories, therefore, would not be as perfect as the results in simulations. However, the convergence of our proposed approach has still been experimentally validated.

5.4.3 Triangular Formation

In the first case, several quadrotor UAVs were commanded to take off from random initial positions and generate a 2D triangular formation at 1m height. We employed 3 quadrotors in the experiments for an intuitive visualization. The switch of experimental tasks on each UAV was operated together via a joystick. Once the formation task was launched, the UAVs started to decide upon their motions online and subsequently moved using the distributed MPC approach that we have presented in previous sections.

The shape of the UAV formation was first decided before the formation task started. In this case, the end-user provided a triangular shape through the finger movement detected via the VR device and then defined the position of vertices. During the experiment, a triangle with vertices of $(0, 1.2, 1)^\top$, $(\frac{3\sqrt{3}}{5}, -0.6, 1)^\top$, and $(-\frac{3\sqrt{3}}{5}, -0.6, 1)^\top$, w.r.t. the prescribed target center, was generated. Despite that our formation control algorithm is theoretically convergent for an arbitrary triangle, we set the formation shape not overly blunt or sharp so that the collisions among multirotors can be avoided.

We carried out 10 trials on the triangular formation, where the initial conditions of multirotors were randomly set. The formation approach successfully converged in all the trials during the experiments. For the sake of brevity, we hereby only illustrate one trial of the triangular formation in Fig. 5.8. The multirotors were commanded to start the formation task from the initial condition of one line array. The UAVs converged to three equilibrium positions (the red bubbles in Fig. 5.8) on the edges of the prescribed triangle whose horizontal projections are the calculated retraction points on the edge of the triangle shape. The velocity reference of each multirotor passed to the MPC was computed negligible at the equilibrium position, thus each multirotor kept hovering.

In the real-world experiments, the UAVs sometimes oscillated due to the external disturbance and the loss of measurements due to the delay of wireless communication. There were also slight drifts during the hovering (within a error ranging in $\pm$5cm). Largely, the 3 multirotors are seen to retract into the target triangle and converge to their equilibrium position. Compared to the auxiliary triangle (solid black lines), the 3 multirotors are seen to retract into a tilted triangle, since in our formation approach the multirotors are commanded to the configuration with equal geodesics on the submanifold rather than to reach the vertice.

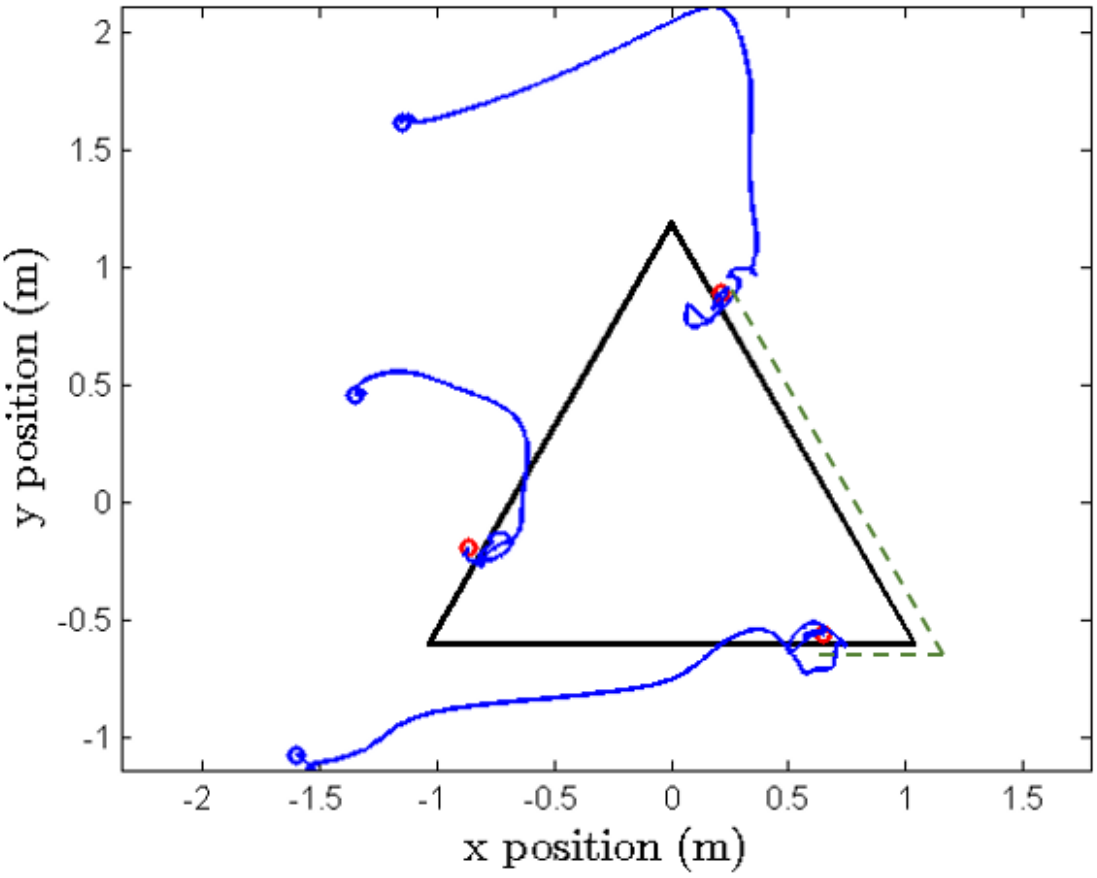


Figure 5.8: Real-robot-experiment result: mutlirotor trajectories of a triangular formation. The assembly of solid black lines represent the prescribed triangle shape. The blue bubbles show the initial position where the multirotors hovered before the start of formation task. The solid blue lines represent the flight trajectories of each multirotor. The red bubbles are the stable positions of UAVs in the target triangular formation. The green dashed line visualizes the geodesic between two multirotors.

In addition, our proposed approach has no limits that the number of a group of robots should be equal to the number of edges on the target polygon to form. An intuitive instance can be seen in Fig. 5.9. In this case, 5 multirotors were commanded to form a triangle from a line initial condition. One of the multirotors converged to the vertex of the target triangle while other 4 UAVs at the end converged to somewhere at the edges, where each agent kept the equal-geodesic condition.

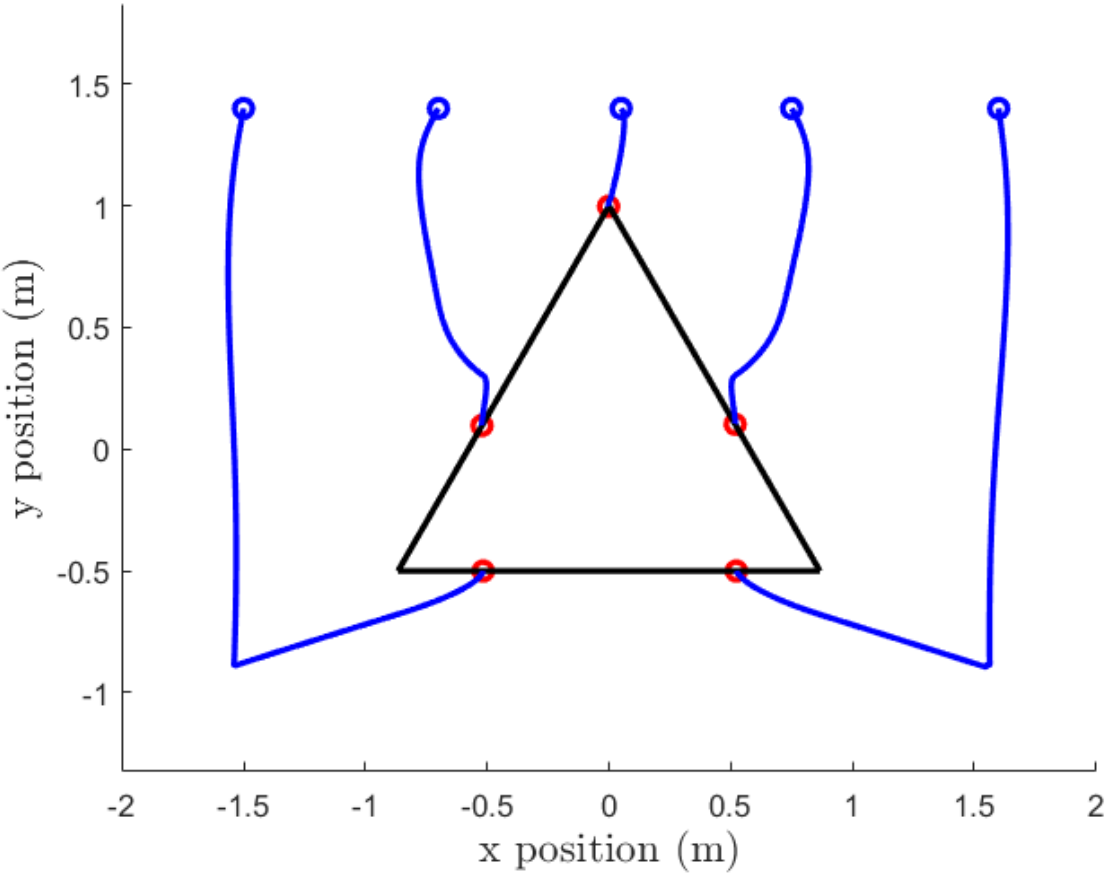


Figure 5.9: Simulation result: mutlirotor trajectories of a triangular formation with 5 UAVs. The assembly of solid black lines represent the prescribed triangle shape. The blue bubbles show the initial position where the multirotors hovered before the start of formation task. The red bubbles are the stable positions of UAVs in the target triangular formation. The solid blue lines represent the flight trajectories of each multirotor.

5.4.4 Rectangular Formation

In the second case, we employed 4 hexarotors to form a rectangle shape. The resulting trajectories of UAVs in a simulation can be found in Fig. 5.10. The shape is decided as a rectangle with vertices of $(-1,-1,1)^\top$, $(-1,1,1)^\top$, $(1,1,1)^\top$, and $(1,-1,1)^\top$ w.r.t. the prescribed target center. In this trial, the multirotors started the formation task from the hovering points $(-1.5,-0.6,1)^\top$, $(-1.5,1,1)^\top$, $(1.5,0.6,1)^\top$, and $(1.5,-1,1)^\top$. Each multirotor converged to an equilibrium position whose horizontal projection is a calculated retraction point on one edge of the rectangle and hovered keeping equivalent distances (on the submanifold) with both of its neighbor UAVs.

In the real-robot experiments, we commanded 4 quadrotors to achieve a rectangular formation. Still, 10 trials have been carried out. The resulting trajectories of UAVs in one

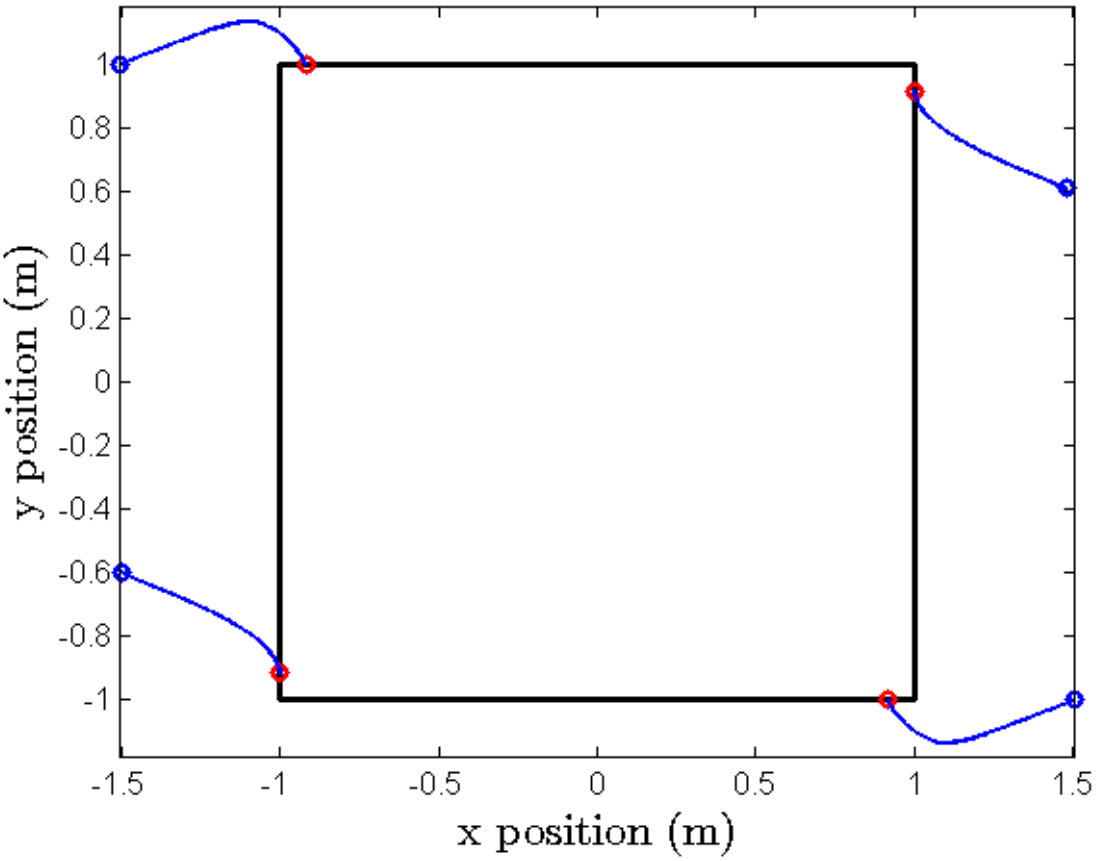


Figure 5.10: Simulation result: mutlirotor trajectories of a rectangular formation. The assembly of solid black lines represent the prescribed rectangle shape. The initial position (blue bubbles) of multirotors are $(-1.5,-0.6,1)^\top$, $(-1.5,1,1)^\top$, $(1.5,0.6,1)^\top$, and $(1.5,-1,1)^\top$, respectively. The solid blue lines represent the flight trajectories. The red bubbles are the positions of UAVs when the formation task ends.

of the trials can be found in Fig. 5.11. The shape is decided as a rectangle with vertices of $(-1,-1,1)^\top$, $(-1,1,1)^\top$, $(1,1,1)^\top$, and $(1,-1,1)^\top$ relative to the shape center. In this trial, the UAVs started the formation task from 3 hovering points on one side of the target rectangle, and the rest from the opposite side. Each multirotor converged to an equilibrium position whose horizontal projection is a calculated retraction point on one edge of the rectangle and hovered with equal distances (on the submanifold) to both of its neighbor UAVs.

Without the auxiliary rectangle (solid black lines), the 4 UAVs are seen to converge to a tilted and smaller rectangular shape, since the euclidian distances between the neighbouring multirotors are shorter than the geodesics on the submanifold.

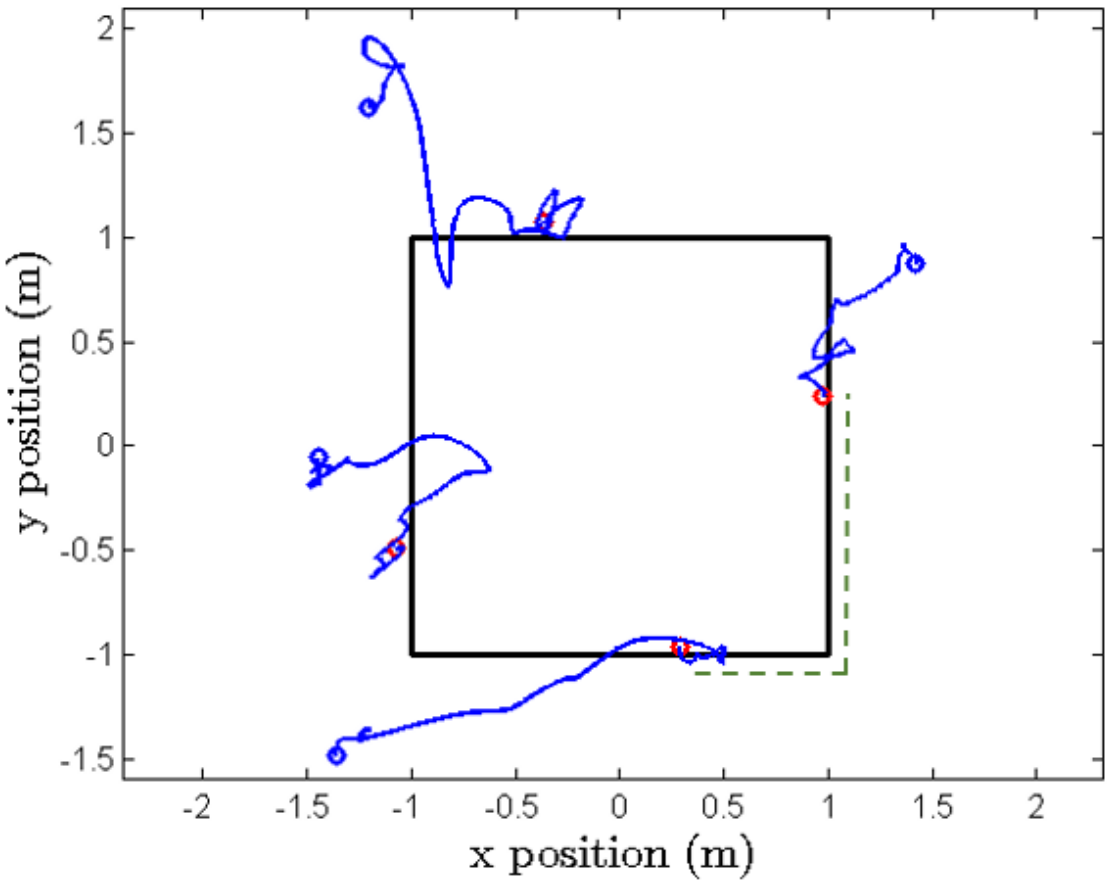


Figure 5.11: Real-robot-experiment result: mutlirotor trajectories of a rectangular formation. The assembly of solid black lines represent the prescribed rectangle shape. The blue bubbles show the initial position where the multirotors hovered before the start of formation task. The solid blue lines represent the flight trajectories of each multirotor. The red bubbles are the stable positions of UAVs in the target rectangular formation. The green dashed line visualizes the geodesic between two multirotors.

5.4.5 Circular Formation

In the third case, 3 quadrotors were employed to form a circle. Similarly as in the cases we presented in the previous sections, the end-user drew a circle and set up the circular center for the group of UAVs. 10 successful trials have been carried out. The resulting trajectories in one circular formation case are displayed in Fig. 5.12.

It can be inferred from the results that the 3 quadrotors enabled to form the prescribed circle in convergence. Thus the quadrotors kept hovering at the arc on the circle in a balanced configuration (equidistant when measured with geodesic arc length).

We hereby study three more simulation cases on circular formation. More specifically, we employed up to 6 onboard computers (with the same configurations as in the real-

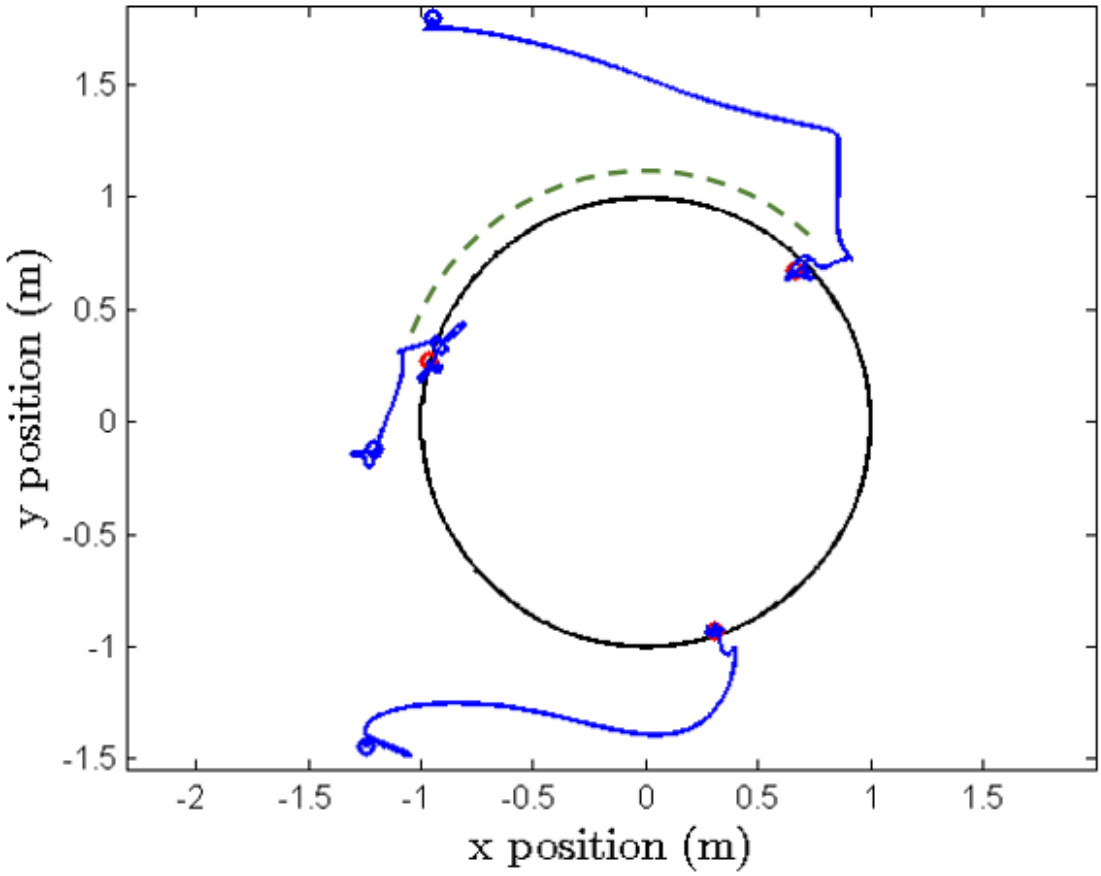


Figure 5.12: Real-robot-experiment result: mutlirotor trajectories of a circular formation. The assembly of solid black lines represent the prescribed circular shape. The blue bubbles show the initial position where the multirotors hovered before the start of formation task. The solid blue lines represent the flight trajectories of each multirotor. The red bubbles are the stable positions of UAVs in the target circular formation. The green dashed line visualizes the geodesic between two multirotors.

world experiments) to simulate up to 6 multirotor UAVs, and commanded them to fly through obstacles in a time-varying circular formation.

In the first simulation trial, 5 multirotors were commanded to hover at their start points $(-2, 1.5, 1)^\top$, $(-1, 1.5, 1)^\top$, $(0, 1.5, 1)^\top$, $(1, 1.5, 1)^\top$, and $(2, 1.5, 1)^\top$, then they were required to form a circle shape (as the solid black lines in Fig. 5.13). The resulting trajectories (solid blue lines) show that the UAVs finally converged to the prescribed circle, though seen as an equidistant pentagon.

In the second simulation, a group of 6 UAVs were commanded to generate a circular formation. Similarly as in the real experiments, the communication between neighboring UAVs is based on the circle graph. The trajectories of multirotors are displayed in

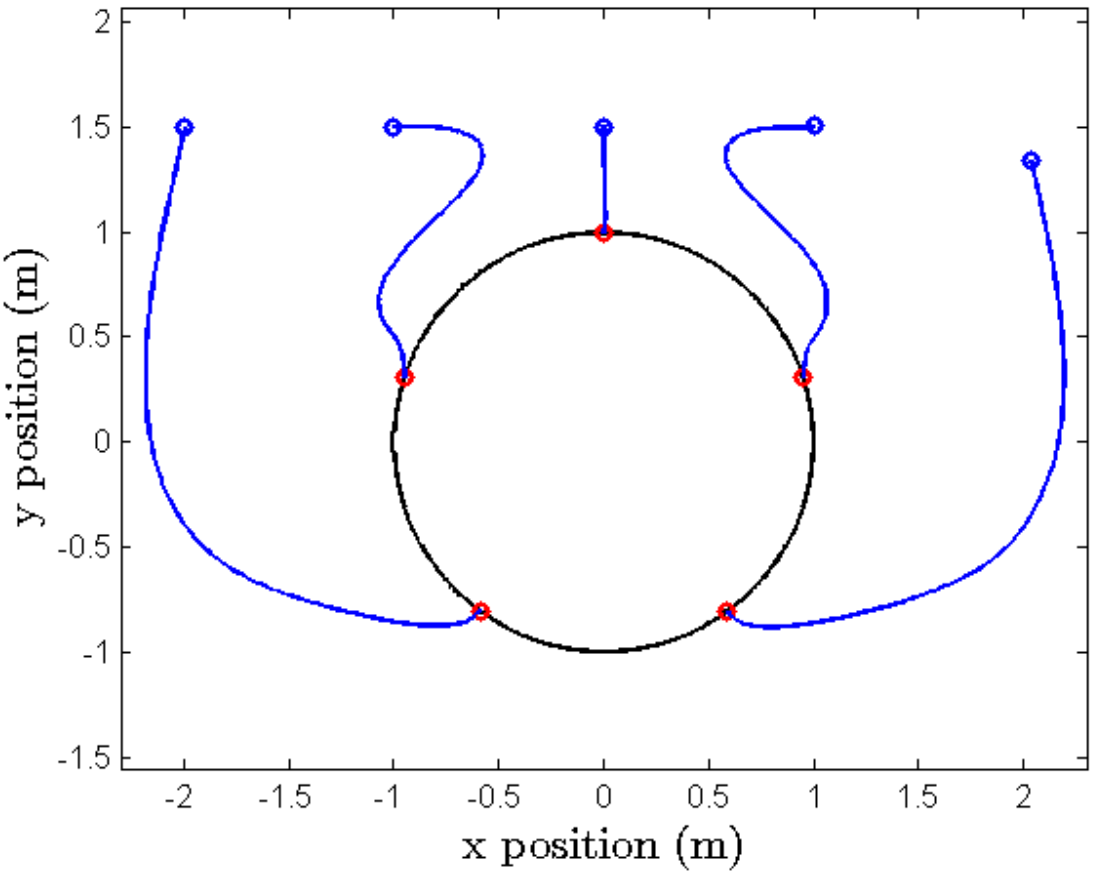


Figure 5.13: Simulation result: mutlirotor trajectories of a circular formation. The assembly of solid black lines represent the prescribed circle shape. The initial position (blue bubbles) of multirotors are $(-2, 1.5, 1)^\top$, $(-1, 1.5, 1)^\top$, $(0, 1.5, 1)^\top$, $(1, 1.5, 1)^\top$, and $(2, 1.5, 1)^\top$, respectively. The solid blue lines represent the flight trajectories. The red bubbles are the positions of UAVs when the formation task ends.

Fig. 5.14.

The multirotors succeeded in converging to a balanced circular shape from arbitrary initial positions.

In additional, by utilizing the characteristic of the time-varying formation shape, we prescribed a trajectory of the center and the radius of the shape (or vertices for polygonal shapes). In this case (shown in Fig. 5.15), the 3 multirotors were first commanded to retract into a circle of 1.3m radius and then were required to fly (in a balanced, circular configuration) through a 2m-wide gap between two gateposts. Since the circle of 1.3m radius is too large for the gatepost obstacles, we prescribed a time-varying (virtual) target that shrinks into a circle of 0.5m radius when the obstacles are approaching, and expands back to a circle of 1.2m radius after the multirotors fly through the obstacles (as shown

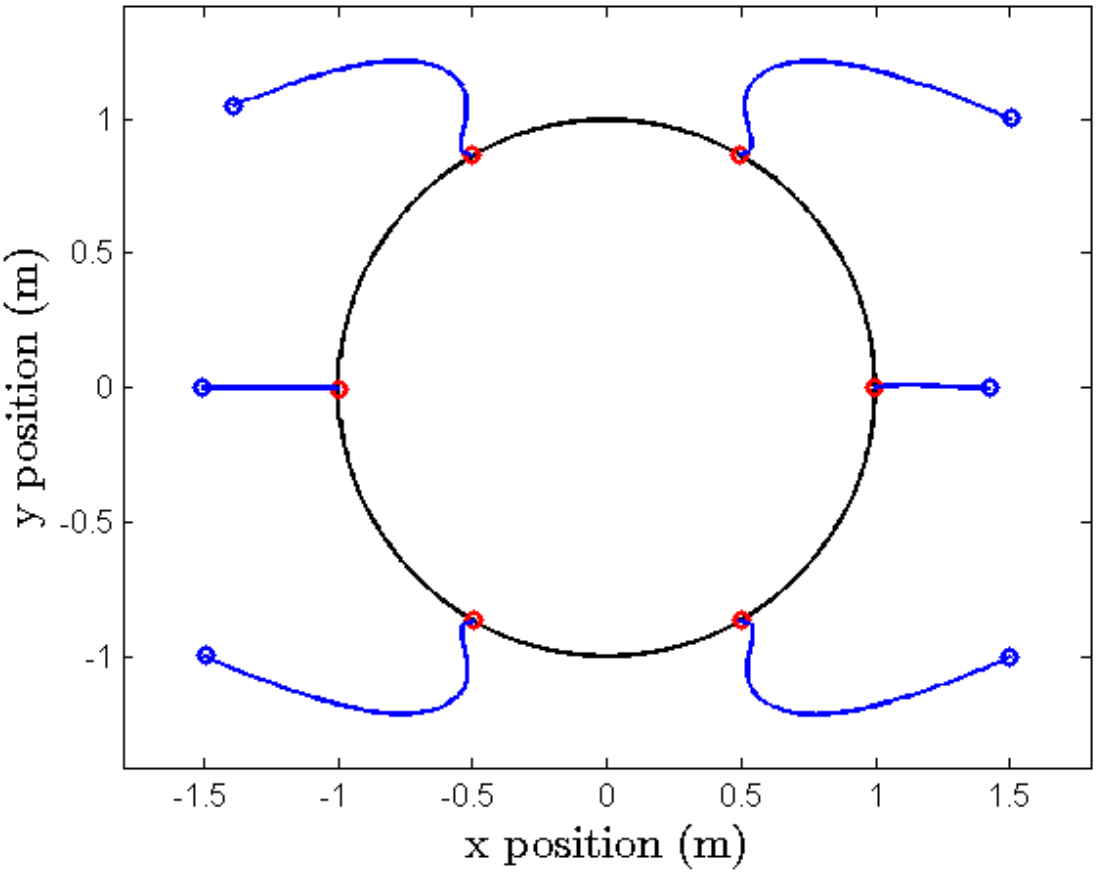


Figure 5.14: Simulation result: mutlirotor trajectories of a circular formation. The assembly of solid black lines represent the prescribed circle shape. The initial position (blue bubbles) of multirotors are $(-1.5,-1,1)^\top$, $(-1.5,0,1)^\top$, $(-1.5,1,1)^\top$, $(1.5,1,1)^\top$, $(1.5,0,1)^\top$, and $(1.5,-1,1)^\top$, respectively. The solid blue lines represent the flight trajectories. The red bubbles are the positions of UAVs when the formation task ends.

in Fig. 5.16).

5.4.6 Spherical Formation

Apart from the experiments on the cases of the formation in $\mathbb{R}^2$, we extended our control algorithm and adapted it to formation maneuvers in a 3D environment. We hereby demonstrate a case study on a spherical formation in $\mathbb{R}^3$ via a simulation of 5 hexarotor UAVs. Differently from the $\mathbb{R}^2$ cases, the communication between the multirotors in $\mathbb{R}^3$ is based on the complete graph as we have briefly introduced in Sec. 5.2.5. The resulting trajectories of the group of 5 UAVs are illustrated in Fig. 5.17.

In the experiment, the multirotors all took off and hovered at 1m height. They were

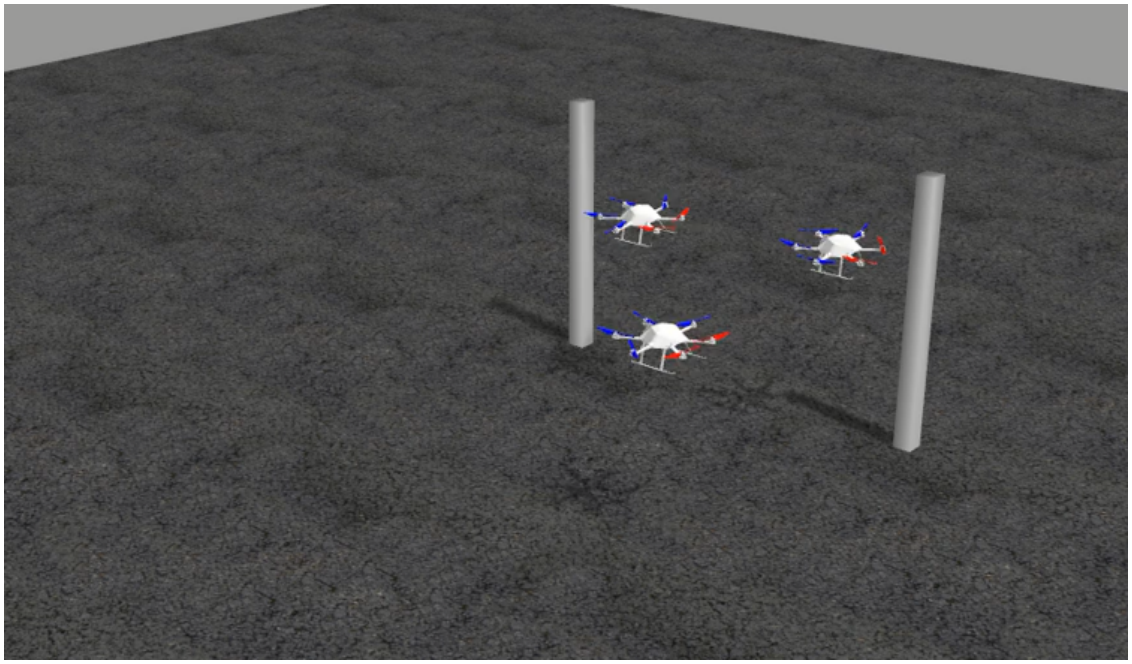


Figure 5.15: Screenshot of the "Moving formation through obstacles" simulation. 3 multirotor UAVs shrank to a circle of 0.5m radius at 1m height in order to fly through the obstacles.

then commanded to form a sphere in a 3D environment. Although the geometry and the radius of the formation shape is usually decided offline, it is not necessary to keep the target shape static. In this case, the center and the radius of the target sphere was set changing.

In the first stage, the target sphere (the color meshed sphere in Fig. 5.17a) was centering at $(0,1,2)^\top$ with a radius of 1m. The UAVs succeeded in converging to the target sphere from their initial positions (the blue bubbles). In the coming second stage, the center of the target sphere was kept but we expanded the radius to 1.3m. In Fig. 5.17b, we see that the UAVs reached balanced positions (the red bubbles) and converged to the surface of the expanded sphere. The multirotors were finally commanded to form a 1m-radius sphere centered at $(0,-0.2,1.6)^\top$ (the color meshed sphere in Fig. 5.17c) in the last stage.

5.5 Conclusion

In this chapter, we proposed a distributed formation control approach for a group of multirotor UAVs. The approach enables them to simultaneously achieve a balanced configuration on a prescribed shape, either in 2-dimensional (2D) or 3D. We combined our formation algorithm with a robust model predictive control method and implemented the

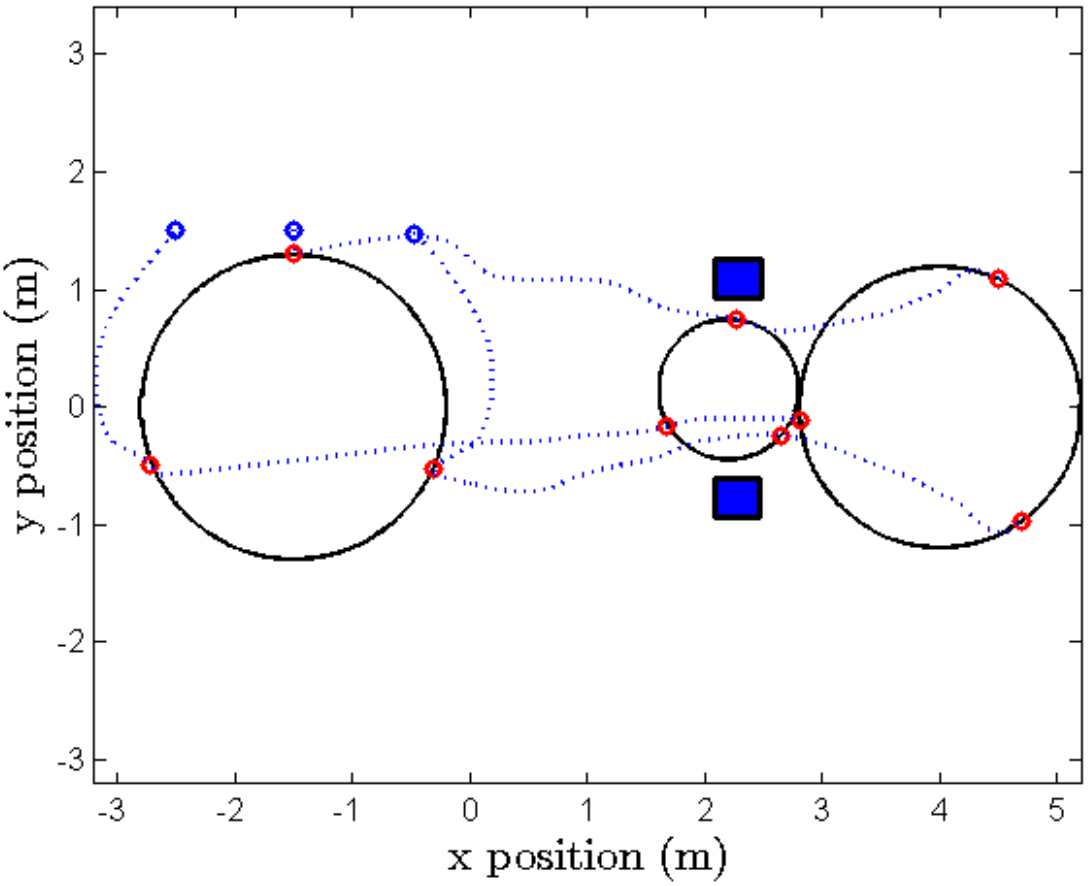


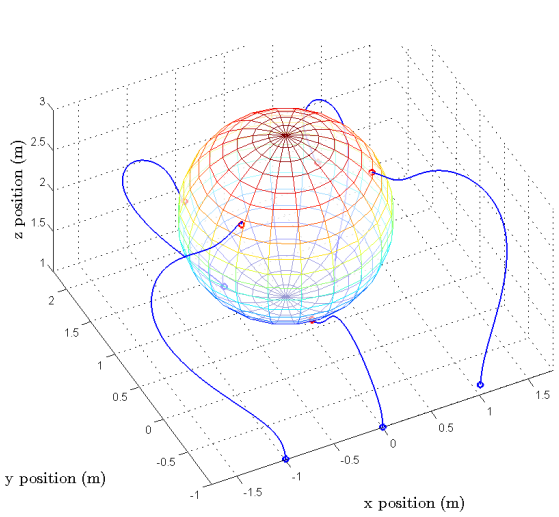
Figure 5.16: Simulation result: the horizontal projection of the resulting trajectories of 3 multirotor UAVs in the "Moving formation through obstacles" experiment. The blue bubbles represent the initial positions of UAVs. The dashed blue lines represent the trajectories of multirotors following the moving circular formation maneuvers. The solid black lines and the red bubbles display the target circle shape submanifold and the positions of 3 multirotors at 10s, 14s and 18s, respectively. The solid blue boxes represent the two gatepost obstacles.

complete framework on low-power onboard computational units. Both hardware-in-the-loop simulations and real-world experiments have validated that the distributed formation control approach is feasible for arbitrary, time-varying circular, triangular, rectangular or spherical formations.

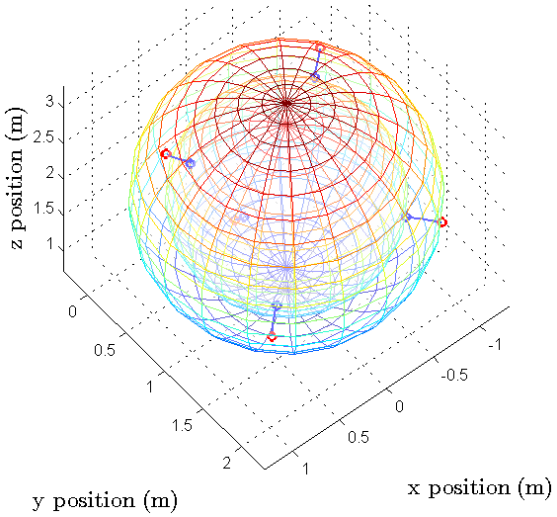
A supplementary video demonstration of this work can be found online[1].

Future work could include an extension of the formation algorithm in the 3D scenarios with multiple obstacles. Another potential improvement is to extend the formation

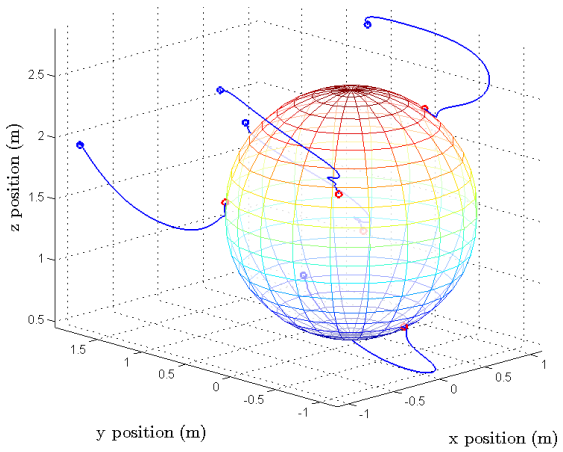
[1] `https://drive.google.com/open?id=0B4QuufsZAdQadEpHRTJ0Rmk3TlU`, accessed March-2017



(a) Stage 1



(b) Stage 2



(c) Stage 3

Figure 5.17: Simulation result: multirotor trajectories in spherical formation in $\mathbb{R}^3$. The solid blue lines represent the trajectories of multirotors following the moving spherical formation maneuvers. The blue bubbles are the initial positions of UAVs. The red bubbles are the positions of UAVs when they converged to the target sphere.

control approach with a human operator directing the swarm by "drawing" the desired (time-varying) formation shape and trajectory online.

Chapter 6

Conclusions

6.1 Summary

This thesis focuses on the control problem for the nontrivial tasks for multirotor UAVs. The major concerns of this thesis have been i) the feasibility of the multirotor to achieve complicated tasks, such as aggressive path following, collision-free navigation, and formation, etc., in the complex scenarios with disturbances and obstacles; ii) the efficiency and the robustness of the onboard implemented flight control approaches. We have worked on nonlinear control approaches to aggressive quadrotor maneuvers, first implemented them on one quadrotor and later extended to multiple multirotors, in order to enable a group of UAVs to achieve nontrivial tasks robustly. The proposed nonlinear control approaches provide versatility and robustness for multiple UAVs to achieve tasks autonomously in the environment with unknown disturbances and obstacles. Furthermore, these control approaches are sufficiently efficient to be implemented onboard the UAVs that have limited computational capability and executed in an online fashion.

In Chapter 3, we have started from solving the control problem for the multirotor UAV to stably achieve aggressive maneuvers during the path following and waypoint navigation tasks. The resulting control framework consists of a nonlinear attitude controller based on the solutions to global regulation problems for rigid body rotations SO3, a backstepping-like position controller, and a model predictive control (MPC) based online trajectory planner. The proposed control approach enables a quadrotor to autonomously follow the prescribed path with large tilts over 40°. We have further extended the nonlinear control framework to an improvement with a combination of a 6-dimensional (6D) force and torque observer, in order to lead the quadrotor to robust flight performance against unknown, time-varying disturbances, such as wind gusts, model mismatches, low battery voltage, etc. The complete nonlinear control approach has been validated not only via theoretical proofs of asymptotic stability, but also via a series of intensive real-robot experiments.

In Chapter 4, we have improved the versatility of the control for multirotor UAV to autonomous, collision-free flights in 3D environment. We have proposed a fast MPC method with nonlinear obstacle avoiding constraints for the multirotor UAV, which enables the multirotor to achieve tasks in the complicated environment. The MPC method

is onboard implemented on the quadrotor with a low-power computer so that the UAV can decide the strategy of avoiding obstacles based on the online update of the obstacle positions. By autonomously choosing to fly over the obstacles that are not too tall, or to fly around the tall obstacles, the UAV can achieve the waypoint navigation tasks in the scenarios with multiple obstacles at a speed that is not slow. The proposed MPC method has been experimentally validated via the trials in the area with randomly located box obstacles.

Finally in Chapter 5, we have extended our research from control for a single multirotor UAV to the control for a group of UAVs. We have proposed a distributed formation algorithm for the multi-agent system to retract and converge to a target shape in 2D and 3D environment and implemented the formation control approach onto multiple multirotors. With the combination of an onboard MPC method, the resulting approach leads a couple of multirotor UAVs to simultaneously converging to a target formation shape. In addition, by applying the human robot interaction device, the operator is able to "draw" a target shape, e.g. circle, triangle, rectangle, etc., via the finger motions. The presented formation control approach has been verified through a series of case studies based on hardware-in-the-loop simulations and real-robot experiments.

As a conclusion, in this thesis, we have focused on developing and validating the efficient and robust nonlinear control approaches for the multirotor UAVs to achieve nontrivial tasks. The control approaches we have proposed enable one multirotor UAV to perform aggressive waypoint navigation and path following tasks in the scenarios with disturbances and obstacles. The proposed approaches also enable a group of UAVs to fly in formation via following the "drawings" by the operator.

6.2 Future Work

Although certain promising results have been obtained, the approaches that we have proposed in this thesis still have their limitations. Meanwhile, several questions on our research topics are open and worthwhile to be exploited. Largely, we have discussed the limitations and potential improvements related to the work in the corresponding conclusion sections in individual chapters. We hereby discuss more general questions, and future research directions related to UAVs.

One open question is the aggressive maneuvers for multirotor UAVs with only onboard sensors. Since in this thesis we focus on the development and the validation of control algorithms, we still rely on the external tracking systems to provide the real-time position data of the UAVs. However, which restricts the flight of the UAVs to the areas that are equipped with infrared cameras or GPS signals. In order to extend the UAVs to a full autonomy in those GPS-denied areas, a potential solution is to equip the vision-based sensors, e.g. monocular and stereo cameras, or light-weight light detection and ranging (LIDAR) sensors onboard the UAVs for the purpose of self-localization. Much research has been carried out on the visual SLAM topic, however in most cases the

accuracy of the position estimates is not high enough to support the aggressive maneuvers of multirotor UAVs. Therefore, the tuning of flight controller has to be very conservative for the purpose of safety, which sacrifices the aggressiveness of the UAV maneuvering. So far, the solutions that combine the vision-based techniques together with the inertial sensors, i.e. IMU, are promising for the aggressive UAV maneuvers Shen *et al.* (2013), but those methods are still not robust in outdoor environment. Hence, compared to the implementation of novel control theories onto a single multirotor UAV, interdisciplinary research on vision-inertial-based aggressive UAV maneuvers is becoming popular.

A second open question can be the development of hybrid aerial robotics. The concept "*hybrid*" should not be restricted to a multirotor UAV that is designed waterproof and is able to dive into the water as a submarine. Instead, an UAV with at least two motion styles would be impressive. A possible solution can be a modular design of UAV that consists of two or more independent mobile robots, one multirotor and one wheeled robot for instance. Meanwhile, bio-inspired designs can be another potential to be exploited.

A third open question that attracts the keen interest from researchers is the cooperative motion and swarm of a networked robotics system, including the human-robot interaction among the operators and the networked system, namely *human-swarm interaction*. The development and implementation of robust formation control algorithms not only lead to the impressive demonstrations in air above the audience, but also enable the UAVs to achieve the complicated tasks that could not be completed separately, such as transporting heavy goods. In this thesis we have carried out early-stage research on distributed formation control for multiple UAVs and human-swarm interaction, while the improvement on the robustness of formation/swarm algorithms, as well as the closer and more user-friendly interaction concepts between the operators and UAVs, are two promising potential topics. It would be worthwhile to exploit these topics, so that in the near future, the end-user can operate a large group of UAVs to swarm in different shapes among the skyscrapers, by only using several fingers.

Bibliography

Albers, A., Trautmann, S., Howard, T., Nguyen, T. A., Frietsch, M., and Sauter, C. (2010). Semi-autonomous flying robot for physical interaction with environment. In *Robotics Automation and Mechatronics (RAM), 2010 IEEE Conference on*, pages 441–446.

Alexis, K., Nikolakopoulos, G., and Tzes, A. (2011). Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Engineering Practice*, **19**(10), 1195–1207.

AMD (2017). Amd zotac mini-pc. `https://www.zotac.com/us/product/mini_pcs/ma760-plus`. Online; accessed 17-Feb-2017.

Anderson, B. D. O., Yu, C., Fidan, B., and Hendrickx, J. M. (2008). Rigid graph control architectures for autonomous formations. *Control Systems Magazine, IEEE*, **28**(6), 48–63.

Antonelli, G., Arrichiello, F., Chiaverini, S., and Giordano, P. (2013). Adaptive trajectory tracking for quadrotor mavs in presence of parameter uncertainties and external disturbances. In *Advanced Intelligent Mechatronics (AIM), 2013 IEEE/ASME International Conference on*, pages 1337–1342.

Aswani, A., Bouffard, P., and Tomlin, C. (2012). Extensions of learning-based model predictive control for real-time application to a quadrotor helicopter. In *Proc. of the* 2012 *American Control Conference*, pages 4661–4666, Montreal, Canada.

Augugliaro, F. and D'Andrea, R. (2013). Admittance control for physical human-quadrocopter interaction. In *Control Conference (ECC), 2013 European*, pages 1805–1810.

Augugliaro, F., Lupashin, S., Hamer, M., Male, C., Hehn, M., Mueller, M. W., Willmann, J. S., Gramazio, F., Kohler, M., and DAndrea, R. (2014). The flight assembled architecture installation: Copperative construction with flying machines. *IEEE Control Systems Magazine*, **34**(4), 46–64.

Bae, Y. and Kim, J. (2004). Obstacle avoidance methods in the chaotic UAV. In *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, volume 2, pages 1.2.D.6–1.21–11 Vol.2.

Bailey, T. and Durrant-Whyte, H. (2006). Simultaneous localization and mapping (SLAM): Part ii. *IEEE Robotics & Automation Magazine*, **13**(3), 108–117.

Bellens, S., De Schutter, J., and Bruyninckx, H. (2012). A hybrid pose / wrench control framework for quadrotor helicopters. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2269–2274.

Boivin, E., Desbiens, A., and Gagnon, E. (2008). UAV collision avoidance using cooperative predictive control. In *Control and Automation, 2008 16th Mediterranean Conference on*, pages 682–688.

Boston Dynamics (2017). The two-legged wheeled robot handle. `https://www.youtube.com/watch?v=-7xvqQeoA8c`. Online; accessed 17-Feb-2017.

Bouabdallah, S. and Siegwart, R. (2005). Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 47–52, Barcelona, Spain.

Bouabdallah, S. and Siegwart, R. (2007). Full control of a quadrotor. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 153–158, San Diego, USA.

Bouabdallah, S., Becker, M., de Perrot, V., and Siegwart, R. (2007). Toward obstacle avoidance on quadrotors. In *Proceedings of the* 12*th International Symposium on Dynamic Problems of Mechanics (DINAME 2007)*, pages 1–10.

Bouffard, P., Aswani, A., and Tomlin, C. (2012). Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *Proc.* 2012 *ICRA*, pages 279–284.

Bruijnen, D., van Helvoort, J., *et al.* (2007). Realtime motion path generation using subtargets in a rapidly changing environment. *Robotics and Autonomous Systems*, **55**, 470–479.

Cesetti, A., Frontoni, E., Mancini, A., Zingaretti, P., and Longhi, S. (2010). A vision-based guidance system for UAV navigation and safe landing using natural landmarks. In *Selected papers from the 2nd International Symposium on UAVs, 2009*, pages 233–257.

Chauffaut, C., Defay, F., Burlion, L., and de Plinval, H. (2016). UAV obstacle avoidance scheme using an output to input saturation transformation technique. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 227–234.

D'Andrea, R. (2016). Meet the dazzling flying machines of the future. `https://www.ted.com/talks/raffaello_d_andrea_meet_the_dazzling_flying_machines_of_the_future`. TED Talk.

De Luca, A. and Mattone, R. (2003). Actuator failure detection and isolation using generalized momenta. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 1, pages 634–639 vol.1.

De Luca, A., Albu-Schaffer, A., Haddadin, S., and Hirzinger, G. (2006). Collision detection and safe reaction with the dlr-iii lightweight manipulator arm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 1623–1630.

Derafa, L., Benallegue, A., and Fridman, L. (2012). Super twisting control algorithm for the attitude tracking of a four rotors UAV. *Journal of the Franklin Institute*, **349**(2), 685–699.

DJI (2017). Uav: Dji inspire 2. `http://www.dji.com/inspire-2`. Online; accessed 17-Feb-2017.

Fekete, M. (1923). über die verteilung der wurzeln bei gewissen algebraischen gleichungen mit ganzzahligen koeffizienten. *Mathematische Zeitschrift*, **17**(1), 228–249.

Franchi, A., Masone, C., VolkerGrabe, Ryll, M., Bülthoff, H. H., and Giordano, P. R. (2012). Modeling and control of UAV bearing formations with bilateral high-level steering. *The international Journal of Robotics Research*, **31**(12), 1504–1525.

Fraundorfer, F., Heng, L., Honegger, D., Lee, G. H., Meier, L., Tanskanen, P., and Pollefeys, M. (2012). Vision-based autonomous mapping and exploration using a quadrotor MAV. In *Proc. of* 2012 *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4557–4564.

Grabe, V., Riedel, M., Bülthoff, H. H., Giordano, P. R., and Franchi, A. (2013). The TeleKyb framework for a modular and extendible ROS-based quadrotor control. In *Proc.* 2013 *European Conference on Mobile Robots*, pages 19–25.

Gultepe, E., Yamanaka, S., Laflin, K. E., Kadam, S., Shim, Y., Olaru, A. V., Limketkai, B., Khashab, M. A., Kalloo, A. N., Gracias, D. H., and Selaru, F. M. (2013). Biologic tissue sampling with untethered microgrippers. *Gastroenterology*, **144**(4), 691–693.

Hacksel, P. and Salcudean, S. (1994). Estimation of environment forces and rigid-body velocities using observers. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 931–936 vol.2.

Hardkernel (2017). Odroid xu4. `http://www.hardkernel.com/main/products/prdt_info.php?g_code=G143452239825`. Online; accessed 17-Feb-2017.

Hartley, E. N., Jerez, J. L., Suardi, A., Maciejowski, J. M., Kerrigan, E. C., and Constantinides, G. A. (2014). Predictive control using an FPGA with application to aircraft control. *IEEE Transactions on Control Systems Technology*, **22**(3), 1006–1017.

Heng, L., Meier, L., Tanskanen, P., Fraundorfer, F., and Pollefeys, M. (2011). Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using on-board processing. In *Proc. of* 2011 *IEEE International Conference on Robotics and Automation*, Shanghai, China.

HiSystems (2017). Mikrokopter quadroxl. `http://wiki.mikrokopter.de/en/MK-QuadroXL`. Online; accessed 17-Feb-2017.

Hoffmann, G. M., Huang, H., Waslander, S. L., and Tomlin, C. J. (2007). Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA Guidance, Navigation and Control Conference*, pages 1–20.

Intel (2017a). Intel nuc mini-pc. `http://www.intel.com/content/www/us/en/nuc/products-overview.html`. Online; accessed 17-Feb-2017.

Intel (2017b). Pepsi zero super bowl LI halftime show. `https://www.youtube.com/watch?v=bT_BqmTgOUw`.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, **82**, 35–45.

Khalil, H. K. (2002). *Nonlinear Systems (Third Edition)*. Prentice Hall.

Khalil, W. and Dombre, E. (2004). *Modeling, Identification and Control of Robots*. Butterworth-Heinemann, Oxford.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*.

Krick, L., Broucke, L., and Francis, B. (2009). Stabilization of infinitesimally rigid formations of multi-robot networks. *International Journal of Control*, **82**(3), 423–439.

Langson, W., Chryssochoos, I., Rakovic, S., and Mayne, D. (2004). Robust model predictive control using tubes. *Automatica*, **40**(1), 125–133.

Leap Motion, I. (2017). Leap motion 3d gesture tracking device. `https://www.leapmotion.com/`. Online; accessed 17-Feb-2017.

Lee, T., Leok, M., and McClamroch, N. H. (2010a). Control of complex maneuvers for a quadrotor UAV using geometric methods on SE(3). arXiv technical report arXiv:1003.2005.

Lee, T., Leok, M., and McClamroch, N. H. (2010b). Geometric tracking control of a quadrotor UAV on SE(3). In *Proc. of* 49*th IEEE Conference on Decision and Control*, pages 5420–5425, Atlanta, USA.

Liu, Y., Montenbruck, J. M., Stegagno, P., Allgöwer, F., and Zell, A. (2015). A robust nonlinear controller for nontrivial quadrotor maneuvers: Approach and verification. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 600–606.

Liu, Y., Rajappa, S., Montenbruck, J. M., Stegagno, P., Bülthoff, H., Allgöwer, F., and Zell, A. (2017). Robust nonlinear control approach to nontrivial quadrotor maneuvers and obstacle avoidance for quadrotor UAV under disturbances. *Robotics and Autonomous Systems*, **98**, 317–332.

Liu, Y., Montenbruck, J. M., Odelga, M., Zelazo, D., Rajappa, S., Bülthoff, H., Allgöwer, F., and Zell, A. (2018). A distributed control approach to formation balancing and maneuvering of multiple multirotor UAVs. *IEEE Transactions on Robotics, Special Issue: Swarm Aerial Robotics (to appear)*.

Lupashin, S., Schöllig, A., Sherback, M., and D'Andrea, R. (2010). A simple learning strategy for high-speed quadrocopter multi-flips. In *Proc. of* 2010 *IEEE International Conference on Robotics and Automation*, pages 1642–1648, Alaska, USA.

Mattingley, J., Wang, Y., and Boyd, S. (2011). Automatic generation of high-speed solvers. *IEEE Control Systems Magazine*, **31**(3), 52–65.

Mayne, D., Seron, M., and Rakovic, S. (2005). Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, **41**(2), 219–224.

Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *Proc. of* 2011 *IEEE International Conference on Robotics and Automation*, pages 2520–2525, Shanghai, China.

Mellinger, D., Michael, N., and Kumar, V. (2012). Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The international Journal of Robotics Research*, **31**(5), 664–674.

Michael, N., Mellinger, D., Lindsey, Q., and Kumar, V. (2010). The grasp multiple micro-UAV testbed. *Robotics & Automation Magazine, IEEE*, **17**(3), 56–65.

Michael, N., Fink, J., and Kumar, V. (2011). Cooperative manipulation and transportation with aerial robots. *Autonomous Robots*, **30**(1), 73–86.

Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., Okada, Y., Kiribayashi, S., Otake, K., Yoshida, K., Ohno, K., Takeuchi, E., and Tadokoro, S. (2012). Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, **29**(5), 832–841.

Montenbruck, J. M., Zelazo, D., and Allgöwer, F. (2017). Fekete points, formation control, and the balancing problem. *IEEE Transactions on Automatic Control (accepted)*.

Moral, P. D. (1996). Non linear filtering: Interacting particle solution. *Markov Processes and Related Fields*, **2**(4), 555–580.

Mueggler, E., Faessler, M., Fontana, F., and Scaramuzza, D. (2014). Aerial-guided navigation of a ground robot among movable obstacles. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–8, Hokkaido.

Naturalpoint (2017). Optitrack infrared-camera tracking system. `http://www.naturalpoint.com/optitrack/products/tracking-tools-bundles`. Online; accessed 17-Feb-2017.

Nguyen, H.-N. and Lee, D. (2013). Hybrid force/motion control and internal dynamics of quadrotors for tool operation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3458–3464.

Nonami, K., Kendoul, F., Suzuki, S., Wang, W., and Nakazawa, D. (2010). *Autonomous Flying Robots: Unmanned Aerial Vehicles and Micro Aerial Vehicles*. Springer.

Odelga, M., Stegagno, P., and Bülthoff, H. H. (2016). Obstacle detection, tracking and avoidance for a teleoperated UAV. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2984–2990.

Oh, K.-K., Park, M.-C., and Ahn, H.-S. (2015). A survey of multi-agent formation control. *Automatica*, **53**, 424–440.

Palunko, I., Cruz, P., and Fierro, R. (2012a). Agile load transportation : Safe and efficient load manipulation with aerial robots. *Robotics Automation Magazine, IEEE*, **19**(3), 69–79.

Palunko, I., Cruz, P., and Fierro, R. (2012b). Agile load transportation: Safe and efficient load manipulation with aerial robots. *IEEE Robotics and Automation Magazine*, **19**(3), 69–79.

Park, J. and Kim, Y. (2012). Stereo vision based collision avoidance of quadrotor UAV. In *Control, Automation and Systems (ICCAS), 2012 12th International Conference on*, pages 173–178.

Paul, T., Krogstad, T. R., and Gravdahl, J. T. (2008). UAV formation flight using 3d potential field. In *Control and Automation, 2008 16th Mediterranean Conference on*, pages 1240–1245.

Pololu (2017). Step-down voltage regulator. `https://www.pololu.com/product/2865`. Online; accessed 17-Feb-2017.

Raffo, G. V., Ortega, M. G., and Rubio, F. R. (2010). An integral predictive/nonlinear H_∞ control structure for a quadrotor helicopter. *Automatica*, **46**(1), 29–39.

Rakovic, S. and Baric, M. (2010). Parameterized robust control invariant sets for linear systems: Theoretical advances and computational remarks. *IEEE Trans. Automatic Control*, **55**(7), 1599–1614.

Regula, G. and Lantos, B. (2014). Formation control of a large group of UAVs with safe path planning and obstacle avoidance. In *Control Conference (ECC), 2014 European*, pages 1522–1529.

Ren, W. R. W. (2006). Consensus based formation control strategies for multi-vehicle systems. In *2006 American Control Conference*, pages 4237–4242.

Ritz, R., Müller, M. W., Hehn, M., and D'Andrea, R. (2012). Cooperative quadrocopter ball throwing and catching. In *Proc. of* 2012 *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4972–4978.

Roberts, A. and Tayebi, A. (2009). Adaptive position tracking of vtol UAVs. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 5233–5238.

Saha, S., Natraj, A., and Waharte, S. (2014). A real-time monocular vision-based frontal obstacle detection and avoidance for low cost UAVs in gps denied environment. In *Aerospace Electronics and Remote Sensing Technology (ICARES), 2014 IEEE International Conference on*, pages 189–195.

Schilling, R. (1990). *Fundamentals of robotics: analysis and control*. Prentice Hall.

Schmidt, G. S., Ebenbauer, C., and Allgöwer, F. (2013a). On the differential equation $\dot{\theta} = (\theta^\top - \theta)\theta$ with $\theta \in \mathrm{SO}(n)$. arXiv technical report arXiv:1308.6669.

Schmidt, G. S., Ebenbauer, C., and Allgöwer, F. (2013b). Output regulation for attitude control: a global approach. In *Proc. of* 2013 *American Control Conference*, pages 5251–5256, Washington, USA.

Shen, S., Mulgaonkar, Y., Michael, N., and Kumar, V. (2013). Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor. In *Robotics: Science and Systems*.

Shin, J. and Kim, H. J. (2009). Nonlinear model predictive formation flight. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, **39**(5), 1116–1125.

Siegwart, R., Nourbakhsh, I., and Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots*. Intelligent robotics and autonomous agents. MIT Press.

Simpson, A. and Sabo, C. (2016). Quadcopter obstacle avoidance using biomimetic algorithms. *American Institute of Aeronautics and Astronautics, AIAA SciTech*.

Softbank (2017). Humanoid robot: Pepper. `https://www.ald.softbankrobotics.com/en/cool-robots/pepper`. Online; accessed 17-Feb-2017.

Stanford Robotics Lab, S. U. (2017). Humanoid robotic diver oceanone. `http://cs.stanford.edu/group/manips/ocean-one.html`. Online; accessed 17-Feb-2017.

Systems, . D. (2017). Catia v5r20. `https://www.3ds.com/products-services/catia/`. Online; accessed 17-Feb-2017.

Takakura, S., Murakami, T., and Ohnishi, K. (1989). An approach to collision detection and recovery motion in industrial robot. In *Industrial Electronics Society, 1989. IECON '89., 15th Annual Conference of IEEE*, pages 421–426 vol.2.

Tomic, T. and Haddadin, S. (2015). Simultaneous estimation of aerodynamic and contact forces in flying robots: Applications to metric wind estimation and collision detection. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 5290–5296.

Turpin, M., Michael, N., and Kumar, V. (2012). Decentralized formation control with variable shapes for aerial robots. In *Proc. 2012 ICRA*.

Vicon (2017). Vero infrared-camera tracking system. `https://www.vicon.com/products/camera-systems/vero`. Online; accessed 17-Feb-2017.

Wächter, A. and Biegler, L. T. (2006). On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, **106**(1), 25–57.

Wang, C., Liu, W., and Meng, M. Q. H. (2015). Obstacle avoidance for quadrotor using improved method based on optical flow. In *Information and Automation, 2015 IEEE International Conference on*, pages 1674–1679.

Wang, J. and Xin, M. (2013). Integrated optimal formation control of multiple unmanned aerial vehicles. *IEEE Transactions on Control Systems Technology*, **21**(5), 1731–1744.

Wang, X., Yadav, V., and Balakrishnan, S. N. (2007). Cooperative UAV formation flying with obstacle/collision avoidance. *IEEE Transactions on Control Systems Technology*, **15**(4), 672–679.

Waymo, A. I. (2017). Google self-driving car. `https://waymo.com/journey/`. Online; accessed 17-Feb-2017.

Xu, F., Chen, H., Gong, X., and Mei, Q. (2016). Fast nonlinear model predictive control on fpga using particle swarm optimization. *IEEE Transactions on Industrial Electronics*, **63**(1), 310–321.

Yuan-yan, H. and Ying-xun, W. (2011). Stereo vision-based fast obstacles avoidance without obstacles discrimination for indoor UAVs. In *Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), 2011 2nd International Conference on*, pages 4332–4337.

Yüksel, B., Secchi, C., Bülthoff, H., and Franchi, A. (2014). A nonlinear force observer for quadrotors and application to physical interactive tasks. In *Advanced Intelligent Mechatronics (AIM), 2014 IEEE/ASME International Conference on*, pages 433–440.

Zelazo, D., Giordano, P. R., and Franchi, A. (2015). Bearing-only formation control using an SE(2) rigidity theory. In *IEEE Conference on Decision and Control*, pages 6121–6126. IEEE.

Zengin, U. and Dogan, A. (2007). Real-time target tracking for autonomous UAVs in adversarial environments: A gradient search algorithm. *IEEE Transactions on Robotics*, **23**(2), 294–307.

Zhao, S. and Zelazo, D. (2015). Bearing rigidity and almost global bearing-only formation stabilization. *IEEE Trans. on Automatic Control*, **61**(5), 1255—1268.

Zhou, D. and Schwager, M. (2015). Virtual rigid bodies for coordinated agile maneuvering of teams of micro aerial vehicles. In *Proc. 2015 ICRA*.